salesforce architects

Following 2.1K Followers About

Diagnostics and Monitoring Tools for Salesforce — Part 1

Improve the quality and performance of your implementations

Salesforce Architects Dec 1, 2020 · 8 min read

Instead, you have a comprehensive set of tools available to you. Admittedly, it can be overwhelming to understand what tools are available and when to best apply them. This two-part series covers tools for diagnostics and monitoring. Based upon their primary use cases, we loosely categorize the tools into four categories:

- **Development and quality** — troubleshooting tools that help developers to debug and improve the quality of application being developed
- **Scale and performance** — measurement tools used to assess key performance and scale metrics
- **Security and compliance** — diagnostics tools that identify security risks and potential loopholes
- **Release and maintenance** — monitoring tools to help administrators track routine jobs and perform system maintenance to avoid technical debt

Part 1 covers the first two categories; Part 2 covers the remaining two. For your reference, all of the tools covered in this series are summarized in a single table shared in Part 2.

The right tool for the right time

Salesforce Application Lifecycle Management models provide the project team with a consistent process and policies to build applications smoothly. The steps in an application lifecycle can have different labels, but they fall into these key activities:

- Plan
- Code and configure
- Merge and test
- Scale and user acceptance testing (UAT)
- Release and maintain

Tool Category	Typical Adoption Timing in ALM	
	Phase-in	Phase-out
Development & Quality	Code & Configure	Release & Maintain
Scale & Performance	Code & Configure	Scale & UAT
Security & Compliance	Merge & Test	Release & Maintain
Operation & Adoption	Release & Maintain	Plan

- **Development and quality tools** are typically used by developers at the beginning of the *code and configure* stage until the post-production support activities are completed during the r*elease and maintain* stage**.**
- **Scale and performance** KPIs should be measured and benchmarked early on in the *code and configure* stage. Since these KPIs should be part of the control tollgate for production releases, we expect that the usage of scale and performance measurement tools to phase out towards the end of the *scale and UAT* stage.
- **Security and compliance** checks are typically part of the scope for non-functional QA during the *merge and test* stage. They are also important tools for administrators who are responsible for application upkeep during the *release and maintain* stage.
- **Operation and adoption** tools are used by administrators to perform their day-to-day Salesforce application support activities during the *release and maintain* stage. This set of tools is also useful to reduce technical debt in existing applications. These findings can be useful inputs to the *plan* stage of next release.

Development and quality use cases

work in progress. Debug logs are one of the most commonly used tools for this. A debug log can record database operations, system processes, and errors that occur when executing a transaction or running unit tests. The debug log is easily accessible in a Salesforce organization's setup section or via Developer Console. Whether you are using Visual Studio Code or the new web-based Salesforce Code Builder, the Apex Interactive Debugger provides a much improved real-time debugging capability. In addition, you can use the Apex Replay Debugger to reproduce a scenario by replaying a debug log interactively against the code base.

What if you could visualize all this great data from debug logs? Wouldn't that make debugging and code optimization so much easier? With **SPEXY** you can start a trace on a specific user flow. The tool will automatically gather all the relevant debug logs in the session and generate a dashboard of call sequences and performance breakdowns by key categories such as SOQL, DML, Callout, Workflow and so on. As a result, you can identify Apex performance bottlenecks much faster and more efficiently than going through multiple large log files manually.

A tool belt for Lightning Experience development

There are two programming models for building reusable components for Salesforce Lightning Experience applications: Aura and Lightning Web Components (LWC). The new LWC model is an implementation of the W3C's Web Components standards. Therefore, LWC developers can use standard browser-based tools to debug front-end issues (JavaScript, HTML, and CSS). Chrome DevTools and Developer Tools in IE are both popular choices. For Aura developers, there is a Chrome DevTools extension, Salesforce Lightning Inspector, that enables you to navigate the component tree, inspect component attributes, and profile component performance. Since JavaScript is a key construct in LWC, Linting can also provide an important guardrail against malformed code or anti-patterns. Salesforce created LWC linting rules that can be enforced with eslint-plugin-lwc.

