Simple and scalable performance testing with JMeter DSL

Improving the DX

https://charlyautomatiza.tech

Carlos Gauto

- 17+ years of IT experience
- Project Leader
- Playwright Ambassador
- K6 Champion
- SelectorsHub Ambassador
- Technology Content Creator

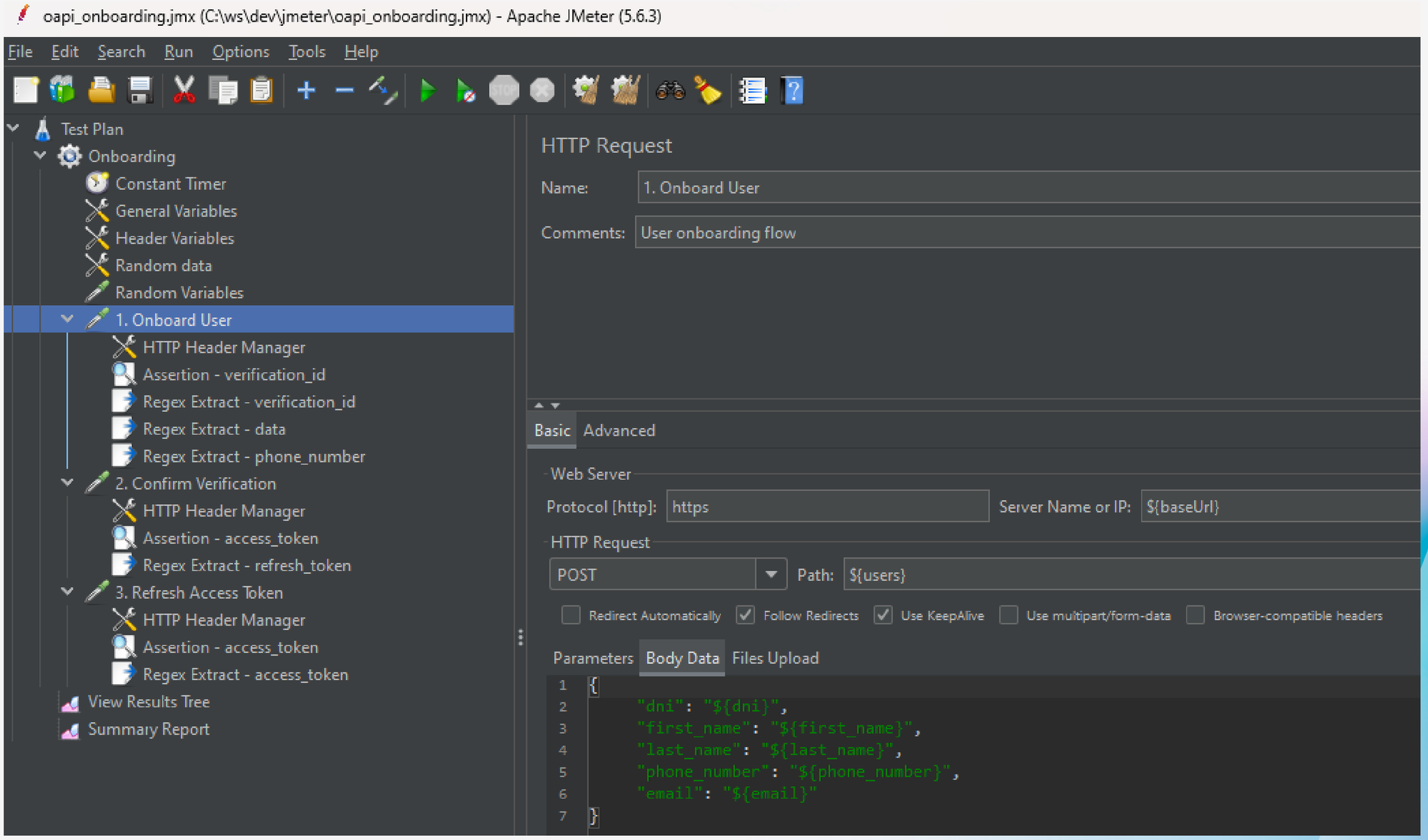
@Char_Automatiza CharlyAutomatiza CharlyAutomatiza

