

## Dynatrace Setup for Application Performance Monitoring

Naveen Muppa\*

Naveen Muppa, 10494 Red Stone Dr Collierville, Tennessee, USA

**Citation:** Muppa N. Dynatrace Setup for Application Performance Monitoring. *J Artif Intell Mach Learn & Data Sci* 2023, 1(2), 371-375. DOI: doi.org/10.51219/JAIMLD/Naveen-muppa/87

**Received:** 03 June, 2023; **Accepted:** 28 June, 2023; **Published:** 30 June, 2023

\***Corresponding author:** Naveen Muppa, 10494 Red Stone Dr Collierville, Tennessee, USA

**Copyright:** © 2023 Muppa N. Enhancing Supplier Relationships: Critical Factors in Procurement Supplier Selection.. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

### ABSTRACT

Dynatrace is an application performance management (APM) software company that specializes in providing insights into the performance and health of software applications. The company offers a comprehensive platform that enables organizations to monitor, analyze, and optimize their digital ecosystems in real-time.

#### Keywords:

**Full-Stack Monitoring:** Dynatrace provides end-to-end visibility across the entire technology stack, including infrastructure, applications, and user experience. It offers monitoring capabilities for cloud-native environments, microservices, containers, and traditional monolithic applications.

**AI-Powered Analytics:** Dynatrace leverages artificial intelligence (AI) and machine learning (ML) algorithms to automatically detect anomalies, identify performance bottlenecks, and provide actionable insights. This proactive approach helps organizations resolve issues before they impact end-users.

**Digital Experience Monitoring (DEM):** Dynatrace enables businesses to understand and optimize the digital experience of their customers by monitoring user interactions with web and mobile applications. DEM includes real-user monitoring (RUM), synthetic monitoring, and session replay capabilities.

**Application Security Monitoring:** Dynatrace integrates security monitoring features into its platform, allowing organizations to detect and respond to security threats in real-time. It provides insights into application vulnerabilities, malicious activity, and compliance violations.

**Auto-Remediation:** Dynatrace automates remediation actions to address performance issues and optimize resource utilization. By leveraging AI-driven insights, it can automatically scale resources, restart services, or adjust configurations to maintain optimal application performance.

**Cloud-Native Support:** With native support for cloud platforms such as AWS, Microsoft Azure, and Google Cloud Platform (GCP), Dynatrace helps organizations monitor and optimize their cloud-native applications and infrastructure. It provides visibility into dynamic, containerized environments and supports technologies like Kubernetes and Docker.

### Introduction

Dynatrace monitors the performance and health of applications across various environments, including on-premises data centers, cloud infrastructure, and hybrid environments.

It helps identify performance bottlenecks, optimize resource utilization, and ensure a smooth user experience. When applications encounter issues or slowdowns, Dynatrace provides deep insights into the root causes of these problems. It correlates data from different layers of the application stack to pinpoint

the exact source of performance issues, allowing IT teams to quickly resolve them.

### 2. Operational Dashboard

Upon logging in, you are presented with the “Operational Dashboard” page, as seen below. One thing to note, all times are based on the time zone settings from your computer, so if your computer’s clock is set to EST (Eastern Standard Time), the times represented will also be set to EST.

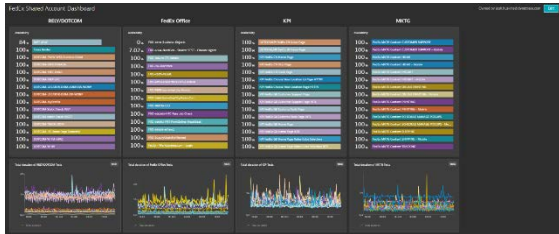


Figure 1: Operational dashboard.

The four Dynatrace grid examples above were chosen because they had a lot of scripts in those Folders and do not necessarily have to be shown if you wish to create your own dashboard. Further details can be gleaned by clicking on any of the panels within the dashboard, for instance, clicking on the window on the top row brings up the following:



Figure 2: Instance view.

This is an expanded view of the clicked-on panel, with further details that can be viewed/drilled down in accordingly. On the top right corner of the screen, where it says “Last 24 hours” in the pic above, you can change the data timeframe that is being displayed (both here and the screen previous) if you need more or less data pulled.

