**Groovy**

1. Groovy is Optionally Type, dynamic language that runs on JVM

Optionally Type: No matter for data type and variable declaration

1. Groovy describe itself feature rich and java friendly
2. Groovy source code complied into java byte code by the groovy complier (we can compile in any platform like java)
3. We can easily integrate with Java and use third party libraries
4. Had features also from Python, Ruby and small talk
5. Operators can be overloaded
6. Literal declaration for maps, arrays, ranges and regular expressions
7. It has efficient object navigation

**What Is The Limitation Of Groovy?**

1. Groovy can be slower
2. Groovy might need lots of memory
3. Groovy start up time requires improvement
4. It requires Java knowledge
5. It takes sometimes to get use to like New Syntax, closures, default typing,
6. Documentation is thin
7. **Definitions:**

**Workspace**: Is a folder, you can store projects, is to group a set of projects together to make an app.

**Package**: Is a folder structure, to manage programs/files easily, package names in lowercases to avoid conflicts from classes/interfaces

Companies use their reversed Internet domain name to begin their packages,

EX: com.ups.smartTrailer

**Class:** blueprint or prototype or templatefrom which objects are created. It represents the set of properties and methods that are common to all objects of one type.

**Method:** Is a collection of statements that performs some specific task and return result to the caller.

Methods allows us to reuse the code without retyping code

**Variable**: Is the name given to a memory location, it is the basic unit of storage in program.

**Data Types**: Based on the type of a variable, OS allocates memory and decides what can be stored in the reserved memory.

**Constructor**: Is a collection of statements that are executed at the time of object creation. Each time an object created using new () keyword at least one constructer is invoked to assign initial values to the data members of the same class. Name of the constructer should be the name class.

**Object:** It is a basic unit of OOP and represents real life entities. A typical class creates many objects, interact by invoking methods. An object consists of

**State:** It represents attributes/properties

**Behavior:** It reps methods of object

**Identity:** It gives unique name to objects.

**Static:** Member belongs to the class instead of a specific instance. When a member is declared static, it can be accessed before any objects of its class created, w/o reference to any object**.**

**This: it’s a reference variable, refers to current object, refer instance object in current class.**

**Super:** used inside a subclass method definition to call a method defined in the super class.

**II) Data Types:**

**Data type define the type of data which you are storing /holding**

1. **Primitive**
   1. Numeric
      1. Integral data type
         1. Byte
         2. Short
         3. Int
         4. long
      2. Floating-Point data type
         1. Float
         2. Double
   2. Non-Numeric
      1. Boolean
      2. Char
2. **Non-Primitive**
   1. String
   2. Class
   3. Interface
   4. Arrays

**III) Variables:**

* To store the data/values
* When we create a variable it create some space in the memory (depends on data type).

1. **Types of Variables**
   1. **Instance(Global variables):** 
      1. Are declared within a class, but outside methods.
      2. These are Non-static variables
   2. **Local variables**
      1. Declared inside method
   3. **Static variables**
      1. A variable which is declared as static is static variable
      2. Known as class variables

**IV) Groovy Data Types:**

* Variables can be defined using keyword called **def**
* In groovy we have only one data type**, i.e def**
* We can use all the data types from Java also
* This why, groovy called as **Optionally Typed Language**

1. **Multiple Assignments:**

**Def (a,b,c) = [10,20,”Groovy”]**

**V) Data Type Conversion:**

Using **“as”**

**VI) Methods:**

* Is a collection of statements that perform some specific task and return result to the caller.
* Method can perform some specific task without returning anything.
* Allow us to reuse the code
* Not allows method inside method
* We can call a method from other method

1. **Simple Method:**

Without any parameters, just print with values

1. **Method with Parameters**
2. **Static Methods**
3. **Methods with Return Type**

No need to mention public, data type for methods, by default all methods are public, and data type we can mention def.

No need to write return also in return type methods.

**EX:**

**JAVA:**

public int sumOfThreeNums(defa, def b, def c){

int total = a+b+C

return total

}

**Groovy:**

def sumOfThreeNums(def a, def b, def c){

def total = a+b+C

total

}

Or

def sumOfThreeNums(a, b, c){

a+b+C

}

1. **Methods with Optional Parameters(Only in groovy)**