@CharlyAutomatiza | @char_automatiza

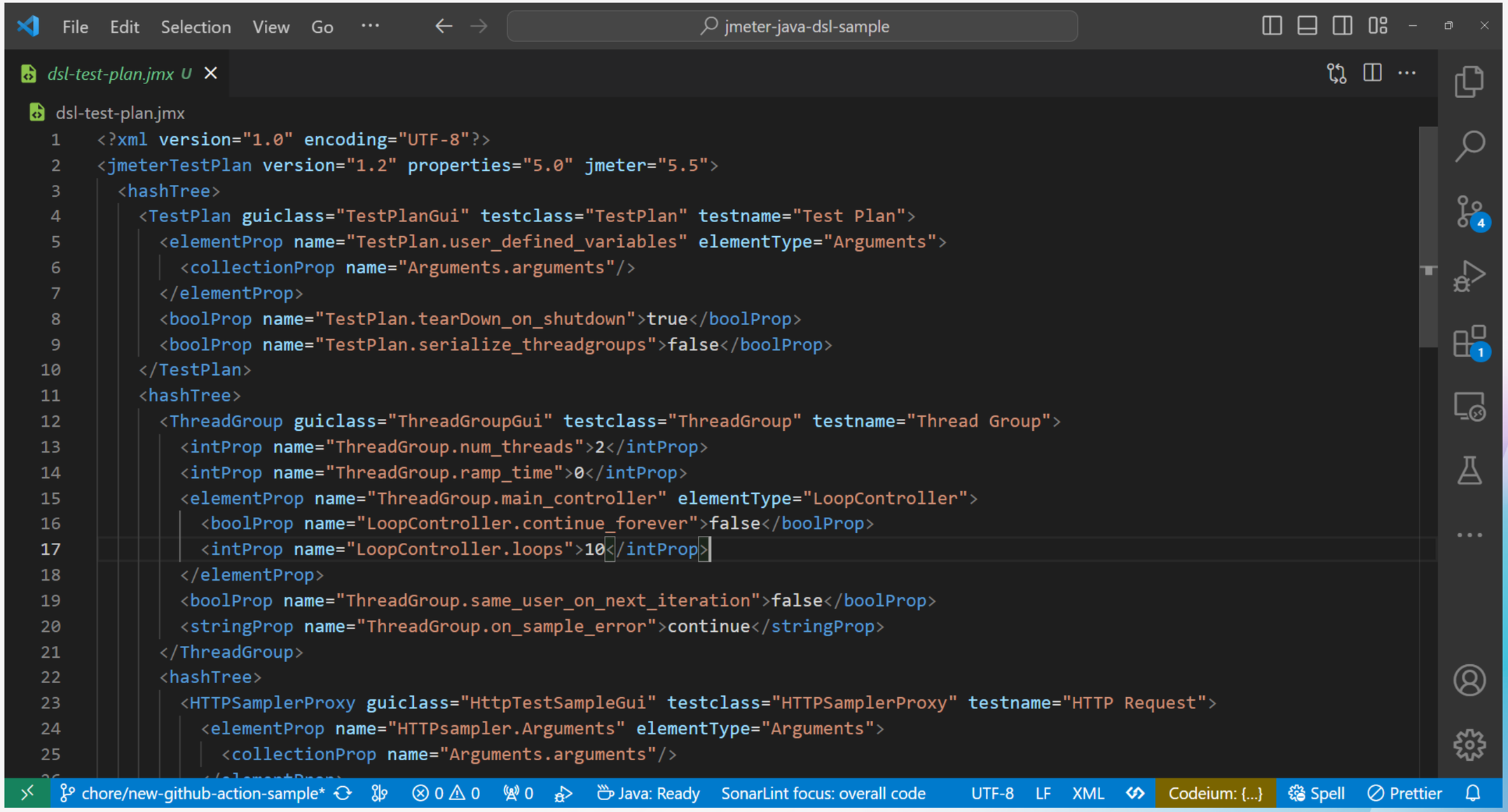
What is JMeter?

A look at what we could do and what it cost us a little.

JMeter - what its user interface looks like



JMeter - what the project code looks like



What is JMeter DSL?

Filling in the blanks to enhance the DX

How to start using JMeter DSL

Maven

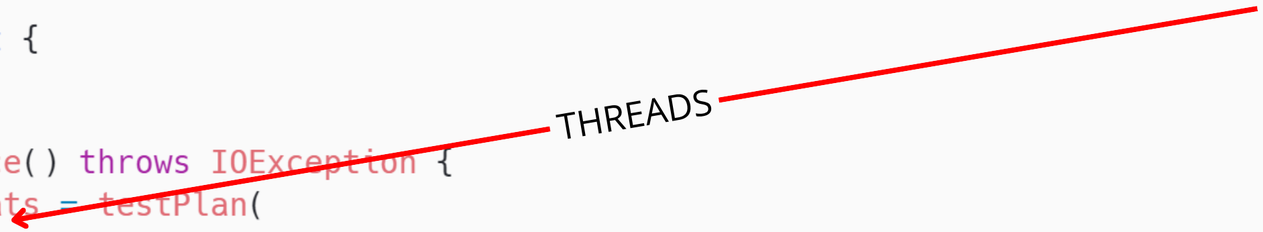
```xml
<dependency>
  <groupId>us.abstracta.jmeter</groupId>
  <artifactId>jmeter-java-dsl-azure</artifactId>
  <version>1.25.3</version>
  <scope>test</scope>
</dependency>
```

```
testImplementation 'us.abstracta.jmeter:jmeter-java-dsl-azure:1.25.3'
```

