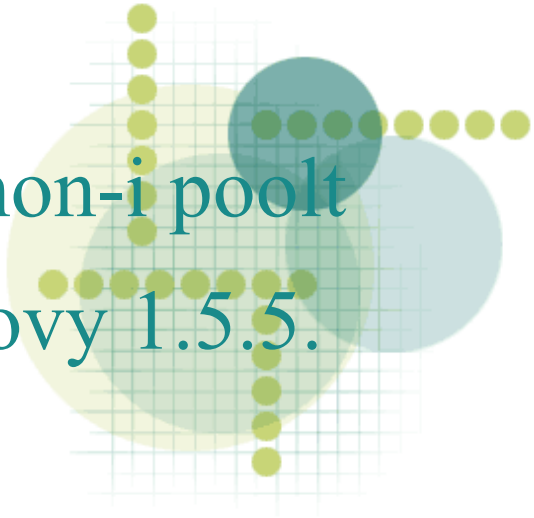
http://groovy.codehaus.org/

Groovy

Murad Kamalov

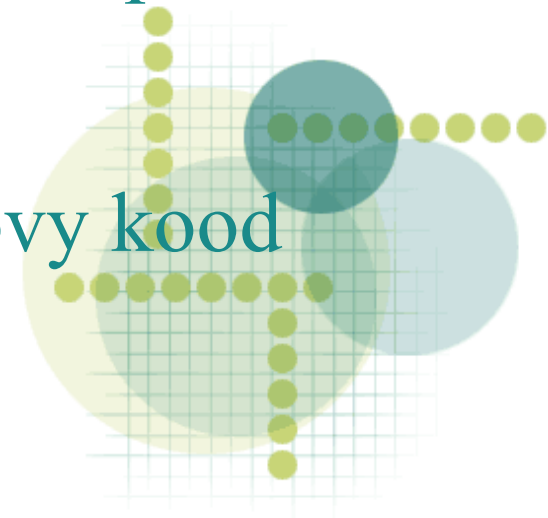
Üldinfo

- Java platvormi keel
- Väle, dünaamiline programmeerimiskeel, mis põhineb Java API-l ja kompileerub Java baitkoodiks
- Loodud 2004 aastal James Strachani ja Bob McWhirteri poolt ja publitseeritud JSR 241 all.
- Tõstab Java arendaja produktiivsust.
- Tugevalt mõjutatud Java, Ruby ja Python-i poolt
- Viimane keele versioon on hetkel Groovy 1.5.5. Väljalastud 14.04.2008



Üldinfo(2)

- Objekt-orienteeritud
- Ideoloogiliselt väga sarnane Ruby-ga
  - kõik on objekt
  - süntaks kohati on väga sarnane Ruby omaga
- Toetab nii staatilist kui ka dünaamilist tüüpimist
- Sujuv integreerimine Javaga
- Peaaegu kogu Java kood on õige Groovy kood
- Võimaldab java teekide kasutamis



Interpreteerimine ja kompileerimine

- Groovy koodi saab interpreteerida, käsuga `groovy`
- Kompileerida java baitkoodiks, käsuga `groovyc` .
  - Selleks et kompileeritud programmi `java` käsuga käima panna peavad teegid `$GROOVY_HOME/lib/groovy-x.x.x.jar` ja `$GROOVY_HOME/lib/asm-x.x.x.jar` olema classpath-is
- Interpreteerida Groovy käsureaprogrammi abil, käsud `groovysh`, või `groovyConsole`

Muutujad

- Kuna kõik on Groovy-is objekt, siis iga muutuja, sealhulgas ka arvulised muutujad on objektid. Ning igal nendest on ka omad meetodit.

```
1.toFloat() // => 1.0
1.toString() // => "1"
```

- Iga lausung (*statement*) tagastab midagi kasvõi null-i.

```
a = println("Tere") // a = null
println(a) // null
```

Muutujad (2)

- Täisarvud
  - Byte – [127, -128]
  - Short – [32767, -32768]
  - Integer [-2147483648, 2147483647]
  - Long [-9223372036854775808, 9223372036854775807]
  - BigInteger – ei ole suuruse piirangut
- Ujukomarvud
  - Float [1.4E-45, 3.4028235E38]
  - Double [4.9E-324, 1.79769313486231570E308]
  - BigDecimal – ei ole suuruse piirangut

```
(1.2).class       // => class java.math.BigDecimal
(1.2f).class      // => class java.lang.Float
(1.2d).class      // => class java.lang.Double
(1.2 as Float)  // => class java.lang.Float
```

