**What is Groovy?**

* Groovy is a dynamic programming language in which most of the program execution processes are done at runtime instead of compile time.
* Groovy can be categorized into the same family of scripting languages such as Ruby, Perl, or JavaScript.
* As we already know, learning a new language is a tedious activity because we need to learn the syntax, control structures, declarations, and so on for the new language.
* However, this is not true for Groovy if you already know the fundamentals of Java. Groovy uses Java-like bracket syntax and most Java code is syntactically valid in Groovy.
* Groovy scripts run on JVM similar to Java programs, hence we do not need to install and configure additional libraries.
* Groovy is a **loosely-typed language**, which means there is no need to define the data types for the variables and for the return types of the methods.

**HelloWorld with Groovy**

The usual HelloWorld script. Write the following script in the soapUI

Groovy Script editor:

//Print "Hello soapUI" string in soap ui console

**log.info "Hello soapUI"**

**Variable and object declaration in Groovy**

As with any programming language, variables or objects must be declared before they are referenced by somewhere else. The variables can be declared with the keyword def, as shown in the following script:

**def name = "soapui"** //declare variable name and assign value soapui

We can even declare variables without the def keyword. You can assign any object to variables defined with def, and return any kind of object if a method is declared returning def. Remember, if you declare the variable with def, there is no need to specify a type. Therefore, the following declaration is unnecessary:

**def String name ="soapui"**

To read the value of a variable, you can just prefix the variable name with $ as in Case 1 or append it as in Java (Case 2).

**Case 1:**

**def name = "soapui"**

**print "Hello $name \n"**

**Case 2:**

**def name ="soapui"**

**print "Hello " +name**

By default the standard Java packages such as java.lang.\*, java.util.\*, java. io.\*, and so on, are included by Groovy interpreter. Thus, the objects in Groovy can be declared in the same way as we do with Java.

The following code will instantiate String object.

**def strObject = new String("soapui")**

Groovy has support for two collection data types:

• **Lists**: These are used to store ordered collections of data. A list can be

declared as follows:

myList = [0, 32, -90, 45, 89923]

The above statement declares a list object, which holds integer values. We

can access a value stored in list with myList[n], where *n* is the index of list.

**Maps**: These store different types of data in key-value pairs. For example,

consider the following script:

**myMap = ["key1":"soapui", "key2":100, "key3":30.05]** //different // types of data are stored in a map

println **myMap["key2"]** //access the value assigned to "key2"

println **myMap.key2** //another way of accessing value assigned to // key2

**Control structures in Groovy**

The syntax of control structures such as "if-else", "for", and "while" are very similar to what we have in Java. Look at the following code snippet:

**status = true**

**strObject = new String("Hello")**

**myList = ["1","2", "3"]**

**if (status && strObject && myList) { //All will evaluate to true**

**println "Condition is true"**

**}else{**

**println "Condition is false"**

**}**

Run the code snippet and you will get Condition is true as the output. Here, strObject (String object) and myList (Collection object) will return false only if either of the two or both are null or empty.

The syntax of the for loop is similar to the following:

**for(Object in IterableObject){**

**// Set of Statements.**

**}**

**def names = ["Saman", "Nethul", "Risith", "Charitha"] // A List**

**// object holdingnames**

**for(name in names){//Iterate over the elements in names list**

**println name**

**}**

**Class and method declarations in Groovy**

Declaration of classes in Groovy is almost the same as it is in Java.

**class Employee{**

**private def id**

**private def name**

**def address**

**public Employee(){**

**}**

**Employee(id, name, address){**

**this.id = id**

**this.name = name**

**this.address = address**

**}**

**public String getId(){**

**return id**

**}**

**def setId(id){**

**this.id = id**

**}**

**public String getName(){**

**return name**

**}**

**def setName(name){**

**this.name = name**

**}**

**static main(arguments){**

**def emp1 = new Employee("100", "Charitha", "Colombo")**

**println("Employee name is "+emp1.getName())**

**}**

**}**

You may notice that in the variable declarations section, we did not explicitly mention the data types of id, name, and address variables. We also did not specify what the access modifiers were. As Groovy is a loosely-typed language, we do not want to specify the data types and access modifiers. The default access modifier of Groovy is public.