Gradle

Hello world using JMeter DSL

```java
package us.abstracta.jmeter.javadsl.sample;

import static org.assertj.core.api.Assertions.assertThat;
import static us.abstracta.jmeter.javadsl.JmeterDsl.*;

import java.io.IOException;
import java.time.Duration;
import org.junit.jupiter.api.Test;
import us.abstracta.jmeter.javadsl.core.TestPlanStats;

class PerformanceTest {

  @Test
  void testPerformance() throws IOException {
    TestPlanStats stats = testPlan(
        threadGroup(8, 2,
            httpSampler("https://restful-booker.herokuapp.com/booking")))
        .run();
    assertThat(stats.overall().sampleTimePercentile99()).isLessThan(Duration.ofSeconds(5));
  }
}
```



Hello world using JMeter DSL

```
package us.abstracta.jmeter.javadsl.sample;

import static org.assertj.core.api.Assertions.assertThat;
import static us.abstracta.jmeter.javadsl.JmeterDsl.*;

import java.io.IOException;
import java.time.Duration;
import org.junit.jupiter.api.Test;
import us.abstracta.jmeter.javadsl.core.TestPlanStats;

class PerformanceTest {

  @Test
  void testPerformance() throws IOException {
    TestPlanStats stats = testPlan(
        threadGroup(8, 2,
            httpSampler("https://restful-booker.herokuapp.com/booking")))
        .run();
    assertThat(stats.overall().sampleTimePercentile99()).isLessThan(Duration.ofSeconds(5));
  }
}
```

Hello world using JMeter DSL

```java
package us.abstracta.jmeter.javadsl.sample;

import static org.assertj.core.api.Assertions.assertThat;
import static us.abstracta.jmeter.javadsl.JmeterDsl.*;

import java.io.IOException;
import java.time.Duration;
import org.junit.jupiter.api.Test;
import us.abstracta.jmeter.javadsl.core.TestPlanStats;

class PerformanceTest {

  @Test
  void testPerformance() throws IOException {
    TestPlanStats stats = testPlan(
        threadGroup(8, 2,
            httpSampler("https://restful-booker.herokuapp.com/booking")))
        .run();
    assertThat(stats.overall().sampleTimePercentile99()).isLessThan(Duration.ofSeconds(5));
  }
}
```

