Java 8 - Nashorn JavaScript

With Java 8, Nashorn, a much improved javascript engine is introduced, to replace the existing Rhino. Nashorn provides 2 to 10 times better performance, as it directly compiles the code in memory and passes the bytecode to JVM. Nashorn uses **invokedynamics** feature, introduced in Java 7 to improve performance.

jjs

For Nashorn engine, JAVA 8 introduces a new command line tool, **jjs,** to execute javascript codes at console.

Interpreting js File

Create and save the file **sample.js** in c:\> JAVA folder.

sample.js

print('Hello World!');

Open console and use the following command.

$jjs sample.js

It will produce the following output:

Hello World!

jjs in Interactive Mode

Open the console and use the following command.

$jjs

jjs> print("Hello, World!")

Hello, World!

jjs> quit()

>>

Pass Arguments

Open the console and use the following command.

$jjs -- a b c

jjs> print('letters: ' +arguments.join(", "))

letters: a, b, c

jjs>

Calling JavaScript from Java

Using ScriptEngineManager, JavaScript code can be called and interpreted in Java.

Example

Create the following Java program using any editor of your choice in, say, C:\> JAVA.

Java8Tester.java

import javax.script.ScriptEngineManager;

import javax.script.ScriptEngine;

import javax.script.ScriptException;

public class Java8Tester {

public static void main(String args[]){

ScriptEngineManager scriptEngineManager = new ScriptEngineManager();

ScriptEngine nashorn = scriptEngineManager.getEngineByName("nashorn");

String name = "Mahesh";

Integer result = null;

try {

nashorn.eval("print('" + name + "')");

result = (Integer) nashorn.eval("10 + 2");

}catch(ScriptException e){

System.out.println("Error executing script: "+ e.getMessage());

}

System.out.println(result.toString());

}

}

Verify the Result

Compile the class using **javac** compiler as follows −

$javac Java8Tester.java

Now run the Java8Tester as follows −

$java Java8Tester

It should produce the following result −

Mahesh

12

Calling Java from JavaScript

The following example explains how to import and use Java classes in java script −

sample.js

var BigDecimal = Java.type('java.math.BigDecimal');

function calculate(amount, percentage) {

var result = new BigDecimal(amount).multiply(

new BigDecimal(percentage)).divide(new BigDecimal("100"), 2, BigDecimal.ROUND\_HALF\_EVEN);

return result.toPlainString();

}

var result = calculate(568000000000000000023,13.9);

print(result);

Open the console and use the following command.

$jjs sample.js

It should produce the following output −

78952000000000000003.20