

9.4 - Appvance Reporting

Appvance Reporting — Knowledge Base articles.

- [AIQ Reporting Knowledge Base Overview](#)
- [All the reports can be viewed by user roles?](#)
- [Root Cause Analysis Reports](#)
- [Can the Script Repository and Results Database Reside in One Location?](#)
- [Difference Between Step Times and Iteration Summary Reports](#)
- [Failure Report](#)
- [Formulas for Total Hits](#)
- [How does iteration summary report calculate step and testcase average duration](#)
- [How 90th Percentile Is Calculated in Reports](#)
- [How is step times calculated for test designer scripts?](#)
- [Share AIQ Result Files for Team Review](#)
- [Understand TPS in AIQ Scalability Reports](#)
- [Download Iteration Summary Reports with AIQ APIs](#)
- [Fewer Requests Reported for a URL](#)
- [Pull Report Details by Execution ID](#)
- [Real Time Reports in Appvance IQ](#)
- [Resource Usage Graph Metric Definitions](#)
- [Sampling Rate in Appvance IQ.](#)
- [Standard Deviation](#)
- [To get the time duration of a particular step using JavaScript](#)
- [Legacy Tool.jar APCStepListener Reference](#)
- [What is Average Size in Iteration/Step times report?](#)
- [When there is blank row in the csv sheet, will it pick the blank row?](#)
- [Performance Report Optimization](#)

AIQ Reporting Knowledge Base Overview

AIQ Reporting Knowledge Base Overview

Use reporting knowledge base content to explain maintained AIQ result, report, and metric interpretation workflows.

Recommended guidance

- Keep metric definitions clear and tied to current AIQ reporting.
- Separate live monitoring from completed-run analysis.
- Avoid duplicating reporting guidance across multiple articles.
- Document environment, load model, and data assumptions when comparing reports.

Validation workflow

1. Identify the report or metric being explained.
2. Confirm the definition still matches current AIQ behavior.
3. Add interpretation guidance and caveats.
4. Link analysis steps to execution output and evidence.

Last updated: September 20, 2026

All the reports can be viewed by user roles?

Yes all the reports can be viewed by all the users.

There is no restrictions on viewing the reports as per the user.

Admin user can view all the executions of all the users.

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Root Cause Analysis Reports

Are Root Cause Analysis (RCA) reports available?

Appvance provides basic monitoring capabilities and can integrate with Application Performance Monitoring (APM) tools such as AppDynamics, New Relic, and Dynatrace.

These integrations can provide monitoring data from approximately **30 seconds before and after a failure**, helping identify the conditions surrounding the failure.

Appvance does not automatically generate a complete Root Cause Analysis report. The monitoring data can be used by developers and support teams, together with application logs and APM data, to investigate and determine the underlying root cause.

Last updated: August 27, 2026

If you have any feedback on this topic, please contact us at documentation@appvance.ai.

Can the Script Repository and Results Database Reside in One Location?

Can the script repository and results database reside in one location and be accessed remotely from another location, such as storing them in Indianapolis and accessing them from AWS Northern Virginia?

Yes. The script repository and results database can be hosted remotely and accessed from another location, provided that the required network connectivity and access permissions are available.

However, for performance and load testing, **low network latency between the Controller, load generators, and results database is critical**. For this reason, we recommend hosting the Controller, load generators, and results database in the same AWS region whenever possible.

For the results database, an **AWS RDS instance** can be used to store test results.

The script repository can be hosted in a **Git or SVN repository** and can be accessed remotely by the Appvance environment.

Last updated: August 17, 2026

If you have any feedback on this topic, please contact us at documentation@appvance.ai.

Difference Between Step Times and Iteration Summary Reports

The **Step Times Report** provides timing information for the individual steps executed within a test case. It can be used to analyze how long each step takes during test execution.

The **Iteration Summary Report** provides a summary of the results for each completed iteration. It can be used to review the overall execution results and performance of each iteration.