Threads configuration

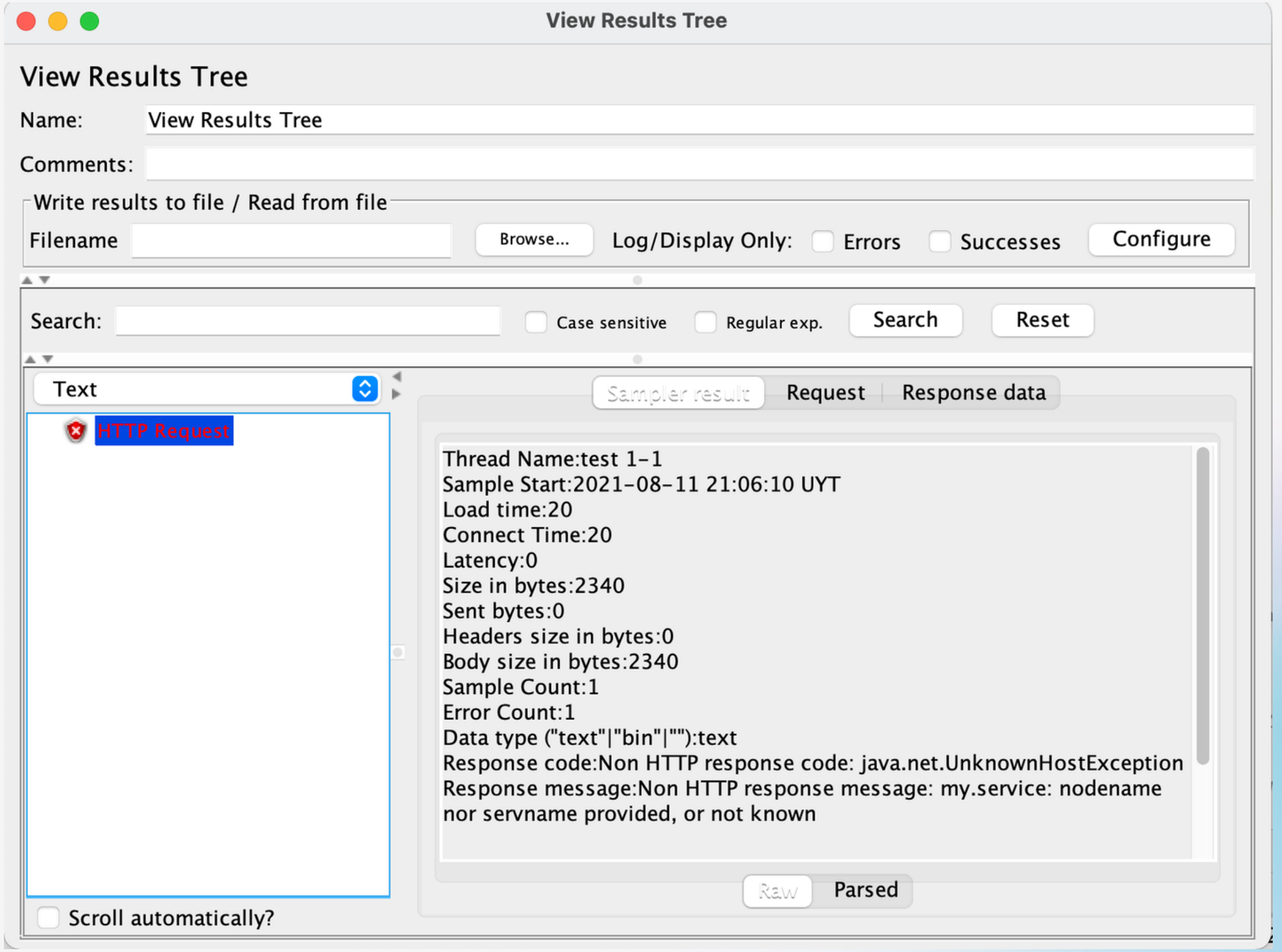
```
// We can use and combine methods to configure more complex scenarios
threadGroup()
    .rampToAndHold(10, Duration.ofSeconds(5), Duration.ofSeconds(20))
    .rampToAndHold(100, Duration.ofSeconds(10), Duration.ofSeconds(30))
    .rampTo(200, Duration.ofSeconds(10))
    .rampToAndHold(100, Duration.ofSeconds(10), Duration.ofSeconds(30))
    .rampTo(0, Duration.ofSeconds(5))
    .children(
      httpSampler("https://restful-booker.herokuapp.com/booking")
    )
```

Threads configuration - Threads timeline

