**Java 8**

**Diagramme**

OOA

OOD

OOP

OOD

OOA

OOP

OOA

OOP

OOD

OOP

OOA

OOD

OOD

OOP

OOA

OOP

OOD

OOA

Johannes Nowak

Johannes Nowak

e-mail: johannes.nowak@t-online.de

Mai 2019

**Inhalt**

[1 Typen von Klassen 5](#_Toc8279952)

[1.1 Top Level Classes 5](#_Toc8279953)

[1.2 Static Member-Classes 6](#_Toc8279954)

[1.3 Non-Static Member Classes 7](#_Toc8279955)

[1.4 Local Classes 8](#_Toc8279956)

[1.5 Anonymous Classes 9](#_Toc8279957)

[1.6 Lambda-Classes 10](#_Toc8279958)

[2 Anonyme Klassen vs. Lambdas 11](#_Toc8279959)

[2.1 Anonyme Klassen 11](#_Toc8279960)

[2.2 Lambdas 12](#_Toc8279961)

[3 DelegatingFocusListener 13](#_Toc8279962)

[3.1 FocusHandler 13](#_Toc8279963)

[3.2 Consumer 14](#_Toc8279964)

[4 Operators 15](#_Toc8279965)

[4.1 Parameter 15](#_Toc8279966)

[4.2 Map 16](#_Toc8279967)

[5 Closures 17](#_Toc8279968)

[6 Worker 18](#_Toc8279969)

[6.1 Top Level Class 18](#_Toc8279970)

[6.2 Local Class – Worker.this 19](#_Toc8279971)

[6.3 Local Class – Closure 20](#_Toc8279972)

[6.4 Anonymous Class 21](#_Toc8279973)

[6.5 Lambda Class 22](#_Toc8279974)

[7 Funktionale Interfaces 23](#_Toc8279975)

[8 ArrayList – XArrayList 24](#_Toc8279976)

[8.1 ArrayList - forEach 24](#_Toc8279977)

[8.2 XArrayList – filter 25](#_Toc8279978)

[8.3 XArrayList – map 26](#_Toc8279979)

[8.4 XArrayList – reduce 27](#_Toc8279980)

[9 SimpleStream 28](#_Toc8279981)

[9.1 Step 1 28](#_Toc8279982)

[9.2 Step 2 29](#_Toc8279983)

[9.3 Step 3 30](#_Toc8279984)

# Typen von Klassen

## Top Level Classes

: JButton

"Plus"

addActionListener

: JButton

"Minus"

addActionListener

: ButtonPlusAdapter

CTOR

*ActionListener*

mathFrame

: ButtonMinusAdapter

actionPerformed

CTOR

*ActionListener*

mathFrame

: MathFrame

onPlus

onMinus

:JTextField

:JTextField

:JTextField

actionPerformed

## Static Member-Classes

: JButton

"Plus"

addActionListener

: JButton

"Minus"

addActionListener

: MathFrame$

ButtonPlusAdapter

actionPerformed

CTOR

*ActionListener*

mathFrame

: MathFrame$

ButtonMinusAdapter

actionPerformed

CTOR

*ActionListener*

mathFrame

: MathFrame

onPlus

onMinus

:JTextField

:JTextField

:JTextField

## Non-Static Member Classes

: JButton

"Plus"

addActionListener

: JButton

"Minus"

addActionListener

: MathFrame$

ButtonPlusAdapter

actionPerformed

CTOR

*ActionListener*

MathFrame.this

: MathFrame$

ButtonMinusAdapter

actionPerformed

CTOR

*ActionListener*

MathFrame.this

: MathFrame

onPlus

onMinus

:JTextField

:JTextField

:JTextField

## Local Classes

: JButton

"Plus"

addActionListener

: JButton

"Minus"

addActionListener

: MathFrame$1

ButtonPlusAdapter

actionPerformed

CTOR

*ActionListener*

MathFrame.this

: MathFrame$1

ButtonMinusAdapter

actionPerformed

CTOR

*ActionListener*

MathFrame.this

: MathFrame

onPlus

onMinus

:JTextField

:JTextField

:JTextField

## Anonymous Classes

: JButton

"Plus"

addActionListener

: JButton

"Minus"

addActionListener

: MathFrame$1

actionPerformed

CTOR

*ActionListener*

MathFrame.this

: MathFrame$2

actionPerformed

CTOR

*ActionListener*

MathFrame.this

: MathFrame

onPlus

onMinus

:JTextField

:JTextField

:JTextField

## Lambda-Classes

: JButton

"Plus"

addActionListener

: JButton

"Minus"

addActionListener

: ??Lambda1??

actionPerformed

CTOR

*ActionListener*

MathFrame.this

: ??Lambda2??

actionPerformed

CTOR

*ActionListener*

MathFrame.this

: MathFrame

onPlus

onMinus

:JTextField

:JTextField

:JTextField

# Anonyme Klassen vs. Lambdas

## Anonyme Klassen

: MathFrame$1

focusGained (e)

MathFrame.this

CTOR

helper ()

foo

42

*FocusListener*

this

focusLost (e)