In summary:

- **Step Times Report:** Provides timing details for individual test steps.
- **Iteration Summary Report:** Provides a summary of the results for each iteration.

Last updated: August 17, 2026

If you have any feedback on this topic, please contact us at documentation@appvance.ai.

Failure Report

The Failure Report provides information about failures that occur during test execution.

If the same error appears in two entries with different average duration values, the entries represent different levels of timing information:

- **First entry:** Shows the total duration of the complete iteration in which the failure occurred.
- **Second entry:** Shows the duration of the specific step that failed.

For example, if an iteration takes **10 seconds** to complete and the failed step takes **4 seconds**, the Failure Report may show:

Entry	Average Duration	Description
Iteration	10 seconds	Total time to execute the complete iteration
Failed Step	4 seconds	Time taken to execute the step that failed

Therefore, the two entries do not represent two separate failures. They provide timing information at two different levels: **the complete iteration and the specific failed step**.

Last updated: August 17, 2026

If you have any feedback on this topic, please contact us at documentation@appvance.ai.

Formulas for Total Hits

This article explains how **Total Hits**, **Total Throughput**, **Average Hits**, and **Average Throughput** are calculated in the **Iteration Summary** report.

These metrics provide information about the number of requests generated during a test and the amount of data transferred.

1. Total Hits

Total Hits represents the total number of requests generated during the test.

Formula:

$$\text{Total Hits} = \text{Total Requests}$$

For example, if the test generates **125 requests**, the Iteration Summary report shows:

$$\text{Total Hits} = 125$$

“ **Note:** A single user action can generate multiple HTTP requests. Therefore, Total Hits does not necessarily represent the number of test steps or user actions.

2. Total Throughput

Total Throughput represents the total amount of data transferred by successful requests during the test.

Formula:

$$\text{Total Throughput} = \text{avgSize} \times \text{success}$$

Where:

- **avgSize** = Average size of the response/request data, in KB.
- **success** = Number of successful requests.

Example

Assume the following values:

Average Size = 2.17 KB
Successful Requests = 25

The Total Throughput is calculated as:

$2.17 \text{ KB} \times 25 = 54.25 \text{ KB}$

Therefore:

Total Throughput $\approx 54.3 \text{ KB}$

The result represents the total volume of data calculated from the average size and the number of successful requests.

3. Average Hits

Average Hits represents the average number of requests generated per second during the test.

Formula:

$\text{Average Hits} = \text{Total Hits} / \text{Test Duration in seconds}$

Example

Assume:

Total Hits = 125
Test Duration = 25 seconds

The Average Hits is:

$$125 / 25 = 5 \text{ hits/s}$$

Therefore:

$$\text{Average Hits} = 5 \text{ hits/s}$$

This means that the test generated an average of **5 requests per second** during the test duration.

“ **Note:** Average Hits is an average value. It does not mean that exactly the same number of requests was generated every second.

4. Average Throughput

Average Throughput represents the average amount of data transferred per second during the test.

Formula:

$$\text{Average Throughput} = \text{Total Throughput} / \text{Test Duration in seconds}$$

Example

Assume:

$$\text{Total Throughput} = 54.3 \text{ KB}$$

$$\text{Test Duration} = 25 \text{ seconds}$$

The Average Throughput is:

$$54.3 \text{ KB} / 25 \text{ seconds} = 2.172 \text{ KB/s}$$

Therefore:

$$\text{Average Throughput} \approx 2.17 \text{ KB/s}$$

This means that the test transferred an average of approximately **2.17 KB of data per second**.

