Problem Statement 4

Method-Trace Analyzer

The problem statement is aimed at developing an application to assist developers in debugging code. Method tracing is one of the commonly used post-mortem diagnostic method to identify problems. It consists of timestamp of entry and exit points for each method invocation. They may also contain stack-trace for each invocation. Depending upon the time for which trace data is collected, the file-size for these traces can be huge. Parsing them manually is a time-consuming and error-prone task.

To learn more about IBM Java Method trace, please read:
https://www.ibm.com/support/knowledgecenter/SSYKE2_8.0.0/com.ibm.java.vm.80.doc/docs/tracing.html

Example and explanation of what method trace looks like:
https://www.ibm.com/support/knowledgecenter/SSYKE2_8.0.0/com.ibm.java.vm.80.doc/docs/trace_method_trace_output.html

Problem Description

For post-mortem debugging, method trace is one of the reliable options. For example, in case of hang / performance related issues - The time taken for each method invocation becomes a key factor. Number of method invocations can be another clue.

In case of functional issues – Change in code-flow / Stack-trace.

Gathering and comparing these data is time-consuming and prone to human error.
The challenge is to create an application to aid developers in debugging Java method trace files.

Expectation

Develop a Java based GUI application which can parse and compare multiple method trace files. The primary goals would be to -

1. Compare two trace files: one from failing and passing case each and find out the anomaly.
2. Parse one or more trace file and suggest anomalies - I.e. Flag methods which are not completing their execution / Taking longer to execute. This will be helpful in addressing hang and performance related problems.
3. Parse one or more trace file and create a tabular and graphical view for the number of times each method is invoked. Comparative view in case of multiple files.
4. Compare code-flow and stack trace for failing and passing case and find anomalies.

Evaluation Criteria

The evaluation parameters are listed on the hackathon landing page.

Tools & Technology

- Java

Resources & References

- https://www.ibm.com/support/knowledgecenter/SSYKE2_8.0.0/com.ibm.java.vm.80.doc/docs/tracing.html
- https://www.ibm.com/support/knowledgecenter/SSYKE2_8.0.0/com.ibm.java.vm.80.doc/docs/trace_method_trace_output.html
- https://adoptopenjdk.net/?variant=openjdk8&jvmVariant=openj9

Frequently asked questions

Q: What are the programming languages?

A: Java

Q: What are mobile platforms allowed?

A: NA

Q: Where to get free access to IBM Cloud?

A: Login here: https://www.ibm.com/cloud/

Q: Is there any documentation available to use IBM Cloud?

A: Yes, each service comes with elaborate documentation with step by step illustration to use the services available on IBM cloud, follow the VIEW DOCS, link available on each service.

Q: Is the knowledge of ML/DL is required?

A: No

Q: Is there any dataset provided?

A: No, there is no dataset made available with this challenge, you can create/mine your own dataset using the instructions here https://www.ibm.com/support/knowledgecenter/SSYKE2_8.0.0/com.ibm.java.vm.80.doc/docs/tracing.html

Note: Java with IBM J9 VM is required to use tracing functionality. It can be downloaded from https://adoptopenjdk.net/?variant=openjdk8&jvmVariant=openj9

Post your technical queries here.