Test plan debugging - Results Tree Visualizer

```
import static us.abstracta.jmeter.javadsl.JmeterDsl.*;

import java.io.IOException;
import org.junit.jupiter.api.Test;

public class PerformanceTest {

  @Test
  public void testPerformance() throws IOException {
    testPlan(
        threadGroup(1, 1,
            httpSampler("https://restful-booker.herokuapp.com/booking")
        ),
        resultsTreeVisualizer() // allows us to do debugging in a simple way
    ).run();
  }
}
```

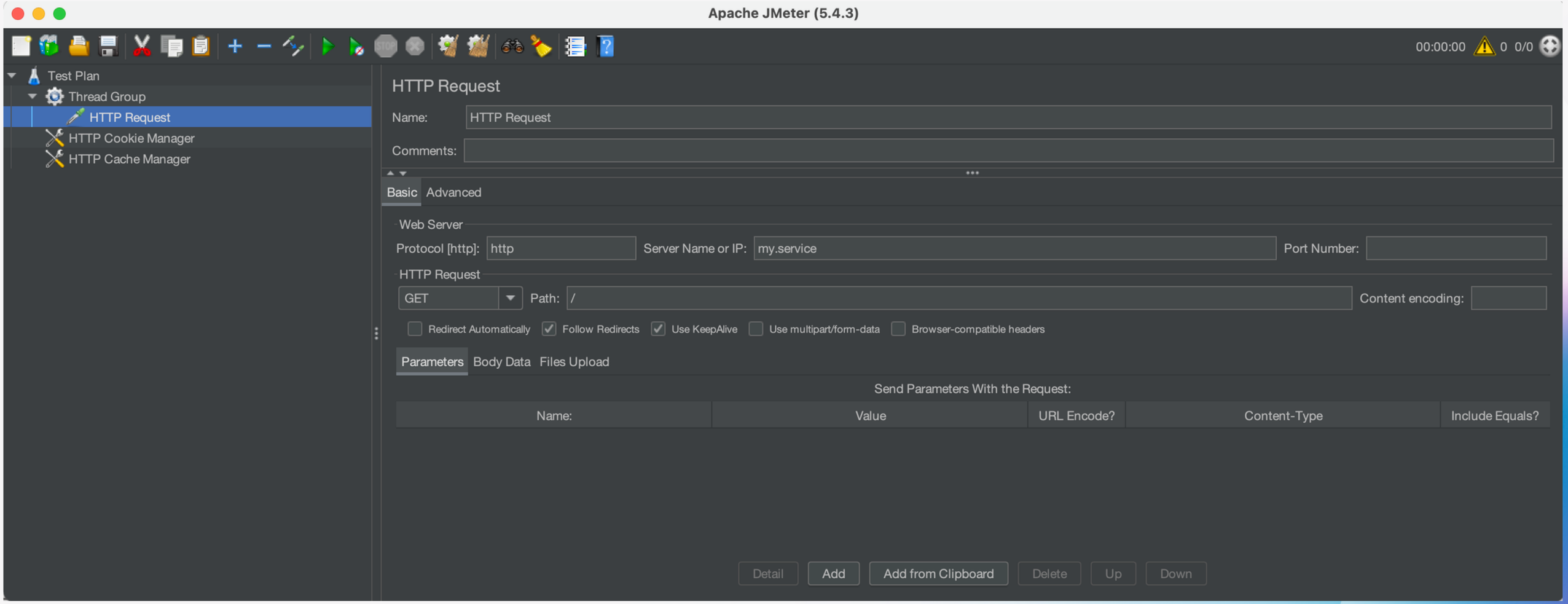
Test plan debugging - Results Tree Visualizer



