**Problem Statement**

Write a calculator program in Java that evaluates expressions in a very simple integer expression language. The program takes an input on the command line, computes the result, and prints it to the console.  For example:

% java calculator.Main "add(2, 2)"

4

Few more examples:

|  |  |
| --- | --- |
| **Input** | **Output** |
| **add(1, 2)** | 3 |
| **add(1, mult(2, 3))** | 7 |
| **mult(add(2, 2), div(9, 3))** | 12 |
| **let(a, 5, add(a, a))** | 10 |
| **let(a, 5, let(b, mult(a, 10), add(b, a)))** | 55 |
| **let(a, let(b, 10, add(b, b)), let(b, 20, add(a, b))** | 40 |

**Logging**

Implemented Logging layer to log all relevant information with 3 levels of verbosity.INFO & ERROR logging is included in the code. DEBUG logging is implemented while running Junit test cases.

**Build**

Created Maven project therefore all the dependencies are in pom.xml fileIt takes input as a string

Dependencies used are

* 1. **Slf4j for logging – stable and non-vulnerable (1.7.25)**
  2. **Junit for testing – stable and non-vulnerable (4.12)**

**Steps to Run Project**

1. Clone the directory from GitHub

***git clone*** [***https://github.com/Shivam1989/Shivam\_Synopsys.git***](https://github.com/Shivam1989/Shivam_Synopsys.git)

1. Clean and build the code

**mvn clean install**

**mvn compile**

1. To run the main file

**mvn -X exec:java -Dexec.mainClass="com.synopsys.calculator.CalculatorApplication " -Dexec.args="let(a,5,add(a,a))”**

**Test Cases**

There are 10 Junit test cases to test the code

* 1. To test the input string
  2. To test the divide by zero
  3. To test the negative numbers
  4. To test the functionality of add, subtract, multiply, division and let function

1. To run the test file

**mvn test**

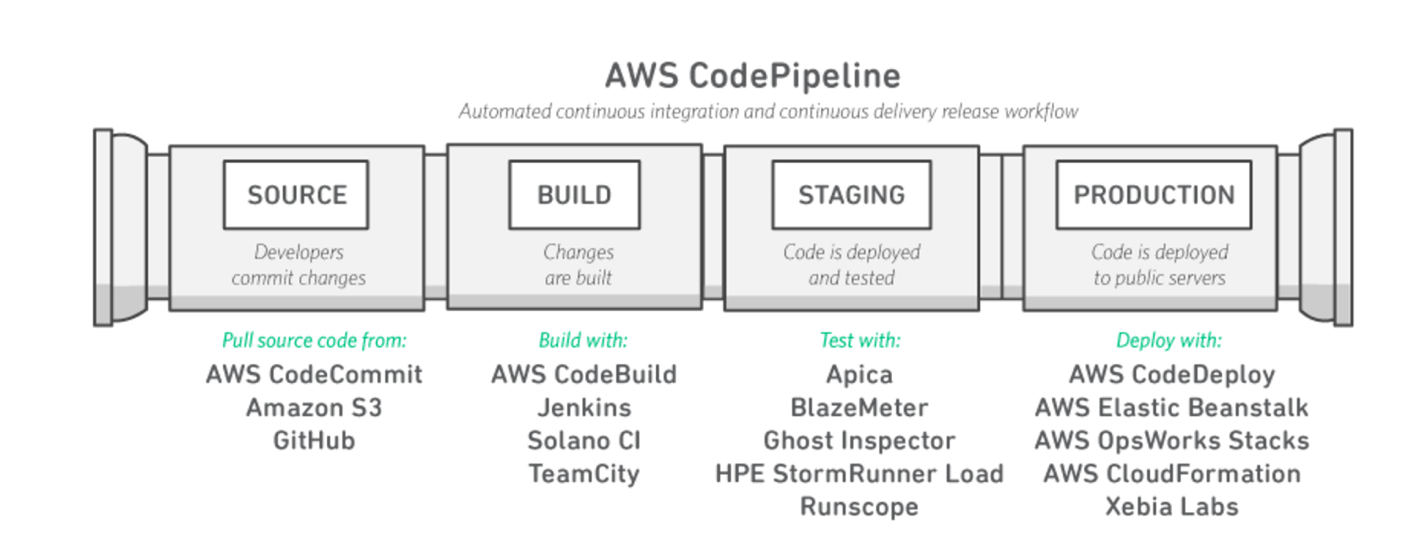
**mvn surefire:test -Dtest=com.synopsys.test.TestSuiteRunner**