Summary

The following table summarizes the metrics and their formulas:

Metric	Description	Formula	Unit
Total Hits	Total number of requests generated	Total Requests	Hits/requests
Total Throughput	Total data volume calculated from successful requests	avgSize × success	KB
Average Hits	Average number of requests generated per second	Total Hits / Duration	Hits/s
Average Throughput	Average amount of data transferred per second	Total Throughput / Duration	KB/s

Complete Example

For a test with the following results:

Total Requests = 125

Successful Requests = 25

Average Size = 2.17 KB

Test Duration = 25 seconds

The metrics are calculated as follows:

Total Hits

= 125 requests

Total Throughput

= 2.17 KB × 25

= 54.3 KB

Average Hits

= 125 / 25

= 5 hits/s

Average Throughput

= 54.3 / 25

= 2.17 KB/s

Therefore, the Iteration Summary metrics would be:

Total Hits	= 125
Total Throughput	= 54.3 KB
Average Hits	= 5 hits/s
Average Throughput	= 2.17 KB/s

Last updated: September 1, 2026

If you have any feedback on this topic, please contact us at documentation@appvance.ai.

How does iteration summary report calculate step and testcase average duration

This article illustrates why the summary of individual steps average duration is not equal to the reported test-case average duration.

Appvance IQ takes only success transactions into account for average duration or sizes calculation.

If there is any failures that is encountered during any iteration, that will not be considered in the calculation.

Please consider below example for better understanding.

Scenario has 1 test case which has about 5 steps.

Number of Execution = 4

Test-case 1 4 Iterations	Execution 1 (in seconds)	Execution 2 (in seconds)	Execution 3 (in seconds)	Execution 4 (in seconds)		Step Average Duration - as seen in APC Reports
STEP 1	20	21	Fail	12		17.67
STEP 2	5	7	25	13		12.5
STEP 3	Fail	8	Fail	7		7.5
STEP 4	Fail	12	11	6		9.67
STEP 5	20	23	21	4		17
Transaction Duration	FAIL	71	FAIL	42		Total from above = 64.34

From the above table, with the average duration that is displayed for all the steps, the total step average duration comes up to 64.34 which is what the user is expected to see.

How Appvance IQ works is it considers only the successful iteration, what AIQ will report is $71 + 42 / 2 = 56.5$ seconds as the average step duration (as mentioned **AIQ considers only the successful transactions**)

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

How 90th Percentile Is Calculated in Reports

Recommended guidance

The 90th percentile is the value below which 90% of measured samples fall. In AIQ reporting, use it to understand typical high-end response time without letting a few extreme outliers dominate the interpretation.

How to interpret it

- If the 90th percentile response time is 2 seconds, 90% of samples completed in 2 seconds or less.
- The remaining 10% were slower than that value.
- Percentiles should be reviewed with average, median, error rate, throughput, and sample count.
- Small sample sizes can make percentile values unstable.

Reporting checklist

1. Confirm the metric and transaction represented by the percentile.
2. Check whether failed or timed-out samples are included or excluded.
3. Compare percentile values only between runs with similar data, load, and environment.
4. Use percentiles with charts and logs to identify sustained slowness versus isolated spikes.

Last updated: October 6, 2026

How is step times calculated for test designer scripts?

When users run a Designer script, reports get generated which have two important columns:

Step Times and Request name

These times are ALL generated through the browser you have chosen, and NOT at the API level and APC will start moving on to the next step as soon as it is ready to be used and those reporting times will be reported.

Now here is how we start and stop the clock on both of these:

STEP TIME:

For the step time, the clock starts as soon as we hit a transaction (like a submit button) until the full page has come in and the DOM has settled.

This includes the submit request placed at the UX in the browser, and then all elements coming back, and then all of them being processed by the browser and displayed and then the page is complete. Just as what a user sees.

REQUESTS:

Now at a lower level are the requests. These are requests made in the browser and through the client side code, then to the server, and then a response coming back. We start the timer at the request time and then stop it as soon as the final bit for that element comes INTO the browser. The reason is that many of the elements (like JS elements) will never be “displayed” so this timing includes sending through client side code but stops when the final bit is delivered TO the client side code.

So the transaction times include all browser code in both directions. Including displaying all elements. Page complete. That is the end to end response time including rendering, latency and everything a user experiences.

