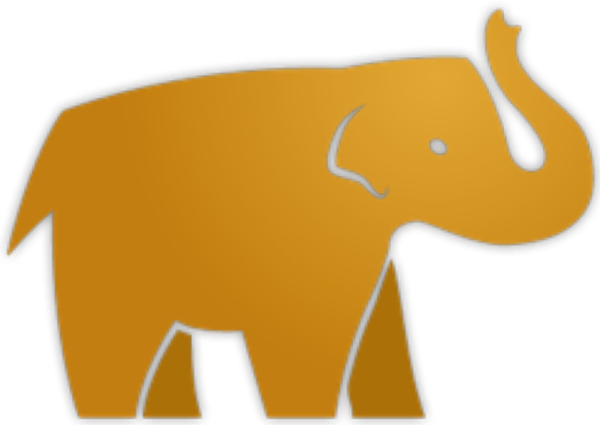
Frischer Wind für die JVM – sechs Programmiersprachen im Vergleich

Simon Olofsson • Content Management AG
@solofs • so@cm4all.com


Groovy




Kotlin
by JetBrains


FANTOM

Groovy

```groovy
def sayHello(name) {
  println "Hallo $name!"
}

sayHello "JavaLand"
```

```
% groovy hello_world
Hallo JavaLand!
```

```groovy
Range range = 1..10

def doubled = range.collect {
  it * 2
}
```

```groovy
def list = ["Star", "Dot", "Operator"]
```

```groovy
def loweredList = list*.toLowerCase()
```

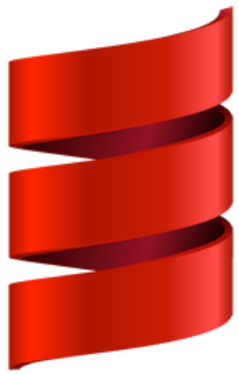
```groovy
def map = [
  function: {
    println "Eine Funktion in einer Map!"
  }
]

map.function()
```

```groovy
def conference = [name: "JavaLand"]

println conference.name ?
  conference.name : "Unknown"

println conference.name ?: "Unknown"

conference.name?.toLowerCase()
```

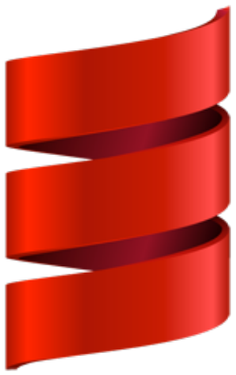
```groovy
def builder =
  new groovy.xml.MarkupBuilder()

builder.html {
  body {
    a(href: 'http://groovy.codehaus.org',
      "Mit Groovy erstellt")
  }
}
```

```
<html>
 <body>
   <a
      href='http://groovy.codehaus.org'>
      Mit Groovy erstellt</a>
 </body>
</html>
```

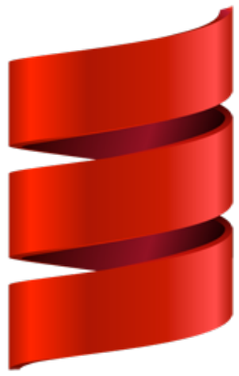
Scala



```
def sayHello(name: String) {
  println(s"Hallo $name!")
}

sayHello("JavaLand")
```

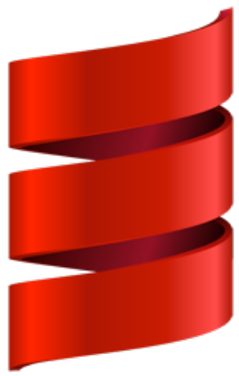


```
% scala helloWorld.scala
Hallo JavaLand!
```

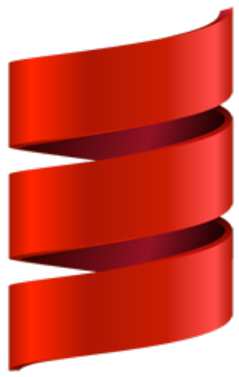


```scala
val setA: Set[Int] = Set(1, 2, 3)
val setB = Set(1, 2, 3)
```

```scala
def doubleA(i: Int): Int = i * 2
def doubleB(i: Int) = i * 2
```