Test plan review in JMeter GUI - Show GUI

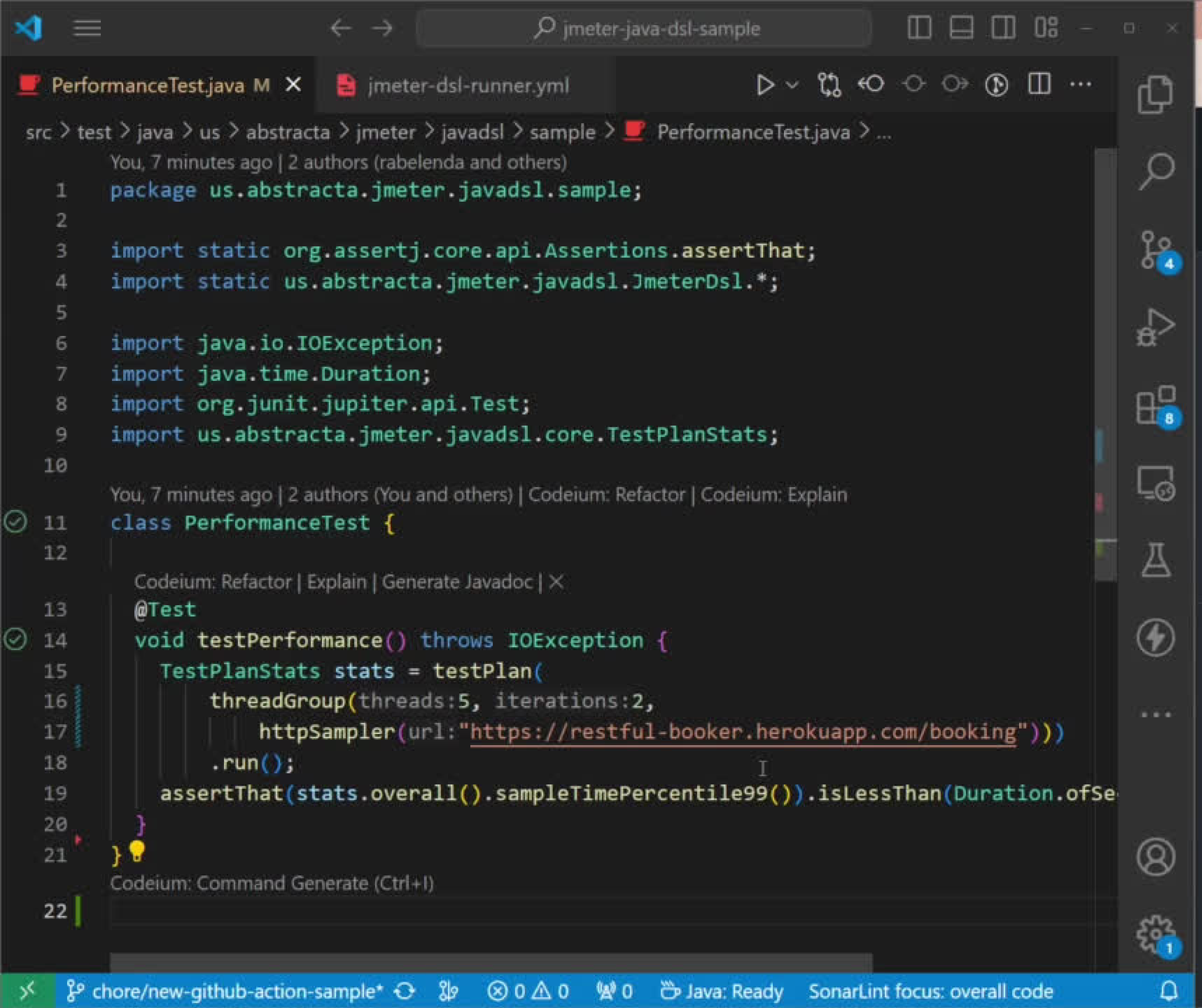
```
import static us.abstracta.jmeter.javadsl.JmeterDsl.*;

import java.io.IOException;
import org.junit.jupiter.api.Test;

public class PerformanceTest {

  @Test
  public void testPerformance() throws IOException {
    testPlan(
        threadGroup(2, 10,
            httpSampler("https://restful-booker.herokuapp.com/booking)
        )
    ).showInGui();
  }
}
```