**Continuous Integration Build**

I have used services like AWS CodePipeline, AWS CodeBuild, AWS CodeDeploy for continuous Integration Build

**AWS CodePipeline**

It builds, tests, and deploys your code every time there is a code change, based on the release process models you define. We can easily build out an end-to-end solution by using our pre-built plugins for popular third-party services like GitHub or integrating our own custom plugins into any stage of our release process. With AWS CodePipeline, we only pay for what you use. There are no upfront fees or long-term commitments.

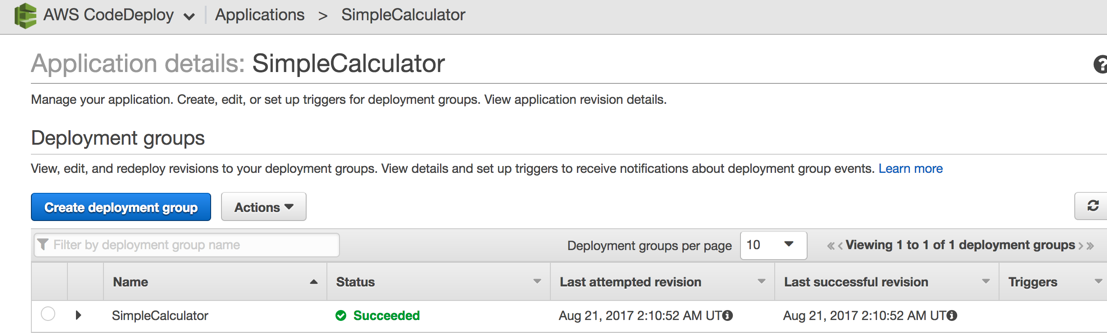


**AWS CodeBuild**

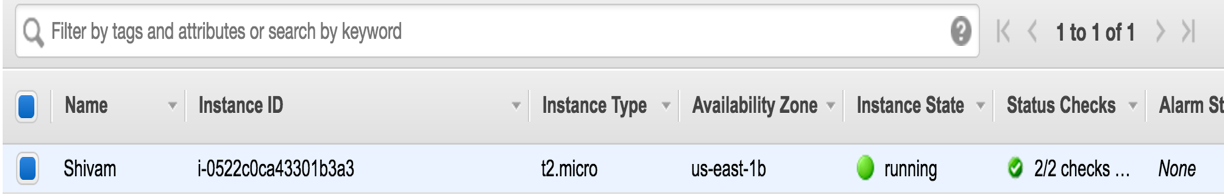
It is fully managed build service that compiles source code, runs tests, and produces software packages that are ready to deploy.

**Screenshots**

1. Hook to GitHub **(Shivam1989/Shivam\_Synopsys)**
2. **AWS CodeBuild**
3. **AWS CodeDeploy**

****

1. **EC2 Instance**

****

**Assumption**

1. When we divide the number by zero, it give +/- Inf when the result is assigned to int but when it is assigned to float it writes Infinity because does not have +/- Inf in it. Therefore, calculator will throw Arithmetic exception using “**throw**” keyword.
2. I am considering negative numbers as invalid argument. It can be extended at later stages.
3. While running application through console, the argument should have no space.
4. I have tried to FindBugs plugin, just to make sure my program is not using any library which are vulnerable.
5. I am assuming maximum length of variable of 100 to avoid malware attack.