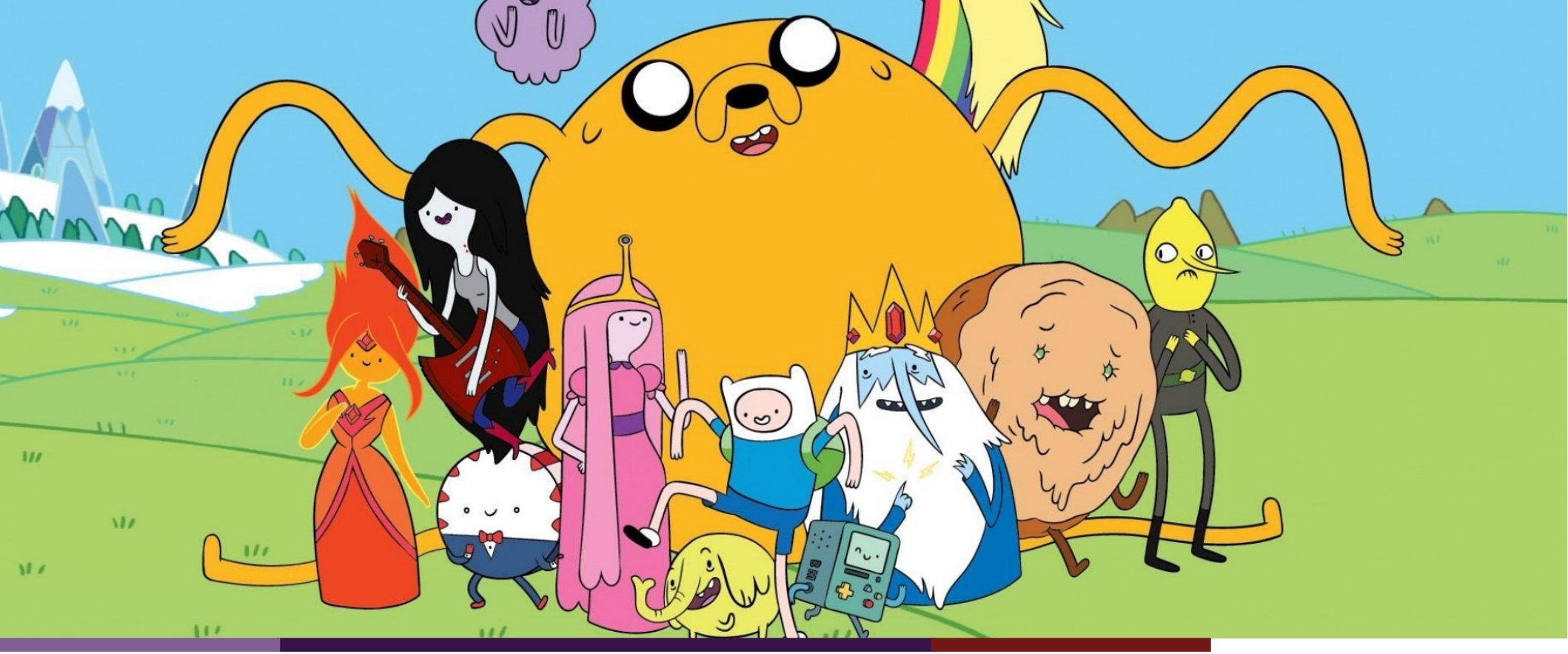


From Java to Groovy: Adventure Time!

JavaCro'16

Iván López @ilopmar

Kaleidos
beautiful code

About me...

- Iván López @ilopmar
- Groovy & Grails developer

- @MadridGUG coordinator

- Greach organizer
  @greachconf
  http://greachconf.com

- Speaker: SpringOne 2GX, GR8Conf, GGX, Codemotion, GeeCon, jDays, Spring IO, Greach, ConFoo, ConFess, RigaDevDays...

1.
What is Groovy?

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

“

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

“

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

“

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

“

Apache Groovy is a powerful, optionally typed and dynamic language, with static-typing and static compilation capabilities, for the Java platform aimed at improving developer productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your application powerful features, including scripting capabilities, Domain-Specific Language authoring, runtime and compile-time meta-programming and functional programming.

– Groovy website

Groovy

2.
Why not Java?

Why not Java?

- Java is solid
- Known by a lot of developers
- Used everywhere
- Is fast

But...

- Java is verbose
- Can be annoying
- It's not dynamic

3.
Making Java Groovy

Greeter.java

Rename

```
public class Greeter {
    private String greeting;

    public void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names);
        System.out.println(helloMessage);
    }

    public String getGreeting() {
        return greeting;
    }

    public void setGreeting(String greeting) {
        this.greeting = greeting;
    }

    private String prepareHelloMessage(String... names) {
        String delimiter = "";
        StringBuilder sb = new StringBuilder();
        for (String name : names) {
            sb.append(delimiter).append(name);
            delimiter = ", ";
        }
        return this.greeting + " " + sb.toString() + "!";
    }

    public static void main(String[] args) {
        final Greeter greeter = new Greeter();
        greeter.setGreeting("Hi");
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard");
    }
}
```

Greeter.groovy

```
public class Greeter {
    private String greeting;

    public void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names);
        System.out.println(helloMessage);
    }

    public String getGreeting() {
        return greeting;
    }

    public void setGreeting(String greeting) {
        this.greeting = greeting;
    }

    private String prepareHelloMessage(String... names) {
        String delimiter = "";
        StringBuilder sb = new StringBuilder();
        for (String name : names) {
            sb.append(delimiter).append(name);
            delimiter = ", ";
        }
        return this.greeting + " " + sb.toString() + "!";
    }

    public static void main(String[] args) {
        final Greeter greeter = new Greeter();
        greeter.setGreeting("Hi");
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard");
    }
}
```