The durations panel brings up the following:

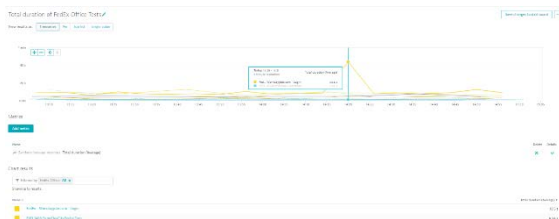
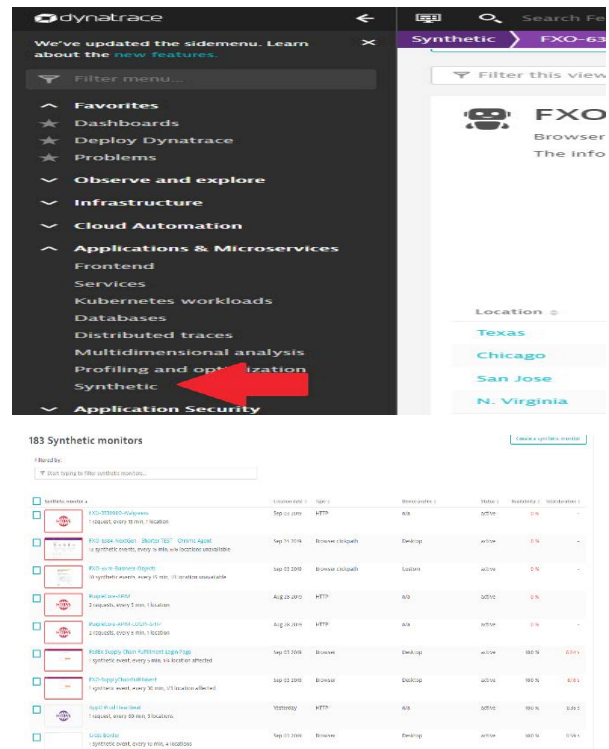


Figure 3: Panel view.

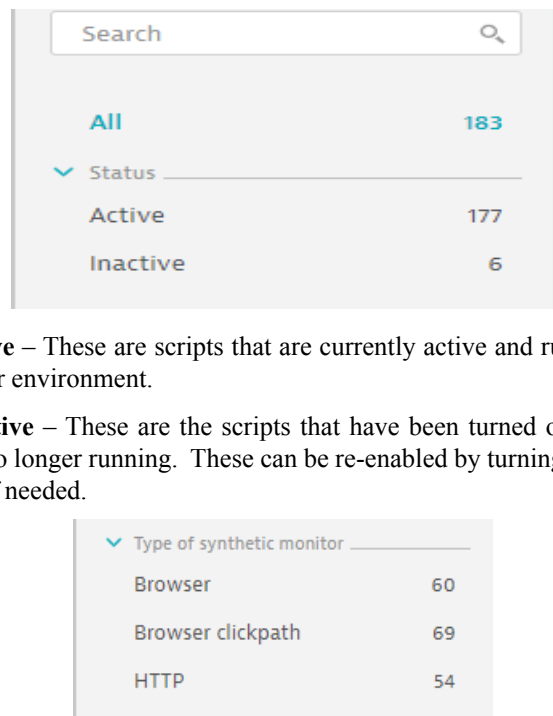
### 3. Synthetic Measures

To go to the Synthetic Measurements section, click the “Applications & Microservices” link on the left side and expand it out, following up with a click on the “Synthetic” link towards the bottom.

This is where all the scripts, both Active and Inactive, are located. The Synthetic monitors page lists all the scripts created for the account. Scripts that are reporting an active issue/problem are represented towards the top, with a red border surrounding the picture next to the script name:



Each section on the left can be toggled by (clicked on) to look for a specific type of script. If you know the script name, you can type in the “Search” field to find it.



**Active** – These are scripts that are currently active and running in our environment.

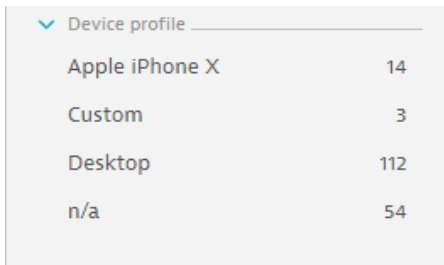
**Inactive** – These are the scripts that have been turned off and are no longer running. These can be re-enabled by turning them on, if needed.

**Browser** – Browser monitors are always one step scripts. These scripts go to one specific page and then stop. They may have a validation action to make sure that the page loaded as intended, but do not go beyond that.

**Browser Click Path** – These are scripts that like above go to a URL, but then go through a flow (these scripts always have more than one step). These are the scripts where a specific path is followed (i.e. go to URL, enter login info, click login, get rating information, etc.).

**HTTP** – These run in a headless browser. They are typically API/SOAP requests, (like POST, GET, etc.) that get a response

back and validate on that response. Because these are browserless transactions, there will not be any screenshots captured for troubleshooting purposes.