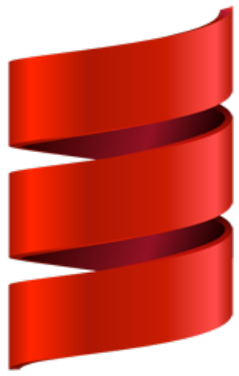


```
val even = for {
  i <- 1 to 10
  if i % 2 == 0
} yield i

assert(List(2, 4, 6, 8, 10) == even)
```



```scala
case class Conference(name: String)

val javaLand = Conference("JavaLand")
```

```scala
javaLand match {
  case Conference("JavaLand") =>
    println("JavaLand!")
  case _ =>
    println("Default Case")
}
```



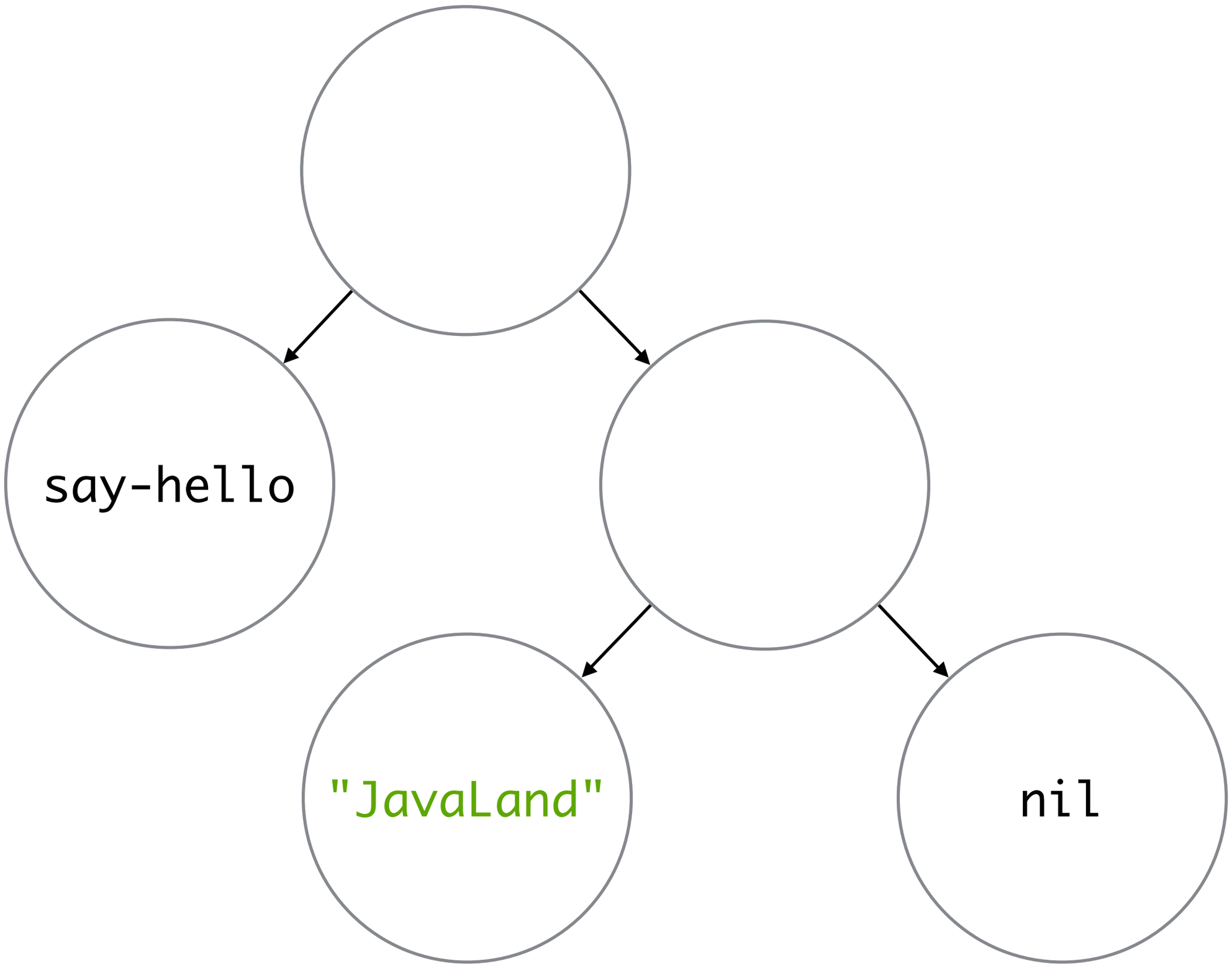
```
trait Schwimmen { def schwimme() {} }

trait Fliegen { def fliege() {} }

abstract class Vogel

class Pinguin extends Vogel with Schwimmen
class Adler extends Vogel with Fliegen
```