The "Request name" times (one level down from step/transaction times) include browser code outgoing and then till the last bit arrives to fulfill that element. Whether or not it is displayed as most elements (especially code segments) do not display. That is the only way to make this timing make sense and all be the same basis.

So an API level test should generally appear faster than a UX level test, all else being the same.

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Share AIQ Result Files for Team Review

Last reviewed: September 28, 2026

Use this article when AIQ result files need to be shared with another team for investigation, approval, or customer review.

What changed in this article

- Corrected the title grammar and made the scope about controlled result-file sharing.
- Added checks for report type, execution ID, recipient team, review purpose, and sensitive data before sending.
- Added a reminder to share links or approved exports rather than ad-hoc files when governance requires traceability.

Validation workflow

1. Identify the execution/result file and why the receiving team needs it.
2. Review the file for sensitive data, tokens, internal URLs, or customer-only information.
3. Share through the approved channel and include execution context, scenario name, and date.

Review note: Added data-handling and traceability guidance so result sharing is review-ready and not just file forwarding.

Last updated: September 28, 2026

If you have feedback on this topic, contact documentation@appvance.ai.

Understand TPS in AIQ Scalability Reports

Understand TPS in AIQ Scalability Reports

TPS values in scalability reports should be interpreted with the test design, valid sample count, throughput, errors, and steady-state period in mind.

Recommended guidance

- Compare TPS only between runs with consistent scenarios, data, environment, and load model.
- Exclude ramp-up, ramp-down, and invalid intervals when analyzing steady-state behavior.
- Review errors and response times with TPS to understand scalability.
- Document the execution ID and report interval used for conclusions.

Validation workflow

1. Open the completed scalability report.
2. Identify the steady-state period.
3. Review TPS with throughput, response time, and error rate.
4. Record the interpretation and assumptions in the report notes.

Last updated: September 20, 2026

Download Iteration Summary Reports with AIQ APIs

Last reviewed: September 28, 2026

Use this article when report data or iteration summaries need to be retrieved programmatically from AIQ.

What changed in this article

- Replaced the Java-project-specific title with API-focused reporting guidance.
- Marked the sample Java project as implementation example content, not the only supported approach.
- Added checks for authentication, execution ID, API endpoint, pagination/response size, and safe storage of downloaded report data.

Validation workflow

1. Confirm the execution ID and report type required by the consumer.
2. Call the approved AIQ API with authorized credentials and capture the response.
3. Validate the downloaded data against the AIQ UI before automating downstream processing.

Review note: Modernized the page from “Java project download” to API-based report retrieval while preserving the iteration-summary use case.

Last updated: September 28, 2026

If you have feedback on this topic, contact documentation@appvance.ai.

Fewer Requests Reported for a URL

When reviewing the **Step Times Report** generated from a Designer script, you may notice that the number of requests reported for a URL is lower than the number of requests shown in browser developer tools such as Firebug.

For example, when accessing `www.appvance.com`, Firebug may show **84 requests**, while the Appvance IQ report may show only **11 requests**.

This difference is expected and is caused by how Appvance IQ determines when a step has completed.

Why does this happen?

Appvance IQ does **not wait for every page resource to finish loading** before completing a step.

When a step starts, Appvance IQ looks for the element required by that step. As soon as the element becomes available, Appvance IQ performs the requested action and proceeds to the next step.

At the same time, the browser may still be loading additional resources in the background, such as:

- CSS files
- JavaScript files
- Images
- AJAX requests
- Other page resources

Because these background requests may continue after Appvance IQ has already completed the step, they may not be associated with that step in the **Step Times Report**.

Why can the request count change between executions?

The requests associated with a step can also vary from one execution to another.

For example, during one execution, a step may take long enough for several background requests to occur while the step is active. Those requests may therefore appear in the report.

During another execution, the required element may become available faster. The step completes before those background requests occur, so fewer requests are associated with the step.