: MathFrame

registerListeners()

creates

## Lambdas

: $$$

actionPerformed (e)

MathFrame.this

CTOR

*ActionListener*

this

: MathFrame

registerListeners()

creates

# DelegatingFocusListener

## FocusHandler

: JTextField

addFocusListener

: Delegating

FocusListener

focusGained (e)

focusLost (e)

*FocusListener*

handle (e)

*FocusHandler*

lostHandler

handle (e)

*FocusHandler*

gainedHandler

CTOR

delegates

delegates

## Consumer

: JTextField

addFocusListener

: Delegating

FocusListener

focusGained (e)

focusLost (e)

*FocusListener*

CTOR

gainedHandler

accept (e)

*Consumer*

*<FpcusEvent>*

*Consumer*

*<FpcusEvent>*

lostHandler

accept (e)

delegates

delegates

# Operators

## Parameter

: $$$

apply(x, y)

*Operator*

x + y

plus

: $$$

apply(x, y)

*Operator*

x - y

minus

: MathFrame

onCalculate (op)

op.apply(x, y)

## Map

: HashMap

: $$$

apply(x, y)

*Operator*

x - y

"Plus"

: $$$

apply(x, y)

*Operator*

x - y

"Minus"

: $$$

apply(x, y)

*Operator*

x \* y

"Minus"

# Closures

: $$$

actionPerformed (e)

MathFrame.this

CTOR

*ActionListener*

this

: MathFrame

registerListeners()

creates

: Point

point

point

# Worker

## Top Level Class

: Combiner

work()

andThen(w)

*Worker*

CTOR

first

second

: $$$

work()

andThen(w)

*Worker*

: $$$

work()

andThen(w)

*Worker*

w1

w3

w2

## Local Class – Worker.this

w3

*Worker*

: $$$

work()

andThen(w)

*Worker*

: $$$

work()

andThen(w)

*Worker*

w1

w2

: Worker$1

Combiner

work()

andThen(w)

CTOR

Worker

.this

second

creates

## Local Class – Closure

: $$$

work()

andThen(w)

*Worker*

: $$$

work()

andThen(w)

*Worker*

w1

w2

: Worker$1

Combiner

work()

andThen(w)

w3

*Worker*

CTOR

Worker

.this

w

copy

## Anonymous Class

: $$$

work()

andThen(w)

*Worker*

: $$$

work()

andThen(w)

*Worker*

w1

w2

: Worker$1

work()

andThen(w)

w3

*Worker*

CTOR

Worker

.this

w

copy

## Lambda Class

: $$$

work()

andThen(w)

*Worker*

: $$$

work()

andThen(w)

*Worker*

w1

w2

: $$$

work()

andThen(w)

w3

*Worker*

CTOR

Worker

.this

w

copy

# Funktionale Interfaces

**Runnable**

void run()

**Supplier**

R get()

R

**Consumer**

void accept(T)

T

**BiFunction**

R apply(T0, T1)

R

T0

T1

**BinaryOperator**

T apply(T, T)

T

T

T

**UnaryOperator**

T apply(T)

T

T

**Function**

R apply(T)

R

T

**Predicate**

boolean test(T)

T

boolean

# ArrayList – XArrayList

## XArrayList - forEach

: XArrayList

"red"

"green"

"blue"

forEach

: $$$

accept

*Consumer*

## XArrayList – filter

: XArrayList

"red"

"green"

"blue"

filter

: $$$

test

*Predicate*

: XArrayList

creates

## XArrayList – map

: XArrayList

"red"

"green"

"blue"

map

: $$$

apply

*Function*

: XArrayList

creates

3

5

4

## XArrayList – reduce

: XArrayList

"red"

"green"

"blue"

reduce

"redgreenblue"

creates

: $$$

apply

*Binary*

*Operator*

# SimpleStream

## Step 1

: StartNode

map

filter

forEach

SimpleStream.of(…)

creates

creates

creates

: MapNode

map

filter

forEach

: FilterNode

map

filter

forEach

*SimpleStream*

*SimpleStream*

*SimpleStream*

StartNode

MapNode

FilterNode

*Node*

map, filter, forEach

*SimpleStream*

of (static)

*map, filter, forEach*

## Step 2

*SimpleStream*

*SimpleStream*

*SimpleStream*

: StartNode

map

filter

forEach

10

11

12

13

14

SimpleStream.of(…)

creates

creates

creates

: MapNode

map

filter

forEach

apply

*Function*

: FilterNode

map

filter

forEach

test

*Predicate*

StartNode

MapNode

FilterNode

Array

Function

Predicate

## Step 3

: MapNode

map

filter

forEach

previous

get

apply

*Function*

: FilterNode

map

filter

forEach

previous

get

test

*Predicate*

: StartNode

map

filter

forEach

previous

get

10

11

12

13

14

SimpleStream.of(…)

creates

creates

creates

*SimpleStream*

*SimpleStream*

*SimpleStream*

Array

Function

Predicate

StartNode

get

MapNode

get

FilterNode

get

*Node*

previous

*get*

map, filter, forEach