Groovy

```
public class Greeter {
    private String greeting;

    public void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names);
        System.out.println(helloMessage);
    }

    public String getGreeting() {
        return greeting;
    }

    public void setGreeting(String greeting) {
        this.greeting = greeting;
    }

    private String prepareHelloMessage(String... names) {
        String delimiter = "";
        StringBuilder sb = new StringBuilder();
        for (String name : names) {
            sb.append(delimiter).append(name);
            delimiter = ", ";
        }
        return this.greeting + " " + sb.toString() + "!";
    }

    public static void main(String[] args) {
        final Greeter greeter = new Greeter();
        greeter.setGreeting("Hi");
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard");
    }
}
```

Groovy

optional

optional

```
public class Greeter {
    private String greeting

    public void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names)
        System.out.println(helloMessage)
    }

    public String getGreeting() {
        return greeting
    }

    public void setGreeting(String greeting) {
        this.greeting = greeting
    }

    private String prepareHelloMessage(String... names) {
        String delimiter = ""
        StringBuilder sb = new StringBuilder()
        for (String name : names) {
            sb.append(delimiter).append(name)
            delimiter = ", "
        }
        return this.greeting + " " + sb.toString() + "!"
    }

    public static void main(String[] args) {
        final Greeter greeter = new Greeter()
        greeter.setGreeting("Hi")
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

Groovy

```
class Greeter {
    private String greeting

        void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names)
        System.out.println(helloMessage)
    }

        String getGreeting() {
        return greeting
    }

        void setGreeting(String greeting) {
        this.greeting = greeting
    }

    private String prepareHelloMessage(String... names) {
        String delimiter = ""
        StringBuilder sb = new StringBuilder()
        for (String name : names) {
            sb.append(delimiter).append(name)
            delimiter = ", "
        }
        return this.greeting + " " + sb.toString() + "!"
    }

        static void main(String[] args) {
        final Greeter greeter = new Greeter()
        greeter.setGreeting("Hi")
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

verbose getter/setter

Groovy

```
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names)
        System.out.println(helloMessage)
    }

    private String prepareHelloMessage(String... names) {
        String delimiter = ""
        StringBuilder sb = new StringBuilder()
        for (String name : names) {
            sb.append(delimiter).append(name)
            delimiter = ", "
        }
        return this.greeting + " " + sb.toString()
    }

    static void main(String[] args) {
        final Greeter greeter = new Greeter()
        greeter.setGreeting("Hi")
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

Accessing as property

Groovy

```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names)
        System.out.println(helloMessage)
    }

    private String prepareHelloMessage(String... names) {
        String delimiter = ""
        StringBuilder sb = new StringBuilder()
        for (String name : names) {
            sb.append(delimiter).append(name)
            delimiter = ", "
        }
        return this.greeting + " " + sb.toString() + "!"
    }

    static void main(String[] args) {
        final Greeter greeter = new Greeter()
        greeter.greeting = "Hi"
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

optional

Groovy

```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names)
        System.out.println(helloMessage)
    }

    private String prepareHelloMessage(String... names) {
        String delimiter = ""
        StringBuilder sb = new StringBuilder()
        for (String name : names) {
            sb.append(delimiter).append(name)
            delimiter = ", "
        }
        this.greeting + " " + sb.toString() + "!"
    }

    static void main(String[] args) {
        final Greeter greeter = new Greeter()
        greeter.greeting = "Hi"
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

shortcut

Groovy

```
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names)
        println(helloMessage)
    }

    private String prepareHelloMessage(String... names) {
        String delimiter = ""
        StringBuilder sb = new StringBuilder()
        for (String name : names) {
            sb.append(delimiter).append(name)
            delimiter = ", "
        }
        this.greeting + " " + sb.toString() + "!"
    }

    static void main(String[] args) {
        final Greeter greeter = new Greeter()
        greeter.greeting = "Hi"
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

Collections API

Groovy

```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names)
        println(helloMessage)
    }

    private String prepareHelloMessage(String... names) {
        String joinedNames = names.join(', ')

        this.greeting + " " + joinedNames + "!"
    }

    static void main(String[] args) {
        final Greeter greeter = new Greeter()
        greeter.greeting = "Hi"
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

String interpolation (GString)

Groovy

```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names)
        println(helloMessage)
    }

    private String prepareHelloMessage(String... names) {
        String joinedNames = names.join(', ')

        "${this.greeting} $joinedNames!"
    }

    static void main(String[] args) {
        final Greeter greeter = new Greeter()
        greeter.greeting = "Hi"
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

Groovy

```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        String helloMessage = prepareHelloMessage(names)
        println(helloMessage)
    }

    private String prepareHelloMessage(String... names) {
        "$greeting ${names.join(', ')}!"
    }

    static void main(String[] args) {
        final Greeter greeter = new Greeter()
        greeter.greeting = "Hi"
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

optional typing

Groovy