Therefore, the number of requests reported for a step should **not be interpreted as the total number of network requests generated by the page**.

This behavior can also affect metrics that depend on the request count or total request size, such as throughput calculations.

Protocol-based technologies

This behavior is specific to how browser-based steps are measured. Protocol-based technologies that use a **HAR (HTTP Archive)** capture network activity at the HTTP level and therefore provide a different view of the requests generated during the transaction.

Last updated: August 17, 2026

If you have any feedback on this topic, please contact us at documentation@appvance.ai.

Pull Report Details by Execution ID

Last reviewed: September 28, 2026

Use this article when a team needs to retrieve report details for a specific AIQ execution ID.

What changed in this article

- Simplified the title and made the retrieval key explicit: execution ID.
- Added validation checks for execution ownership, report availability, query/API permissions, and matching UI results.
- Added caution against exposing raw query output that contains customer data, internal URLs, or environment names.

Validation workflow

1. Locate the exact execution ID from AIQ results or reporting.
2. Run the approved query or API call using an account with reporting access.
3. Compare returned report metadata with the AIQ UI before sharing or storing it.

Review note: Added governance checks around query output so report details can be used safely outside the UI.

Last updated: September 28, 2026

If you have feedback on this topic, contact documentation@appvance.ai.

Real Time Reports in Appvance IQ

Real Time Analysis with Appvance IQ.

Below are the list of real time reports in IQ.

Live Reports: (Sampling Rate - Reports takes about 10 seconds)

- Active Users
- Total test case execution.
- Hit Count
- Test case average duration.
- Scalability Report.
- Average Duration.
- Total Successes.
- Total Failures.
- Hits/Second.
- Resource Usage (APC 2.1)
- CPU
- Memory
- Net
- Thread Count
- Heap Usage
- Garbage Collector
- User-time percentage.

Step Times/Iteration Summary Report (Sampling Rate - Saving summary to database is 1 minute)

- Total Throughput
- Average Hits
- Average Throughput/s
- Test-cases passed
- Test-cases failed
- Step passed
- Step failed
- User Percentage
- Step name
- Request name
- Duration detail
- Response Code

- Success Count
- Failure Count
- Total Step
- Total Request
- Average Duration
- Standard Deviation Duration
- Minimum Duration
- Duration 85 Percentile
- Duration 90 Percentile
- Duration 95 Percentile
- Maximum Duration
- Average Size
- Standard Deviation Size
- Minimum Size
- Size 85 Percentile
- Size 85 Percentile
- Size 85 Percentile
- Maximum Size

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Resource Usage Graph

Metric Definitions

The **Resource Usage** graph provides information about the system resources consumed during test execution. These metrics help identify potential resource constraints and correlate resource usage with test performance.

The metrics are divided into two categories:

- **JVM Metrics** — Metrics related to the Java Virtual Machine (JVM) running the Appvance process.
- **OS Metrics** — Metrics reported by the operating system for the machine where the test is running.

JVM Metrics

1. Heap Usage

Heap Usage shows the amount of Java heap memory currently being used by the JVM.

The Java heap is a runtime memory area used by the JVM to store objects and arrays created during application execution. For example, test execution may create objects to process requests, responses, variables, and other runtime data.

Appvance initializes the JVM heap with a default maximum heap size of **1500 MB**, according to the configuration documented for this version.

Heap usage can increase as the test creates more objects. The JVM's Garbage Collector can subsequently reclaim memory occupied by objects that are no longer being used.

For example:

Heap Usage

400 MB → 700 MB → 1100 MB → Garbage Collection → 600 MB

An increase in heap usage does not necessarily indicate a problem. When analyzing this metric, look at its behavior over time.

If heap usage continuously increases and approaches the maximum heap size without being reclaimed, this may indicate memory pressure and should be investigated.

“ **Note:** The default heap size can vary depending on the Appvance version and configuration. Verify the JVM configuration for the specific environment being tested rather than assuming that 1500 MB applies to every installation.