You might be wondering how to debug issues with Salesforce applications for mobile devices. Salesforce provides mobile develper tools and resources that enable you to build and debug mobile-ready Lightning Web Components both locally from Visual

App Builder is a point-and-click tool that makes it easy to create custom pages for the Salesforce mobile and desktop app. You can get tips in a docked prompt within App Builder to improve your pages in areas such as performance, usability, and structure. For Salesforce communities, the Community Page Optimizer is a Chrome plug-in that analyzes your community and identifies issues that impact performance.

Other diagnostic tools in the Salesforce ecosystem

Salesforce takes an API-first approach to building features on the Salesforce Platform. There is a vibrant community using those APIs to build third-party tools that extend the platform's diagnostics capabilities. For example, **Workbench** is a popular, community-supported suite of tools designed for administrators and developers to interact with Salesforce organizations via the Force.com APIs. **Toolkit for Salesforce** provides a set of tools and applications to support common diagnostic tasks.

AppExchange is another good source for solutions that extend the standard diagnostics provided by the platform. For example, the **Streaming Monitor** and **Platform Event Usage Monitor** apps published by Salesforce Labs let developers or administrators monitor stream events such as PushTopic, platform, and CDC events, as well as usage statistics.

Tool	Features
Apex Interactive Debugger and Apex Replay Debugger (Salesforce Extension Pack for VS Code)	Troubleshoot your Apex code by inspecting debug logs using Visual Studio Code as the client.
AppExchange	Find ready-to-install solutions that extend Salesforce capabilities in this marketplace
Browser Developer Tools (e.g. Chrome DevTools)	Help LWC developers to inspect, profile and audit lightning components
Linting - Lightning	Identify errors about malformed code against established

Monitor	Take further action with Einstein Analytics notifications.
Salesforce Community Page Optimizer	Identify common performance anti-pattern
Salesforce Debug Log	Identify problems with a Salesforce process. Check out these trailheads for more information : Generate and Analyze Logs in Developer Console and Find and Fix Bugs with Apex Replay Debugger
Salesforce Lightning Inspector	Enables Aura component developers to navigate the component tree, inspect component attributes, and profile component performance. This Chrome DevTools extension also helps you to understand the sequence of event firing and handling.
Salesforce Mobile Tools	Enable developers to build and debug mobile ready Lightning web components both locally, and from within the Salesforce mobile app virtual device builds.
SPEXY	Identify and visualize performance hotspots by automatically gathering all the relevant debug logs in the session and generate a dashboard of call sequence with breakdowns by key categories such as SOQL, DML, Callout, Workflow and etc.
Streaming Monitoring	Monitor streaming events (PushTopic, generic, standard/custom platform events, CDC and monitoring events)
Tips in Lightning App Builder	Gives you feedback for enhancing performance and usability when you design a page, via tips in a docked prompt within the Salesforce App Builder
Toolkit for Salesforce	Assist common consulting processes and diagnosing tasks
Workbench	Allows users to describe, query, manipulate, and migrate both data and metadata in Salesforce.com organizations

Scale and performance use cases

For any mission-critical applications, scale and performance requirements should be an integral part of the project plan from inception. One of the primary goals for scale testing is to determine the application's ability to scale up to the maximum number of expected users or transaction volume. It is especially important to test CPU-intensive operations or database-intensive transactions to ensure they don't violate governor limits during peak load. Additionally, performance testing should be conducted to identify performance bottlenecks under anticipated user loads.

Measure Lightning Experience performance

Experienced Page Time (EPT) is a performance metric Salesforce uses in Lightning applications or communities. EPT measures how long it takes for a page to load into a state that a user can meaningfully interact with. You can display EPT in the header of your application by appending ?eptVisible=1 to the URL. The aggregate page EPT statistics are stored in Lightning Usage App objects (e.g. LightningUsageByPageMetrics). You can monitor these metrics with the standard Lightning Usage App or by building a custom report.