```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        def helloMessage = prepareHelloMessage(names)
        println(helloMessage)
    }

    private String prepareHelloMessage(String... names) {
        "$greeting ${names.join(', ')}!"
    }

    static void main(String[] args) {
        def greeter = new Greeter()
        greeter.greeting = "Hi"
        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

named constructor

Groovy

```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        def helloMessage = prepareHelloMessage(names)
        println(helloMessage)
    }

    private String prepareHelloMessage(String... names) {
        "$greeting ${names.join(', ')}!"
    }

    static void main(String[] args) {
        def greeter = new Greeter(greeting: 'Hi')

        greeter.sayHiTo("Sheldon", "Leonard", "Raj", "Howard")
    }
}
```

optional parentheses

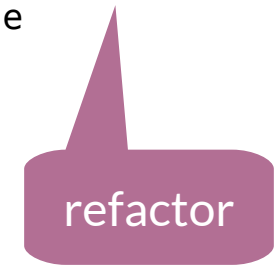
optional parentheses

Groovy

```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        def helloMessage = prepareHelloMessage(names)
        println helloMessage
    }

    private String prepareHelloMessage(String... names) {
        "$greeting ${names.join(', ')}!"
    }

    static void main(String[] args) {
        def greeter = new Greeter(greeting: 'Hi')

        greeter.sayHiTo "Sheldon", "Leonard", "Raj", "Howard"
    }
}
```

Groovy

```
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        def helloMessage = prepareHelloMessage(names)
        println helloMessage
    }

    private String prepareHelloMessage(String... names) {
        "$greeting ${names.join(', ')}!"
    }
}

greeter = new Greeter(greeting: 'Hi')

greeter.sayHiTo "Sheldon", "Leonard", "Raj", "Howard"
```



Groovy

```groovy
        class Greeter {
             String greeting

            void sayHiTo(String... names) {

                    println "$greeting ${names.join(', ')}!"
    }
```


```groovy
}

            greeter = new Greeter(greeting: 'Hi')

         greeter.sayHiTo "Sheldon", "Leonard", "Raj", "Howard"
```

Groovy: Final version

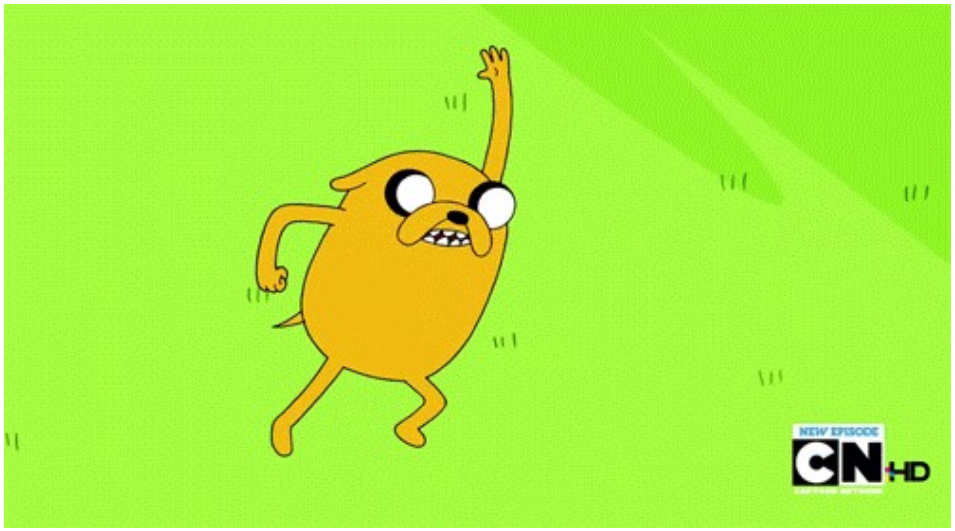
```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        println "$greeting ${names.join(', ')}!"
    }
}

greeter = new Greeter(greeting: 'Hi')
greeter.sayHiTo "Sheldon", "Leonard", "Raj", "Howard"
```

Groovy: Final version

```groovy
class Greeter {
    String greeting

    void sayHiTo(String... names) {
        println "$greeting ${names.join(', ')}!"
    }
}

greeter = new Greeter(greeting: 'Hi')
greeter.sayHiTo "Sheldon", "Leonard", "Raj", "Howard"
```

Java	32 lines of code	880 characters
Groovy	10 lines of code	221 characters
Difference	**69%**	**75%**

Groovy: Adventure Time!



4.
Groovy

Differences between Java and Groovy

- Dynamic
- Optional static compilation
- Everything is an object
- Operator overloading
  - + is a call to *.plus()*
  - * is a call to *.multiply()*
- Native syntax for lists, maps and ranges
- Methods and classes are *public* by default
- All exceptions are *unchecked*
- http://groovy-lang.org/differences.html

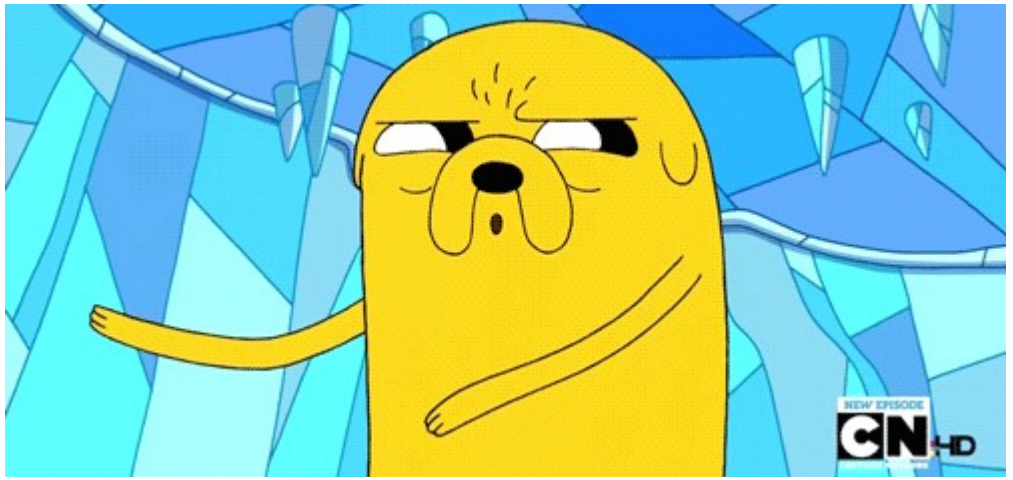
Getters y setters


```
public class Person {

    private String name;
    private int age;

    String getName() {
        return name;
    }

    void setName(String name) {
        this.name = name;
    }

    int getAge() {
        return age;
    }

    void setAge(int age) {
        this.age = age;
    }
}
```