2. Garbage Collector

The **Garbage Collector (GC)** is a JVM process responsible for identifying and reclaiming memory occupied by objects that are no longer required.

The Resource Usage graph reports the CPU usage associated with the Garbage Collector.

For example:

```
graph TD; A[Application creates objects] --> B[Heap usage increases]; B --> C[Garbage Collector runs]; C --> D[Unused objects are reclaimed]; D --> E[Available heap memory increases];
```

Higher Garbage Collector CPU usage can occur when the JVM needs to perform garbage collection more frequently or for longer periods.

A sustained increase in Garbage Collector activity, particularly when accompanied by increasing heap usage, high CPU usage, or increasing response times, may indicate memory pressure.

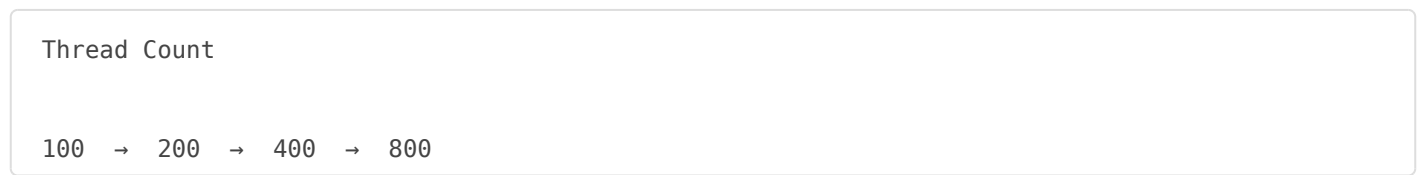
3. Thread Count

Thread Count shows the current number of active threads in the JVM.

A thread is an execution unit within a process. The JVM can use multiple threads to perform work concurrently.

Thread count can increase as the amount of concurrent work increases. However, a continuously increasing or unusually high thread count may require investigation, especially when CPU or memory usage is also high.

For example:



Thread count should be evaluated together with the workload and other resource metrics. A high thread count alone does not necessarily indicate a problem.

OS Metrics

1. CPU Usage

CPU Usage shows CPU utilization as reported by the operating system.

CPU usage indicates how much processing capacity is being consumed by the system during test execution.

For example:



Sustained high CPU utilization can indicate that the machine is approaching its processing capacity.

If CPU usage remains close to maximum while response times increase or throughput stops increasing, CPU may be a resource bottleneck.

2. MEM

MEM shows the percentage of system memory being used relative to the total memory available, as reported by the operating system.

This metric represents system-level memory usage and should not be confused with **Heap Usage**.

For example, a machine with 16 GB of RAM using approximately 8 GB would report approximately 50% memory utilization.

The distinction is important:

Heap Usage

↓

Memory used by the JVM heap

MEM

↓

System-level memory usage reported by the OS

System memory usage can include memory used by the JVM, other processes, the operating system, and other memory consumers.

Sustained high memory utilization may indicate memory pressure and should be investigated together with Heap Usage and Garbage Collector activity.

3. NET

NET shows network traffic as reported by the operating system.

Network traffic represents data being transferred by the machine during test execution.

For example, test activity can generate network traffic when:

- Requests are sent to the application.
- Responses are received from the application.
- Large files or responses are transferred.
- Multiple virtual users generate concurrent network activity.

An increase in network traffic is not necessarily a problem. The metric should be evaluated against the expected workload and the available network capacity.

Sustained high network utilization combined with increasing response times or reduced throughput may indicate a potential network bottleneck.

Interpreting Resource Usage Metrics

Resource Usage metrics should be analyzed together rather than individually.

For example:

CPU ↑
Heap ↑
Garbage Collector ↑
Response Time ↑

This combination may indicate that the JVM and system are experiencing increased resource pressure.

Another example:

CPU ≈ 100%
Heap stable
MEM normal
NET normal
Response Time ↑

In this case, CPU saturation may be a potential bottleneck.