Clojure

```clojure
(defn say-hello [name]
 (println (format "Hallo %s!" name)))

(say-hello "JavaLand")
```

```
% lein hello-world
Hallo JavaLand!
```

(say-hello "JavaLand")



```
'(println "Eine literale Liste")

(println "Eine evaluierte Liste (Form)")

(eval '(println "Eine literale Liste"))
```

```clojure
["Ein" "Vektor"]
(vector "Ein" "Vektor")

#{"Ein" "Set"}

{:schluessel "Eine Map"}

(let [name "JavaLand"] (println name))
```

```
(defn double-from [n]
 (cons (* n 2) (double-from (inc n))))

(println (take 4 (double-from 4)))
```

```
% lein non-lazy
Exception in thread “main"
java.lang.StackOverflowError
```

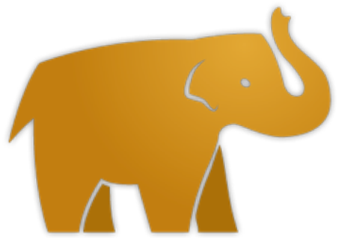
```clojure
(defn double-from [n]
 (cons (* n 2)
       (lazy-seq (double-from (inc n)))))

(println (take 4 (double-from 4)))
```

```
% lein lazy
(8 10 12 14)
```

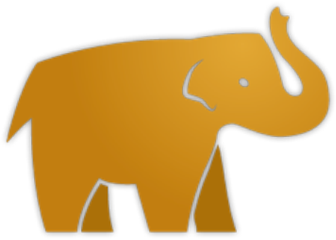
```clojure
(defmacro dbg [body]
  `(let [x# ~body]
     (println "dbg:" '~body "=" x#)
     x#))

(dbg (+ 1 2))
```

http://www.jayway.com/2011/03/13/dbg-a-cool-little-clojure-macro/

```
% lein macro
dbg: (+ 1 2) = 3
```

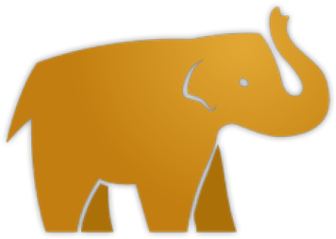


Ceylon

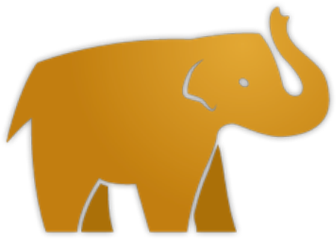


```
void sayHello(String name) {
  print("Hallo ``name``!");
}
```

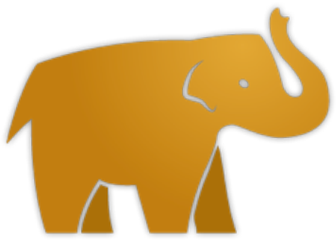
```
void run() {
  sayHello("JavaLand");
}
```



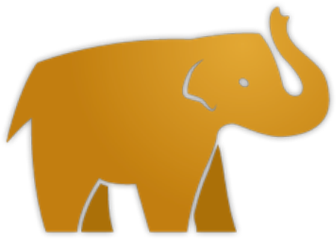
```
% ceylon compile de.olofsson
% ceylon run de.olofsson
Hallo JavaLand!
```



```
// module.ceylon:
module de.olofsson "1.0.0" {
  import ceylon.collection "1.0.0";
}


// useMap.ceylon:
import ceylon.collection { HashMap }
```



```
import ceylon.language.meta
  { annotations }

"Eine Funktion mit Annotationen."
by ("Simon Olofsson")
void annotationsDemo() {

  value a = annotations(`DocAnnotation`,
    `function annotationsDemo`);
}
```