Listid

- Groovy vaikimisi toetab liste

```
1: l = [] // uus tühi list
2: ls = [1,2,3,4,5]
3: println( ls[1..3] ) // [2, 3, 4]
4: println( ls[2,-2] ) // [3, 4]
5: ls = [1, *2..5]     // [1,2,3,4,5]
6:
7: strLs = ['Tartu', 'Ülikool']
8: strLs*.size() // => [5, 7] meetod rakendatakse
9:                  // igale listi elemendile
10: strLs.each{ println it } // Tartu\nÜlikool\n
11: strLs[0][4..3] // => ut
```

Map

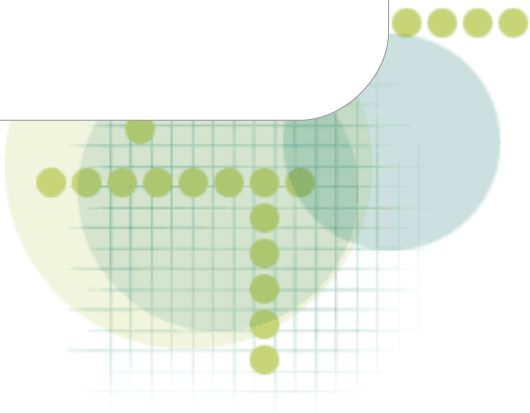
- Groovy vaikimisi toetab Mappe

```
1: m = [:] // uus tühi map
2:
3: m = [yks: 1, kaks: 2] // või ["yks":1,"kaks":2]
4: m["yks"] // => 1
5: m.kaks // => 2
6:
7: m.each { key, value ->
8:    println key + " : " + value
9: } // yks : 1\nkaks : 2
```

Sulundid (Closures)

- Sulundid võimaldavad täidetavat koodi { } sulgude vahele panna, muutujale väärtustada ning hiljem ka välja kutsuda.

```
ruut = { it * it } // it on muutuja mida
                   // sulundile ette antakse
ruut(2)            // => 4

ruut2 = { x -> x * x} // x annakse ette
ruut2(2)              // => 4
```



Sulundid (2)

- Kõige kasulikum sulundite puhul on see, et neid saab meetoditele ette anda ja hiljem nendes kasutada

```
def itereeri(ls, sulund){
   for(i = 0; i < ls.size(); i++){
      ls[i] = sulund(ls[i])
   }
   ls
}

itereeri([1,2,3,4,5]){x -> x*2} //=> [2,4,6,8,10]
```

Regulaaravaldised

- Groovy toetab regulaaravaldisi
- Operaator **==~** võimaldab kontrollida sõne vastavust regulaaravaldisele
- Regulaaravaldised pannakse „**/**“ märkide vahele
- operaatori **=~** abil saab luua uue Macher objekti
- operaatori **~** abil saab luua uue Pattern objekti

```
"roovy" ==~ /g?roovy/  // => true
"groovy?" ==~ /groovy\?/ // => true

m = "groovy roovy" =~ /g?roovy/ // Matcher
m.replaceAll("tere") // => "tere tere"
m = "groovy roovy" =~ /g?roovy/
m.each{ println(it) } // groovy\nroovy

p = ~/groovy/  // Pattern
switch("groovy"){case p: println("matches")} // matches
```

Tingimusavaldised

- Tingimusavaldised on Groovy-s Javaga identsed

```
if(true){
    //...
} else if(true){
    //...
} else{
    //...
}
```

```
switch(x){
    case: 1
        //...
        break
    case: 2
        //...
        break
    default:
        //...
}
```

Tsüklid

- Java for ja while tsüklide semantika töötab ka Groovys
- Groovy lisab uusi võimalusi itereerimiseks

```
1: for(x in [1,2,3,4,5]){ // 1, 2, 3, 4, 5,
      print(x + ", ")
   }

2: for(x in [ysk:1, kaks:2]){ // yks : 1, kaks : 2
      print(x.key + " : " + x.value + ", ")
   }

3: [1,2,3,4,5].each(){ // 1, 2, 3, 4, 5,
      print(it + ", ")
   }

   ls = [1,2,3,4,5]
4: ls.eachWithIndex(){ x, i -> // 1, 2, 3, 4, 5,
      ls[i] = x * 2
   } // ls = [2,4,6,8,10]
```