Getters y setters


```java
public class Person {

    private String name;
    private int age;

    String getName() {
        return name;
    }

    void setName(String name) {
        this.name = name;
    }

    int getAge() {
        return age;
    }

    void setAge(int age) {
        this.age = age;
    }
}
```


```groovy
class Person {
    String name
    int age
}
```



Named constructors

```
class Person {
    String name
    int age
}

def p = new Person(name: 'Iván', age: 36)
assert p.name == 'Iván'

p.age = 37
assert p.age == 37
```

Constructor Builder

```
import groovy.transform.builder.Builder

@Builder
class Person {
    String name
    int age
}

Person.builder()
    .name('Iván')
    .age(36)
    .build()
```

Numbers that...


```
System.out.println(2.00 - 1.1);
```

Numbers that...


```java
System.out.println(2.00 - 1.1);
// 0.8999999999999999
```

Numbers that...

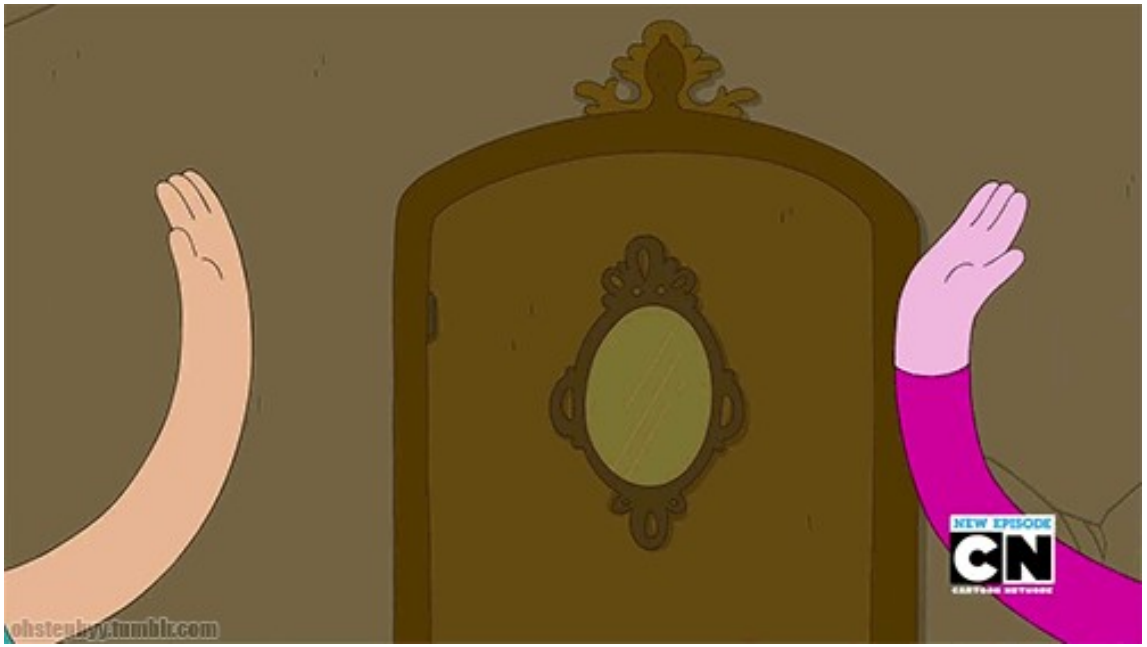

```
System.out.println(3 / 2);
```

Numbers that...


```
System.out.println(3 / 2);
// 1
```

BigDecimal by default!


```
assert 2.0 - 1.1 == 0.9
assert 3 / 2 == 1.5
```



Equals and ==


```
state != null &&
state.equals(State.COMPLETED);
```

Equals and ==


```
state != null &&
state.equals(State.COMPLETED);
```


```
state == State.COMPLETED
```

Strings and GStrings

```
def name = 'Iván'
def age = 36

println "¡Hi ${name}, you are ${age} years old!"
```

Strings and GStrings

```groovy
def name = 'Iván'
def age = 36

println "¡Hi ${name}, you are ${age} years old!"



def query = """
insert into people
        (firstName, age)
values (${name}, ${age})
"""

new Sql(datasource).execute query
```

Lists

```
def list = ['a', 'b', 'c']
```

Lists

```
def list = ['a', 'b', 'c']

list << 'd' // list.add("d")
assert list.contains('d')
```

Lists

```groovy
def list = ['a', 'b', 'c']

list << 'd' // list.add("d")
assert list.contains('d')

assert list.collect { it.toUpperCase() } == ['A', 'B', 'C', 'D']
```