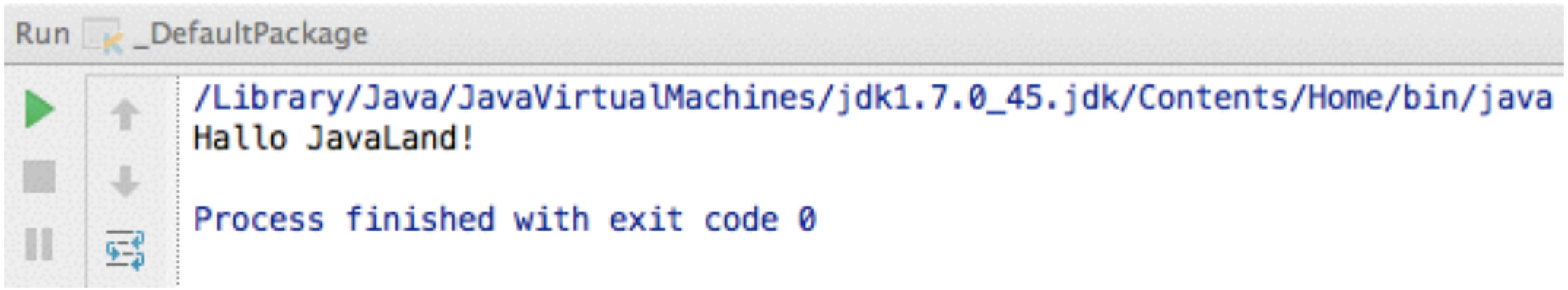
```
List<String> l1 = ["Eins"];

String? s1 = l1.get(0);
String|Null s2 = l1.get(0);

List<String|Integer> l2 = ["Eins", 1];
```

Kotlin


```
fun sayHello(name: String) {
    println("Hallo $name!")
}
```

```
fun main(args: Array<String>) {
    sayHello("JavaLand")
}
```

Kotlin
by JetBrains



Kotlin
by JetBrains

```
fun double(a: Any): Int {
    if (a is Int) {
        return a * 2
    }
    return 0
}
```


```
class MyList(
        l: List<String>): List<String> by l {

    fun printSize() {
        println(size())
    }
}
```


```
fun List<String>.printFirst() {
    println(first)
}

fun testPrintFirst() {
    val list = listOf("Eins", "Zwei")
    list.printFirst()
}
```

Fantom

FANTOM

```
class HelloWorld {

  static Void sayHello(Str name) {
    echo("Hallo $name!")
  }

  static Void main() {
    sayHello("JavaLand")
  }
}
```

FANTOM

```
% fan hello_world.fan
Hallo JavaLand!
```

FANTOM

```
class DynamicProgramming {
  static Void main() {

    Num n := 2
    // n.mult(2)
    // Unknown method 'sys::Num.mult'
    n->mult(2)

    Obj o := "4"
    Int i1 := Int.fromStr((Str) o)
    Int i2 := Int.fromStr(o)
  }
}
```

FANTOM

```
@Serializable class Conference {
  Str? name
  Int year

  static Void main() {
    javaLand := Conference {
      name = "JavaLand"
      year = 2014
    }
    Env.cur.out.writeObj(javaLand)
  }
}
```

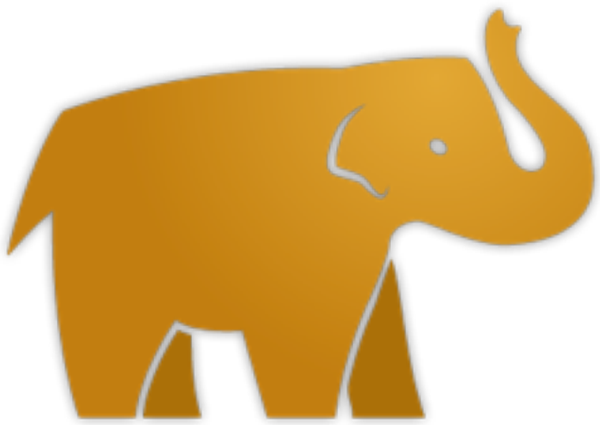
FANTOM

```
serialization_0::Conference {
  name = "JavaLand"
  year = 2014
}
```

FANTOM

```
using concurrent

class Actors {
  static Void main() {

    p := ActorPool()
    actor := Actor(p) { "Hallo $it!" }

    echo(actor.send("JavaLand").get)
  }
}
```


Groovy




Kotlin
by JetBrains


FANTOM

Vielen Dank!

Simon Olofsson • Content Management AG
@solofs • so@cm4all.com

Die Logos wurden den Webseiten der jeweiligen Projekte entnommen.