Device profile	Count
Apple iPhone X	14
Custom	3
Desktop	112
n/a	54

**Apple iPhone X** – These are the mobile web scripts. (For clarification, these are scripts running on a \*software simulated\* iPhone X, NOT a mobile-native application).

**Custom** – This could be that it is running at a different resolution than default. 1366x768 resolution is typically the default for this group (this can be modified to reflect different device profiles).

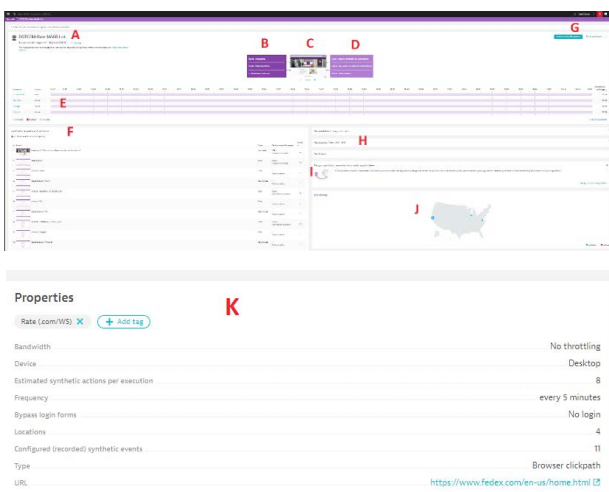
**Desktop** – These are running at the default 1920x1080 resolution.

**n/a** – These are the scripts currently aligned with the HTTP monitors mentioned above.

#### 4. Tags

The Tags section represents the “Folders” where each script lives. Certain users have views for certain folders, and the dashboard previously referenced can also be tailored to reflect scripts for tags. It is best practice to ensure when saving a new script, you tag it for your respective folder to ensure visibility for all team members to be able to see it. Tags can also be used for grouping alerts.

Once you have found a script you would like to investigate, click it, and the window changes to give you a test detail analysis for the selected script. See the legend below for an explanation of how this page is laid out.



A) This is the synthetic test script name, as well as the tags that are associated with the test. You can have multiple tags and can create or remove tags from here as well. Additionally, the type of synthetic monitor is identified here.

B) This is where some of the high-level information is located (Percent Availability / Total Downtime / Number of locations the script is running from). The data for Total Downtime is the total amount of time, across all monitor locations, for the given time period, added up.

C) Screenshots of the measurement taken from recent runs. This step can be changed by clicking the arrow located underneath the picture. Changing the step can also be done by clicking the picture, and then clicking the arrow in the top right of the page that loads, so you can see a visual step by step guide of what the script is doing/entering as it runs. Screenshots are available for both successful and failed runs.

D) Performance Metrics: Average of all the load actions and XHR actions displayed for the specified time is given here. Load actions are an average of all the load actions that take place in the running of the script (these are the actions that load a full page, ie CSS files, etc.). XHR actions are more along the lines of the backend calls that are occurring during the loading of the page (these can be occurring during the page load or after the page load). A lot of XHR actions are along the line of metrics or AJAX calls. Total duration is the total time it takes for the measurement to complete.

E) Availability is shown in the light purple color, while downtime is indicated by the color red. The location(s) that a script is running from are also displayed here, along with the Cloud Provider. In the bottom right of this section is a button labeled “Analyze Availability,” which will be covered below.

F) Synthetic events and actions - Actions taken by the script during the run – This includes going to a URLs, entering data, clicking buttons, etc. In the case of API/SOAP calls, this is where the data is displayed for the run of that script (POST, etc.). These are basically the same screenshots that are referenced in the section labeled “C” above. Each of the actions can be further broken out by clicking the arrow in the bottom corner of the action, to give a more detailed view of that step for the duration selected.

G) The Time Range can be changed at the top right corner – The level of detail for the data displayed on this page will be based on the selected timeframe. It can be changed by simply clicking on the time shown, and selecting whatever timeframe (Minutes, Hours, Days or Weeks) is relevant to the information desired.

H) The Problems Section labeled here is where the Alerts are displayed that are generated during the problem(s).