Maps

```
def map = [name: 'Iván', age: 36]
assert map.name == 'Iván'
```

Maps

```
def map = [name: 'Iván', age: 36]
assert map.name == 'Iván'

map.childs = ['Judith', 'Adriana']
assert map['childs'].contains('Adriana')
```

Ranges

```
def range = 'a'..'z'
assert range.contains('i')
assert range.contains('z')
```

Ranges

```
def range = 'a'..'z'
assert range.contains('i')
assert range.contains('z')


def exclusive = 1..<10
assert !exclusive.contains(10)
```

Ranges

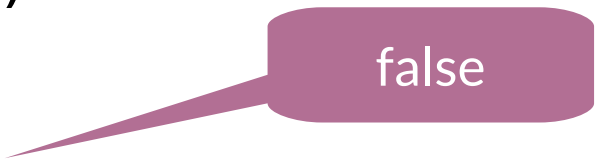
```
def range = 'a'..'z'
assert range.contains('i')
assert range.contains('z')


def exclusive = 1..<10
assert !exclusive.contains(10)


def inverse = 10..1
assert inverse[0] == 10
assert inverse[-1] == 1
```

Groovy truth

```
assert !( null )
assert !( "" )
assert !( '' )
assert !( [] )
assert !( [:] )
assert !( 0 )
```

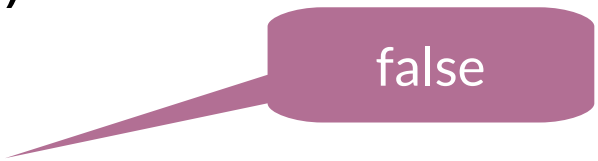
Groovy truth

```
assert !( null )
assert !( "" )
assert !( '' )
assert !( [] )
assert !( [:] )
assert !( 0 )
```



Groovy truth

```
assert !( null )
assert !( "" )
assert !( '' )
assert !( [] )
assert !( [:] )
assert !( 0 )
```




```
assert new Object()
assert "string"
assert 'string'
assert [1, 2]
assert [a: 1]
assert 1
```

Groovy truth

```
assert !( null )
assert !( "" )
assert !( '' )
assert !( [] )
assert !( [:] )
assert !( 0 )
```

false

true

```
assert new Object()
assert "string"
assert 'string'
assert [1, 2]
assert [a: 1]
assert 1
```

Power asserts

```
assert (2 + 7) * 5 != (2 * 5) + (7 * 5)
```

Power asserts

```
assert (2 + 7) * 5 != (2 * 5) + (7 * 5)
```

```
(2 + 7) * 5 != (2 * 5) + (7 * 5)
   |    |   |      |    |     |
   9    45  false 10   45    35
```

Power asserts

```
def info = [
  name: 'Iván', age: 36,
  childs: [
    [name: 'Judith', age: 8], [name: 'Adriana', age: 5]
  ]
]
```

Power asserts

```
def info = [
  name: 'Iván', age: 36,
  childs: [
    [name: 'Judith', age: 8], [name: 'Adriana', age: 5]
  ]
]

assert info.childs.name.first() == 'Adriana'
```

Power asserts

```
def info = [
  name: 'Iván', age: 36,
  childs: [
    [name: 'Judith', age: 8], [name: 'Adriana', age: 5]
  ]
]

assert info.childs.name.first() == 'Adriana'
```

```
info.childs.name.first() == 'Adriana'
|
|
|
|
|
|
|
[name:Iván, age:36, childs:[[name:Judith, age:8], [name:Adriana,
age:5]]]
```

Power asserts

```
def info = [
  name: 'Iván', age: 36,
  childs: [
    [name: 'Judith', age: 8], [name: 'Adriana', age: 5]
  ]
]

assert info.childs.name.first() == 'Adriana'
```

```
info.childs.name.first() == 'Adriana'
|    |
|    |
|    |
|    |
|    |
|    |
|    [[name:Judith, age:8], [name:Adriana, age:5]]
[name:Iván, age:36, childs:[[name:Judith, age:8], [name:Adriana,
age:5]]]
```

Power asserts

```
def info = [
  name: 'Iván', age: 36,
  childs: [
    [name: 'Judith', age: 8], [name: 'Adriana', age: 5]
  ]
]

assert info.childs.name.first() == 'Adriana'
```

```
info.childs.name.first() == 'Adriana'
|    |      |
|    |      |
|    |      |
|    |      |
|    |      |
|    |      [Judith, Adriana]
|    [[name:Judith, age:8], [name:Adriana, age:5]]
[name:Iván, age:36, childs:[[name:Judith, age:8], [name:Adriana,
age:5]]]
```

Power asserts

```
def info = [
  name: 'Iván', age: 36,
  childs: [
    [name: 'Judith', age: 8], [name: 'Adriana', age: 5]
  ]
]

assert info.childs.name.first() == 'Adriana'
```

```
info.childs.name.first() == 'Adriana'
|    |      |      |       |
|    |      |      Judith  false
|    |      |              6 differences (14% similarity)
|    |      |              (Ju)d(-)i(th-)
|    |      |              (A-)d(r)i(ana)
|    |      [Judith, Adriana]
|    [[name:Judith, age:8], [name:Adriana, age:5]]
[name:Iván, age:36, childs:[[name:Judith, age:8], [name:Adriana,
age:5]]]
```

Elvis

```
List result =
     (names != null && names.size() > 0) ?
       names : Collections.emptyList();
```

Elvis

```
List result =
    (names != null && names.size() > 0) ?
      names : Collections.emptyList();
```


```
def result = names ? names : []
```

