JavaScript - Teil 3

ANWENDUNG FUNKTIONALER PROGRAMMIERUNG

MARINUS NOICHL | MICHAEL HÄUSLMANN
FH ROSENHEIM 2016

Agenda

1. Versuch mit JavaScript
2. Groovy als Alternative
3. Lösungsversuch

Ziel

"Vorhandene Java API auf eine der OPL nähere Form abbilden"

→ einführen einer "Domain Specific Language"

Versuch mit JavaScript

Java API von JavaScript aufrufen?

- Nashorn (ab Java 8) ←
- Rhino (veraltet)

Eigenschaften:

- JavaScript Engine
- Java Klassen von JavaScript aus aufrufbar
- JavaScript Scripte von Java evaluierbar

Versuch mit JavaScript

```
var Solver = Java.type("optim.Solver");
var Assignment = Java.type("optim.Assignment");

var solver = new Solver();

var x = solver.variable('x');
var y = solver.variable('y');

solver.maximize(x.sum(y));

solver.add(solver.constant(2).prod(x).sum(y.sum(solver.constant(10))).leq(solver.constant(20)));
solver.add(x.sum(solver.constant(3).prod(y)).geq(solver.constant(10)));

var as = new Assignment();
as.put("x",0.0);
as.put("y",10.0);

solver.solve(as);
```

```
private ScriptEngine engine = new ScriptEngineManager().getEngineByName("nashorn");
engine.eval(file);
```

Versuch mit JavaScript

Ziel: "schönere" API

→ Operatoren überladen

↯ *Operatoren Überladung in JavaScript NICHT möglich* ↯

→ als DSL nicht geeignet!

http://www.2ality.com/2011/12/fake-operator-overloading.html

Groovy

Allgemeine Informationen

- basiert auf JVM
- "Skriptsprache"
- dynamische sowie statische Typisierung
- bietet Typinferenz
- "gut für DSLs geeignet"

Operatoren Überladung

Operator	Methode
a + b	a.plus(b)
a - b	a.minus(b)
a * b	a.multiply(b)
a / b	a.div(b)
a == b	a.equals(b) oder a.compareTo(b) == 0
a > b	a.compareTo(b) > 0
a >= b	a.compareTo(b) >= 0
…	

Command Chain

```groovy
// equivalent to: turn(left).then(right)
turn left then right

// equivalent to: take(2.pills).of(chloroquinine).after(6.hours)
take 2.pills of chloroquinine after 6.hours

// equivalent to: paint(wall).with(red, green).and(yellow)
paint wall with red, green and yellow

// with named parameters too
// equivalent to: check(that: margarita).tastes(good)
check that: margarita tastes good
```

Quelle: http://docs.groovy-lang.org

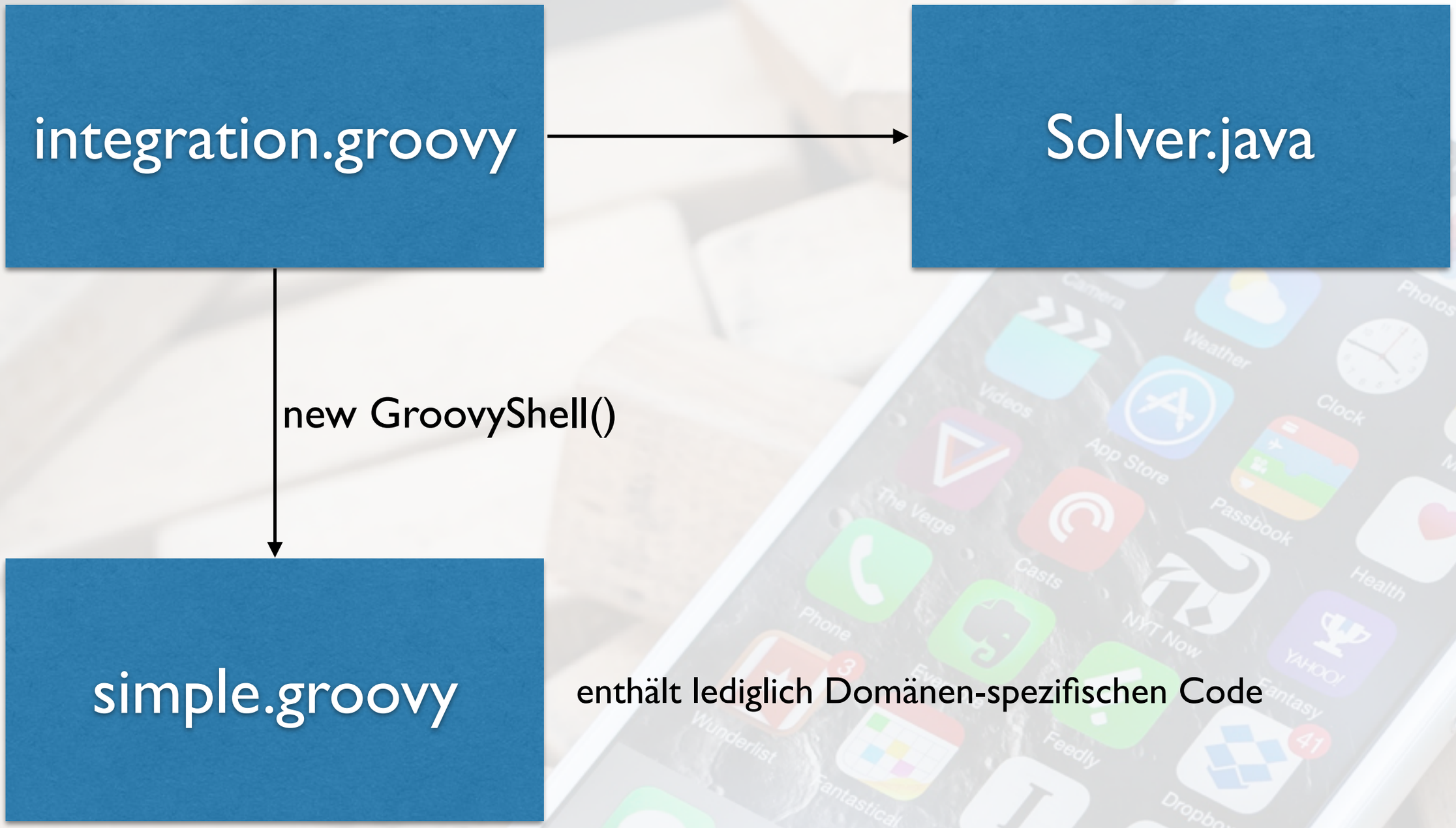
Command Chain

```
show = { println it }
square_root = { Math.sqrt(it) }

def please(action) {
  [the: { what ->
    [of: { n -> action(what(n)) }]
  }]
}

// equivalent to: please(show).the(square_root).of(100)
please show the square_root of 100
// ==> 10.0
```

Quelle: http://docs.groovy-lang.org

→ *Reihenfolge entscheidend!*

Aufbau



enthält lediglich Domänen-spezifischen Code

Lösungsvorschlag

Live-Demo