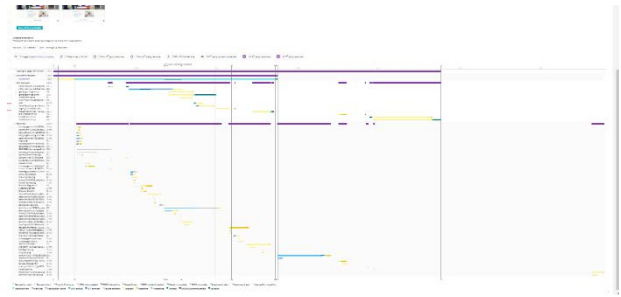
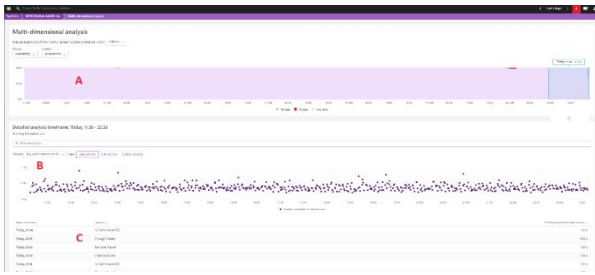
I) Events are like the Problems Section mentioned above but provides a more detailed analysis from the script side of what happened.

J) Worldmap - This is the geographical locations of where the scripts are running from and where the Dynatrace agents are located. It will display in blue if it is currently running successfully, while it will show as red if there are problems being reported.

K) Test Script Properties: Located below Section “F” referenced above, this is a summary view the script and details pertaining to it. This includes: the type of script, the time between runs, the number of locations it is running from, and the starting URL being used, not to mention other data as seen in the picture.

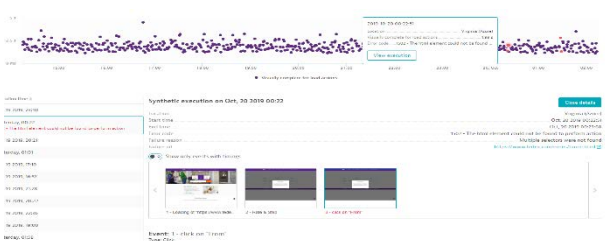
#### 5. Analyze Availability

Clicking the “Analyze Availability” link referenced in Section “E” above, brings us to a multi-dimensional analysis of the script for the given time period specified. For this example, I changed the timeframe to the last 7 days, to capture failures.



A) This is an expanded view of the section we came from, with the same purplish color indicating successes and red color indicating failures. In this view, it is representing ALL of the agents instead of individually listed as before. Note that the areas can be moused over to give percentage of availability, both the success areas as well as the failure areas. If there is a red bar ABOVE the areas marked as failures, that means that an Alert was generated during the failure(s).

B) This is a representation for the script runs (like Scatterplot Summary in the Classic environment) and can be changed by sliding the bar located on the top right side of the shaded area. (In the example, it is displaying the last 12 hours). Sliding the bar to change the time selected also changes the number of data points being shown, depending on the amount of data selecting. This is very useful for trying to troubleshoot problems that recently occurred or going back to determine root cause, after the fact. This also helpful to view what the end user may be seeing in real time, if that time frame is selected. Purple dots represent a successful run, while red dots indicate a failed run. Clicking on any of the data points represented here will give a detailed analysis of that run, and that run only (rather than an average), and will generate more details below the “B” section, expanding out it’s election from the “C” selection automatically, as seen below.



This detail given us the location, the time, the type of error, and the URL that the error occurred from. If you click the pictures shown, it will show you the expected response, taken from recent successful runs, versus the actual page that was loaded.

C) These are the actual individual test runs for each node. These can be expanded out individually or do so automatically if clicking on a data point from Section “B” in the diagram above. The times displayed, like all times within the Portal, are based on the computer viewing the analysis from (meaning, if you are viewing it from your computer located in the CST time zone, the time show will be in CST time accordingly). Below this section is a section dedicated to the waterfall chart for the run, as well as all the data collected accordingly as seen below. The data is broken down by type, and can be collapsed and expanded, depending on the user’s desired view.

Calls represented in black are calls that were successful, while calls in red indicate calls that failed to resolve. The grey circles at the top of the waterfall chart indicate browser events that occurred during the run of the script.

## 6. How to Decide What to Index

In Dynatrace, “waterfalls” refer to waterfall charts that provide a detailed visualization of the loading process of a web page or web application. These charts display the sequence of HTTP requests and responses that occur when a user accesses a web page, showing the timing and dependencies of each resource (such as HTML, CSS, JavaScript, images, etc.) as they are fetched from the server and rendered in the browser.

Here’s how you can interpret waterfalls in Dynatrace:

**Resource Timing:** Each row in the waterfall chart represents an individual resource (e.g., HTML file, CSS stylesheet, JavaScript file, image). The chart displays the start time, duration, and end time of each resource’s loading process. This information helps you understand how long it takes for each resource to be fetched and processed by the browser.