Elvis

```
List result =
    (names != null && names.size() > 0) ?
      names : Collections.emptyList();
```


```
def result = names ? names : []
```


```
def result = names ?: []
```

Safe navigation


```
if (order != null) {
  if (order.getCustomer() != null) {
    if (order.getCustomer().getAddress() != null) {
      System.out.println(order.getCustomer().getAddress());
    }
  }
}
```

Safe navigation


```
if (order != null) {
  if (order.getCustomer() != null) {
    if (order.getCustomer().getAddress() != null) {
      System.out.println(order.getCustomer().getAddress());
    }
  }
}
```

```
println order?.customer?.address
```

Closures

```
def multiplier = { a, b -> a * b }

assert multiplier(2, 3) == 6
assert multiplier.call(2, 3) == 6
assert multiplier('=', 8) == '========'
```

Closures

```groovy
def multiplier = { a, b -> a * b }

assert multiplier(2, 3) == 6
assert multiplier.call(2, 3) == 6
assert multiplier('=', 8) == '========'
```

```groovy
def multiplier = { int a, int b -> a * b }
```

Closures

```
def multiplier = { a, b -> a * b }

assert multiplier(2, 3) == 6
assert multiplier.call(2, 3) == 6
assert multiplier('=', 8) == '========'
```

```
def multiplier = { int a, int b -> a * b }
```

```
def adder = { ... numbers -> numbers.sum() }
assert adder(1, 2, 3) == 6
assert adder('a', 'b', 'c') == 'abc'
```

Closures: Default values

```
def multiplier = { int a, int b = 10 -> a * b }

assert multiplier(2, 3) == 6
assert multiplier(2) == 20
```

Closures: Methods as functions

```
def logBase10 = Math.&log10
assert logBase10(10) == 1
```

Closures: map, filter, reduce

```
def persons = [
    new Person('Iván', 36),
    new Person('Judith', 8),
    new Person('Adriana', 5)
]
```

Closures: map, filter, reduce

```
def persons = [
    new Person('Iván', 36),
    new Person('Judith', 8),
    new Person('Adriana', 5)
]

def names = persons.findAll { it.age < 18 }
    .collect { it.name.toUpperCase() }
    .sort()
    .join(', ')
```

Closures: map, filter, reduce

```
def persons = [
    new Person('Iván', 36),
    new Person('Judith', 8),
    new Person('Adriana', 5)
]

def names = persons.findAll { it.age < 18 }
    .collect { it.name.toUpperCase() }
    .sort()
    .join(', ')

assert names == 'ADRIANA, JUDITH'
```

Groovy Closures and Java 8 Lambdas

```
import static java.util.Arrays.asList;

public class JavaLambdas {
    public static void main(String[] args) {
        asList(1, 2, 3).stream()
            .map(i -> i * 2)
            .filter(i -> i > 3)
            .findFirst()
            .orElseThrow(IllegalArgumentException::new);
    }
}
```

Groovy Closures and Java 8 Lambdas

```
import static java.util.Arrays.asList;

public class JavaLambdas {
    public static void main(String[] args) {
        asList(1, 2, 3).stream()
            .map(i -> i * 2)
            .filter(i -> i > 3)
            .findFirst()
            .orElseThrow(IllegalArgumentException::new);
    }
}
```


```
[1, 2, 3].stream()
    .map { i -> i * 2 }
    .filter { i -> i > 3 }
    .findFirst()
    .orElseThrow(IllegalArgumentException.&newInstance)
```

Json builder

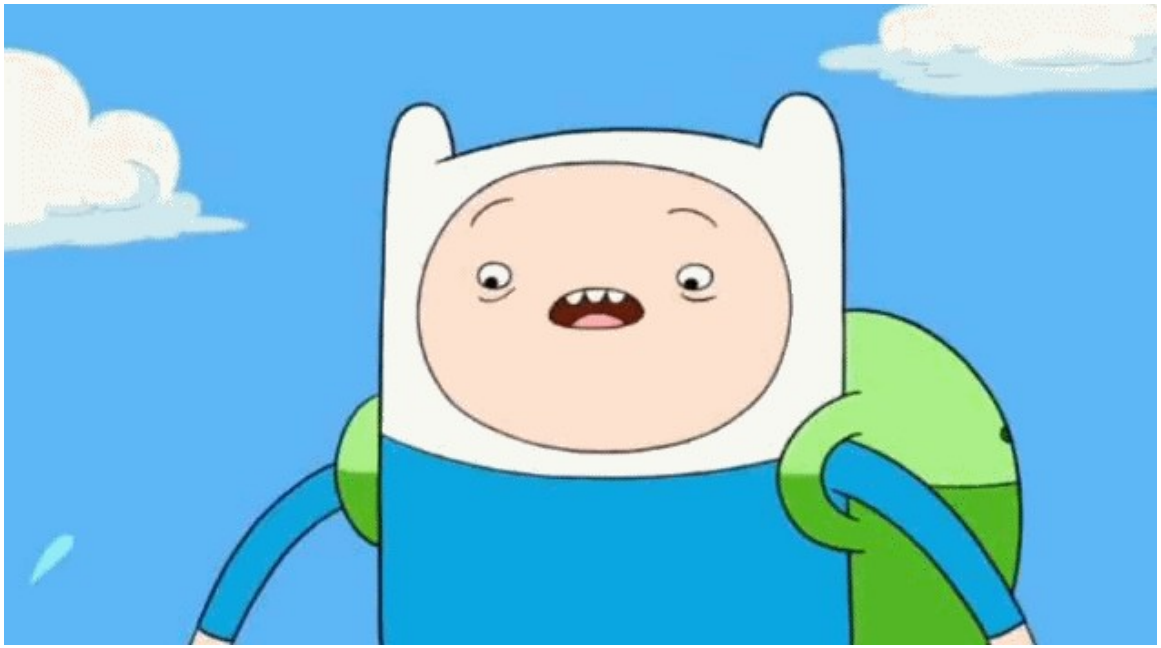
```
{
  "speaker": {
    "firstName": "Iván",
    "lastName": "López",
    "address": {
      "city": "Madrid",
      "country": "España",
      "zip": 12345
    },
    "conferences": [
      "JavaCro",
      "SpringOne 2GX",
      "Greach"
    ]
  }
}
```