Operaatorite ülelaadimine

Operaator	Meetod
a + b	a.plus(b)
a - b	a.minus(b)
a * b	a.multiply(b)
a ** b	a.power(b)
a / b	a.div(b)
a % b	a.mod(b)
a | b	a.or(b)
a & b	a.and(b)
a ^ b	a.xor(b)
a++ või ++a	a.next()
a-- või --a	a.previous()
a[b]	a.getAt(b)
a[b] = c	a.putAt(b, c)
a << b	a.leftShift(b)
a >> b	a.rightShift(b)
switch(a){case(b):}	b.isCase(a)
~a	a.bitwiseNegate()
-a	a.negative()
+a	a.positive()

Operaator	Meetod
a == b	a.equals(b)
a != b	! a.equals(b)
a <=> b	a.compareTo(b)
a > b	a.compareTo(b) > 0
a >= b	a.compareTo(b) >= 0
a < b	a.compareTo(b) < 0
a <= b	a.compareTo(b) <= 0

Klassid

- Klasside loomine võtmesõnaga **`class`**
- **`private`**, **`public`**, **`protected`**, **`static`**, **`final`** võtmesõnad nagu Javas
- **`extends`**, **`implements`**, **`abstract`** võtmesõnad nagu Javas
- Praegune keele versioone ei toeta alamklasse, tavaliselt nende asemel saab kasutada sulundeid
- Interfeisi loomine võtmesõnaga **`interface`**
- Meetodeid ja muutujaid saab deklareerida kasutades võtmesõna **`def`**

Klassi näide

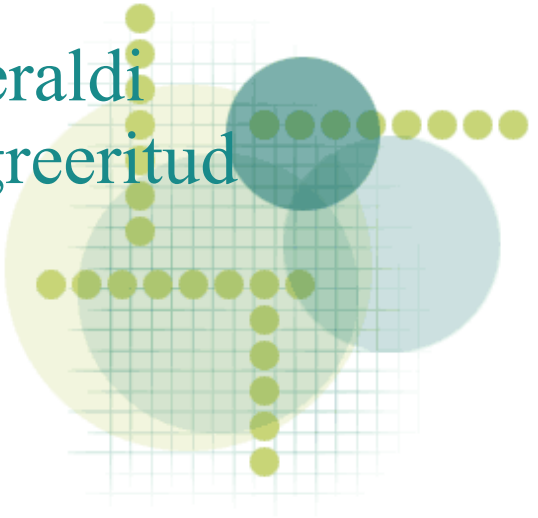
```
class Main{
        private nimi
        public Main(nimi){
            this.nimi = nimi
        }
        def tere(){
            println("Tere " + nimi + "!")
        }
        public setNimi(nimi){
            this.nimi = nimi
        }
        public getNimi(){
            nimi
        }
}

m = new Main("maailm")
m.tere() // Tere maailm!
m.nimi = "Tartu" //kasutab piilumeetodit setNimi()
m.tere() // Tere Tartu!
```

Miks Groovy ja mitte Java

- Groovy pakkub asju, mida Javas ei ole
  - Dünaamiline tüüpimine
  - Kergem ja arusaadavam süntaks
  - Käsurea interpretaator
  - Väledus
  - Tõstab arendaja produktiivsust
  - Teeb koodi kirjutamist lihtsamaks

Miks Groovy

- On ju juba olemas Ruby ja Python, mis on edukalt porditud Java platvormile?
  - Groovy on nagu Java aga palju väiksemate piirangutega
  - Groovy põhineb Java API-l, kõik mis on standardses javas olemas on kasutatav Groovyst
  - Ruby ja Python on algselt loodud Javast eraldi keeltena ja ainult hiljem olid sellega integreeritud



Küsimused?

Lingid

- http://groovy.codehaus.org/
- http://java.sun.com/developer/technicalArticles/JavaLP/groovy/
- http://groovy.codehaus.org/api/
- http://java.sun.com/developer/technicalArticles/JavaLP/groovy/
- http://www.projectzero.org/
- http://grails.codehaus.org/