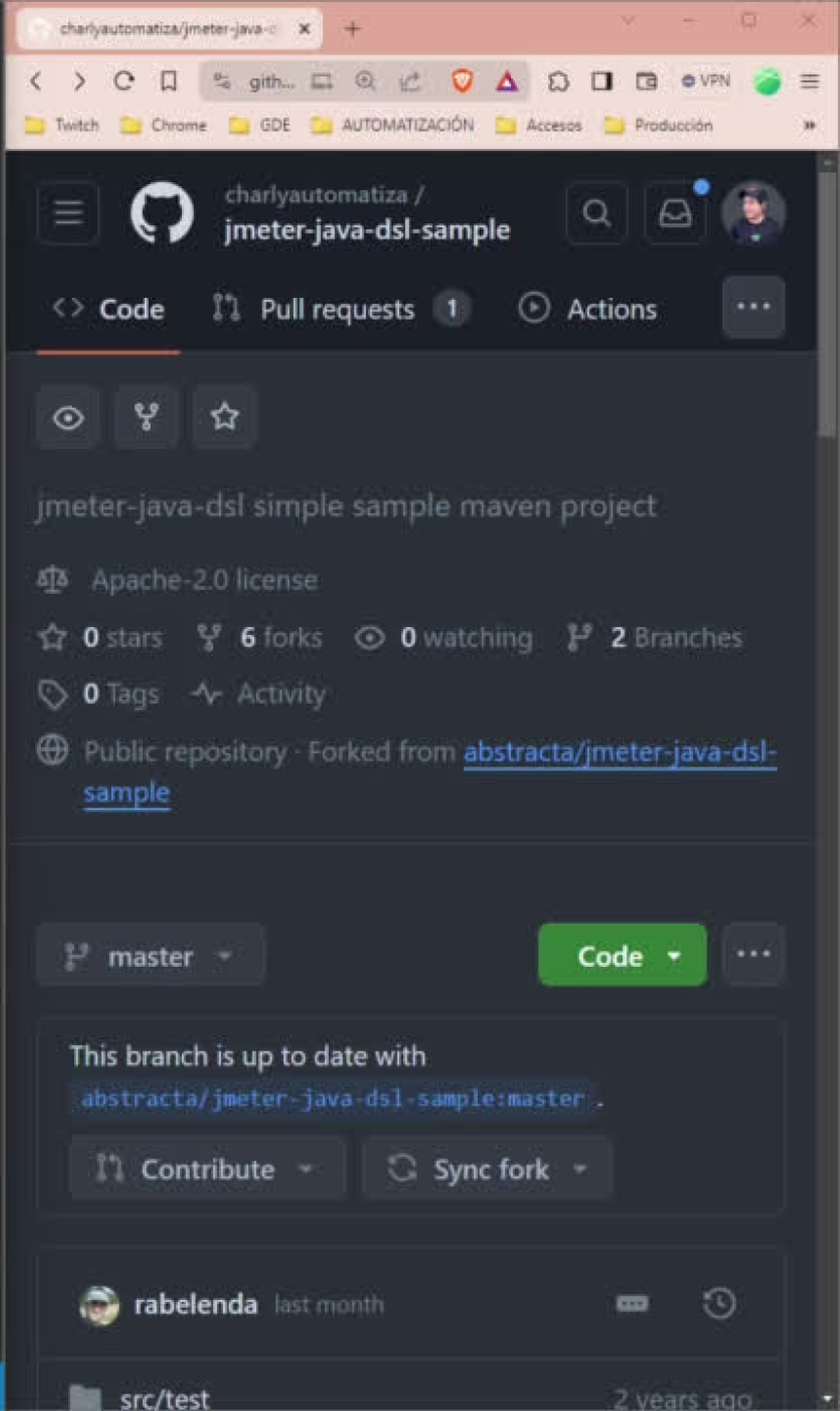
Test plan review in JMeter GUI



JMeter DSL - Demo

Filling in the blanks to enhance the DX


jmeter-java-dsl-sample
PerformanceTest.java M
jmeter-dsl-runner.yml
src > test > java > us > abstracta > jmeter > javadsl > sample > PerformanceTest.java > ...
You, 7 minutes ago | 2 authors (rabelenda and others)
1 package us.abstracta.jmeter.javadsl.sample;
2
3 import static org.assertj.core.api.Assertions.assertThat;
4 import static us.abstracta.jmeter.javadsl.JmeterDsl.*;
5
6 import java.io.IOException;
7 import java.time.Duration;
8 import org.junit.jupiter.api.Test;
9 import us.abstracta.jmeter.javadsl.core.TestPlanStats;
10
You, 7 minutes ago | 2 authors (You and others) | Codeium: Refactor | Codeium: Explain
11 class PerformanceTest {
12
Codeium: Refactor | Explain | Generate Javadoc |
13 @Test
14 void testPerformance() throws IOException {
15 TestPlanStats stats = testPlan(
16 threadGroup(threads:5, iterations:2,
17 httpSampler(url:"https://restful-booker.herokuapp.com/booking")))
18 .run();
19 assertThat(stats.overall().sampleTimePercentile99()).isLessThan(Duration.ofSe
20 }
21 }
Codeium: Command Generate (Ctrl+I)
22
chore/new-github-action-sample*
0 0
0
Java: Ready
SonarLint focus: overall code


charlyautomatiza / jmeter-java-dsl-sample
Code
Pull requests 1
Actions
jmeter-java-dsl simple sample maven project
Apache-2.0 license
0 stars 6 forks 0 watching 2 Branches
0 Tags Activity
Public repository · Forked from abstracta/jmeter-java-dsl-sample
master
Code
This branch is up to date with abstracta/jmeter-java-dsl-sample:master .
Contribute
Sync fork
rabelenda last month
src/test
2 years ago

How to run test at scale with JMeterDSL?

Run test at scale - JMeter remote testing

```
import static org.assertj.core.api.Assertions.assertThat;
import static us.abstracta.jmeter.javadsl.JmeterDsl.*;

import java.time.Duration;
import org.junit.jupiter.api.Test;
import us.abstracta.jmeter.javadsl.core.engines.DistributedJmeterEngine;
import us.abstracta.jmeter.javadsl.core.TestPlanStats;

public class PerformanceTest {

  @Test
  public void testPerformance() throws Exception {
    TestPlanStats stats = testPlan(
        threadGroup(200, Duration.ofMinutes(10),
            httpSampler("https://restful-booker.herokuapp.com/booking")
        )
    ).runIn(new DistributedJmeterEngine("host1", "host2"));
    assertThat(stats.overall().sampleTimePercentile99()).isLessThan(Duration.ofSeconds(5));
  }
}
```