Json builder

```groovy
import groovy.json.JsonBuilder

def builder = new JsonBuilder()
builder.
    speaker {
        firstName 'Iván'
        lastName 'López'
        address(
            city: 'Madrid',
            country: 'España',
            zip: 12345,
        )
        conferences(
            'JavaCro',
            'SpringOne 2GX',
            'Greach'
        )
    }

println builder.toPrettyString()
```

```json
{
  "speaker": {
    "firstName": "Iván",
    "lastName": "López",
    "address": {
      "city": "Madrid",
      "country": "España",
      "zip": 12345
    },
    "conferences": [
      "JavaCro",
      "SpringOne 2GX",
      "Greach"
    ]
  }
}
```

Parse XML and Json in Java



Json parser

http://api.openweathermap.org/data/2.5/weather?units=metric&q=Rovinj&appid=...


```
{
  coord: {
    lon: 13.64,
    lat: 45.08
  },
  weather: [
    {
      id: 801,
      main: "Clouds",
      description: "few clouds",
      icon: "02d"
    }
  ],
  base: "cmc stations",
  main: {
    temp: 19.06,
    pressure: 1014,
    humidity: 55,
    temp_min: 18,
    temp_max: 20
  },
  wind: {
    speed: 2.6,
    deg: 170
  },
  clouds: {
    all: 20
  },
  dt: 1460728800,
  sys: {
    type: 3,
    id: 5456,
    message: 0.0028,
    country: "HR",
    sunrise: 1460693949,
    sunset: 1460742728
  },
  id: 3191518,
  name: "Rovinj",
  cod: 200
}
```

Json parser

```
http://api.openweathermap.org/data/2.5/weather?
units=metric&q=Rovinj&appid=...
```


```
{
  coord: {
    lon: 13.64,
    lat: 45.08
  },
  weather: [
    {
      id: 801,
      main: "Clouds",
      description: "few clouds",
      icon: "02d"
    }
  ],
  base: "cmc stations",
  main: {
    temp: 19.06,
    pressure: 1014,
    humidity: 55,
    temp_min: 18,
    temp_max: 20
  },
  wind: {
    speed: 2.6,
    deg: 170
  },
  clouds: {
    all: 20
  },
  dt: 1460728800,
  sys: {
    type: 3,
    id: 5456,
    message: 0.0028,
    country: "HR",
    sunrise: 1460693949,
    sunset: 1460742728
  },
  id: 3191518,
  name: "Rovinj",
  cod: 200
}
```

Json parser

```
{
  weather: [
    {
      description: "few clouds",
    }
  ],
  main: {
    temp: 19.06
  },
  sys: {
    country: "HR",
  },
  name: "Rovinj",
}
```

Json parser

```
{
  weather: [
    {
      description: "few clouds",
    }
  ],
  main: {
    temp: 19.06
  },
  sys: {
    country: "HR",
  },
  name: "Rovinj",
}
```

```
import groovy.json.JsonSlurper

def url = "http://api.openweathermap.org/data/2.5/weather?
units=metric&q=Rovinj&appid=...".toURL()

def response = new JsonSlurper().parse(url)
```

Json parser

```
{
  weather: [
    {
      description: "few clouds",
    }
  ],
  main: {
    temp: 19.06
  },
  sys: {
    country: "HR",
  },
  name: "Rovinj",
}
```

```
import groovy.json.JsonSlurper

def url = "http://api.openweathermap.org/data/2.5/weather?
units=metric&q=Rovinj&appid=...".toURL()

def response = new JsonSlurper().parse(url)

String weather = response.weather.collect { it.description }.join(', ')
String country = response.sys.country
String temp = response.main.temp
String city = response.name
```

Json parser

```
{
  weather: [
    {
      description: "few clouds",
    }
  ],
  main: {
    temp: 19.06
  },
  sys: {
    country: "HR",
  },
  name: "Rovinj",
}
```

```
import groovy.json.JsonSlurper

def url = "http://api.openweathermap.org/data/2.5/weather?
units=metric&q=Rovinj&appid=...".toURL()

def response = new JsonSlurper().parse(url)

String weather = response.weather.collect { it.description }.join(', ')
String country = response.sys.country
String temp = response.main.temp
String city = response.name

println "Weather in ${city} (${country}): ${weather}. Temp: ${temp} ºC"
// Weather in Rovinj (HR): few clouds. Temp: 19.06 ºC
```

Read text file


```java
static String readFile(File file) throws IOException {
    byte[] bytes = Files.readAllBytes(file.toPath());
    return new String(bytes, "UTF-8");
}

public static void main(String[] args) {
    File file = new File("foo.txt");
    try {
        String content = readFile(file);
        System.out.println(content);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Read text file