**Dependencies:** Waterfalls show the dependencies between resources, indicating which resources are required to load others. For example, JavaScript files may depend on CSS files, and CSS files may depend on HTML files. Understanding these dependencies can help identify potential bottlenecks and optimize resource loading order.

**HTTP Status Codes:** Waterfalls display the HTTP status codes returned by the server for each resource request. Common status codes include 200 (OK), 404 (Not Found), 500 (Internal Server Error), etc. By examining status codes, you can identify failed requests, server errors, and other issues affecting resource loading.

**Initiator:** Waterfalls indicate the initiator of each resource request, which can be the browser (e.g., when fetching HTML, CSS, or JavaScript files), an external domain (e.g., when loading third-party scripts or resources), or a redirect from another resource. Understanding initiators helps identify the sources of resource requests and potential performance optimizations.

**Timeline View:** In addition to the waterfall chart, Dynatrace provides a timeline view that aggregates resource loading times and categorizes them into different phases (e.g., DNS, TCP, SSL, Request, Response, etc.). This view offers a high-level overview of the loading process and helps identify which phases contribute most to overall page load time.

Interpreting waterfalls in Dynatrace allows you to identify performance bottlenecks, optimize resource loading, and improve the overall user experience of your web applications. By analyzing the timing, dependencies, and characteristics of resource loading, you can pinpoint areas for optimization and ensure fast and reliable page loading times.

## 7. Conclusion

Dynatrace provides an Error Analysis page that offers insights into errors and exceptions occurring within your applications.



This page aggregates error data from various sources, such as application logs, monitored transactions, and user sessions, to help you understand the frequency, impact, and root causes of errors. Here's an overview of the key features and functionalities of the Error Analysis page in Dynatrace:

**Error Overview:** The Error Analysis page typically starts with an overview section that summarizes key error metrics, such as the total number of errors, error rate, and error distribution over time. This section provides a high-level understanding of the error landscape within your applications.

**Error Timeline:** Dynatrace presents an error timeline that visualizes the distribution of errors over time. You can identify spikes or trends in error occurrence and correlate them with specific events or changes in your application environment.

**Error Details:** The Error Analysis page allows you to drill down into individual error occurrences to view detailed information, including the error message, stack trace, affected user sessions, and associated performance metrics. This information helps you diagnose the root cause of errors and prioritize remediation efforts.

**Error Grouping and Aggregation:** Dynatrace automatically groups similar errors based on common characteristics, such as error message, exception type, or stack trace. This grouping simplifies error analysis by consolidating related errors and providing aggregated statistics for each error group.

**Error Severity and Impact:** Errors are often categorized based on severity levels, such as critical, warning, or informational. Dynatrace assigns severity levels to errors based on their impact on application performance and user experience, helping you prioritize critical issues requiring immediate attention.

**Error Trend Analysis:** The Error Analysis page offers trend analysis capabilities to track changes in error rates and patterns over time. You can identify whether error rates are increasing or decreasing, analyze the impact of code deployments or configuration changes, and monitor the effectiveness of error mitigation measures.

**Integration with Monitoring and Alerting:** Dynatrace integrates error data with its broader monitoring and alerting capabilities, allowing you to set up alerts based on specific error conditions or thresholds. You can receive real-time notifications when error rates exceed predefined thresholds or when new types of errors are detected.

Overall, the Error Analysis page in Dynatrace provides a comprehensive view of errors within your applications, enabling you to quickly identify, diagnose, and resolve issues to ensure the reliability and performance of your digital services.

## 8. References

1. <https://docs.dynatrace.com/docs/platform-modules/digital-experience/web-applications/additional-configuration/configure-errors>
2. [https://www.dynatrace.com/?utm\\_source=google&utm\\_medium=cpc&utm\\_term=dynatrace&utm\\_campaign=us-brand-brand&utm\\_content=none&utm\\_campaign\\_id=236796727&gclid=CjwKCAjwz42xBhB9EiwA48pT7\\_m1dQDRYpKD0q3gKXSZTIGoLal5\\_tQ8-ND8b93pAtvbzyiJxrb0hxoC71kQAvD\\_BwE](https://www.dynatrace.com/?utm_source=google&utm_medium=cpc&utm_term=dynatrace&utm_campaign=us-brand-brand&utm_content=none&utm_campaign_id=236796727&gclid=CjwKCAjwz42xBhB9EiwA48pT7_m1dQDRYpKD0q3gKXSZTIGoLal5_tQ8-ND8b93pAtvbzyiJxrb0hxoC71kQAvD_BwE)