



Learn Performance Testing with JMeter, RedLine13 & BlazeMeter

Powered by

testscenario

testscenario.com

perfect QA

perfectqaservices.com

Performance Testing

Module 1: Introduction to Performance Testing

What performance testing is and why it matters

Types: load, stress, spike, soak (endurance), scalability, volume

Non-functional requirements: SLAs, response-time targets, expected users and throughput

Installing Java and JMeter; Plugins Manager

JMeter UI tour

Components: Test Plan, Thread Group, Sampler, Config Element, Pre/Post Processor, Assertion, Timer, Listener, Logic Controller

Execution order and scoping rules

Lab: first test plan with one HTTP Request against blazedemo.com

Module 2: Recording and Playback

HTTP(S) Test Script Recorder; proxy setup in the browser

Installing the JMeter root CA certificate for HTTPS recording

Recording filters: exclude static resources and third-party domains

Config elements: HTTP Request Defaults, Cookie Manager, Header Manager, Cache Manager

Parameterization with CSV Data Set Config (sharing mode, recycle, stop on EOF)

Replaying the recording and spotting why it fails

Lab: record a login and booking flow, parameterize users from CSV

Module 3: Building Robust Test Scripts

Correlation: finding dynamic values (session IDs, CSRF tokens, auth tokens)

Extractors: Regular Expression, JSON Extractor, Boundary, CSS Selector

Assertions: Response, JSON, Duration, Size

Logic controllers: Transaction, If, Loop, While, Once Only

Timers and think time: Constant, Uniform Random, Gaussian Random

Variables vs properties; functions __P(), __Random(), __time(), __UUID()

JSR223 Groovy basics for small pre/post-processing tasks

Debugging: View Results Tree, Debug Sampler, jmeter.log

Lab: fully correlated, asserted, transaction-grouped script that replays clean with 5 users

Module 4: Execution and Results Analysis

Non-GUI mode: `jmeter -n -t test.jmx -l results.jtl -e -o report/`

Passing users, ramp-up and duration as properties with `-J`

Listeners: when to use them, and why to disable them under load

Key metrics: response time, latency, connect time, throughput, error rate

Percentiles (90th, 95th, 99th) and why averages mislead

HTML dashboard report walkthrough

Reading graphs to spot bottlenecks: response time vs active threads, throughput plateau, error spikes

Lab: 25-user run in non-GUI mode, generate and interpret the dashboard

Module 5: Workload Modeling and Advanced JMeter

Workload modeling: Little's Law, converting business volume into users and throughput

Thread groups: standard, Concurrency, Ultimate and Arrivals Thread Groups

Throughput control: Constant Throughput Timer, Throughput Shaping Timer

Server-side monitoring during tests: CPU, memory, DB, APM and log tools

Real-time monitoring with Backend Listener, InfluxDB and Grafana (demo)

JMeter tuning: heap size, lean listeners, CSV results only

Distributed testing concept (controller and workers) as lead-in to cloud tools

JDBC Request sampler (demo)

Lab: model a workload from given business numbers and build matching thread groups

Module 6: Load Testing with RedLine13

RedLine13 overview and AWS account linking

Uploading a JMeter plan, CSV and plugins

Choosing regions, instance size and number of servers

Passing properties to the test

Running, monitoring and downloading results

Cost awareness: AWS instance charges

Lab: run the Module 5 script on RedLine13 from 2 regions

Module 7: Load Testing with BlazeMeter

BlazeMeter overview and free-tier limits

BlazeMeter Chrome recorder as an alternative to the JMeter proxy

Uploading and configuring a JMeter test

Load configuration and geo-distribution

Reports: timeline, request stats, errors, comparison between runs

Lab: run the same script on BlazeMeter, compare results with the RedLine13 run

Module 8: Best Practices and Group Project

Script best practices: naming, modular Test Fragments, reusable config, version control

Refactoring a messy recorded script

Common scenarios: login storm, search-heavy, checkout, API-only

CI/CD integration demo: run JMeter from Jenkins or GitHub Actions with pass/fail thresholds

Writing the performance test report: summary, findings, bottlenecks, recommendations

Group project: requirements → workload model → script → execution → report; peer review

Module 9: Final Assessment and Wrap-Up

Individual practical test: script, run and analyze a given scenario

Report review and feedback

Q&A, next steps and further learning resources

Module 10: Prerequisites and Tools

HTTP basics: methods, status codes, headers, cookies, sessions

Browser DevTools Network tab; basic command line (Windows or Linux)

Java (JDK) 17 or later; Apache JMeter 5.6.x with the Plugins Manager

Chrome or Firefox for proxy recording; Docker (optional) for a local practice app

Practice sites: blazedemo.com, or JPetStore / OWASP Juice Shop in Docker

Cloud accounts before Module 6: RedLine13 linked to AWS, BlazeMeter free tier



Talk to a Consultant

+91 82091 66170

hello@testeracademy.com

Address

305 & 411, North Plaza

Near 4D Square Mall

Motera, Ahmedabad

Gujarat 380005