Similarly:

Heap ↑
Garbage Collector ↑
MEM ↑
Response Time ↑

may indicate increasing memory pressure.

The Resource Usage graph is therefore most useful when its metrics are correlated with other test results, such as:

- Response time
- Throughput
- Error rate
- Virtual user load
- Request volume

Summary

Category	Metric	Description
----------	--------	-------------

JVM	Heap Usage	Amount of Java heap memory currently being used
JVM	Garbage Collector	CPU usage associated with JVM garbage collection
JVM	Thread Count	Current number of active JVM threads
OS	CPU Usage	CPU utilization reported by the operating system
OS	MEM	Percentage of system memory being used
OS	NET	Network traffic reported by the operating system

When investigating performance issues, use these metrics to determine whether the test environment may be approaching a **CPU, memory, JVM, thread, or network resource limit**.

Last updated: August 28, 2026

If you have any feedback on this topic, please contact us at documentation@appvance.ai.

Sampling Rate in Appvance IQ.

What is sampling rate during Appvance IQ executions. Like, real time graphs and values getting refreshed at what interval?

Real-time reports takes about 10 seconds.
Saving summary to the database is each minute.

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Standard Deviation

How to calculate standard Deviation duration, minimum and maximum duration, Duration 85/90/95 Percentile, Standard deviation size, Minimum and Maximum size, size 85 /90/95 percentile in the iteration summary report?

All the above mentioned calculation's in the iteration summary are done at back-end i.e. it is all based on algorithms which calculates the values for each of it and no Mathematical calculation to be done in the front end to obtain the results in the iteration summary report.

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

To get the time duration of a particular step using JavaScript

Javascript to find the time duration of a particular step:

```
var $startTime = new Date().getTime(); ---- Before the step starts  
--- Actual Steps  
var $endTime = new Date().getTime(); ---- After the step  
var $diff = $startTime - $endTime — Log the difference  
_log("" + $diff); — Display it
```

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Legacy Tool.jar

APCStepListener Reference

Legacy Tool.jar

APCStepListener Reference

This page is legacy reference material for older Tool.jar and APCStepListener usage. Retain it only for maintained assets that still depend on this integration.

Recommended guidance

- Do not use this as default guidance for new reporting or Scenario Editor work.
- Document the exact legacy asset that still requires it.
- Avoid publishing internal implementation details that are not needed for maintenance.
- Plan migration to current reporting and execution evidence when possible.

Validation workflow

1. Confirm the dependency is still used.
2. Validate the legacy asset in a non-production environment.
3. Capture execution output and reporting evidence.
4. Document replacement or retirement status.

Last updated: September 20, 2026

What is Average Size in Iteration/Step times report?

Average size is the size of the object, html page, js page, image etc.

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

When there is blank row in the csv sheet, will it pick the blank row?

Will a blank row in a CSV file be picked up during execution?

Yes. **Blank rows in a CSV file are treated as entries and are picked up during iteration execution.**

If a CSV file contains a blank row, Appvance IQ will include that row as an iteration. Any corresponding fields with blank values will be processed as empty values.

For example, if the CSV contains:

```
username,password
user1,password1
,
user3,password3
```

The blank row will still be processed as an iteration.

If blank rows are not intended to be used during execution, remove them from the CSV file before running the test.

Last updated: August 28, 2026

If you have any feedback on this topic, please contact us at documentation@appvance.ai.

Performance Report Optimization

If you have a situation where you are having trouble downloading the execution summary file for a large number of users because of the amount of data. A example of this situation is where you can download the data for 200 users, but you have problems downloading the data for 500 or 1000 users.

