Technical Blog

Contents

[Reusable master template 1](#_Toc417842097)

[How this word document template is created 1](#_Toc417842098)

[Why Groovy / Features of Groovy 2](#_Toc417842099)

[Installing Groovy 2](#_Toc417842100)

[Package & Import statements 2](#_Toc417842101)

[HelloWorld example 2](#_Toc417842102)

[Data Types 2](#_Toc417842103)

[Class 2](#_Toc417842104)

[Constructor 3](#_Toc417842105)

[Methods 3](#_Toc417842106)

[Operators 3](#_Toc417842107)

[Closure 4](#_Toc417842108)

[If statement / Groovy truth 4](#_Toc417842109)

[For loop 4](#_Toc417842110)

[Number 4](#_Toc417842111)

[String 4](#_Toc417842112)

[List 5](#_Toc417842113)

[Map 5](#_Toc417842114)

[Processing Files 6](#_Toc417842115)

[Processing http request 6](#_Toc417842116)

[AST Transformations 7](#_Toc417842117)

[Reusable master template 7](#_Toc417842118)

# Reusable master template

This is reusable master template that can be copied and reused for this entire document. All content in this document need to be as per this template.

# How this word document template is created

Reference >> Table of contents to generate table of contents.

Home >> small arrow under change styles to select headings which are used for table content generation.

<https://www.youtube.com/watch?v=awRxezHJpSM>

# Why Groovy / Features of Groovy

* In groovy everything is an object and there are no primitive types. Even numbers are objects. So it is pure object oriented programming.
* Less code and more work when compared to java
* Easy to learn and any java developer can learn it in easy time. You can write pure java code in groovy files
* There are no checked exceptions

# Installing Groovy

Download binary file and extract it and include bin in path and groovy folder in GROOVY\_HOME variable.

Groovy –version in cmd

For plugin : <http://docs.codehaus.org/display/GROOVY/Eclipse+Plugin>

# Package & Import statements

Package and import statements have same significance as in java.

Default imports are java.lang , java.util , java.net , java.io , groovy.lang, groovy.util...

# HelloWorld example

When creating variables we use types. All java types can be use typed in groovy as well or no type is required

Ex: int a, def name

# Data Types

When creating variables we use types. All java types can be use typed in groovy as well or no type is required. Variables can be used without declaring but they become part of script binding

Ex: int a, def name

Supports Range data type like 0..9

# Class

By default class is public and syntax is

**class** Greetings{

**int** toName;

**def** address;

}

Every class is a pojo by default. Setter and Getter methods are available for all fields in my class. Though we access by field internally it calls getter method. If you mark a field as final then there is no setter method for that field.

# Constructor

**package** com.vogella.groovy.first

**public** **class** Person{

String firstName

String lastName

**int** age

def address

**static** **void** main(def args) {

Person p = **new** Person()

*// use the generated access methods*

p.setFirstName("Lars")

*// this will still use the generated access method, it is not a direct access!*

p.lastName = "Vogel"

p.address = ("Homestreet 3");

println(p.firstName + " " + p.lastName);

*// use the generated constructor*

p = **new** Person(firstName: "Peter", lastName:"Mueller");

println(p.firstName + " " + p.lastName);

}

}

# Methods

Return type can be java type or def.

Access modifier can be used and default one is public.

Always return value , if no value is returned then last calculated value is returned.

You can have optional arguments as defined below.

**static** sum(a,b,c=0){

a+b+c;

}

# Operators

== , Calls compareTo method or equals method

Is() used for ref comparision.

Str?.reverse

0..2 >> Range operator

# Closure

Closure is like a function without name / anonymous block kind of thing.

**Define closure**

Define multiply = {

Int a , int b -> a \* b;

}

Last calculated value is always returned by closure

**Use closure**

Assert multiply(2,5) == 10

Because of these closure number of for loops in your code comes down

# If statement / Groovy truth

Empty list = false

Null = false

# For loop

3.times{

}

For (i in 0..2){

}

# Number

Every number is object in groovy

There are many methods in Number class like 10.times{ closure}

# String

“” >> This is GString and it evaluates the expressions in it

‘ ‘ >> String and doesn’t evaluate expressions

def date = **new** Date()

println "We met at $date"

println "We met at ${date.format('MM/dd/yy')}"

# List

List list = new List[]

List<Integer> list = [1,2,3,4]

println list[0]

println list[1]

println list[2]

//To Add and element

List << e

To perform some operation on list object

findAll(closure) / find / grep

// grep for elements which are larger than 40 based on the closure

assert [50, 100, 300] == [10, 12, 30, 50, 100, 300].grep({ it > 40 })

No for loop required to iterate on all elements

# Map

def emptyMap = [:]

def anotherMap = [Jim:"Knopf", Thomas:"Edison"]

def map = ["Jim":"Knopf", "Thomas":"Edison"]

get(key) or get(key,default)

mymap.each { key, value ->

print key + " "

println value

}

# Processing Files

File file = **new** File("./input/test.txt")

file.eachLine{ line -> println line }

// adds a line number in front of each line to the console

def lineNumber = 0

file = **new** File("./input/test.txt")

file.eachLine{ line ->

lineNumber++

println "$lineNumber: $line"

}

// read the file into a String

String s = **new** File("./input/test.txt").text

println s

// write the content of the file to the console

File file = **new** File("output.txt")

file.write "Hello\n"

file.append "Testing\n"

file << "More appending...\n"

File result = **new** File("output.txt")

println (result.text)

// clean-up

file.delete()

# Processing http request

def data = **new** URL(http://www.vogella.com).text

# AST Transformations

The @Builder can be applied to a class and generates transparently a builder for this class.

*@Grab(group='org.eclipse.jetty.aggregate', module='jetty-all', version='7.6.15.v20140411')*

@GrabResolver(name='myrepo', root='http://myrepo.my-company.com/')

|  |  |  |
| --- | --- | --- |
| *@Singleton* | *Makes annotated class a Singleton, access via ClassName.instance.* | |
| *@Immutable* | | *Makes all fields as final and do not generate setters.* |

*@Sortable(excludes = ['duration'])*

*@ToString(includeFields=true)*

*@EqualsAndHashCode* (excludes=["summary","description"])

If a class is annotated with @TupleConstructor Groovy generates a constructor using all fields

# Reusable master template

This is reusable master template that can be copied and reused for this entire document. All content in this document need to be as per this template.