Navigate through the performance triangle: client, network, and server

With the Winter '21 release, you can analyze the performance of your page against configuration best practices right inside Lightning App Builder. In order to improve the performance of a suboptimal Lightning page, a logical next step is to learn additional information about the performance characteristics. Lightning page performance can be categorized into three areas:

- **Client** — time to execute JavaScript or render HTML locally in the browser
- **Network** — time to communicate between the browser and Salesforce
- **Server** — time Salesforce needs to perform various logic or database operations

You can use browser developer tools to gain insights on Lightning page performance characteristics in these areas. In addition, the profiling capabilities in these tools are pivotal in identifying specific culprits at client or network bottlenecks. Other than the

power and memory available on the client machine, and the choice of browser. You can find the browser's octane score, latency, and download speed by running a speed test on the target user's machine. To run the test, append "speedtest.jsp" to the Salesforce organization's URL (e.g. https://MyDomainName.lightning.force.com/speedtest.jsp)

For Salesforce server-side performance metrics, the debug log is a good starting point. However, it can be challenging to fish out specific measurements from a monolithic debug log. Event Monitoring is powerful tool for exploring the granular details of user activity in your organization. It captures and stores key system events into an EventLogFile object. The event types address a variety of monitoring use cases such as security, compliance, and performance. There is a standard Event Monitoring Analytics App with a set of prebuilt dashboards that let you to explore the event log data. An AppExchange package named Eagle Eyes provides further visualization and interpretation of performance events based upon established best practices. You can also download the event files using the Salesforce Event Log File Browser and import them to other log analytics platforms (e.g. Splunk) for further analysis. The server performance metrics are typically tied to CPU and database usage. For SOQL-related performance issues, the Query Plan tool in Developer Console provides you with insight on different query operations and their related costs.

A subset of events gathered by Event Monitoring is available in near real time. These real time events are streamed as platform events and can be consumed by the Streaming Monitoring App from App Exchange, or any third-party monitoring application that supports the streaming API (e.g. CometD). There are five main categories of real-time events: authentication, data access, page access, threat detection, and mobile security events.
Among them, there are two streaming events that are especially useful for performance monitoring:

- **LightningUriEventStream**, with EPT attribute, detects when a user creates, accesses, updates, or deletes a record in Lightning Experience.

Tool	Features
Developer Console Query Plan Tool	View query plans for SOQL queries, SOSL searches, reports, and list views in order to optimize and speed up queries done over large numbers of records. Refer to the Query Plan Tool FAQ and Lightning Platform Query Optimization FAQ
Eagle eyes-Performance monitoring powered by Event Monitoring	View and interpret performance from Event Monitoring based on established best practices
Event Log Browser	A Salesforce connected web app to access and download event log files
Event Monitoring Analytics App	A set of prebuilt dashboards for events log captured by Event Monitoring
Lightning Usage App	Monitor adoption metrics, like daily active Lightning Experience users, the number of users switching back to Salesforce Classic, and the most visited pages in Lightning Experience. Trailhead: Lightning Experience Performance Optimization Help Article: Improve Performance and Speed in Lightning Experience
?eptVisible=1	Add '?eptVisible=1' string to your URL in the Address Bar to show the page Lightning page load times
speedtest.jsp	Measure your lightning performance by running the Salesforce Performance Test by appending `speedtest.jsp` to your org's domain.

Table 2. Diagnostic tools for scale and performance

performance diagnosis. In Part 2, we will complete the checklist for your toolbox with tools for security and system maintenance.

About the Authors

Ivan Yeung works as a Success Architect at Salesforce. He helps innovative customers architect, build, and manage enterprise-scale applications on the Salesforce Platform. He is passionate about applying leading edge technology, such as NLP AI and blockchain, in Salesforce solutions.

Salesforce Architect Ask An Architect Monitoring Diagnostics