```java
static String readFile(File file) throws IOException {
    byte[] bytes = Files.readAllBytes(file.toPath());
    return new String(bytes, "UTF-8");
}

public static void main(String[] args) {
    File file = new File("foo.txt");
    try {
        String content = readFile(file);
        System.out.println(content);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```


```groovy
String content = new File('foo.txt').text
```

Read text file


```
static String readFile(File file) throws IOException {
    byte[] bytes = Files.readAllBytes(file.toPath());

}

publ

}
```


```
String content = new File('foo.txt').text
```

Read the content of an URL


```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLConnection;

public class GetURLContent {
    public static void main(String[] args) {

        try {
            URL url = new URL("http://www.google.com");
            URLConnection conn = url.openConnection();
            BufferedReader br = new BufferedReader(new InputStreamReader(conn.getInputStream()));
            String inputLine;

            while ((inputLine = br.readLine()) != null) {
                System.out.println(inputLine);
            }

            br.close();
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Read the content of an URL


```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLConnection;

public class GetURLContent {
    public static void main(String[] args) {

        try {
            URL url = new URL("http://www.google.com");
            URLConnection conn = url.openConnection();
            BufferedReader br = new BufferedReader(new InputStreamReader(conn.getInputStream()));
            String inputLine;

            while ((inputLine = br.readLine()) != null) {
                System.out.println(inputLine);
            }

            br.close();
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```


```groovy
println 'http://www.google.com'.toURL().text
```

Read the content of an URL

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import ja

public cl
    publi

        t

                                                                    tInputStream()));

        }
        }
        }
    }
}
```

```
println 'http://www.google.com'.toURL().text
```

GDK

- Groovy JDK
- Adds new types and methods
- http://www.groovy-lang.org/gdk.html

5.
I want more!

DSLs: Domain Specific Languages

```
def mailer = new Mailer()
mailer.setTo('admin@example.com', 'user@example.com')
mailer.setSubject('Urgent!')
mailer.setBody('Bla, bla, bla')
mailer.setHeaders(spam: 'no', important: true)
```

DSLs: Domain Specific Languages

```
def mailer = new Mailer()
mailer.setTo('admin@example.com', 'user@example.com')
mailer.setSubject('Urgent!')
mailer.setBody('Bla, bla, bla')
mailer.setHeaders(spam: 'no', important: true)
```

```
def mailer = new Mailer()
    .setTo('admin@example.com', 'user@example.com')
    .setSubject('Urgent!')
    .setBody('Bla, bla, bla')
    .setHeaders(spam: 'no', important: true)
```

DSLs: Domain Specific Languages

```
mail {
    to 'admin@example.com', 'user@example.com'
    subject 'Urgent!'
    body 'Bla, bla, bla'
    headers spam: 'no', important: true
}
```

DSLs: Domain Specific Languages

```
mail {
    to 'admin@example.com', 'user@example.com'
    subject 'Urgent!'
    body 'Bla, bla, bla'
    headers spam: 'no', important: true
}

class MailComposer {
    void to(String... addresses) { println "to: $addresses"}
    void subject(String subject) { println "subject: $subject" }
    void body(String body) { println "body: $body" }
    void headers(Map headers) { println "headers: $headers" }
}
```

DSLs: Domain Specific Languages

```
mail {
    to 'admin@example.com', 'user@example.com'
    subject 'Urgent!'
    body 'Bla, bla, bla'
    headers spam: 'no', important: true
}

class MailComposer {
    void to(String... addresses) { println "to: $addresses"}
    void subject(String subject) { println "subject: $subject" }
    void body(String body) { println "body: $body" }
    void headers(Map headers) { println "headers: $headers" }
}

void mail(@DelegatesTo(MailComposer) Closure composer) {
    // Same as:
    //   new MailComposer().with(composer)
    Closure cl = composer.clone()
    cl.delegate = new MailComposer()
    cl.resolveStrategy = Closure.DELEGATE_FIRST
    cl()
}
```

AST Transformations

- +50 transformations out-of-the-box
- @ToString, @EqualsAndHashCode, @InheritConstructors, @Sortable, @Delegate, @Immutable, @CompileStatic,...

@EqualsAndHashCode


```
public class User extends java.lang.Object {

    private java.lang.String name
    private java.lang.Integer age

    public int hashCode() {
        java.lang.Object _result = org.codehaus.groovy.util.HashCodeHelper.initHash()
        if (!(this.getName().is(this))) {
            _result = org.codehaus.groovy.util.HashCodeHelper.updateHash(_result, this.getName())
        }
        if (!(this.getAge().is(this))) {
            _result = org.codehaus.groovy.util.HashCodeHelper.updateHash(_result, this.getAge())
        }
        return _result
    }

    public boolean canEqual(java.lang.Object other) {
        return other instanceof User
    }

    public boolean equals(java.lang.Object other) {
        if ( other == null) {
            return false
        }
        if (this.is(other)) {
            return true
        }
        if (!( other instanceof User)) {
            return false
        }
        User otherTyped = (( other ) as User)
        if (!(otherTyped.canEqual( this ))) {
            return false
        }
        if (!(this.getName() == otherTyped.getName())) {
            return false
        }
        if (!(this.getAge() == otherTyped.getAge())) {
            return false
        }
        return true
    }
}
```

@EqualsAndHashCode


```groovy
@groovy.transform.EqualsAndHashCode
class User {
    String name
    Integer age
}
```


FEMPUTATIONS | TUMBLR
THANK YOU!
CN HD

Thank you!

Questions?

Iván López

@ilopmar

lopez.ivan@gmail.com

https://github.com/ilopmar

http://bit.ly/javacro-groovy