docker-compose sample: https://bit.ly/jmeter-dsl-distributed

Run test at scale - Azure Load Testing

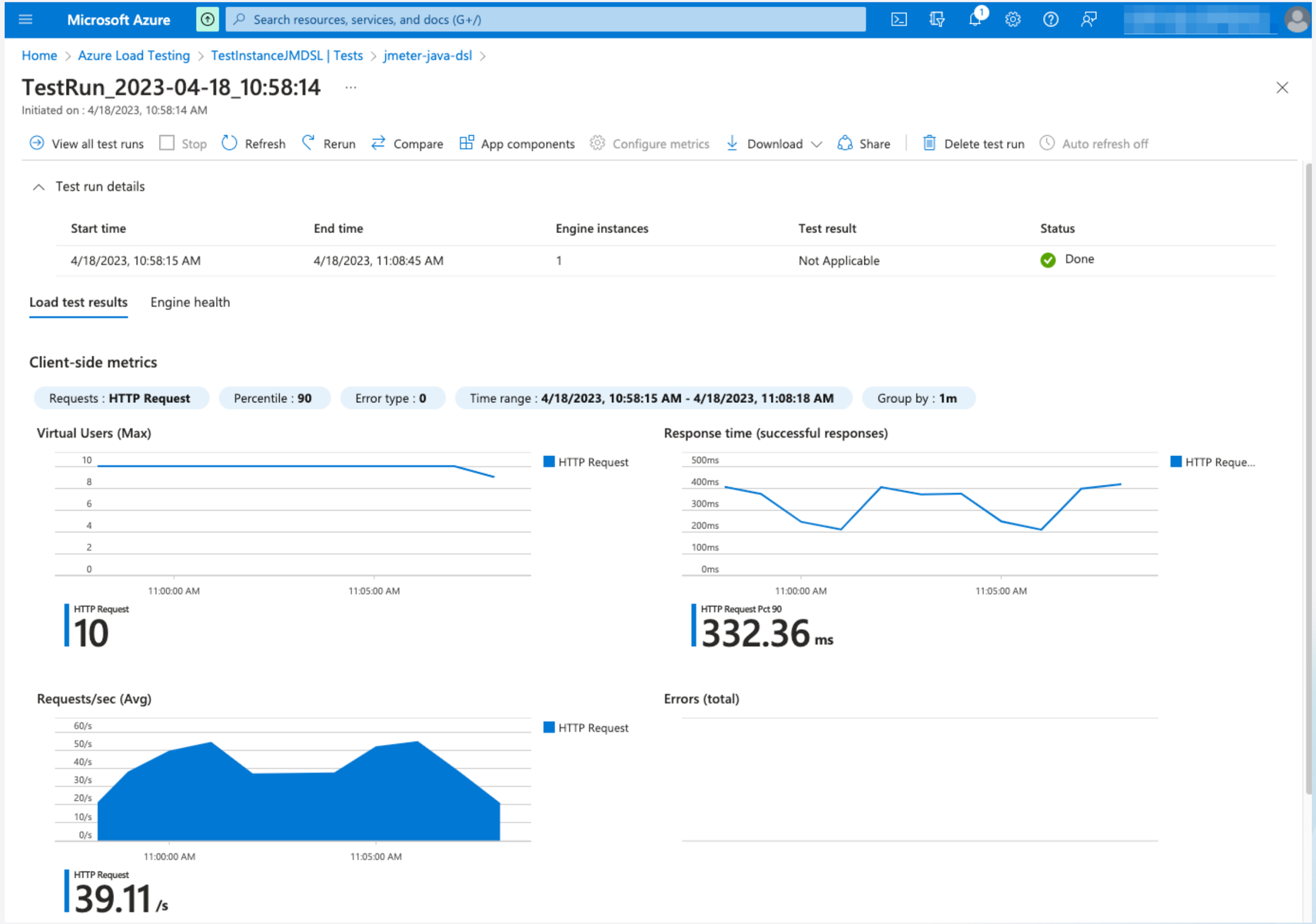
```java
import static org.assertj.core.api.Assertions.assertThat;
import static us.abstracta.jmeter.javadsl.JmeterDsl.*;

import java.time.Duration;
import org.junit.jupiter.api.Test;
import us.abstracta.jmeter.javadsl.azure.AzureEngine;
import us.abstracta.jmeter.javadsl.core.TestPlanStats;

public class PerformanceTest {

  @Test
  public void testPerformance() throws Exception {
    TestPlanStats stats = testPlan(
        threadGroup(2, 10,
            httpSampler("https://restful-booker.herokuapp.com/booking")
        )
    ).runIn(new AzureEngine(System.getenv("AZURE_CREDS")) // AZURE_CREDS=tenantId:clientId:secretId
        .testName("dsl-test")
        .engines(2)
        .testTimeout(Duration.ofMinutes(20)));
    assertThat(stats.overall().sampleTimePercentile99()).isLessThan(Duration.ofSeconds(5));
  }
}
```

Run test at scale - Azure Load Testing

```java
import static org.assertj.core.api.Assertions.assertThat;
import static us.abstracta.jmeter.javadsl.JmeterDsl.*;

import java.time.Duration;
import org.junit.jupiter.api.Test;
import us.abstracta.jmeter.javadsl.azure.AzureEngine;
import us.abstracta.jmeter.javadsl.core.TestPlanStats;

public class PerformanceTest {

  @Test
  public void testPerformance() throws Exception {
    TestPlanStats stats = testPlan(
        threadGroup(2, 10,
            httpSampler("https://restful-booker.herokuapp.com/booking")
        )
    ).runIn(new AzureEngine(System.getenv("AZURE_CREDS")) // AZURE_CREDS=tenantId:clientId:secretId
        .testName("dsl-test")
        .engines(2)
        .testTimeout(Duration.ofMinutes(20)));
    assertThat(stats.overall().sampleTimePercentile99()).isLessThan(Duration.ofSeconds(5));
  }
}
```

ENGINES - AGENTS

Azure Load Testing - Reports



JMeterDSL - Summary

Pros and cons

JMeterDSL - Summary

Pros

- 👍 IDE friendly + CI/CD integration
- 👍 Code modularization
- 👍 Support for JMeter supported protocols
- 👍 Interact with JMX files and take advantage of JMeter ecosystem
- 👍 All details of simple test plans at a glance
- 👍 Simple way to do assertions on statistics

Cons

- 👎 Basic Java knowledge required
- 👎 Same resources (CPU & RAM) usage as JMeter

https://charlyautomatiza.tech

Thank you

@Char_Automatiza

CharlyAutomatiza

CharlyAutomatiza