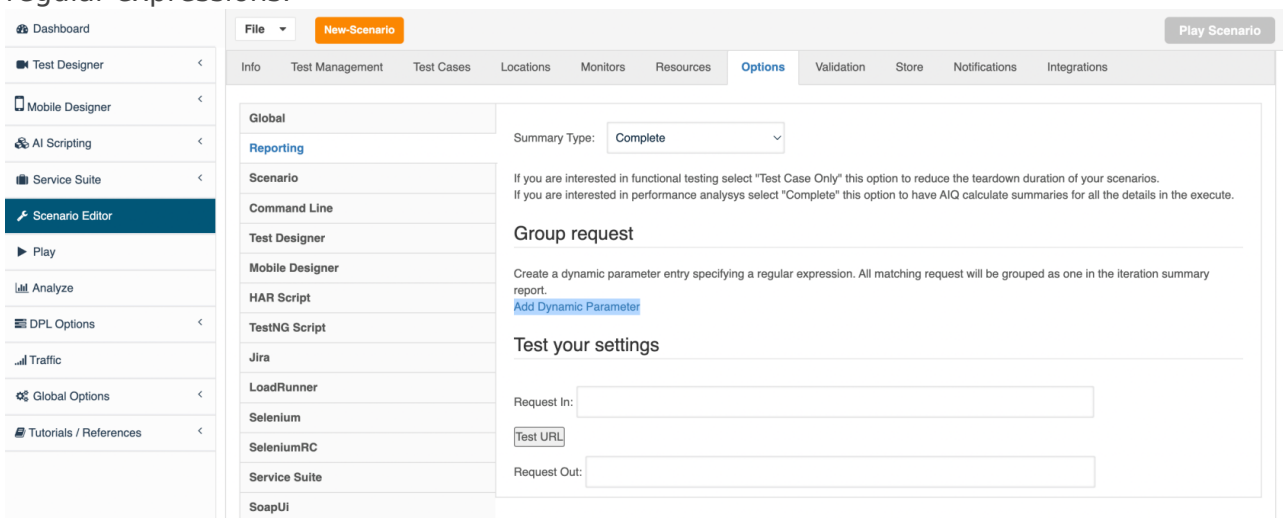
Here are some guidelines for suppressing some of the data by replacing dynamic values with regular expressions.

For more information on handling dynamic requests with regular expressions see [Best practices to improve AIQ performance.](#)

Report Optimization

The following optimization will greatly reduce the tear down duration of the scenario and also improve the time required to download the iteration summary report because it will reduce the number of rows.

1. In the **Reporting** section on the **Options** tab of the Scenario Editor, there is an option called “Add Dynamic Parameter” that you can use to replace dynamic values using regular expressions.



For example:

```
db=[\w\-\!\!]+
```

```
policyId=[0-9]+
```

```
_windowId%3DW[0-9]+
```

```
customerId=[0-9]+
```

```
customerNumber=[\w]+
```

```
ln=org[\w\.\!]+
```

```
_=[0-9]+
```

2. Open the IDE, and in the script(s) locate the steps listed below and add a description to those steps. The descriptions that you add will be listed in the iteration summary report instead of the current step name. Until you add the description the report is listing some dynamic values which results in the steps being listed multiple times. Generally you should not see duplicate step numbers in the iteration summary report.

```
1.13)
```

```
SetValue(textbox({'locators':['crmForm:generalInfo_firstName'],'DS3_Any':true}),'Jack  
kbadmn')
```

```
1.43) Log([Jackaslnj Jillilkkti, , New Person, New Corporation] // this comes after a  
step which name
```

```
1.42)
```

```
Store(getSelectTexts(select('policyDataGatherForm:sedit_PreconfigInsured_partySelecti  
on')))
```

```
1.44)
```

```
SetSelected(select('policyDataGatherForm:sedit_PreconfigInsured_partySelection'),'Jack  
avzpc Jillzrvwg')
```

```
1.48) Log([Jackaslnj Jillilkkti, , Create New Driver])
```

```
1.49)
```

```
SetSelected(select('policyDataGatherForm:sedit_PreconfigAutoDriver_partySelection'),'J  
ackaslnj Jillilkkti')
```

Last updated: April 19, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai