

AppDynamics Pro Documentation

Complete Documentation Suite



AppDynamics

Application Management for the Cloud Generation

1. AppDynamics Pro Documentation	7
1.1 AppDynamics Essentials	8
1.1.1 Features Overview	9
1.1.2 Architecture	12
1.1.3 Logical Model	14
1.1.3.1 Hierarchical Configuration Model	15
1.1.3.2 Mapping Application Services to the AppDynamics Model	16
1.1.4 Getting Started	17
1.1.4.1 Get Started with AppDynamics SaaS	18
1.1.4.1.1 Use a SaaS Controller	21
1.1.4.1.2 SaaS Availability and Security	22
1.1.4.2 Get Started With AppDynamics On-Premise	23
1.1.4.3 Download AppDynamics Software	26
1.1.4.4 Quick Start for DevOps	27
1.1.4.5 Quick Start for Architects	28
1.1.4.6 Quick Start for Administrators	29
1.1.4.7 Quick Start for Operators	30
1.1.4.8 Set User Preferences	30
1.1.5 Glossary	31
1.1.6 AppDynamics Support	37
1.1.6.1 Controller Dump Files	37
1.1.6.2 Log Files for Troubleshooting	38
1.1.6.3 Configure Remote Monitoring of an On-Premise Controller	38
1.1.6.4 Download Doc PDFs	41
1.1.6.5 Use the Documentation Wiki	42
1.1.6.6 License Information	44

1.1.7 Documentation Map	44
1.2 AppDynamics Features	46
1.2.1 Application Performance Monitoring	47
1.2.1.1 Visualize App Performance	48
1.2.1.1.1 Dashboards	48
1.2.1.1.2 Transaction Analysis Tab	87
1.2.1.1.3 Flow Maps	89
1.2.1.1.4 Time Ranges	97
1.2.1.1.5 Key to the AppDynamics Icons	99
1.2.1.1.6 Scorecards	101
1.2.1.1.7 Events	102
1.2.1.1.8 KPI Graphs	116
1.2.1.1.9 Metric Points in Graphs	116
1.2.1.1.10 Metric Browser	118
1.2.1.2 Behavior Learning and Anomaly Detection	123
1.2.1.2.1 Configure Baselines	133
1.2.1.3 Reports	135
1.2.2 Business Transaction Monitoring	138
1.2.2.1 Organizing Traffic as Business Transactions	141
1.2.2.1.1 Monitor All Other Traffic	144
1.2.2.2 Measure Business Transaction Performance	145
1.2.2.3 Apparent Business Transaction Flow Anomalies	146
1.2.2.4 Measure Distributed Transaction Performance	148
1.2.2.5 Transaction Snapshots	150
1.2.2.5.1 Configure Transaction Snapshots	155
1.2.2.6 Alerting for Business Transaction Health Problems	155
1.2.2.7 Data Collectors	156
1.2.2.7.1 Configure Data Collectors	158
1.2.2.7.2 Data Collectors Versus Information Points	162
1.2.2.8 Configure Business Transaction Detection	163
1.2.2.8.1 Match Rule Conditions	170
1.2.2.8.2 Regular Expressions In Match Conditions	171
1.2.2.8.3 Messaging Entry Points	171
1.2.2.8.4 Web Entry Points	173
1.2.2.8.5 Creating New Tiers	177
1.2.2.9 Thresholds	178
1.2.2.9.1 Configure Thresholds	178
1.2.3 End User Experience	180
1.2.3.1 EUM Metrics	181
1.2.3.2 Monitor End User Experience (EUM)	184
1.2.3.2.1 Monitor End User Experience by Browser	185
1.2.3.2.2 Monitor End User Experience by Device	187
1.2.3.3 Pages	189
1.2.3.3.1 Page List	190
1.2.3.3.2 Page Dashboard	192
1.2.3.3.3 Browser Snapshots	195
1.2.3.4 Geo Dashboard	201
1.2.3.4.1 Hosting a Geo Server	207
1.2.3.5 Configure End User Experience	207
1.2.3.5.1 Configure the .NET App Agent for EUM	209
1.2.3.5.2 Configure Page Identification and Naming	209
1.2.3.5.3 Configure JavaScript and Ajax Error Detection	212
1.2.3.5.4 Configure EUM Thresholds	215
1.2.3.5.5 Configure Browser Snapshot Collection	216
1.2.3.5.6 Customize Your EUM Deployment	217
1.2.3.5.7 Inject the JavaScript Agent for EUM	221
1.2.3.6 EUM Supported Countries and Regions	233
1.2.3.6.1 EUM Supported Countries	233
1.2.3.6.2 EUM Supported Regions	237
1.2.3.7 EUM License	294
1.2.4 Background Task Monitoring	294
1.2.4.1 Configure Background Tasks	295
1.2.5 Backend Monitoring	297
1.2.5.1 Monitor Databases	297
1.2.5.2 Monitor Remote Services	301
1.2.5.3 Configure Backend Detection	304
1.2.5.3.1 Configure Custom Exit Points	310
1.2.5.4 Configure Stale Backend Removal	314
1.2.6 Infrastructure Monitoring	315
1.2.6.1 Infrastructure Metrics	316
1.2.6.2 Monitor App Servers	323
1.2.6.3 Monitor Hardware	323
1.2.7 Alert and Respond	325
1.2.7.1 Policies	327
1.2.7.1.1 Configure Policies	330

1.2.7.2 Health Rules	333
1.2.7.2.1 Configure Health Rules	340
1.2.7.2.2 Troubleshoot Health Rule Violations	344
1.2.7.3 Actions	347
1.2.7.3.1 Notification Actions	350
1.2.7.3.2 Diagnostic Actions	351
1.2.7.3.3 Remediation Actions	352
1.2.7.3.4 Cloud Auto-Scaling Actions	357
1.2.7.3.5 Custom Actions	357
1.2.7.4 Email Digests	358
1.2.7.4.1 Configure Email Digests	358
1.2.7.5 Alerting Wizard	360
1.2.8 Rapid Troubleshooting	361
1.2.8.1 Troubleshoot Slow Response Times	362
1.2.8.2 Troubleshoot Errors	364
1.2.8.3 Troubleshoot Node Problems	366
1.2.8.4 Call Graphs	368
1.2.8.4.1 Configure Call Graphs	372
1.2.8.5 Diagnostic Sessions	374
1.2.8.6 Analyze	376
1.2.8.6.1 Scalability Analysis	376
1.2.8.6.2 Correlation Analysis	378
1.2.8.6.3 Compare Releases	379
1.2.9 Information Points	380
1.2.9.1 Business Metrics	380
1.2.9.1.1 Configure Business Metric Information Points	382
1.2.9.2 Code Metrics	385
1.2.9.2.1 Configure Code Metric Information Points	386
1.2.10 System Integrations	386
1.2.10.1 Integration Basics	387
1.2.10.2 Integrate using Custom Action Scripts	389
1.2.10.3 Use the AppDynamics REST API	393
1.2.10.3.1 Example Integrations Using the AppDynamics REST API	413
1.2.10.4 Integrate AppDynamics with Splunk	413
1.2.10.5 Integrate AppDynamics with BMC End User Experience Management	417
1.2.10.6 Integrate with AppDynamics for Databases	423
1.2.10.7 Integrate AppDynamics with DB CAM	424
1.2.10.8 Integrate with Apica	427
1.3 AppDynamics for Java	427
1.3.1 Tutorials for Java	428
1.3.1.1 Overview Tutorials for Java	428
1.3.1.1.1 Use AppDynamics for the First Time with Java	428
1.3.1.2 Monitoring Tutorials for Java	431
1.3.1.2.1 Tutorial for Java - Events	431
1.3.1.2.2 Tutorial for Java - Flow Maps	432
1.3.1.2.3 Tutorial for Java - Server Health	436
1.3.1.2.4 Tutorial for Java - Transaction Scorecards	439
1.3.1.3 Troubleshooting Tutorials for Java	441
1.3.1.3.1 Super-Simple Java Troubleshooting using Events	442
1.3.1.3.2 Tutorial for Java - Business Transaction Health Drilldown	447
1.3.1.3.3 Tutorial for Java - Exceptions	448
1.3.1.3.4 Tutorial for Java - Slow Transactions	453
1.3.1.2 Install the App Agent for Java	455
1.3.2.1 Multi-Agent Deployment for Java	459
1.3.2.2 Java Server-Specific Installation Settings	460
1.3.2.2.1 Apache Cassandra Startup Settings	460
1.3.2.2.2 Apache Tomcat Startup Settings	461
1.3.2.2.3 Glassfish Startup Settings	466
1.3.2.2.4 IBM WebSphere Startup Settings	468
1.3.2.2.5 JBoss Startup Settings	474
1.3.2.2.6 Jetty Startup Settings	480
1.3.2.2.7 Oracle WebLogic Startup Settings	480
1.3.2.2.8 OSGi Infrastructure Configuration	483
1.3.2.2.9 Resin Startup Settings	485
1.3.2.2.10 Solr Startup Settings	487
1.3.2.2.11 Standalone JVM Startup Settings	488
1.3.2.2.12 Tanuki Service Wrapper Configuration	489
1.3.2.2.13 Tibco BusinessWorks Configuration	489
1.3.2.3 Upgrade the App Agent for Java	489
1.3.2.4 Uninstall the App Agent for Java	490
1.3.3 Administer App Agents for Java	490
1.3.3.1 App Agent for Java Configuration Properties	490
1.3.3.2 App Agent for Java Diagnostic Data	497
1.3.3.3 App Agent for Java Directory Structure	499
1.3.3.4 App Agent for Java Performance Tuning	500

1.3.3.5 Configure App Agent for Java for Batch Processes	501
1.3.3.6 Configure App Agent for Java for JVMs that are Dynamically Identified	502
1.3.3.7 Configure App Agent for Java in Restricted Environments	503
1.3.3.8 Configure App Agent for Java in z-OS or Mainframe Environments	503
1.3.3.9 Configure App Agent for Java on Multiple JVMs on the Same Machine that Serve Different Tiers	505
1.3.3.10 Configure App Agent for Java on Multiple JVMs on the Same Machine that Serves the Same Tier	507
1.3.3.11 Configure App Agent for Java to Use Existing System Properties	509
1.3.3.12 IBM App Agent for Java	511
1.3.3.13 Move an App Agent for Java Node to a New Application or Tier	512
1.3.3.14 Troubleshoot App Agent for Java	513
1.3.3.15 Administer App Agent for Java FAQ	514
1.3.4 Configure AppDynamics for Java	515
1.3.4.1 Configure Custom Exit Points (Java)	515
1.3.4.1.1 Configurations for Custom Exit Points	520
1.3.4.2 Code Metric Information Points (Java)	523
1.3.4.3 Configure JMX Metrics from MBeans	523
1.3.4.3.1 Exclude JMX Metrics	530
1.3.4.3.2 Exclude MBean Attributes	530
1.3.4.3.3 Configure JMX Without Transaction Monitoring	531
1.3.4.3.4 Resolve JMX Configuration Issues	531
1.3.4.3.5 Import or Export JMX Metric Configurations	536
1.3.4.4 Configure Custom Memory Structures (Java)	539
1.3.4.5 Configure Object Instance Tracking (Java)	543
1.3.4.6 Configure Memory Monitoring (Java)	544
1.3.4.7 Configure Multi-Threaded Transactions (Java)	545
1.3.4.8 Configure Background Tasks (Java)	547
1.3.4.9 Web Application Entry Points	549
1.3.4.9.1 Servlet Entry Points	550
1.3.4.9.2 Struts Entry Points	567
1.3.4.9.3 Web Service Entry Points	568
1.3.4.9.4 POJO Entry Points	570
1.3.4.9.5 Spring Bean Entry Points	575
1.3.4.9.6 EJB Entry Points	576
1.3.4.9.7 POCO Entry Points	579
1.3.4.10 Getter Chains in Java Configurations	583
1.3.5 Troubleshoot Java Application Problems	585
1.3.5.1 Detect Code Deadlocks (Java)	585
1.3.5.2 Troubleshoot Java Memory Issues	587
1.3.5.2.1 Troubleshoot Java Memory Leaks	587
1.3.5.2.2 Troubleshoot Java Memory Thrash	592
1.3.6 Monitor Java Applications	595
1.3.6.1 Monitor JVMs	595
1.3.6.2 Monitor Java App Servers	602
1.3.6.2.1 Monitor JMX MBeans	603
1.3.6.3 Trace Multi-Threaded Transactions (Java)	608
1.4 AppDynamics for .NET	613
1.4.1 Install the App Agent for .NET	614
1.4.1.1 Upgrade the App Agent for .NET	616
1.4.1.2 Configure the App Agent for .NET	616
1.4.1.3 Uninstall the App Agent for .NET	630
1.4.1.4 Enable the App Agent for .NET for Non-IIS Applications	630
1.4.1.5 Troubleshoot App Agent for .NET Installation and Configuration	634
1.4.1.6 How to Configure the .NET Agent Manually	644
1.4.1.7 Multi-Agent Deployment for .NET	648
1.4.1.8 Naming Conventions for .NET Nodes	650
1.4.2 Administer App Agents for .NET	651
1.4.2.1 App Agent for .NET Configuration Properties	652
1.4.2.2 Disable Instrumentation for an IIS Application Pool	657
1.4.2.3 App Agent for .NET Logs	657
1.4.3 Monitor .NET Applications	658
1.4.3.1 Monitor CLRs	659
1.4.3.2 Monitor IIS	661
1.4.3.3 Windows Performance Counters	663
1.4.4 Configure AppDynamics for .NET	663
1.4.4.1 Configure Custom Exit Points (.NET)	663
1.4.4.2 Getter Chains in .NET Configurations	669
1.4.4.3 Enable Thread Correlation (.NET)	669
1.4.4.4 Configure the .NET Machine Agent	670
1.4.4.5 Enable Correlation for .NET Remoting	672
1.4.5 Tutorials for .NET	673
1.4.5.1 Monitoring Tutorials for .NET	673
1.4.5.2 Overview Tutorials for .NET	673
1.4.5.3 Troubleshooting Tutorials for .NET	673
1.4.6 Automate Scaling in Windows Azure	673
1.5 AppDynamics Administration	676

1.5.1 Release Notes for AppDynamics Pro	676
1.5.1.1 Upgrade Policies from 3.6 to 3.7	682
1.5.1.2 Upgrade End User Monitoring	686
1.5.1.3 Agent - Controller Compatibility Matrix	687
1.5.2 Supported Environments and Versions	688
1.5.3 Install and Upgrade AppDynamics	696
1.5.3.1 Name Business Applications, Tiers, and Nodes	697
1.5.3.2 Install Agents for 5 or fewer JVMs or CLRs (Self-Service Installations)	698
1.5.3.2.1 Self-Service On-Premise Install - Controller and App Agents for Java	698
1.5.3.2.2 Self-Service On-Premise Install - Controller and App Agents for .NET	699
1.5.3.2.3 Self-Service Install - App Agents for .NET	699
1.5.3.2.4 Self-Service Install - App Agents for Java	700
1.5.3.3 Install Agents for 5 or more JVMs or CLRs (Standard Installations)	702
1.5.3.3.1 Automate Multi-Agent Deployment	702
1.5.3.4 Install the Controller	703
1.5.3.4.1 Controller System Requirements	703
1.5.3.4.2 Install the Controller on Linux	708
1.5.3.4.3 Upgrade the Controller	722
1.5.3.4.4 Uninstall the Controller	725
1.5.3.4.5 Migrate the Controller between Machines	725
1.5.3.4.6 Controller Install and Admin FAQ	727
1.5.3.4.7 Install the Controller on Windows	730
1.5.3.4.8 Install an On-Premise Controller Silently	737
1.5.3.4.9 Install the Self-Service Controller on Linux	739
1.5.3.4.10 Install the Self-Service Controller on Windows	743
1.5.3.5 Verify App Agent - Controller Communication	746
1.5.3.6 Install the Machine Agent	747
1.5.3.6.1 Configure the Machine Agent to Automatically Start on Linux	750
1.5.4 Administer Agents	751
1.5.4.1 Metrics Limits	751
1.5.4.2 App Agent Node Properties	752
1.5.4.2.1 App Agent Node Properties Reference	753
1.5.4.2.2 App Agent Node Properties Reference By Type	771
1.5.4.3 Manage App Agents	772
1.5.4.4 Administer Machine Agents	774
1.5.4.4.1 Add Metrics Using Custom Monitors	776
1.5.4.4.2 Configure Custom Metrics for the z-OS Machine Agent	780
1.5.4.4.3 Machine Agent Configuration Properties	782
1.5.4.4.4 Machine Agent HTTP Listener	787
1.5.4.4.5 Machine Agent Install and Admin FAQ	788
1.5.4.4.6 Modify Machine Agent Data Collection Metrics	790
1.5.4.4.7 Configure Multiple Machine Agents for One Machine	791
1.5.5 Administer the Controller	792
1.5.5.1 Modify Glassfish JVM Options	792
1.5.5.2 Access the AppDynamics UI	794
1.5.5.3 Start or Stop the Controller	794
1.5.5.4 Administrative Users	795
1.5.5.5 Access the Administration Console	796
1.5.5.6 Configure an On-Premise Controller	797
1.5.5.6.1 Controller Licenses	797
1.5.5.6.2 Controller Tenant Mode	797
1.5.5.6.3 Controller Port Settings	799
1.5.5.6.4 Controller SSL and Certificates	801
1.5.5.6.5 Controller Logs	801
1.5.5.6.6 Controller Performance	804
1.5.5.6.7 Configure the SMTP Server	804
1.5.5.6.8 Controller High Availability	806
1.5.5.6.9 Controller Data and Backups	819
1.5.5.7 Controller Version	827
1.5.5.8 Remove Unused Nodes	827
1.5.5.9 Controller Info Configuration for Agents	828
1.5.5.10 Controller Infrastructure	829
1.5.5.11 Controller Audit Log	829
1.5.5.12 Configure Controller VIP	830
1.5.5.13 Controller Error Messages	830
1.5.5.14 Tuning for Large Enterprise Environments	831
1.5.6 Configure Authentication, User Permissions and Integrations	834
1.5.6.1 Configure Authentication Provider	835
1.5.6.2 Configure Authentication Using SAML	835
1.5.6.2.1 Disable SAML Authentication for an Account	837
1.5.6.2.2 SAML Configuration for OneLogin	838
1.5.6.3 Configure Authentication Using LDAP	841
1.5.6.4 Configure Users	846
1.5.6.5 Configure Groups	847
1.5.6.6 Configure Roles	848

1.5.6.6.1 Access Role Configuration	850
1.5.6.6.2 View Predefined Roles	850
1.5.6.6.3 Configure Custom Roles	851
1.5.6.6.4 Assign Roles to Users and Groups	854
1.5.6.7 Configure Integrations	855
1.5.7 AppDynamics for Large Enterprises	855
1.5.8 Best Practices for Failover Scenarios	855
1.5.9 Implement SSL	858
1.5.10 Export and Import Business Application Configurations	859
1.5.11 Internationalization	860
1.5.12 Automation	861
1.5.12.1 Workflow Automation	861
1.5.12.1.1 Create a Workflow	862
1.5.12.2 Workflow for Cloud Orchestration	873
1.5.12.2.1 Compute Clouds	873
1.5.12.2.2 Machine Images and Instances	873
1.5.12.2.3 Use the ITask Interface for Cloud Orchestration	874
1.6 AppDynamics Best Practices	875
1.6.1 Best Practices for Performance and Quality Assurance Engineers	875
1.6.2 Best Practices for Operations Professionals	880
1.6.3 Best Practices for Application Developers	889

AppDynamics Pro Documentation

AppDynamics is crazy
simple to install and use

[Release Notes](#)
current release is 3.7.0

[PDFs](#)

[All Releases](#)



GET STARTED

[Self-Service Installations for 5 or fewer app servers](#)

[Standard Installations](#)



TROUBLESHOOT

[Slow Response Times](#)

[Errors](#)

[Health Rule Violations](#)



MONITOR

Application

[Overall Business Application](#)

[Business Transactions](#)

[Databases \(SQL, NoSQL, etc.\)](#)

[Remote Services \(Web Services, HTTP etc.\)](#)

[Code Metrics](#)

[Business Metrics](#)

Custom Dashboards

End User Experience

Pages
Geo Dashboard

Infrastructure

JVMs
CLRs
Java App Servers (JMX)
IIS
Hardware
Windows Performance Counters



ALERT & AUTOMATE

Alert
Integrate with Third Party Alerting Systems
Automate Diagnostics Capture
Automate Local Scripts (Runbook Automation)
Scale Applications on Compute Clouds

Analyze

Metric Browser
Compare Releases

Administer

SaaS
On-Premise

Extend

REST API
Custom Actions
Custom Metrics and Events

Integrate

Splunk
Apica
BMC End User Experience Management

AppDynamics Essentials

AppMan Advice



Isn't it about time you mapped your app?

AppDynamics provides application performance management for modern application architectures. Designed for distributed SOA environments, AppDynamics helps you to manage service levels, reduce mean-time-to-repair for problems, plan for application efficiency, and automate typical life-cycle actions for distributed applications.

Basics

[Features Overview](#)
[AppDynamics in Action Videos](#)
[Architecture](#)
[Logical Model](#)
[Glossary](#)

Getting Started

[Get Started with AppDynamics SaaS](#)
[Get Started With AppDynamics On-Premise](#)
[Download AppDynamics Software](#)
[Set User Preferences](#)

[Quick Start for Operators](#)
[Quick Start for DevOps](#)
[Quick Start for Architects](#)
[Quick Start for Administrators](#)

Support

[Release Notes](#)
[Supported Environments](#)

[AppDynamics Support](#)
[Documentation Map](#)

Features Overview

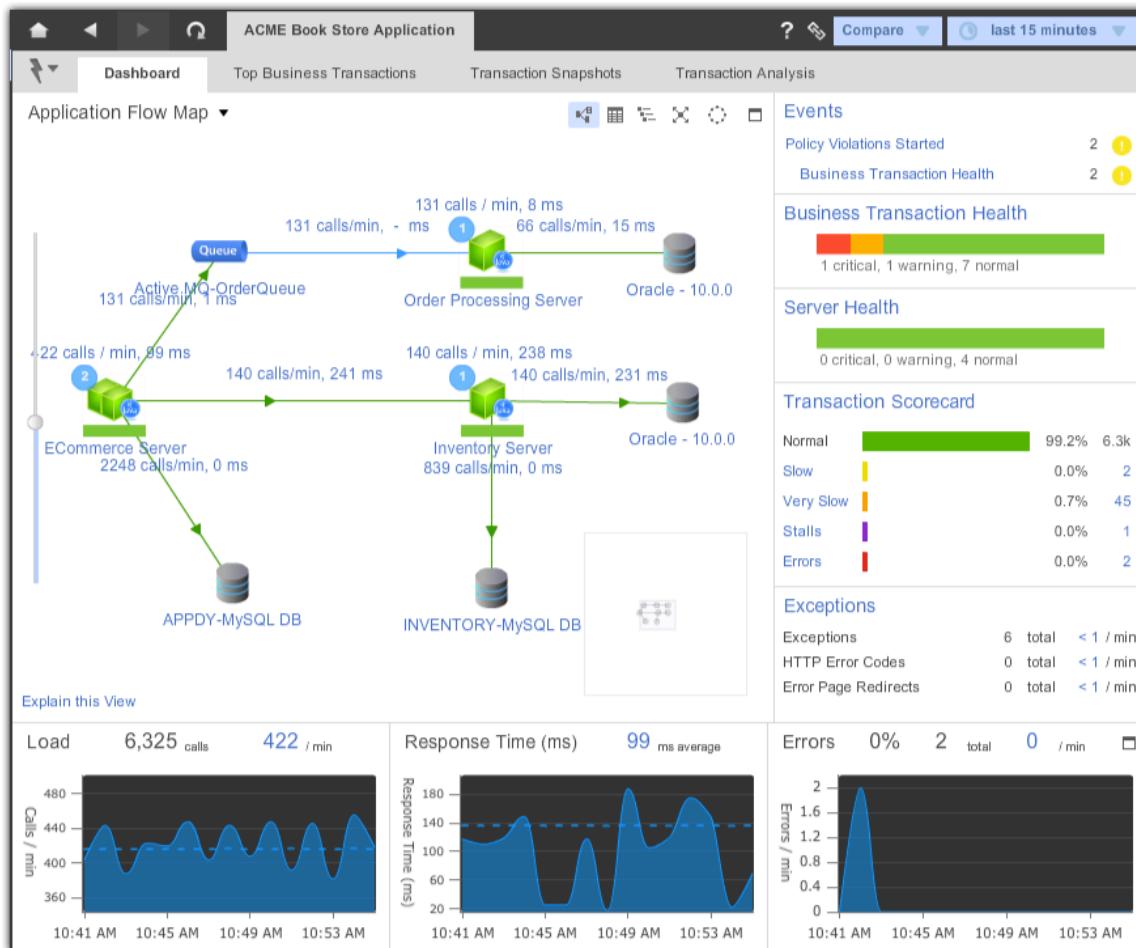
- Continuous Discovery, Visibility, and Problem Detection
- Real-Time Business Transaction Monitoring
- End User Monitoring
- Hardware and Server Monitoring
- Health Rules, Policies, and Actions
- Troubleshooting and Diagnostics
- Systems Integration
- Learn More

This topic describes high-level benefits and features of AppDynamics Pro.

Continuous Discovery, Visibility, and Problem Detection

AppDynamics continuously discovers and monitors all modules in your application environment using advanced tag-and-follow tracing across your distributed transactions. With this information, AppDynamics provides a simple intuitive view of live application traffic and you can see where bottlenecks exist.

Dashboards show the health of your entire business application. Health indicators are based on configurable thresholds and they update based on live traffic. When new services are added to the system AppDynamics discovers them and adds them to the dashboards and flow maps. See [Visualize App Performance](#).



AppDynamics observes normal performance patterns so that it knows when application performance becomes abnormal. It automatically identifies metrics whose current values are out of the normal range, based on dynamic baselines it has observed for these metrics. See [Behavior Learning and Anomaly Detection](#).

Real-Time Business Transaction Monitoring

An AppDynamics business transaction represents a distinct logical user activity such as logging in, searching for items, buying an item, etc. Organizing application traffic into business transactions aligns the traffic with the primary functions of a web business. This approach focuses on how your users are experiencing the site and provides real-time performance monitoring.

The screenshot shows the 'Business Transactions' section of the AppDynamics interface. At the top, there are navigation icons and a search bar set to 'last 15 minutes'. Below this is a header row with columns: Name, Health, End User, Page Render, Network Time, Server Time, Calls, Calls / min, Errors, Error %, Slow Transact, Very Slow, Stalled Transact, Tier, and Type. The main body of the table lists nine transactions, each with a small icon and a link to its details. The transactions are:

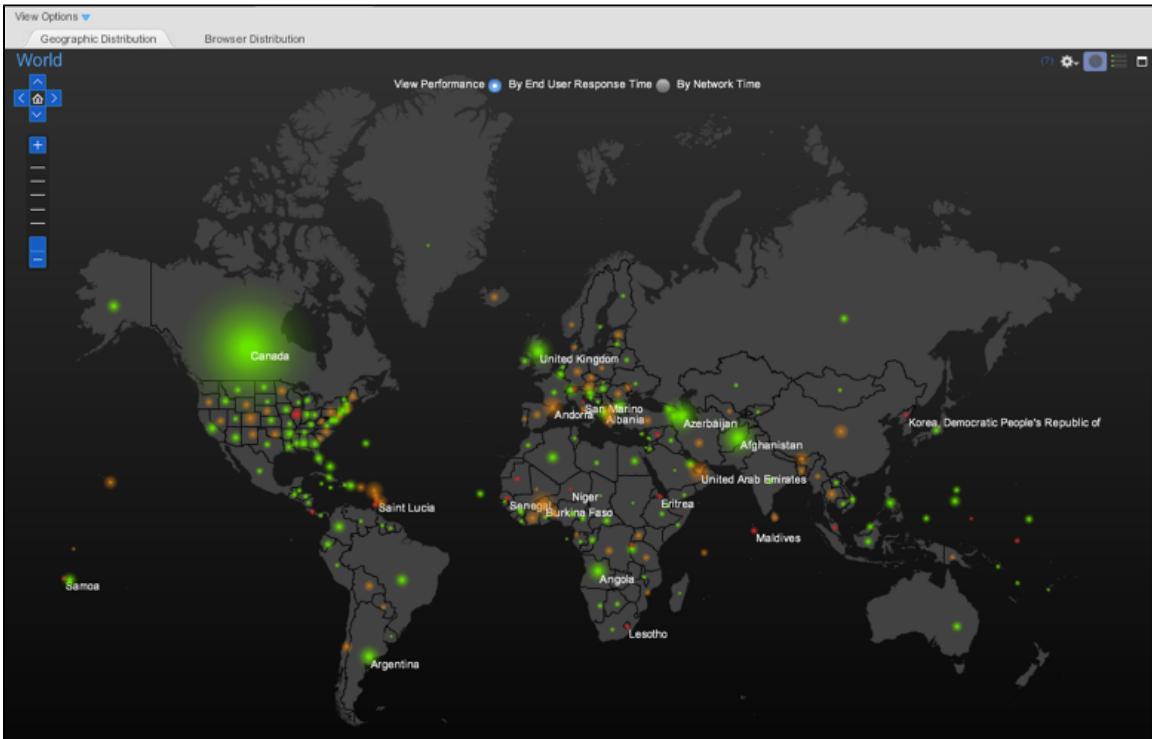
Name	Health	End User	Page Render	Network Time	Server Time	Calls	Calls / min	Errors	Error %	Slow Transact	Very Slow	Stalled Transact	Tier	Type
Fetch catalog	Green	0	0	0	11	1,061	71	0	0	1	0	0	ECommerce...	Struts Action
Homepage	Green	0	0	0	5	1,061	71	0	0	1	0	0	ECommerce...	Servlet
Login	Green	0	0	0	5	1,061	71	0	0	1	0	0	ECommerce...	Struts Action
Add to cart	Yellow	0	0	0	14	1,061	71	0	0	2	0	0	ECommerce...	Struts Action
Checkout	Red	0	0	0	667	1,060	71	2	0.2	0	63	0	ECommerce...	Struts Action
Logout	Green	0	0	0	4	1,060	71	0	0	0	0	0	ECommerce...	Struts Action

See [Business Transaction Monitoring](#) and [Background Task Monitoring](#).

End User Monitoring

End user monitoring (EUM) provides information about your end users' experience starting from the users' web browsers. It gives you visibility across geographies and browser types, answering questions such as:

- Where are the heaviest loads?
- Where are the slowest end-user response times?
- How does end user performance vary by Web browser?



See [End User Experience](#).

Hardware and Server Monitoring

AppDynamics machine agents gather information about the operating systems and machines, such as CPU activity, memory usage, disk reads and writes, etc. AppDynamics agents monitor JVM and CLR metrics including heap usage and collections. See [Infrastructure Monitoring](#).

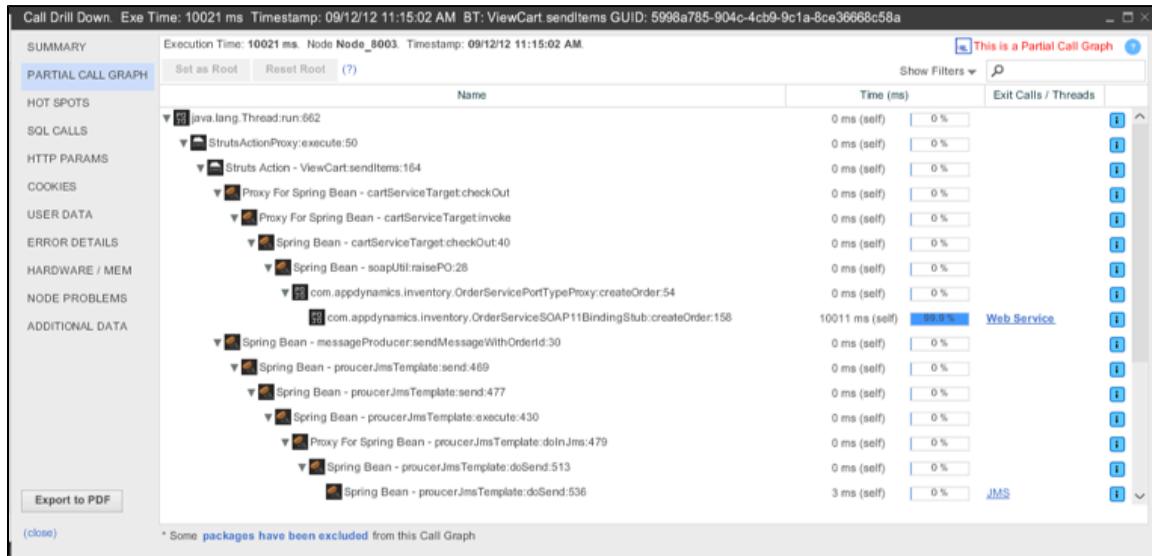
Health Rules, Policies, and Actions

Dynamic baselines combined with policies and health rules help you proactively detect and troubleshoot problems before customers are affected. Health rules define metric conditions to monitor, such as when the "average response time is four times slower than the baseline". AppDynamics supplies default health rules that you can customize, and you can create new ones.

You can configure policies to trigger automatic actions when a health rule is violated or when any event occurs. Actions include sending email, scaling-up capacity in a cloud or virtualized environment, taking a thread dump, or running a local script. See [Alert and Respond](#).

Troubleshooting and Diagnostics

You can examine transaction snapshots for slow and error transactions and drill down into the snapshot with the slowest response time to begin deep diagnostics to discover the root cause of the problem.



See [Rapid Troubleshooting](#).

Systems Integration

AppDynamics is designed to interface with other systems in your organization. You can add data to AppDynamics, retrieve data from AppDynamics, and integrate AppDynamics actions into your alerting system. See [System Integrations](#).

Learn More

- [Product Features and Benefits](#)

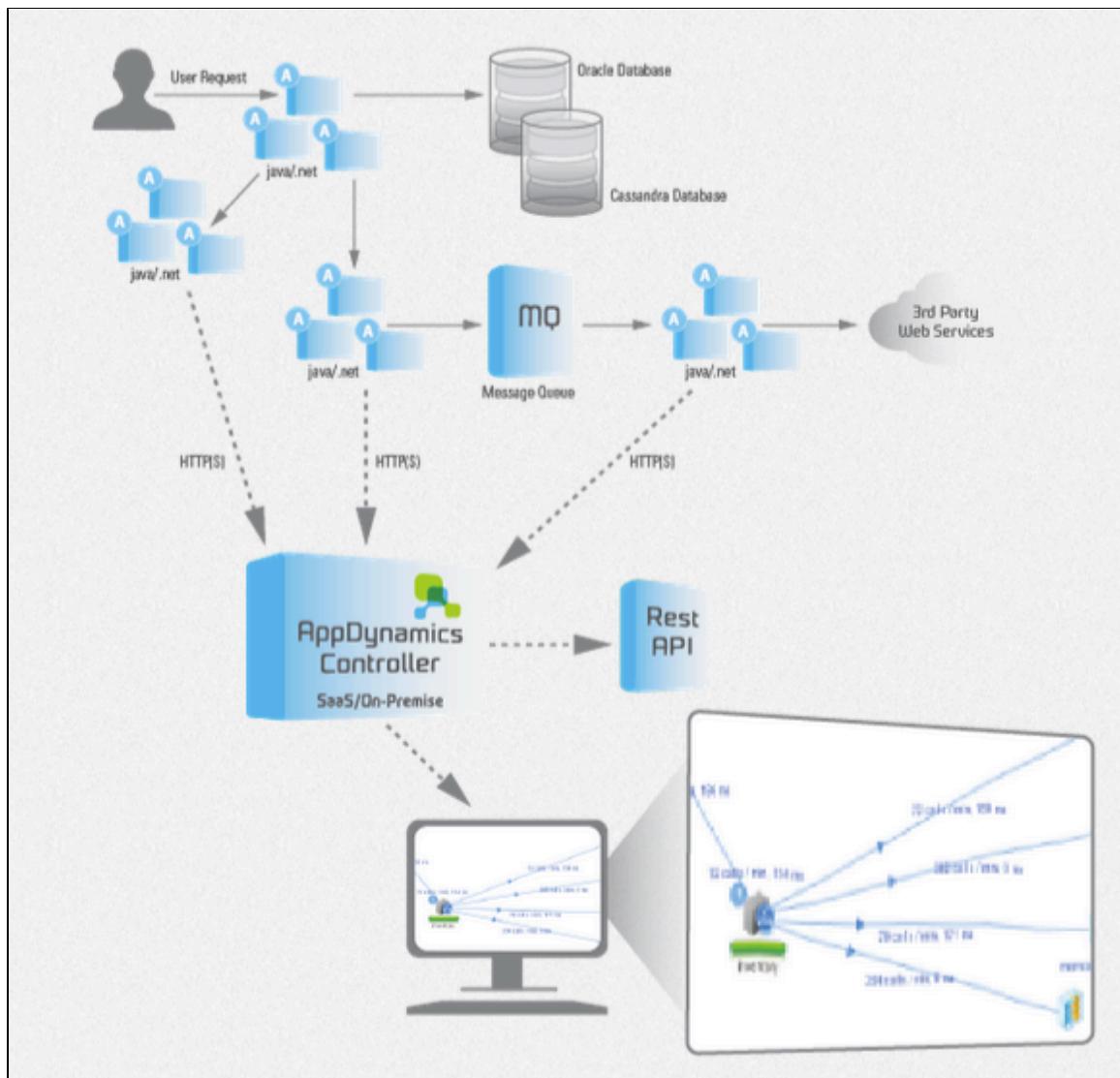
Architecture

- [AppDynamics Pro Architecture](#)
 - [AppDynamics Controller and UI](#)
 - [AppDynamics App Agents](#)
 - [AppDynamics Machine Agents](#)
- [Learn More](#)

This topic summarizes the components of AppDynamics and how they work together to monitor your application environment.

AppDynamics Pro Architecture

An AppDynamics deployment consists of a Controller (either on-premise or SaaS) and its UI, app agents, and machine agents.



AppDynamics Controller and UI

The AppDynamics Controller is the central repository and analytics engine where all performance data is stored, baselined, and analyzed. The Controller is specially designed for large-scale production environments, and can scale to manage hundreds to thousands of application servers.

The AppDynamics Controller can be installed on-premise or it can be accessed as software as a service (SaaS). A SaaS Controller is managed at AppDynamics and you connect to it from a web browser using HTTP/HTTPS. An on-premise Controller is managed by you on your server in a data center or in the cloud.

You access performance data interactively using the Controller UI or programmatically using the AppDynamics REST API.

AppDynamics App Agents

AppDynamics app agents are installed on your JVM or .NET application. They automatically inject instrumentation in application bytecode at runtime.

Patent-pending Dynamic Flow Mapping (TM) technology continuously discovers, maps, and tracks all business transactions, services, and backends in your web application architecture 24x7.

Patent-pending Deep-on-Demand Diagnostics (TM) technology learns code execution behavior for each business transaction. It automatically detects problems and collects deep diagnostics data to troubleshoot them.

AppDynamics Machine Agents

One or more machines (real or virtual) constitute the hardware and operating system on which your application runs. Machines can be instrumented by an AppDynamics machine agent, which collects data about machine performance and sends it to the Controller.

Learn More

- [Logical Model](#)
- [AppDynamics Administration](#)

Logical Model

- [Business Application](#)
- [Tiers](#)
- [Nodes](#)
- [Learn More](#)

This topic describes the basic elements of the AppDynamics model.

Before deploying AppDynamics, also see [Mapping Application Services to the AppDynamics Model](#).

Business Application

An AppDynamics business application models all components or modules in an application environment that provide a complete set of functionality. Think of it as all the web applications, databases, and services that interact or "talk" to each other or to a shared component. When web applications, databases, and services interact, AppDynamics can correlate their activities to provide useful and interesting performance data.

A business application usually does not map directly to only one Java or .NET application, and often it maps to more than one. See [Mapping Application Services to the AppDynamics Model](#).

AppDynamics lets you monitor multiple business applications, though it does not correlate events between them.

Because a single node belongs to a single business application, you can also think of a business application as a kind of namespace for all your nodes. See [Nodes](#).

Business applications contain tiers, and tiers contain nodes.

Tiers

A tier represents a key module in an application environment, such as a website or processing application or a virtual machine. Tiers help you logically organize and manage your business application so that you can scale multiple nodes, partition metrics, define performance thresholds, and respond to anomalies. The metrics from one tier tell a different story than those from another tier; AppDynamics helps you define different policies and processes for each tier. A tier can belong to only one business application.

A tier is composed of one node or a group of nodes. For example, in the Acme sample application the Inventory tier has one node whereas the E-Commerce tier has 2 nodes.

Nodes grouped into a tier may have redundant functionality or may not. An example of a multi-node tier with redundant nodes is when you have a set of clustered application servers or services. An example of a multi-node tier with different nodes is when you have a set of services that do not interact with each other though you want to roll up their performance metrics together.

Keep in mind that an environment can have similar nodes that are used by different applications, so similar nodes should not always belong to the same tier. An example is a complex environment that has two HTTP web servers that serve two separate applications.

Business applications contain tiers. The traffic in a business application flows between tiers. This flow is represented in AppDynamics flow maps along with performance data for the traffic. There is always a tier that is the starting point for a Business Transaction, indicated by a Start label on the flow map.

Nodes

A node is the basic unit of processing that AppDynamics monitors. By definition a node is instrumented by an AppDynamics agent, either an app agent or machine agent or both. App agents are installed on JVMs and Windows .NET applications (IIS, executables, or services). Machine agents are installed on virtual or physical machine operating systems.

Nodes belong to tiers. An app agent node cannot belong to more than one tier. A machine agent cannot belong to more than one tier; however you can install more than one machine agent on a machine.

Learn More

- Mapping Application Services to the AppDynamics Model
- Name Business Applications, Tiers, and Nodes
- Features Overview
- Glossary

Hierarchical Configuration Model

- Entry Point and Exit Point Inheritance
- Node Inheritance
- Switching Configuration Levels
- Learn More

Transaction detection (entry point), backend detection (exit point), and node property configurations are applied on a hierarchical inheritance model. This model provides a default configuration for new tiers as well as the ability to re-use custom configurations in all tiers or tiers that you specify, eliminating the need to configure custom entry and exit points for all tiers.

A tier can inherit all its transaction detection and backend detection configuration from the application, or it can override the application configuration to use a custom configuration.

Similarly, a node can inherit its entire node property configuration from its parent, or it can override the parent configuration to use a custom configuration.

Entry Point and Exit Point Inheritance

By default, tiers inherit the entry point and exit point configurations of the application. You can copy the application-level configuration to specific tiers or explicitly configure all tiers to use the application-level configuration.

The screenshot shows the 'Transaction Detection' tab of the configuration interface. At the top, there are tabs for 'Transaction Detection', 'Backend Detection', 'Error Detection', 'Diagnostic Data Collectors', 'Call Graph Settings', 'JMX', and 'Memory Monitoring'. Below these tabs, a section titled 'Select Application or Tier' shows the 'Acme Online Book Store' application selected. Underneath the application, the 'ECommerce-Server' tier is expanded, showing its three sub-nodes: 'InventoryServer', 'Order-Processing-Server', and 'ECommerce-Server'. The 'ECommerce-Server' node has a green checkmark next to it, indicating it is customized. To the right of the tier list, there are two buttons: 'Java - Transaction Detection' and '.NET - Transaction Detection'. Below the tier list, there is a button labeled 'Copy' and another labeled 'Configure all Tiers to use this Configuration'.

At the tier level, you can specify that the tier should use the application-level configuration.

Or you can override the application-level configuration by creating a custom configuration for the specific tier.

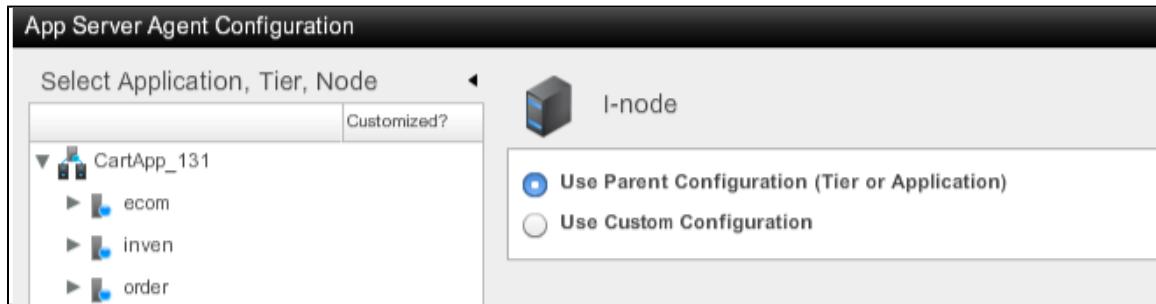
You can configure all tiers to use the custom configuration or copy the configuration for re-use in specific tiers. You can also reset a tier that is currently using a custom configuration to use the application-level configuration.

The screenshot shows the 'Transaction Detection' tab of the configuration interface. At the top, there are tabs for 'Transaction Detection', 'Backend Detection', 'Error Detection', 'Diagnostic Data Collectors', 'Call Graph Settings', 'JMX', and 'Memory Monitoring'. Below these tabs, a section titled 'Select Application or Tier' shows the 'Acme Online Book Store' application selected. Underneath the application, the 'ECommerce-Server' tier is expanded, showing its three sub-nodes: 'InventoryServer', 'Order-Processing-Server', and 'ECommerce-Server'. The 'ECommerce-Server' node has a green checkmark next to it, indicating it is customized. To the right of the tier list, there are three radio buttons: 'Use the Application Configuration' (unchecked), 'Use Custom Configuration for this Tier' (checked), and 'Configure all Tiers to use this Configuration' (unchecked). Below the tier list, there are three buttons: 'Copy', 'Reset to Application Configuration', and 'Configure all Tiers to use this Configuration'.

Node Inheritance

By default a node inherits the node properties of its parent tier (or of the application).

When you configure node properties you can specify that all nodes in a tier inherit the node properties of the parent (tier or application) or that the node should use a custom configuration.



If you create a custom configuration for a node, you can copy that configuration to the application, tier or to another node.

Switching Configuration Levels

If you customize configuration at the tier or node level and then switch back to the application-level configuration, you will not see the old configuration in the UI. However, the old tier or node level configuration is stored, and if you will see these old settings if you switch to the lower-level configuration again.

Learn More

- Configure Backend Detection
- Configure Business Transaction Detection
- App Agent Node Properties

Mapping Application Services to the AppDynamics Model

- Your Application and the AppDynamics Model
- The AppDynamics Business Application
 - One or More Business Applications
 - Tier
 - Node
- Naming Guidelines
 - Learn More

This topic discusses how to map your application environment's services to AppDynamics business applications, tiers, and nodes. For an overview see [Logical Model](#).

Your team may use different terminology to describe your application environment such as "site", "system" or "managed environment". You may have other terms for the services, such as "resource", "module" or "component". For the purposes of this discussion the terms "application environment" and "services" are used.

Your Application and the AppDynamics Model

Your distributed application environment most likely consists of various services, including:

- Web applications served from a JVM, IIS or other platform
- Databases or other data stores
- Services such as message queues and caches

In a large organization, different business units may be responsible for different services. When you finish the model and configure AppDynamics, you can also create specific alerts and dashboards for the different service owners.

Design how your services are mapped and modeled in AppDynamics so that the flow map shows:

- All relevant tiers
- All relevant databases and remote services
- The business transactions flowing through the tiers

AppDynamics models your application environment into a hierarchical set of business applications, tiers, and nodes. The hierarchy is important: a node cannot belong to more than one tier, and a tier cannot belong to more than one business application.

Once you understand the model and configure the AppDynamics business application, tiers, and nodes, you can monitor your application traffic through all its services to rapidly identify existing and potential problems.

The AppDynamics Business Application

An AppDynamics business application models all of your application environment's services that provide a complete set of functionality. Think of it as all the web applications, databases, and services that interact or "talk" to each other or to a shared service. When web applications, databases, and services interact, AppDynamics can correlate their activities to provide useful and interesting performance data. If services exchange information, then they should be monitored in the same business application. You would give them the same business application name when you configure AppDynamics agent properties during installation.

For example, a business application maps to a shopping application environment that is composed of an inventory application, e-commerce frontend application, and databases. All of these services interact with one or more of the others.

One or More Business Applications

There are no hard-and-fast rules for deciding what you want to monitor as a business application for your application environment. Each environment has its own needs. Here are some guidelines:

- AppDynamics lets you monitor multiple business applications; however activities cannot be correlated between business applications. If you want events to correlate make sure the services are in the same business application.
- If all of your services interact, then you can have one big business application.
- If a service doesn't interact with any others, then it can have its own business application.
- If there are multiple sets of services that can be grouped, you can have multiple business applications.

A good example of using multiple business applications is when you have staging, testing, and production environments for the same website. In this case the three business applications are essentially copies of each other.

Tier

Business applications contain tiers. The traffic in a business application flows between tiers. This flow is represented in AppDynamics flow maps along with performance data for the traffic.

An AppDynamics tier represents an instrumented service (such as a web application) or multiple services that perform the exact same functionality, and may even run the same code.

In the AppDynamics model, a tier is composed of one node or multiple redundant nodes. For example the inventory tier may have one node whereas the ecommerce tier may use 3 nodes. Each node in a multi-node tier provides identical functionality, and there is no traffic between nodes.

One example of a multi-node tier is when you have a set of clustered application servers or services.

A difference between a business application and a tier is that there is no interaction between the nodes in a tier. There is interaction between tiers in an application.

Business transactions flow through tiers. This means that there is always a tier that is the starting point for a business transaction. The method call that starts the business transaction is called the entry point.

Node

A node is the basic unit of processing that AppDynamics monitors. By definition a node is instrumented by an AppDynamics agent, either an app agent or machine agent or both. App agents are installed on JVMs and Windows .NET applications (IIS, executables or services). Machine agents are installed on virtual or physical machine operating systems.

Nodes belong to tiers. An app agent node cannot belong to more than one tier. A machine agent cannot belong to more than one tier; however you can install more than one machine agent on a machine.

Naming Guidelines

Use names that are recognizable to the people in your group or company.

For details see [Name Business Applications, Tiers, and Nodes](#).

Learn More

- [Logical Model](#)

Getting Started

- Initial Installation
 - Self-Service Trial or Standard?
 - On-premise or SaaS?
 - Get Started with AppDynamics SaaS
 - Get Started With AppDynamics On-Premise
- Monitoring, Troubleshooting, and Analyzing Application Performance

This section gives you a roadmap to using AppDynamics.

Initial Installation

Self-Service Trial or Standard?

If you are using the self-service trial see [Install Agents for 5 or fewer JVMs or CLRs \(Self-Service Installations\)](#).

If you are using a standard installation see [Install and Upgrade AppDynamics](#).

On-premise or SaaS?

To get started with installing, configuring, and using AppDynamics, first determine whether you will use an on-premise or SaaS Controller.

For information about the different approaches see:

- SaaS and On-Premise Deployment Options
- SaaS Availability and Security
- Differences when using a SaaS Controller

Get Started with AppDynamics SaaS

If you are using or going to use the AppDynamics SaaS Controller, see [Get Started with AppDynamics SaaS](#).

Get Started With AppDynamics On-Premise

If you are going to host your own Controller on premise, see [Get Started With AppDynamics On-Premise](#).

Monitoring, Troubleshooting, and Analyzing Application Performance

To get started using AppDynamics after it is installed see:

- [AppDynamics Essentials](#)

Get Started with AppDynamics SaaS

AppMan Advice



"Deploying APM in the Enterprise... the Path of the Rock Star"

Follow these steps to get started with AppDynamics.

If you are reading a PDF of this document, use your Help Center login to access the documentation at <http://docs.appdynamics.com>.

- Get Your SaaS Account Information from AppDynamics
- Design Your AppDynamics Deployment
- Download the AppDynamics Agents
- Install the AppDynamics Agents
- SaaS Login Credentials
- Connecting Agents to Your SaaS Controller Service
- Access the AppDynamics UI from a Browser
- Review the Dashboards and Flow Maps
- Review Defaults and Configure Business Transactions, if Needed
- Review Defaults and Configure Databases and Remote Services, if Needed
- Review Default Health Rules and Set Up Policies
- Review Default Error Detection
- Explore Additional Data and Metric Features
- Configure Advanced Features
- Start Monitoring and Troubleshooting
- Questions?

Get Your SaaS Account Information from AppDynamics

After signing up for AppDynamics SaaS, you receive a Welcome email containing important account information, including the [Account Owner](#) login. Save this information.

Design Your AppDynamics Deployment

- Learn about [Business Transaction Monitoring](#) and identify which critical business transactions you want to monitor.
- Learn about [End User Experience](#) and decide whether you want to use this feature.
- Learn about how to map your application components to the AppDynamics business application, tier, and node model. See [Logical Model](#) and [Name Business Applications, Tiers, and Nodes](#).
- Based on the model, plan how you will specify AppDynamics application, tier, and node names during installation.
- For Java environments, decide whether you want to use [object instance tracking](#).

Download the AppDynamics Agents

Download the AppDynamics agents from the [Download Center](#). AppDynamics agents collect data from your application servers and other monitored systems and report to the Controller. Select the agents that are appropriate for your environment:

- Java Agent
- .NET Agent
- Machine Agent

For details see [Download AppDynamics Software](#).

Install the AppDynamics Agents

Install agents on the application servers you want to instrument and any other machines you want to monitor. Follow the [instructions to install the AppDynamics Agents](#).

SaaS Login Credentials

SaaS Controller login credentials are included in the welcome email from AppDynamics.

To add additional login accounts contact the [AppDynamics Support Team](#).

The SaaS Controller login is an Account Administrator credential. The Account Administrator can create other users for the account. See [Account Administrator](#).

Connecting Agents to Your SaaS Controller Service

For agents to successfully connect to the Controller, configure the Controller host and port information using either the controller-info.xml file or the system properties of your JVM startup script.

To use HTTPS communication, enable SSL by setting the <controller-ssl-enabled> agent configuration property to "True". For details see [App Agent for Java Configuration Properties](#), [App Agent for .NET Configuration Properties](#), and [Machine Agent Configuration Properties](#).

- The default ports for the SaaS Controller service are:
 - Port 80 for HTTP
 - Port 443 for HTTPS

If you need to specifically open up the communication ports (80 or 443) for the AppDynamics SaaS Controller IP address please request the IPs from the [AppDynamics Support Team](#).

Access the AppDynamics UI from a Browser

Once you have installed the agents, launch your web browser and connect to the AppDynamics User Interface (UI). For SaaS, the URL includes the account name from the Welcome email:

`http://<account-name>.saas.appdynamics.com/controller`

When using SSL, use port 443 or https to access the Controller.

Review the Dashboards and Flow Maps

AppDynamics automatically discovers the [Business Transactions](#) in your application environment. Browse the [Application Dashboard](#) and see the [Flow Maps](#) to visualize your application. You can resize and move icons around on the flow maps.

Review Defaults and Configure Business Transactions, if Needed

The default configurations may need to be further customized for your environment. For example, AppDynamics may have discovered transactions that you want to group together or even exclude, because you want to concentrate on the most important transactions. There may be business transactions that are not yet discovered for which you need to configure detection rules. See:

- [Monitor Business Transactions](#)
- [Configure Business Transaction Detection](#)

Review Defaults and Configure Databases and Remote Services, if Needed

AppDynamics automatically discovers "backends" such as databases, message queues, etc. by following calls in the Java or .NET code. Look at the [databases](#) and [remote services](#) dashboards to make sure all necessary backends are revealed. If needed, configure [how backends are detected](#).

Review Default Health Rules and Set Up Policies

AppDynamics provides default [health rules](#) that define performance parameters for business transactions, such as the conditions that indicate a slow transaction, or when too much memory is being used. You can adjust the thresholds that define when a health rules is

violated, create new health rules, and set up policies to specify actions to automate when health rules are violated.

Review Default Error Detection

AppDynamics detects errors and exceptions. You can review and, if needed, modify the error detection rules. For example, some errors you may want to ignore.

Explore Additional Data and Metric Features

Explore these features to gain more insight into application performance:

- Data Collectors
- Business Metrics
- (for Java environments) JMX Metrics
- Machine Agent Custom Metrics

Configure Advanced Features

Additional features you may want to use include:

- End User Monitoring
- Custom Dashboards
- Automation
- System Integrations

Start Monitoring and Troubleshooting

Start getting the benefits of AppDynamics! See:

- [AppDynamics in Action Videos](#)
- [AppDynamics Features](#)

Questions?

For questions about using AppDynamics contact the [AppDynamics Support Team](#).

Use a SaaS Controller

- Your SaaS Controller URL
- Login Credentials
- Connecting Agents to Your SaaS Controller Service
- SMTP Service for SaaS
- SaaS Limitations
- Contact Support

If you are using the SaaS service for the AppDynamics Controller, simply open a web browser at the URL of the AppDynamics UI and log in with your AppDynamics credentials.

Your SaaS Controller URL

Your SaaS Controller URL is included in the welcome email from AppDynamics.

The URL is of the following form:

`http(s)://<customer>.saas.appdynamics.com/controller`

Login Credentials

Login credentials are included in the welcome email from AppDynamics.

To add additional login accounts contact the [AppDynamics Support Team](#).

Connecting Agents to Your SaaS Controller Service

For agents to successfully connect to the Controller, configure the Controller host and port information using either the controller-info.xml file or the system properties of your JVM startup script.

To use HTTPS communication, enable SSL by setting the <controller-ssl-enabled> agent configuration property to "True". For details see [App Agent for Java Configuration Properties](#), [App Agent for .NET Configuration Properties](#), and [Machine Agent Configuration Properties](#).

- The default ports for the SaaS Controller service are:
 - Port 80 for HTTP
 - Port 443 for HTTPS

 **Important:** If you need to specifically open up the communication ports (80 or 443) for the AppDynamics SaaS Controller IP address please request the IPs from the [AppDynamics Support Team](#).

SMTP Service for SaaS

To enable email and SMS notifications you must configure SMTP. See [Configure the SMTP Server](#).

For SaaS users, AppDynamics has an SMTP service running on every machine.

The configuration is:

host: localhost
port: 25

No authentication is needed.

SaaS Limitations

Compared to an on-premise installation, the following are limitations in the SaaS environment:

- Flow maps show a maximum of the last 60 minutes of data, regardless of whether the Time Range is set to a larger range. Other graphs will display according to the selected Time Range.
- There is a minimum time lag of 4 minutes before data is displayed in the UI.
- Custom action scripts are not supported.
- LDAP integration is limited.
- The business transaction limit is 200 per business application, unless you are using a dedicated Controller.
- Configuration settings related to data retention is limited.

Contact AppDynamics Support for details.

See also <http://www.appdynamics.com/products-saas-on-premise.php>.

Contact Support

For questions about the service contact the [AppDynamics Support Team](#).

SaaS Availability and Security

- Service Availability
- Customer Account Login Security
- Hosting
- Data Access
- Data Collection
- Data Communication

This topic summarizes the service availability and security AppDynamics provides for customers who use the AppDynamics SaaS platform.

Service Availability

AppDynamics makes every best effort to operate and manage the AppDynamics SaaS platform with a goal of 99.5% uptime Service Level Agreement (SLA), excluding planned maintenance windows. AppDynamics actively monitors the latency of the SaaS platform 24/7 from different locations around the world to ensure AppDynamics delivers the best quality of service.

Customer Account Login Security

The AppDynamics user interface (UI) uses TLS 1.0 with AES 256 bit encryption terminated at the server to ensure end-to-end security over the wire. For additional security, AppDynamics can restrict UI access to customer corporate networks.

Hosting

The AppDynamics SaaS platform (servers, infrastructure and storage) is hosted in one of the largest Tier III data centers in North America. The data center is designed and constructed to deliver world-class physical security, power availability, infrastructure flexibility, and growth capacity. The data center provider is SSAE 16 SOC 1 Type II compliant, which means that it has been fully independently audited to verify the validity and functionality of its control activities and processes.

Every server is operated in a fully redundant fail-over pair to ensure high availability. Data is backed up nightly, stored redundantly and can be restored rapidly in case of failure. AppDynamics also provides an off-site backup service that is available at additional cost.

Security updates and patches are actively evaluated by engineers and are deployed based upon the security risks and stability benefits they offer to the AppDynamics SaaS platform and customers.

Data Access

Access to the AppDynamics SaaS platform infrastructure and data is secured by multiple authentication challenges including RSA and DSA key pairs, passwords, and network access control lists. Infrastructure and data access is restricted to AppDynamics employees and contractors, all of whom are under strict confidentiality agreements.

System and Network activity is actively monitored by a team of engineers 24/7. Failed authentication attempts are audited and engineers are paged immediately so that any possible intrusion or threat can be investigated promptly. Standard firewall policies are deployed to block all access except to ports required for AppDynamics SaaS platform and agent communication.

Data Collection

AppDynamics agents collect metrics that relate to the performance, health and resources of an application, its components (transactions, code libraries) and related infrastructure (nodes, tiers) that service those components.

Data Communication

AppDynamics agents typically push data using one-way HTTP or HTTPS connections to a single host (a Controller) which has been allocated to one or more customer accounts. AppDynamics offers dedicated Controllers for customers who require their data to be isolated.

For added security, agents can be configured to send data using encrypted transmission by simply selecting HTTPS port 443 and setting "controller-ssl-enabled" to true in the agent configuration. AppDynamics agents also have built-in support for outbound HTTP proxies for customers using these security mechanisms.

A single agent with the default configuration will typically push between 300KB to 500KB of data per minute depending on application characteristics. AppDynamics uses random staggering on agent data communication to the AppDynamics SaaS platform so traffic is spread evenly to minimize bursts and spikes of network traffic from your data center to the AppDynamics SaaS platform.

# of Agents	Typical Network Bandwidth Used (per min)
1	300KB to 500KB
100	4Mbit to 6.4Mbit
1000	40Mbit to 64Mbit

These figures assume a 1:1 relationship between an agent and a JVM/CLR.

Get Started With AppDynamics On-Premise

AppMan Advice



"Deploying APM in the Enterprise... the Path of the Rock Star"

Follow these steps to get started with AppDynamics.

If you are reading a PDF of this document, use your Help Center login to access additional documentation at <http://docs.appdynamics.com>.

- Design Your AppDynamics Deployment
- Size and Verify the Controller Environment
- Download AppDynamics
- Install the AppDynamics Controller
- Install the AppDynamics Agents
- Access the AppDynamics UI from a Browser
- Review the Dashboards and Flow Maps
- Review Defaults and Configure Business Transactions, if Needed
- Review Defaults and Configure Databases and Remote Services, if Needed
- Review Default Health Rules and Set Up Policies
- Review Default Error Detection
- Explore Additional Data and Metric Features
- Configure Advanced Features
- Start Monitoring and Troubleshooting

Design Your AppDynamics Deployment

- Learn about [Business Transaction Monitoring](#) and identify which critical business transactions you want to monitor.
- Learn about [End User Experience](#) and decide whether you want to use this feature.
- Learn about how to map your application components to the AppDynamics business application, tier, and node model. See [Logical Model](#) and [Name Business Applications, Tiers, and Nodes](#).
- Based on the model, plan how you will specify AppDynamics application, tier, and node names during installation.
- For Java environments, decide whether you want to use object instance tracking.

Size and Verify the Controller Environment

- Verify that you have the resources to support system requirements and the Controller performance profile. The profile reflects the number of nodes and AppDynamics applications that the Controller will monitor. For details see [Controller System Requirements](#).

Download AppDynamics

- Download the AppDynamics software components from the [Download Center](#). For details see [Download AppDynamics Software](#).

Install the AppDynamics Controller

The AppDynamics Controller is the central management server where all data is stored and analyzed. All AppDynamics Agents connect to the Controller to report data, and the Controller provides a browser-based user interface for monitoring and troubleshooting.

application performance. A wizard installs the Controller in just a few minutes. Install the AppDynamics Controller only if you are using the on-premise Controller deployment option.

- Follow the [instructions to install an on-premise Controller](#).
- Important installation and configuration considerations include:
 - High Availability
 - Backups
 - SSL and Certificates
 - User Authentication with LDAP or SAML

Install the AppDynamics Agents

AppDynamics Agents collect data from your application servers and other monitored systems and report to the Controller. Install them on the application servers you want to instrument and any other machines you want to monitor. Follow the [instructions to install the AppDynamics Agents](#).

Access the AppDynamics UI from a Browser

Once you have installed the Controller and agents, launch your web browser and connect to the AppDynamics User Interface (UI).

- For an on-premise Controller, the URL pattern is:

`http://<controller-host>:<controller-port>/controller`

When using SSL, use port 443 or https to access the Controller.

Review the Dashboards and Flow Maps

AppDynamics automatically discovers the [Business Transactions](#) in your application environment. Browse the [Application Dashboard](#) and see the [Flow Maps](#) to visualize your application. You can resize and move icons around on the flow maps.

Review Defaults and Configure Business Transactions, if Needed

The default configurations may need to be further customized for your environment. For example, AppDynamics may have discovered transactions that you want to group together or even exclude, because you want to concentrate on the most important transactions. There may be business transactions that are not yet discovered for which you need to configure detection rules. See:

- [Business Transaction Monitoring](#)
- [Configure Business Transaction Detection](#)

Review Defaults and Configure Databases and Remote Services, if Needed

AppDynamics automatically discovers "backends" such as databases, message queues, etc. by following calls in the Java or .NET code. Look at the [databases](#) and [remote services](#) dashboards to make sure all necessary backends are revealed. If needed, [configure how backends are detected](#).

Review Default Health Rules and Set Up Policies

AppDynamics provides default [health rules](#) that define performance parameters for business transactions, such as the conditions that indicate a slow transaction, or when too much memory is being used. You can adjust the thresholds that define when a health rule is violated, create new health rules, and set up policies to specify actions to automate when health rules are violated.

Review Default Error Detection

AppDynamics detects errors and exceptions. You can review and, if needed, modify the [error detection rules](#). For example, some errors you may want to ignore.

Explore Additional Data and Metric Features

Explore these features to gain more insight into application performance:

- [Data Collectors](#)
- [Business Metrics](#)

- (for Java environments) JMX Metrics
- Machine Agent Custom Metrics

Configure Advanced Features

Additional features you may want to use include:

- End User Monitoring
- Custom Dashboards
- Automation
- System Integrations

Start Monitoring and Troubleshooting

Start getting the benefits of AppDynamics! See:

- [AppDynamics in Action Videos](#)
- [AppDynamics Features](#)

Download AppDynamics Software

- Accessing the AppDynamics Download Center
 - Download Tips
- AppDynamics Software Components
- Access to Older Versions
- Downloading from the Linux Shell
- Learn More

Accessing the AppDynamics Download Center

You should have received a Welcome email from AppDynamics. The Welcome email contains credentials for you to log in to the AppDynamics Support Center.

If you have not received this Welcome email, contact your AppDynamics Sales Representative or email support@appdynamics.com.

Access the [AppDynamics Download Center](http://download.appdynamics.com) (<http://download.appdynamics.com>) and browse to the appropriate section on the Download Center to download the relevant files.

Download Tips

Always copy or transfer the downloaded files in binary mode.

If you have downloaded a binary on Windows, and you are moving it to a Unix environment, the transfer program must use binary mode.

 For each file you download, verify that the download is complete and that the file is not corrupted. Run a checksum tool and compare the results against the checksum information on the download site.

AppDynamics Software Components

AppDynamics Software Component	Description	SaaS	On-Premise	Files
Controller	Central management server where all data is stored and analyzed.	N/A	Required	
Java App Server Agent	Instrumentation Agent for Java virtual machines. This component must be installed on each Java application server you want to instrument through AppDynamics.	Required for Java	Required for Java	<ul style="list-style-type: none"> • For Sun and JRockit JVMs: AppServerAgent-x.x.x.zip • For IBM JVMs: AppServerAgent-ibm-x.x.x.zip

.NET App Server Agent (includes a Machine Agent by default)	Instrumentation Agent for .NET Common Language Runtime (CLR). This component must be installed on those worker processes that you want to instrument through AppDynamics.	Required for .NET	Required for .NET	<ul style="list-style-type: none"> For Windows 32-bit: dotNetAgentSetup.msi For Windows 64-bit: dotNetAgentSetup64.msi If you are using 32-bit Application on a 64-bit machine, also install the "32-bit add-on for 64-bit" binary.
Machine Agent	Collects hardware performance metrics and can be installed on any machine in your environment. The Machine Agent can be extended to collect data from other subsystems.	Optional	Optional	MachineAgent-x.x.x.xGA.zip

Access to Older Versions

The [AppDynamics Download Center](#) provides downloads of older versions of the products.

- For On-Premise installations: Go to "AD Pro-OnPremise".
- For SaaS installations: Go to "AD Pro-SaaS".

On the top-right corner, click on the drop-down list to select the version that you want to download.

Downloading from the Linux Shell

To download AppDynamics software from a Linux shell, use the wget utility.

Use the following parameters for wget:

```
wget --content-disposition '<URL_TO_FILE>?username=<USERNAME>&password=<PASSWORD>'
```

Ensure that the URL enclosed by single quotes, for example:

```
wget --content-disposition
'http://download.appdynamics.com/onpremise/public/AppDynamics_GA/latest/controller_64bit_linux_v3'
```

The --content-disposition option saves a file with a proper name that does not have the query string attached to the file name. On some Linux environments, wget may not have the content-disposition option. In this case, use following parameters for wget:

```
wget
'[http://download.appdynamics.com/onpremise/public/AppDynamics_GA/latest/controller_64bit_linux_v3']'
```

Learn More

- [Supported Environments and Versions](#)
- [Agent - Controller Compatibility Matrix](#)

Quick Start for DevOps

Get Started

[Get Started on SaaS](#)

[Get Started On-Premise](#)

[Features Overview](#)

Tutorials for Java

Use AppDynamics for the First Time with Java
Understanding Events
Understanding Flow Maps
Understanding Slow Transactions
Understanding the Transaction Scorecard
Understanding Server Health
Understanding Exceptions

Learn More

Best Practices for Application Developers
Best Practices for Performance and Quality Assurance Engineers
Best Practices for Operations Professionals

Monitor Your Applications

Business Transaction Monitoring
Monitor End User Experience (EUM)
Monitor Events
Monitor Application Change Events
Background Task Monitoring
Backend Monitoring
Infrastructure Monitoring
Monitor CLRs
Monitor Hardware

Troubleshoot Application Performance

Troubleshoot Slow Response Times
Troubleshoot Health Rule Violations
Transaction Snapshots
Troubleshoot Node Problems
Troubleshoot Errors
Diagnostic Sessions
Troubleshoot Java Memory Issues

Quick Start for Architects

Get Started

Supported Environments and Versions
Get Started on SaaS
Get Started On-Premise

Concepts

Features Overview
Architecture
Logical Model
Mapping Application Services to the AppDynamics Model
Behavior Learning and Anomaly Detection
Thresholds
Glossary

Basic Configuration

Configure Business Transaction Detection
Configure Policies
Configure Baselines
Configure Thresholds
Configure Error Detection

Configure End User Experience
Alerting Wizard
Custom Dashboards

Analyze

Business Metrics
Infrastructure Metrics
Reports
Compare Releases

Learn More

Advanced Configuration

Hierarchical Configuration Model
Configure Data Collectors
Configure Code Metric Information Points
Configure Backend Detection
Configure Call Graphs
Configure Background Tasks
Configure Stale Backend Removal
Configure Custom Memory Structures (Java)
Configure JMX Metrics from MBeans
Configure Multi-Threaded Transactions (Java)
Configure Object Instance Tracking (Java)
Configure Transaction Snapshots
Configure Memory Monitoring (Java)
Internationalization
Integrate using Custom Action Scripts
Export and Import Business Application Configurations

Integration

System Integrations
Use the AppDynamics REST API

Automation

Workflow Automation
Workflow for Cloud Orchestration

Quick Start for Administrators

Get Started

Basic Components
Get Started on SaaS
Get Started On-Premise
Basic Concepts

Basic Administration

Release Notes for AppDynamics Pro
Supported Environments and Versions
Install and Upgrade AppDynamics
Name Business Applications, Tiers, and Nodes

Learn More

Advanced Administration

Implement SSL
Best Practices for Failover Scenarios

Administer Agents
Administer the Controller
Configure Authentication, User Permissions and Integrations
AppDynamics for Large Enterprises

Quick Start for Operators

Get Started

[AppDynamics Essentials](#)

Tutorials for Java

[Use AppDynamics for the First Time with Java](#)
[Understanding Events](#)
[Understanding Flow Maps](#)
[Understanding Slow Transactions](#)
[Understanding the Transaction Scorecard](#)
[Understanding Server Health](#)
[Understanding Exceptions](#)

Learn More

[Best Practices for Operations Professionals](#)
[Set User Preferences](#)
[Custom Dashboards](#)

Set User Preferences

- [Accessing My Preferences](#)
 - To user preferences
 - Account Information
 - Advanced Features
 - View Preferences
 - To set Help Popups
 - To change the navigation panel color scheme
 - To change the Metric Browser graph color scheme
 - To change the Dashboard graph color scheme
 - To specify the minimal number of backends to display
 - To set the date format
 - To use the 24 hour time format
 - To enable debug mode
 - To display server call data
 - To enable server call logging

This topic describes how users can change their password and set various preferences in the My Settings view.

Accessing My Preferences

To user preferences

1. Click the Setup menu in the upper right section of the screen:
2. From the drop-down menu, click **My Preferences**.

The My Preferences screen opens.

Account Information

In the My Account pane, you can update your user name and password.

Advanced Features

If you want to access AppDynamics cloud automation features, such as workflows, check the Show Cloud Auto-Scaling features

checkbox. The Automate menu appears in the left navigation pane.

Automation features are triggered by policy actions.

See [Workflow Automation](#) for information about automation. See [Policies](#) for information about policies.

View Preferences

The View Preferences pane lets you configure various preferences in the user interface.

To set Help Popups

By default Help popups are enabled. Not every screen has these but those that do will display the tips every time you open the screen. When that happens, on a per-screen basis you can check Don't Show Again. If you've checked Don't Show Again for many screens and want to re-enable them, check Enable Help Pop-ups and click **Reset All**.

To disable Help popups for the whole UI at all times, clear **Enable Help Pop-ups** on the View Preferences pane.

To change the navigation panel color scheme

Select either Light or Dark from the Navigation Panel Color Scheme pulldown.

To change the Metric Browser graph color scheme

Select either Light or Dark from the Metric Browser Graph Color Scheme pulldown.

To change the Dashboard graph color scheme

Select either Dark or Light from the Dashboard Graph Color Scheme pulldown.

To specify the minimal number of backends to display

Enter a number in the Maximum number of backends field. The default is 20.

To set the date format

Select a format from the **Date Format** pulldown menu.

To use the 24 hour time format

Click the **Use 24 Hour Format** check box.

To enable debug mode

Click the **Enable Debug Mode** check box.

To display server call data

Click the **Display Server call data in the lower right of the screen** check box.

To enable server call logging

Click the **Enable Advanced Server Call Logging** check box.

 Warning: Enabling call logging can slow down the CPU, especially if there are large datasets.

Glossary

- Action
- Agent
- Alert
- Alert Digest

- Anomaly Detection
- Ajax Request
- Application Performance Management
- APM
- App Server
- App Server Agent
- Application
- Application Dashboard
- Backend
- Background Task
- Baseline
- Baseline Deviation
- Baseline Pattern
- BCI
- Browser Snapshot
- Business Application
- Business Transaction
- Bytecode Injection
- Call Graph
- Compute Cloud
- Controller
- Detection
- Diagnostic Session
- Discovery
- Distributed Application
- Entry Point
- Error
- Error Transaction
- Exception
- Event
- Exit Point
- Flow Map
- Health
- Health Rule
- Health Rule Violation
- High Availability (HA) Cluster
- Histogram
- iFrame
- Information Point
- Key Performance Indicator
- KPI
- Machine
- Machine Agent
- Managed Application
- Match Condition
- Node
- Pageview
- Policy
- Remote Service
- Request
- Scorecard
- Task
- Tier
- Tag and follow
- Threshold
- Trace, tracing
- Transaction Snapshot
- Workflow

Action

An action automatically responds to an event, usually without operator interaction. There are various types of actions which can be notification, diagnostic, or remedial.

Agent

In AppDynamics an agent collects data about an application (and optionally machine) performance. AppDynamics has application server agents (app agents) and machine agents. A Java agent intercepts the bytecode loading at the classloader and enhances it before it is loaded in the JVM. See also Bytecode Injection.

Alert

An alert notifies a recipient list of a problem or event; can be email, SMS or customized to interface with external notification systems.

Alert Digest

An alert digest compiles alerts and sends the compilation sent by email or SMS to a recipient list at a configured time interval.

Anomaly Detection

Anomaly detection refers to the identification of metrics whose values are out of the normal range, where normal range is based on dynamic baselines that AppDynamics has observed for these metrics.

Ajax Request

An Ajax request is a request for data sent from a page using Asynchronous JavaScript and HTML (Ajax). The Ajax request is tracked as a child page using EUM.

Application Performance Management

Application performance management monitors and manages the performance and availability of software applications in a production environment, focusing on relating IT metrics to business values.

According to Gartner research, application performance management includes: end user experience monitoring, application runtime architecture discovery and modeling, business transaction management, application component deep-dive monitoring, and application data analytics.

APM

See Application Performance Management.

App Server

An app server or application server provides software applications with services such as security, data services, transaction support, load balancing, and management of large distributed systems. Examples are a Java Virtual machine (JVM) or a Common Language Runtime (CLR).

App Server Agent

An app server agent monitors an application server. A Java app server agent monitors a JVM. A .NET app server agent monitors a CLR. See also Node.

Application

See Business Application.

Application Dashboard

The application dashboard graphically represents high-level structural and status information for a single business application. The application dashboard includes a flow map, grid view, summary of key performance indicators.

Backend

A backend or backend node is not instrumented directly, but you can monitor the flow of traffic to it. Typical examples are a database or messaging service.

Background Task

A background task or a batch job is a scheduled program that runs without user intervention.

Baseline

A baseline provides a defined known point of reference. The baseline is established by configuration or by observing current performance. Baselines can be static or dynamic.

Baseline Deviation

Baseline deviation represents the degree of deviation from the baseline at a point in time as an integer value. Baseline deviation can be used to configure health rule conditions based on the degree of deviation. For example, you can configure a warning condition as 2 deviations from the baseline and a critical condition as 4 deviations from baseline.

Baseline Pattern

A baseline pattern defines how a dynamic baseline is calculated. Time period baselines use a fixed range of time ("1/1/2011 - 1/31/2011"). Trend baselines use a rolling time range (last 3 months, last year).

BCI

See Bytecode Injection.

Browser Snapshot

A browser snapshot presents set of diagnostic data for an individual base page, iFrame or Ajax request. The data reports the end-user's experience starting with the Web browser. Browser snapshots are taken at a certain point in time.

Business Application

A business application serves some major business functionality. In AppDynamics a business application does not necessarily map one-to-one to a single Java/J2EE application (WAR / EAR) or to a single .NET application (IIS/ASP.NET).

Business Transaction

A business transaction represents an aggregation of similar user requests to accomplish a logical user activity. Examples of these activities include: logging in, searching for items, adding items to the cart, checking out (e-commerce); content sections that users navigate such as sports, world news, entertainment (content portal); viewing a quote, buying and selling stocks, placing a watch (brokerage). A single request is a business transaction instance.

Bytecode Injection

Bytecode injection modifies a compiled class at runtime by injecting code into it immediately before it is loaded and run.

Call Graph

A call graph represents the calling relationships among subroutines in an application. Part of a transaction snapshot that is used to identify root cause of a performance problem.

Compute Cloud

A compute cloud delivers computing and storage capacity as a service. Examples are Amazon EC2, OpenStack, etc.

Controller

The Controller collects stores, baselines and analyzes performance data collected by agents. Can be installed on-premise or using the AppDynamics SaaS model.

Detection

Detection is the process by which AppDynamics identifies a business transaction or backend in a managed application. Detection is also referred to as discovery.

Diagnostic Session

A diagnostic session is a session in which transaction snapshots are captured, with full call graphs. A diagnostic session can be started manually through the user interface or configured to start automatically when thresholds for slow or error transactions are reached.

Discovery

See Detection.

Distributed Application

A distributed application runs on multiple computers in a network. Some of the computers can be virtual machines.

Entry Point

An entry point begins a business transaction. Typically an entry point is a method or operation. Auto-detected for most common frameworks but configurable for customized business transaction detection.

Error

An error codifies a departure from the expected behavior of a business transaction, which prevents the transaction from working properly.

Error Transaction

An error transaction is an error that occurred during transaction execution. An error transaction can be a logged error or a thrown exception.

Exception

An exception is a code-logged message outside the context of a business transaction.

Event

An event represents a change in application state. There are different event types.

Exit Point

An exit point initiates an external call from an app server to a database, remote service or to another app server. An exit point is auto-detected at startup and is configurable for customized backend detection.

Flow Map

A flow map graphically represents the tiers and backends and the flows between them in a managed application.

Health

A visual summary of the extent to which a business transaction or server is experiencing critical and warning health rule violations.

Health Rule

A health rule defines a condition or set of conditions in terms of metrics. The condition compares the performance metrics that AppDynamics collects with some static or dynamic threshold that you define. If performance exceeds the threshold, a health rule violation event is triggered.

Health Rule Violation

A health rule violation exists if the conditions of a health rule are true.

High Availability (HA) Cluster

A high availability cluster of computers hosts server applications with the purpose of reducing down time. The HA cluster, also called a failover cluster, is enabled by redundant systems that guarantee continued delivery of service during system failure.

Histogram

A histogram graphically represents a frequency distribution using rectangles whose widths represent class intervals and whose areas are proportional to the corresponding frequencies.

iFrame

An iFrame is an HTML element embedded in another HTML element and is tracked as a child page using EUM.

Information Point

An information point instruments a method in application code that is outside the context of a business transaction. Used for monitoring the performance of the method itself or for capturing data from the method's return value or parameters.

Key Performance Indicator

Key performance indicators are main metrics that an organization uses to measure its success. In AppDynamics, the key performance indicators are assumed to be load (number of calls and calls per minute), average response time, and errors (number of errors and errors per minute.)

KPI

See Key Performance Indicator.

Machine

A machine consists of hardware and an operating system. It hosts application services. Can be virtual.

Machine Agent

A machine agent instruments a machine to report data about hardware and the network to the Controller.

Managed Application

A managed application is an application with servers that are instrumented by AppDynamics.

Match Condition

A match condition frames a test consisting of a match criterion (such as a method name, servlet name, URI, parameter, hostname, etc.), a comparison operator typically selected from a drop-down list, and a value. Used in many types of AppDynamics configuration to specify entities to be included in or excluded from monitoring.

Node

A node is an instrumented Java application server or an instrumented Windows .NET application (IIS, executable, or service.) Instrumentation is accomplished by installing an AppDynamics App Agent. Nodes belong to tiers. See Tier.

Pageview

A pageview is an instance of a web page being loaded into a Web browser.

Policy

A policy consists of a trigger based on events and an action respond to the event. A policy provides a mechanism for automating monitoring, alerting, and problem remediation.

Remote Service

A remote service provides a service to a distributed application outside of the JVM or CLR. Examples are a Java Message Service or Web Service. See Backend.

Request

A request is a single instance of a business transaction; for example, 500 requests per minute for the "Checkout" business transaction.

Scorecard

A visual summary of the performance of a business transaction within a specified time range, covering percentage of instances that are normal slow, very slow, stalled or errors.

Task

A task codifies a unit of work as a set of instructions. Component of a step in a workflow

Tier

A tier represents a key service in an application environment, such as a website or processing application. A tier is composed of one or more nodes. See Node.

Tag and follow

Tag and follow is a methodology used for tracing code execution.

Threshold

A threshold is a configurable boundary of acceptable or normal business transaction or background task performance.

Trace, tracing

Tracing is following the execution of software code and recording information about the execution, usually for debugging or performance monitoring purposes.

Transaction Snapshot

A transaction snapshot depicts a set of diagnostic data for an individual business transaction across all app servers through which the business transaction has passed. The data reports the user's experience starting with the application server. A transaction snapshot is taken at a specific point in time.

Workflow

A workflow builds a sequence of steps in which each step consists of one or more tasks that are executed on a machine instrumented by a machine agent.

AppDynamics Support

If you have questions about using AppDynamics please [file a support ticket](http://help.appdynamics.com/tickets/new) (<http://help.appdynamics.com/tickets/new>) and attach any logs that are related to your issue.

To get log files from the Controller, see [Log Files for Troubleshooting](#).

To get heap, histogram, and thread dumps see [Controller Dump Files](#).

If you want the Support team to remotely monitor your Controller, see [Configure Remote Monitoring of an On-Premise Controller](#).

Controller Dump Files

- Controller Troubleshooting Information Needed by the AppDynamics Support Team
 - To get the heap and histogram dump files
 - To take four thread dumps at three second intervals
 - Provide the AppDynamics Support Team with the files

Controller Troubleshooting Information Needed by the AppDynamics Support Team

If requested, collect the following files to send to the AppDynamics Support Team.

To get the heap and histogram dump files

- Get the **process id of the Controller** to use in subsequent commands.

```
ps -ef | grep java
```

- Get the **heap dump before garbage collection** using following command:

```
<java-jdk-install-dir>/bin/jmap -dump:format=b,file=heap_before_live.bin  
<Controller_pid>
```

- Get the **histogram before garbage collection** using following command:

```
<java-jdk-install-dir>/bin/jmap -histo <Controller_pid> | head -200 >  
histo_before_live.txt
```

- Get the **histogram before garbage collection** using following command:

```
<java-jdk-install-dir>/bin/jmap -histo:live <Controller_pid> | head -200 >  
histo_after_live.txt
```

2. Save the generated files:

- heap_before_live.bin
- histo_before_live.txt
- histo_after_live.txt

To take four thread dumps at three second intervals

- Using the Controller process ID, execute following command:

```
kill -3 <Controller_pid>
```

- Save the <Controller_Installation_Directory>/appserver/domains/domain1/logs/jvm.log file.

Provide the AppDynamics Support Team with the files

- heap_before_live.bin
- histo_before_live.txt
- histo_after_live.txt
- jvm.log

Log Files for Troubleshooting

- [Log Files that Record Possible Problems](#)

Use installer.log and the files in the Controller logs directory to troubleshoot Controller errors.

Log Files that Record Possible Problems

The following log files may reveal errors:

- **installer.log:**
This file contains information about events of the install process such as extraction, preparation and other post-processing tasks. It is located at <Controller_Installation_directory>.
- **server.log:**
This file contains log information for the embedded Glassfish application server used by the Controller. It is located at <Controller_Installation_directory>/logs.
- **database.log:**
This file contains log information for the MySQL database that is used by the Controller. It is located at <Controller_Installation_directory>/logs.
- **installation.log**
This file contains log information about the installation. It is located at <Controller_Installation_directory>/.install4j.

If after analyzing the logs you have questions, please [file a support ticket](#) and attach any logs that are related to your issue. See [AppDynamics Support](#) for suggestions.

Configure Remote Monitoring of an On-Premise Controller

- Prerequisites

- Configuring Remote Monitoring
 - Make a formal request to AppDynamics Support
 - Shut down your Controller and its application server
 - Modify the Controller configuration
 - Create a new controller-info.xml file for the application agent
 - Sample controller-info.xml file
 - Restart the Controller's application server
 - Install the Machine Agent for the Controller
 - Monitor the performance of the Controller
- Learn More

This topic describes how to configure your Controller so that it can be monitored remotely for optimization purposes.

Prerequisites

Install the AppDynamics Machine Agent on the machine where your Controller is installed. See [Install the Machine Agent](#).

Configuring Remote Monitoring

Follow these instructions to configure an on-premise Controller so that you can remotely monitor your AppDynamics environment.

Make a formal request to AppDynamics Support

To help you optimize your Controller, AppDynamics Support will set up an account for you on our monitoring system.

You will receive the following information from the AppDynamics Support team:

- Account name
- Account key
- Application name

 **Important:** Wait for the response from the AppDynamics Support before proceeding.

Shut down your Controller and its application server

1. Shut down your Controller. Open a command line console and execute following command:

- For Linux:

```
./controller.sh stop
```

- For Windows:

```
controller.bat stop
```

2. Shut down the Controller's application server. Open a console and navigate to the <Controller_Install_Directory>/bin directory and execute following command:

For Linux:

```
./controller.sh stop-appserver
```

For Windows:

```
controller.bat stop-appserver
```

Modify the Controller configuration

1. Open the <Controller_Installation_Directory>/appserver/domains/domain1/conf/domain.xml file, the configuration file for the Controller's application server.

2. Update the values of the following JVM options as shown:

```
-Dappdynamics.controller.hostName=saas-monitor.saas.appdynamics.com  
-Dappdynamics.controller.port=80
```

If SSL is required add:

```
-Dappdynamics.controller.ssl.enabled=true
```

See [Controller SSL and Certificates](#).

Create a new controller-info.xml file for the application agent

This step requires the account name and key sent to you by the AppDynamics Support Team.

1. Create a new controller-info.xml file. Refer to the sample controller-info.xml file shown below.
If SSL is required set controller.ssl.enabled to "true".

2. Add the new controller-info.xml file to the <Controller_Installation_Directory>/appserver/domains/domain1/appagent/conf folder.

3. Set the account name to the account name supplied by AppDynamics Support.

4. Set the application name to the application name supplied by AppDynamics Support.

Optional: Modify the node name to the name of the machine hosting the Controller.

 **Important:** The tier name must be "App Server". Do not change this value.

Sample controller-info.xml file

```
<?xml version="1.0" encoding="UTF-8"?>  
<controller-info>  
    <controller-host>saas-monitor.saas.appdynamics.com</controller-host>  
    <controller-port>80</controller-port>  
    <controller-ssl-enabled>false</controller-ssl-enabled>  
    <disable-virtualization-resolvers>true</disable-virtualization-resolvers>  
    <account-name>testing</account-name>  
    <account-access-key>BlueCust!20</account-access-key>  
    <application-name>Controller</application-name>  
    <tier-name>App Server</tier-name>  
    <node-name>mynode</node-name>  
    <agent-install></agent-install>  
    <agent-runtime-dir></agent-runtime-dir>  
</controller-info>
```

Modify only those values that are highlighted in the sample file given above.

Restart the Controller's application server

1. Open a console and navigate to the <Controller_Install_Directory>/bin directory and execute following command:

- For Linux:

```
./controller.sh start-appserver
```

- For Windows:

```
controller.bat start-appserver
```

Install the Machine Agent for the Controller

1. Install the Machine Agent on the Controller machine. See [Install the Machine Agent](#).
2. Use the same controller-info.xml file used by the App Agent, to point the Machine Agent at the AppDynamics Monitor.

Monitor the performance of the Controller

1. Open a browser at <http://saas-monitor.saas.appdynamics.com/controller/>.
2. Use the following information to access the Controller UI:
 - Account: <account-name> (same as specified in the URL)
 - User: admin (You can request an alternative user name to the one supplied by Support in step 1).
 - Password: (supplied by Support in step 1)
3. View the performance of your Controller.

Learn More

- [Controller SSL and Certificates](#)

Download Doc PDFs

Release Notes

[Release Notes \(337 kB\)](#) Updated for 3.7 on May 13, 2013

All Docs

[Complete Documentation Suite \(MB\)](#)

Doc Subsets

[Release Notes \(337 kB\)](#) Updated May 13, 2013

[AppDynamics Essentials \(904 kB\)](#)

[AppDynamics Features \(13,159 kB\)](#)

[Get Started \(121 kB\)](#)

[Get Started with AppDynamics SaaS \(84 kB\)](#)

[Get Started With AppDynamics On-Premise \(74 kB\)](#)

[Introduction and Tutorials \(5.87 MB\)](#)

[Use AppDynamics \(13.31 MB\)](#)

[AppDynamics for Java \(14,664 kB\)](#)
[AppDynamics for .Net \(1,161 kB\)](#)
[End User Experience \(2.82 MB\)](#)
[Configure AppDynamics \(7,379 kB\)](#)
[Integrate With Other Systems - includes REST and 3rd Parties \(699 kB\)](#)
[Monitor \(7,040 kB\)](#)
[Alert and Respond \(1.22 MB\)](#)
[Alert and Automate \(5.49 MB\)](#)
[AppDynamics Administration \(3.11 MB\)](#)
[Automate AppDynamics \(354 kB\)](#)
[Troubleshoot \(1,034 kB\)](#)
[AppDynamics Best Practices \(3,359 kB\)](#)

Use the Documentation Wiki

- Locate Information Using Search
- Locate Information Using Labels
- Use Comments for Feedback
- Generate a PDF or Word File from a Page
- Download a PDF of the Documentation
- Generate PDF or HTML Versions of Pages
- Known Issues
 - "Duplicate headers received from server" Error on PDF Export

Locate Information Using Search

1. Type a word or phrase in the Search box located at the upper right of the wiki pages.
2. Scroll through the list of results.

Locate Information Using Labels

Labels are keywords or tags that are added to pages. To view the labels defined in the system,

1. Click **Browse -> Labels**. The wiki displays a list of popular labels.
2. Click on any label to find pages tagged with that label.

Note: Labels are handy but may not return all instances of a topic. Use Search for more fine-grained results.

Use Comments for Feedback

AppDynamics is happy to receive your feedback! We strongly encourage you to leave comments on pages. Scroll to the bottom of a page and click **Add Comment** to add your comment. Comments are regularly monitored.

Generate a PDF or Word File from a Page

1. Navigate to the page.
2. Click **Tools -> Export to PDF** or **Tools -> Export to Word**.

Download a PDF of the Documentation

The entire documentation is available to download in PDF format as a ZIP file.

1. In the left navigation bar, scroll to the bottom.

2. Click the most recent ZIP file.

Generate PDF or HTML Versions of Pages

You can generate PDF or HTML versions of part or all of the documentation.

1. Go to the "Browse" menu.

2. Click **Browse -> Advanced**.

3. Click **PDF/HTML Export**.

4. Select those pages that you want to be available offline.

5. Click on "Export" button.



Known Issues

"Duplicate headers received from server" Error on PDF Export

When exporting a single page as PDF the follow error occurs:

Duplicate headers received from server
The response from the server contained duplicate headers. This problem is generally the result of a misconfigured website or proxy. Only the website or proxy administrator can fix this issue.
Error 349 (net::ERR_RESPONSE_HEADERS_MULTIPLE_CONTENT_DISPOSITION): Multiple distinct Content-Disposition headers received. This is disallowed to protect against HTTP response splitting attacks.

This is most likely due to a problem in the Chrome browser. In Chrome, files with non-standard alpha-numeric characters in the file name, such as , \$ % & * characters, can cause this issue. Try another browser.

License Information

In the Controller UI, click **Setup -> License** to see information about the AppDynamics licenses installed on your system.

The License window shows the name of your customer account and the type of license (Trial, etc.) used for the account. It lists the number of agent licenses and expiration date.

The App Agent for Java consumes a license when the JVM is instrumented, regardless of whether the node is used in the UI. To remove an agent license see [Manage App Agents#Deleting App Agents](#).

- Controller Licenses
- EUM License

Documentation Map

Get Started

[Get Started on SaaS](#)

[Get Started On-Premise](#)

[Features Overview](#)

[Architecture](#)

[Logical Model](#)

[Quick Start for DevOps](#)

[Quick Start for Operators](#)

[Quick Start for Administrators](#)

[Quick Start for Architects](#)

Tutorials for Java

[Use AppDynamics for the First Time with Java](#)

[Understanding Events](#)

[Understanding Flow Maps](#)

[Understanding Slow Transactions](#)

[Understanding the Transaction Scorecard](#)

[Understanding Server Health](#)

[Understanding Exceptions](#)

Monitor and Troubleshoot

[Best Practices for Application Developers](#)

Monitor Your Applications

[Business Transaction Monitoring](#)

Monitor End User Experience (EUM)
Monitor Events
Monitor Application Change Events
Background Task Monitoring
Health Rules

Troubleshoot Application Performance

Troubleshoot Slow Response Times
Troubleshoot Health Rule Violations
Transaction Snapshots
Troubleshoot Node Problems
Troubleshoot Errors
Diagnostic Sessions
Troubleshoot Java Memory Issues

Analysis

Business Metrics
Infrastructure Metrics
Reports
Compare Releases

Administration

Basic Administration

Release Notes for AppDynamics Pro
Supported Environments and Versions
Install and Upgrade AppDynamics
Name Business Applications, Tiers, and Nodes

Advanced Administration

AppDynamics for Large Enterprises
Implement SSL
Best Practices for Failover Scenarios
Administer Agents
Administer the Controller
Configure Authentication, User Permissions and Integrations

Concepts

Architecture
Logical Model
Mapping Application Services to the AppDynamics Model
Behavior Learning and Anomaly Detection
AppDynamics for Large Enterprises
Glossary

Configuration

Basic Configuration

Configure Business Transaction Detection
Configure Policies
Configure Baselines
Configure Thresholds
Configure Error Detection
Configure End User Experience
Alerting Wizard
Custom Dashboards

Advanced Configuration

Hierarchical Configuration Model
Configure Data Collectors
Configure Code Metric Information Points
Configure Backend Detection
Configure Call Graphs
Configure Background Tasks
Configure Stale Backend Removal
Configure Custom Memory Structures (Java)
Configure JMX Metrics from MBeans
Configure Multi-Threaded Transactions (Java)
Configure Object Instance Tracking (Java)
Configure Transaction Snapshots
Configure Memory Monitoring (Java)
Internationalization
Integrate using Custom Action Scripts
Export and Import Business Application Configurations

Integration

System Integrations
Use the AppDynamics REST API

Automation

Workflow Automation
Workflow for Cloud Orchestration

Reference

User Interface Reference

Dashboards
Transaction Analysis Tab
Flow Maps
Time Ranges
Call Graphs
Scorecards
KPI Graphs
Key to the AppDynamics Icons

FAQs

Controller Install and Admin FAQ
Controller High Availability FAQ
Machine Agent Install and Admin FAQ

AppDynamics Support

Controller Dump Files
Log Files for Troubleshooting
Configure Remote Monitoring of an On-Premise Controller
Download Doc PDFs
Use the Documentation Wiki

AppDynamics Features

Application Performance Monitoring

Business Transaction Monitoring

Background Task Monitoring

End User Experience

Rapid Troubleshooting

Infrastructure Monitoring

Databases and Services Monitoring

Alert and Respond - Runbook Automation

System Integrations

Search AppDynamics 3.7

Application Performance Monitoring

AppDynamics continuously discovers and monitors all modules in your application environment using advanced tag-and-follow tracing across your distributed transactions. With this information, AppDynamics provides a simple intuitive view of live application traffic and you can see where bottlenecks exist.

Dashboards show the health of your entire business application. Health indicators are based on configurable thresholds and they update based on live traffic. When new services are added to the system AppDynamics discovers them and adds them to the dashboards and flow maps. See [Visualize App Performance](#).

AppDynamics observes normal performance patterns so that it knows when application performance becomes abnormal. It

automatically identifies metrics whose current values are out of the normal range, based on dynamic baselines it has observed for these metrics. See [Behavior Learning and Anomaly Detection](#).

To learn more see:

Visualize App Performance

The AppDynamics UI provides a rich visual display of your application's performance. In addition to the default dashboards you can create customized views of the key performance metrics of your applications.

Dashboards

- [AppDynamics Dashboards](#)
- [Elements Common to all Dashboards](#)
- [Learn More](#)

AppDynamics Dashboards

Elements Common to all Dashboards



The top row menu is available on every view.

- The Setup drop-down list provides access to various system-level configurations.
- The Help drop-down list links to the documentation and support sites, and enables you to disable the help pop-ups.
- The Logout button shows the current user and enables you to log out.

The second row menu is available on every dashboard to help you navigate.

- The up arrow icon opens the All Applications dashboard.
- The right and left icons go to previously accessed views, just like in a browser.
- The refresh icon causes the display to update to the latest data.
- The breadcrumbs indicate the path to the current view.
- The help icon opens the wiki documentation.
- The link icon copies a link to the current view to the clipboard.
- The global Time Ranges drop-down list in the upper right sets the time range for all metrics shown for the application.

Learn More

- [Custom Dashboards](#)

All Applications Dashboard

- [Accessing the Applications Dashboard](#)
- [How the Applications Dashboard is Organized](#)
 - Top Menus
- [Application Panel Links and Menus](#)
- [Application Summary Panels](#)
 - Health Rule Violations
 - Key Metrics
 - Business Transaction Health
 - Server Health
- [Learn More](#)

This topic describes the Applications Dashboard, also called Home. You can monitor all AppDynamics business applications from the

Applications Dashboard.

Accessing the Applications Dashboard

To first access the Applications Dashboard, log into the AppDynamics Controller at the URL given in your welcome email and on your AppDynamics account home page.



To return to the Applications Dashboard, click the **Home - All Applications** (up arrow) icon.

Policy Violations	Load	Response Time	Errors
3 3 Critical 0 Warning 1 Open Now	1.1m calls 3.3k calls / min	104 ms	3.7 % 40.1k errors 124 errors / min

Business Transaction Health: 2 critical, 0 warning, 15 normal, 0 unknown.

Server Health: 0 critical, 0 warning, 4 normal, 0 unknown.

How the Applications Dashboard is Organized

The Applications Dashboard displays all business applications and the key performance indicators of each. By default, business applications are listed in descending order based on load (calls per minute), with the highest load listed first.

Top Menus

The top two menus are common to all dashboards. See [Dashboards](#).

The third row menu is specific to each dashboard. In the All Applications dashboard you can:

- Click **Create Application** to add a new business application to AppDynamics.
- Click **Import Application** to import a business application configuration to AppDynamics. See [Export and Import Business Application Configurations](#).
- Use the **View** icons to change between graphical and list views.
- Use the **Sort By** drop down menu helps you sort a longer list of business applications.

Application Panel Links and Menus

Each business application listing has a link to the application's dashboard.

For convenience each application listing has menus for commonly-performed actions.

- **Troubleshoot** opens the Troubleshoot main menu screen.
- **Analyze** opens the Analyze main menu screen.
- **Configure** opens the Configuration main menu screen.
- **Actions -> Troubleshoot**
 - **View Agent Diagnostics** opens the Agent Diagnostics Data screen.
- **Actions -> Configure**
 - **Edit Application Properties** lets you edit the name and description of the application.
 - **Delete**
 - **Copy**
 - **Export** saves an application configuration for reuse. See [Export and Import Business Application Configurations](#).

Application Summary Panels

For each business application, the Applications Dashboard shows these panels:

- Health Rule Violations
- Key Metrics
- Business Transaction Health
- Server Health

Health Rule Violations

The Health Rule panel shows how many health rule violations, if any, occurred during the selected time range. Click **Health Rule Violations** to [Troubleshoot Health Rule Violations](#).

Key Metrics

The key metrics panels show key performance indicators for each business application:

- Load: number of calls and calls per minute
- Response Time: average response time
- Errors: number of errors

Click **Errors** to [Troubleshoot Errors](#).

Business Transaction Health

The Business Transaction Health panel shows the combined health for all business transactions in the application. The health of each business transaction is based on any health rule violations triggered during the selected time range. The color bar indicates:

- Green - no violations
- Yellow - at least one violation
- Red - at least one critical violation

Click **Business Transaction Health** to open the [Business Transactions List](#) and find more details. See [Troubleshoot Health Rule Violations](#).

Server Health

The Server Health panel shows combined health for all tiers and nodes in the application. The health of each tier and node is based on any health rule violations triggered during the selected time range. The color bar indicates:

- Green - no violations
- Yellow - at least one violation
- Red - at least one critical violation

Click **Server Health** to see details about all the tiers and nodes in the App Servers List. See [Troubleshoot Health Rule Violations](#).

Learn More

- Dashboards
- Application Dashboard

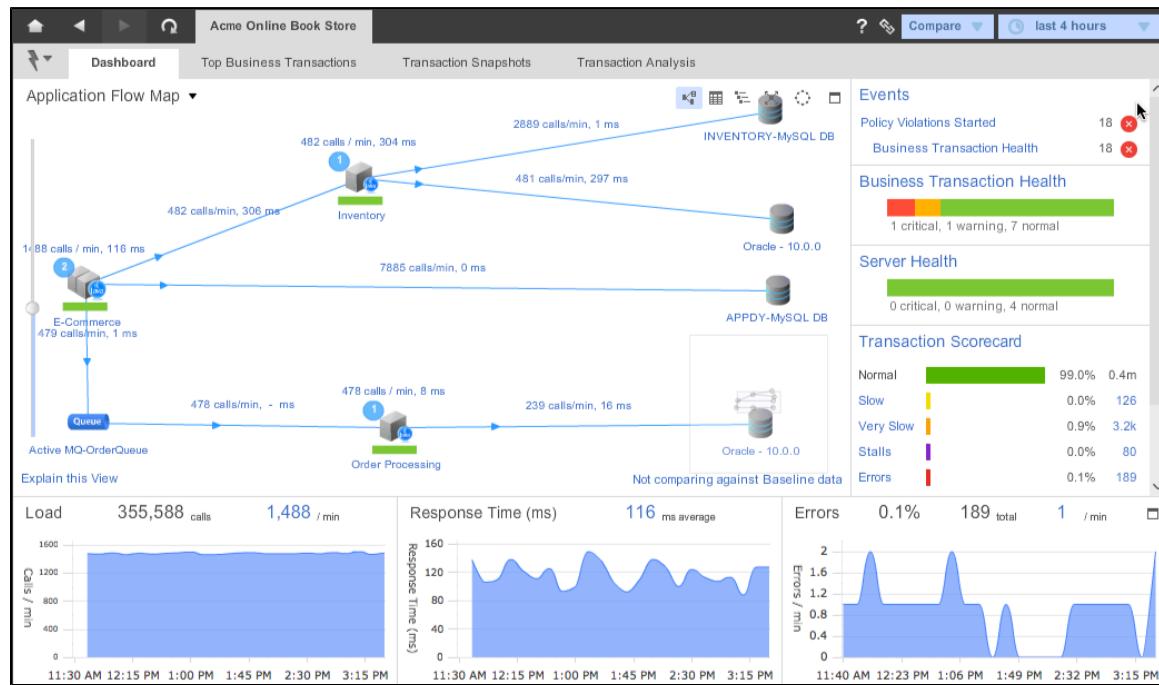
Application Dashboard

- Accessing the Application Dashboard
- How the Application Dashboard is Organized
 - Action Menu
 - Application Dashboard Tabs
 - Dashboard Tab
 - Compare Menu
 - Top Business Transactions Tab
 - Transaction Snapshots Tab
 - Transaction Analysis Tab
- Learn More

Each business application has its own Application Dashboard, which shows high-level status information for a single business application in the selected time range.

Accessing the Application Dashboard

On the All Applications Dashboard, click an application name to view the dashboard for that application. The Application Dashboard opens.



How the Application Dashboard is Organized

The Application Dashboard shares the same common components as other Dashboards, including Time Ranges.

Action Menu

The Action menu provides commonly-performed actions for a business application.

- **Edit Application Properties:** Lets you edit the name and description of the application.
- **Delete** Deletes the application and all its data from AppDynamics. Not reversible.
- **Copy** Copies the entire application to a new application.
- **Export:** Exports an application configuration for reuse. See [Export and Import Business Application Configurations](#).
- **Export as PDF report:** Exports a report in PDF format containing the flow map and KPIs for the selected time range.

Application Dashboard Tabs

The Application Dashboard has these tabs:

- [Dashboard](#)
- [Top Business Transactions](#)
- [Transaction Snapshots](#)
- [Transaction Analysis](#)

Dashboard Tab

The Dashboard Tab displays summary panels similar to the All Applications Dashboard:

- **Application flow map:** Graphically depicts performance across the all tiers, nodes and backends for all business transactions in the application. See [Flow Maps](#).
- **Events:** Lists events that are by default configured to display in the dashboard: Health Rule Violation Started, Code Problems, Application Changes and Custom events. See [Events](#)
- **Business Transaction Health:** shows the combined health for all business transactions in the application. The health of each business transaction is based on any health rule violations triggered during the selected time range. The color bar indicates:
 - Green - no health rule violations
 - Yellow - at least one health rule violation
 - Red - at least one critical health rule violation
 Click [Business Transaction Health](#) to open the Business Transactions List and find more details. See [Troubleshoot](#)

- Health Rule Violations.**
- **Server Health:** shows combined health for all tiers and nodes in the application. The health of each tier and node is based on any health rule violations triggered during the selected time range. The color bar indicates:
 - Green - no health rule violations
 - Yellow - at least one health rule violation
 - Red - at least one critical health rule violation
 Click **Server Health** to see details about all the tiers and nodes in the App Servers List. See [Troubleshoot Health Rule Violations](#).
 - **Transaction Scorecard:** number and percentage of transactions that are normal, slow, very slow, stalled or errors. See [Scorecards](#).
 - **Exceptions:** list the total number and number-per-minute of:
 - Exceptions
 - HTTP Error Codes
 - Error Page Redirects
 Click the **Exceptions** link to see the list of exceptions. See [To Troubleshoot Exceptions](#) for more information.
 - **Key Performance Indicators:** See [KPI Graphs](#).

Compare Menu

The **Compare** menu is visible when the Dashboard tab is selected.

By default the performance displayed in the dashboard is compared against the daily trend, which is the performance over the last 30 days. From the Compare menu you can direct AppDynamics to use a different baseline or no baseline. You can also configure baselines. For more information about baselines see [Behavior Learning and Anomaly Detection](#) and [Configure Baselines](#).

Top Business Transactions Tab

The Top Business Transactions Tab shows the key performance indicators for the most expensive business transactions sorted by the following criteria:

- Load
- Response Time
- Errors
- Number of Slow Transactions
- Number of Stalls
- Number of Health Rule Violations

If you click **View All** in any of the panels in this tab, the business transaction list opens displaying all the key performance indicators for the business transactions in one panel. See [Business Transactions List](#).

Transaction Snapshots Tab

The Transaction Snapshots Tab displays the transaction snapshots for the selected time range. From a transaction snapshot you can drill down to the root cause of a performance problem. See [Transaction Snapshots](#).

Transaction Analysis Tab

The Transaction Analysis tab displays application performance over the selected time range as a graph. Use the graph to analyze the impact of different events on the application's response time. See [Transaction Analysis Tab](#).

Learn More

- Dashboards
- Flow Maps
- Configure Baselines
- Time Ranges
- Transaction Analysis Tab
- Business Transactions List
- Transaction Snapshots

Business Transaction Dashboard

- Accessing a Business Transaction Dashboard
- How the Business Transaction Dashboard is Organized
 - Business Transaction Flow Map
 - Action Menu
 - Business Transaction Dashboard Tabs
- Dashboard Tab
- Events Tab

- Slow Response Times Tab
- Errors Tab
- Transaction Snapshots Tab
 - All Snapshots Subtab
 - Slow and Error Transactions Subtab
 - Diagnostic Sessions Subtab
 - Periodic Collection Subtab
- Transaction Analysis Tab
- Learn More

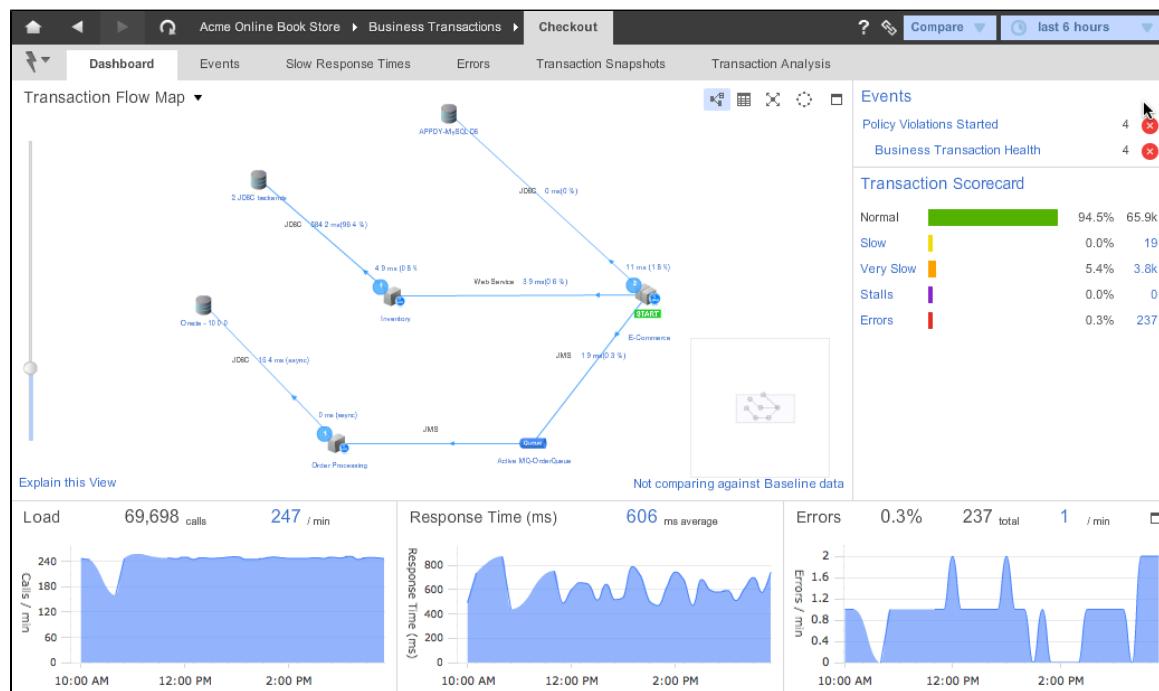
The Business Dashboard provides single-click access to all performance indicators for any detected business transaction.

Each business transaction has its own dashboard.

Accessing a Business Transaction Dashboard

To view a dashboard for a particular business transaction:

1. Select the business application.
2. In the left navigation pane, click **Business Transactions**. AppDynamics displays the business transaction list.
3. From the list select the business transaction for which you want to see the dashboard.
4. Either double-click on the business transaction or click **View Dashboard**.



How the Business Transaction Dashboard is Organized

The Business Transaction Dashboard shares the same components as other Dashboards, including Time Ranges.

Business Transaction Flow Map

The Business Transaction Dashboard contains a Flow Map of the discovered tiers, nodes, databases, remote services. See [Flow Maps](#).

Action Menu

The Action menu provides commonly-performed actions for a business transaction.

- **Monitor -> View Metrics By Individual Nodes** displays metrics for each node in the tier in which the business transaction

- started.
- **Troubleshoot -> View Health Rule Violations** displays all the health rule violations for the business transaction for the selected time range.
 - **Troubleshoot -> Start Diagnostic Session** starts a diagnostic session on the business transaction. See [Diagnostic Sessions](#).
 - **Analyze -> Analyze Response Time vs Load** See [Scalability Analysis](#).
 - **Report -> Export PDF Report** instantly generates a PDF report summarizing the business transaction performance. See [Reports](#).
 - **Configure -> Thresholds** See [Configure Thresholds](#).
 - **Configure -> Data Collectors** See [Configure Data Collectors](#).
 - **Rename** Rename the business transaction.
 - **Delete** The delete action prevents the transaction from being monitored. However, it will be discovered and added to the business transactions list if it is discovered again by AppDynamics. To prevent a business transaction from being monitored at all, exclude the transaction.
 - **Exclude** Use the exclude action to prevent the selected transaction from being discovered by AppDynamics.
 - **Set as Background Task** Converts the business transaction to a background task. See [Background Task Monitoring](#).

Business Transaction Dashboard Tabs

The Business Transaction Dashboard has these tabs:

- Dashboard Tab
- Events Tab
- Slow Response Times Tab
- Errors Tab
- Transaction Snapshots Tab
- Transaction Analysis Tab

Dashboard Tab

Because the design for Business Transaction Dashboard is modeled on the [Application Dashboard](#), you can drill down for transaction snapshots, errors and events directly from the dashboard.

Events Tab

The Business Transaction Dashboard Events tab lists the events for the transaction and summary information about each.

- **Type:** type of event. See [Events](#).
- **Summary:** description of the event.
- **Time:** date and time when the event occurred.
- **Business Transaction:** link to the [Business Transaction Dashboard](#) for the event.
- **Tier:** link to the tier on which the event occurred.
- **Node:** link to the [Node Dashboard](#) for the node on which the event occurred.

This is an embedded copy of the event list in which only events for the selected business transaction are reported. You can modify the event types reported in this list. For more information about the events list see [Filter and Analyze Events](#).

Slow Response Times Tab

The Slow Response Times tab graphs slow transactions and displays the transaction snapshots for the slow and stalled transactions for the selected time range.

Select a transaction snapshot from the list and click **View Transaction Snapshot** to drill down to the cause of the error. See [Transaction Snapshots](#).

Errors Tab

The Errors tab graphs and displays the error transaction snapshots for the selected time range.

Select a transaction snapshot from the list and click **View Transaction Snapshot** to drill down to the cause of the error. See [Transaction Snapshots](#).

Transaction Snapshots Tab

The Transaction Snapshots tab has these subtabs:

- All Snapshots
- Slow and Error Transactions

- Diagnostic Sessions
- Periodic Collection

The **Slow and Error Transactions** or **Diagnostic Sessions** subtabs are the quickest route to drill down to the root cause of slow, stalled or error transactions.

All Snapshots Subtab

This subtab displays all the transaction snapshots captured for the selected time range. Error transaction snapshots are coded red, slow transaction snapshots are coded yellow, and stalled transaction snapshots are coded purple.

You can select a snapshot from the list and double-click it or click **View Snapshot** to drill down to the root cause of the problem. See [Transaction Snapshots](#) for more information.

Slow and Error Transactions Subtab

This subtab displays only snapshots for slow and error transactions.

Diagnostic Sessions Subtab

This subtab displays diagnostic sessions that have already been started. It also lets you start a diagnostic sessions by clicking **Start Diagnostic Session**.

A diagnostic session captures detailed data about the processing of a transaction as transaction snapshots over a defined period of time. The snapshots include full call graphs. See [Diagnostic Sessions](#).

Periodic Collection Subtab

This subtab displays transaction snapshots that have been configured to be collected periodically, whether or not the transactions are slow or error. See [Configure Transaction Snapshots](#) for information about enabling and disabling periodic snapshot collection.

Transaction Analysis Tab

The Transaction Analysis tab shows the [Transaction Analysis Tab](#) graph for the business transaction.

Learn More

- [Flow Maps](#)
- [Filter and Analyze Events](#)

Business Transactions List

- [Accessing the Business Transactions List](#)
- [How the Business Transactions List is Organized](#)
 - [View Performance Data](#)
 - [Filters](#)
 - [Business Transactions List Operations](#)
 - [Monitor](#)
 - [Troubleshoot](#)
 - [Analyze](#)
 - [Report](#)
 - [Configure](#)
 - [Other](#)
- [Learn More](#)

This topic describes the business transactions list.

Accessing the Business Transactions List

AppDynamics lists all business transactions on the business transactions list view.

To access this dashboard,

In the left navigation pane, click **Business Transactions**.

How the Business Transactions List is Organized

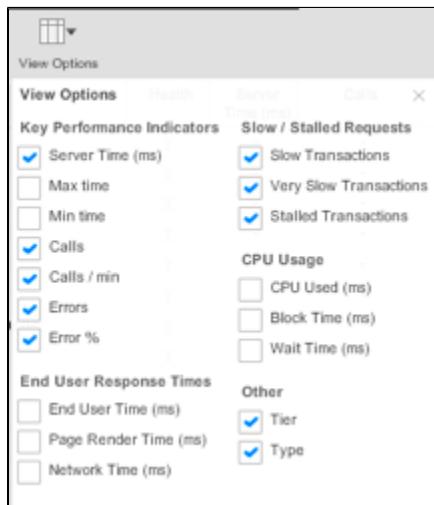
The business transactions list shows all the transactions for a business application.

Click on a column to sort the business transactions based on the data in that particular column.

To view the business transaction dashboard for a specific business transaction, select the business transaction and click **View Dashboard**.

View Performance Data

Select **View Options** to configure which performance data is displayed in the business transaction list.



- Server Time = average response time that the application server spends processing the business transaction requests over the configured time period.
- Max Time = the longest response time that the application server has spent processing the business transaction request over the configured time period.
- Min Time = the shortest response time that the application server has spent processing the business transaction request over the configured time period.

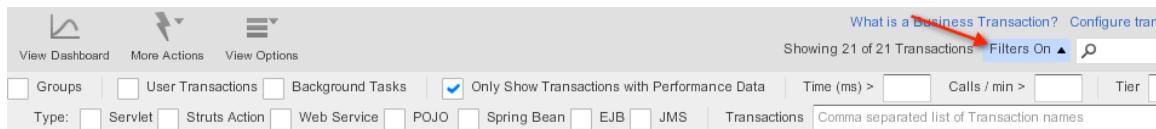
Filters

You can filter the display of business transactions based on different criteria such as:

- **User transactions:** Enable this option to retrieve only user transactions.
- **Background tasks:** Enable this option to retrieve only those transactions that are identified as background tasks.
- **Performance data:** Use this option to filter the transactions based on response time and calls/minute.
- **Tier:** Provide the tier name to filter the transactions for a particular tier.
- **Transaction Name:** Provide a comma-separated list of transaction names.
- **Transaction Type:** You can choose to filter transactions based on transaction types like Servlet, Struts Action, Web Service, etc.

To filter the transactions:

1. Click **Filters** next to the Time Range drop-down menu in the upper right corner of the list.
2. Check the checkboxes for the criteria that define the business transactions to display.



Business Transactions List Operations

Select a particular business transaction and right-click, or click **More Actions**. You can perform following operation on the selected business transaction.

Monitor	Configure
View Metrics by Individual Nodes	Only Show Top 10 Requests Configure Thresholds Configure Data Collectors
Troubleshoot	Health Rule Violations Rename (ms) Delete Transaction Exclude Transactions
View Policy Violations	View Excluded Transactions
Start Diagnostic Session	Set as Background Task
ToCart	
Analyze	
Analyze Response Time vs Load	
Report	Create Group Delete Group
Export Grid Data	

Monitor

- **View Metrics by Individual Nodes:** Use this option to see metrics by node in the tier in which the business transaction started. You can view the metrics in the node dashboard or the metric browser.

Troubleshoot

- **View Health Rule Violations:** Use this option to view all the health rule violations for the selected transaction(s) for the particular time-range.
- **Start Diagnostic Session:** Use this option to manually start a diagnostic session on the selected transaction (by default, diagnostic sessions are triggered only when the request experiences problems or when it violates the thresholds).

Analyze

- **Analyze Response Time vs Load Time**

Report

- **Export Grid Data:** You can export the data displayed on the grid and save it in a file.

Configure

- **Configure Thresholds:** Use this option to configure thresholds for the selected transaction(s).
- **Configure Data Collectors:** Use this option to configure data collectors for the selected transaction(s).

Other

- **Rename:** By default, all the user requests are identified using the default naming scheme for different types of requests. For example, all Servlets based requests are identified using the first two segments of the URI. You can rename these transactions.
- **Delete:** You can also delete the selected transaction(s) from being monitored. However, it will be discovered and added to the business transactions list if it is discovered again by AppDynamics. To remove a particular transaction from being monitored at all, you must exclude that transaction.
- **Exclude Transactions:** Use this option to remove the selected transaction from being discovered by AppDynamics.
- **View Excluded Transactions**
- **Set as Background Task:** The metrics (average response time and calls per minute) for a background tasks are typically much higher than the user requests. And therefore, AppDynamics does **not** collect tier and application level metrics for the background tasks. It is therefore important to identify a transaction serving the functionality of background tasks separately. As a result, the calls and response time metrics for background task will not be counted on application or tier dashboard.
- **Create Group:** Use this option to group different business transactions in a single logical group.
- **Delete Group:**

Learn More

- Business Transaction Dashboard

Tier Dashboard

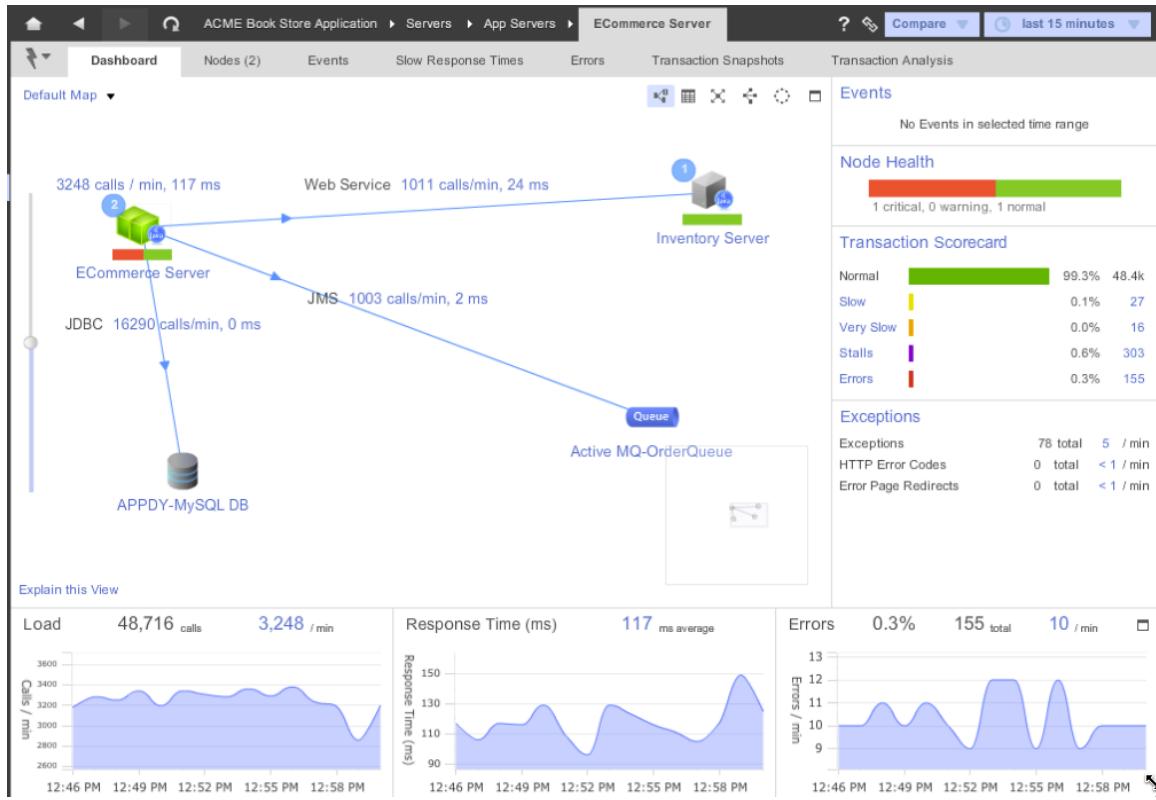
- Accessing the Tier Dashboard
- How the Tier Dashboard is Organized
 - Action Menu
 - Tier Dashboard Tabs
 - Dashboard Tab
 - Compare Menu
 - Nodes Tab
 - Health Subtab
 - Hardware Subtab
 - Memory Subtab
 - Events Tab
 - Slow Response Times Tab
 - Slow Transactions Subtab
 - Slow DB & Remote Service Calls Subtab
 - Errors Tab
 - Transaction Snapshots Tab
 - All Snapshots Subtab
 - Slow and Error Transactions Subtab
 - Diagnostic Sessions Subtab
 - Periodic Collection Subtab
 - Transaction Analysis Tab
 - IIS AppPools Tab (.NET Only)
 - Learn More

The Tier Dashboard provides single-click access to all performance indicators for a particular tier (server), including health and key metric information for each node. You can also view the Metric Browser from the Tier Dashboard.

Accessing the Tier Dashboard

To view the Tier Dashboard for a particular tier:

1. Select the business application.
2. In the left navigation pane, click **Servers -> App Servers -> <Tier>**.
AppDynamics displays the Tier Dashboard for the selected tier.



How the Tier Dashboard Is Organized

The Tier Dashboard shares the same common components as other Dashboards, including Time Ranges.

Action Menu

The Action menu provides commonly-performed actions for a tier.

- **Analyze** -> **Analyze Response Time vs Load** and **Analyze CPU vs Load** open the Scalability Analysis window.
- **Report** -> **Export PDF Report** instantly generates a PDF report. See [Reports](#).
- **Configure** opens the Configuration main menu screen.
 - **Configure transaction detection** opens the Transaction Detection tab of the Configure Instrumentation window. See [Configure Business Transaction Detection](#).
 - **Configure App Server Agent** opens the App Server Agent Configuration window. See [App Agent Node Properties](#).
 - **Configure Backends resolving to this Tier** lets you change the association of a remote service or backend with this tier. You can either associate the backend with another tier by clicking **Resolve to a Different Tier** and selecting the new tier from the dropdown menu or by clicking the **Delete** button to disassociate it from all tiers, in which case the backend appear as an independent component ("unresolved backend") on the application flow map.
 - **Edit Properties** opens the Tier Properties window. You can change the name (label) of the tier and write a description for it. If the tier is an instrumented app server, the Type is automatically set.
 - **Delete Tier** deletes the tier from the flow map.

Tier Dashboard Tabs

The Tier Dashboard has these tabs:

- Dashboard Tab
- Nodes Tab
- Events Tab
- Slow Response Times Tab
- Errors Tab
- Transaction Snapshots Tab
- Transaction Analysis Tab
- IIS AppPools Tab (.NET Only)

Dashboard Tab

The Dashboard Tab displays:

- **Tier Flow Map:** The Tier Dashboard contains a Flow Map of the selected tier. See [Flow Maps](#).
- **Events:** Lists events that are by default configured to display in the dashboard: Health Rule Violation Started, Code Problems, Application Changes and Custom events. See [Events](#)
- **Node Health:** shows the combined health for all nodes in the tier. The health of each node is based on any health rule violations triggered during the selected time range. The color bar indicates:
 - Green - no health rule violations
 - Yellow - at least one health rule violation
 - Red - at least one critical health rule violation
- **Transaction Scorecard:** number and percentage of transactions that are normal, slow, very slow, stalled or errors. See [Scorecards](#).
- **Exceptions:** list the total number and number-per-minute of:
 - Exceptions
 - HTTP Error Codes
 - Error Page RedirectsClick the [Exceptions](#) link to see the list of exceptions. See [To Troubleshoot Exceptions](#) for more information.
- **Key Performance Indicators:** See [KPI Graphs](#).

Compare Menu

The **Compare** menu is visible when the Dashboard tab is selected.

By default the performance displayed in the dashboard is compared against the daily trend, which is the performance over the last 30 days. From the Compare menu you can direct AppDynamics to use a different baseline or no baseline. You can also configure baselines. For more information about baselines see [Behavior Learning and Anomaly Detection](#) and [Configure Baselines](#).

Nodes Tab

The Nodes Tab has these subtabs:

- [Health](#)
- [Hardware](#)
- [Memory](#)

On any subtab, select a node and click **View Dashboard** or double-click on a node in the list to open its [Node Dashboard](#).

Health Subtab

The Nodes Health subtab lists nodes that are part of the tier and summary information about each.

- **Name:** the name of the node.
- **Health:** health rule violation status.
- **App Agent Status:** whether the agent is up (running and connected to the Controller) or not.
- **App Agent Version:** version of the agent.
- **JVM:** version of the JVM or CLR.
- **Last JVM Restart:** date and time of the last JVM/CLR restart.
- **Machine Agent Status:** if there is a machine agent on the node server, whether it is up (running and connected to the Controller) or not.

Hardware Subtab

The Nodes Hardware subtab lists information about the physical server that is hosting the node. If the Machine Agent is not installed on the physical device, no data will be available.

Memory Subtab

The Memory subtab lists information about the JVM or CLR of the node.

Events Tab

The Tier Dashboard Events tab lists the events for the tier and summary information about each.

- **Type:** type of event. See [Events](#).
- **Summary:** description of the event.
- **Time:** date and time when the event occurred.
- **Business Transaction:** link to the [Business Transaction Dashboard](#) for the event.
- **Tier:** link to the tier on which the event occurred.
- **Node:** link to the [Node Dashboard](#) for the node on which the event occurred.

This is an embedded copy of the event list in which only events for the selected tier are reported. You can modify the event types reported in the list. For more information about the event list see [Filter and Analyze Events](#).

Slow Response Times Tab

The Slow Response Times tab has two subtabs:

- [Slow Transactions](#)
- [Slow DB & Remote Service Calls](#)

Slow Transactions Subtab

This subtab is also accessible from the Troubleshoot menu.
See [Troubleshoot Slow Response Times](#).

Slow DB & Remote Service Calls Subtab

This subtab is also accessible from the Troubleshoot menu.
See [Troubleshoot Slow Response Times](#).

Errors Tab

The Errors tab has two subtabs:

- [Error Transactions](#)
- [Exceptions](#)

These subtabs are also accessible from the Troubleshoot menu.
See [Troubleshoot Errors](#).

Transaction Snapshots Tab

The Transaction Snapshots tab has these subtabs:

- [All Snapshots](#)
- [Slow and Error Transactions](#)
- [Diagnostic Sessions](#)
- [Periodic Collection](#)

The **Slow and Error Transactions** or **Diagnostic Sessions** subtabs are the quickest route to drill down to the root cause of slow, stalled or error transactions.

All Snapshots Subtab

This subtab displays all the transaction snapshots captured for the selected time range. Error transaction snapshots are coded red, slow transaction snapshots are coded yellow, and stalled transaction snapshots are coded purple.

You can select a snapshot from the list and double-click it or click **View Snapshot** to drill down to the root cause of the problem. See [Transaction Snapshots](#) for more information.

Slow and Error Transactions Subtab

This subtab displays only snapshots for slow and error transactions.

Diagnostic Sessions Subtab

This subtab displays diagnostic sessions that have already been started. It also lets you start a diagnostic sessions by clicking **Start Diagnostic Session**.

A diagnostic session captures detailed data about the processing of a transaction as transaction snapshots over a defined period of time. The snapshots include full call graphs. See [Diagnostic Sessions](#).

Periodic Collection Subtab

This subtab displays transaction snapshots that have been configured to be collected periodically, whether or not the transactions are slow or error. See [Configure Transaction Snapshots](#) for information about enabling and disabling periodic snapshot collection.

Transaction Analysis Tab

The Transaction Analysis tab shows the Transaction Analysis Tab graph for the selected tier.

IIS AppPools Tab (.NET Only)

This tab displays the nodes in the IIS App Pool. See Monitoring IIS Application Pools.

Learn More

- Dashboards
- Flow Maps
- Configure Baselines
- Time Ranges
- Transaction Analysis Tab
- Business Transactions List
- Transaction Snapshots
- Events

Node Dashboard

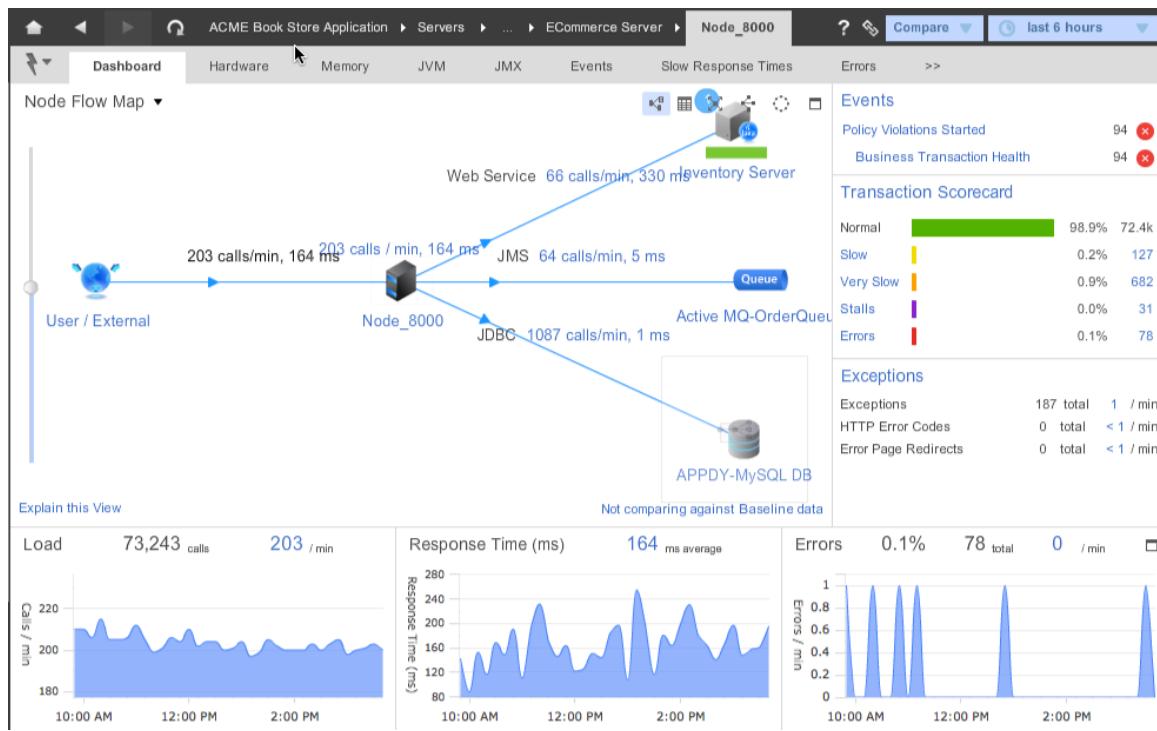
- Accessing the Node Dashboard
 - To view the Node Dashboard for a particular node
- How the Node Dashboard is Organized
 - Action Menu
 - Node Dashboard Tabs
 - Dashboard Tab
 - Compare Menu
 - Hardware Tab
 - Memory Tab
 - JVM Tab (Java Only)
 - JMX Tab (Java Only)
 - CLR Tab (.NET Only)
 - Events Tab
 - Slow Response Times Tab
 - Slow Transactions Subtab
 - Slow DB & Remote Service Calls Subtab
 - Errors Tab
 - Transaction Snapshots Tab
 - All Snapshots Subtab
 - Slow and Error Transactions Subtab
 - Diagnostic Sessions Subtab
 - Periodic Collection Subtab
 - Transaction Analysis Tab
 - Agents Tab
 - App Server Agent Subtab
 - To Request Agent Log Files
 - To Configure and Start an Agent Logging Session (Java only)
 - Machine Server Agent Subtab
 - Agent Diagnostic Stats Subtab
 - Learn More

The Node Dashboard displays the overall health, key metrics, and provides single-click access to all of the performance indicators for a particular node.

Accessing the Node Dashboard

To view the Node Dashboard for a particular node

1. Select the business application.
2. In the left navigation pane, click **Servers -> App Servers -> <Tier> -> <Node>**.
AppDynamics displays the Node Dashboard for the selected node.



How the Node Dashboard is Organized

The Node Dashboard shares the same common components as other Dashboards, including Time Ranges.

Action Menu

The Action menu provides commonly-performed actions for a node

- **Troubleshoot -> View Slow and Error Transactions** displays transaction snapshots for the selected time range by node. See [Transaction Snapshots](#).
- **Troubleshoot -> View Health Rule Violations** displays the list of health rule violations. See [Troubleshoot Health Rule Violations](#).
- **Troubleshoot -> View Agent Diagnostic Events** displays the list of agent diagnostic events. See [App Agent for Java Diagnostic Data](#).
- **Analyze -> Analyze Response Time vs Load and Analyze CPU vs Load** open the Scalability Analysis window.
- **Analyze -> Analyze Node Problems** displays the node problem viewer. See [Troubleshoot Node Problems](#).
- **Report -> Export PDF Report** instantly generates a PDF report. See [Reports](#).
- **Configure -> Configure App Server Agent** opens the Node Properties configuration window. See [App Agent Node Properties](#)
- **Edit Properties** lets you modify the type of the node.
- **Move Node** lets you move the node to another tier.
- **Delete Node** deletes the node from the flow map.

Node Dashboard Tabs

The Node Dashboard has these tabs:

- Dashboard Tab
- Hardware Tab
- Memory Tab
- JVM Tab (Java Only)
- JMX Tab (Java Only)
- CLR Tab (.NET Only)
- Events Tab
- Slow Response Times Tab
- Errors Tab
- Transaction Snapshots Tab
- Transaction Analysis Tab
- Agents Tab

Dashboard Tab

The Dashboard Tab displays:

- **Node Flow Map:** The Node Dashboard contains a Flow Map for the selected node. The User/External symbol in the Flow Map represents all traffic to the selected node in the selected tier that is not accounted for by calls from other tiers. See [Flow Maps](#).
- **Events:** Lists events that are by default configured to display in the dashboard: Health Rule Violation Started, Code Problems, Application Changes and Custom events. See [Events](#)
- **Transaction Scorecard:** Number and percentage of transactions that are normal, slow, very slow, stalled or errors. See [Scorecards](#).
- **Exceptions:** list the total number and number-per-minute of:
 - Exceptions
 - HTTP Error Codes
 - Error Page RedirectsClick the [Exceptions](#) link to see the list of exceptions. See [To Troubleshoot Exceptions](#) for more information.
- **Key Performance Indicators:** See [KPI Graphs](#).

Compare Menu

The **Compare** menu is visible when the Dashboard tab is selected.

By default the performance displayed in the dashboard is compared against the daily trend, which is the performance over the last 30 days. From the Compare menu you can direct AppDynamics to use a different baseline or no baseline. You can also configure baselines. For more information about baselines see [Behavior Learning and Anomaly Detection](#) and [Configure Baselines](#).

Hardware Tab

The Hardware tab graphs metrics about the physical server that hosts the node. These metrics include:

- CPU Utilization
- Memory Utilization
- Disk I/O
- Network I/O

These metrics are available if a machine agent is installed on the machine that hosts the node.

Memory Tab

The Memory tab has four subtabs:

- Heap and Garbage Utilization
- Automatic Leak Detection
- Object Instance Tracking
- Custom memory Structures

See [Troubleshoot Java Memory Issues](#).

JVM Tab (Java Only)

The JVM tab displays the JVM version, startup options, system options and environment properties for the node. See [App Agent for Java Configuration Properties](#) for information about the startup options.

Java only.

JMX Tab (Java Only)

The JMX tab lets you monitor JMX metrics. See [Monitor JMX MBeans](#).

Java only.

CLR Tab (.NET Only)

The CLR tab displays CLR properties, CLR startup properties, CLR metadata, and environment variables.

Events Tab

The Events tab lists the events for the node and summary information about each.

- **Type:** type of event. See [Events](#).
- **Summary:** description of the event.
- **Time:** date and time when the event occurred.
- **Business Transaction:** link to the [Business Transaction Dashboard](#) for the event.
- **Tier:** link to the tier on which the event occurred.
- **Node:** link to the [Node Dashboard](#) for the node on which the event occurred.
- **Actions Executed:** action executed in response to the event, if any

This is an embedded copy of the event list in which only events for the selected node are reported. You can modify the event types reported in the list. For more information about the event list see [Filter and Analyze Events](#).

Slow Response Times Tab

The Slow Response Times tab has two subtabs:

- [Slow Transactions](#)
- [Slow DB & Remote Service Calls](#)

Slow Transactions Subtab

This subtab is also accessible from the Troubleshoot menu.
See [Slow and Stalled Transactions](#).

Slow DB & Remote Service Calls Subtab

This subtab is also accessible from the Troubleshoot menu.
See [Slow Database and Remote Service Calls](#).

Errors Tab

The Errors tab has two subtabs:

- [Error Transactions](#)
- [Exceptions](#)

These subtabs are also accessible from the Troubleshoot menu.
See [Troubleshoot Errors](#).

Transaction Snapshots Tab

The Transaction Snapshots tab has these subtabs:

- [All Snapshots](#)
- [Slow and Error Transactions](#)
- [Diagnostic Sessions](#)
- [Periodic Collection](#)

The **Slow and Error Transactions** or **Diagnostic Sessions** subtabs are the quickest route to drill down to the root cause of slow, stalled or error transactions.

All Snapshots Subtab

This subtab displays all the transaction snapshots captured for the selected time range. Error transaction snapshots are coded red, slow transaction snapshots are coded yellow, and stalled transaction snapshots are coded purple.

You can select a snapshot from the list and double-click it or click **View Snapshot** to drill down to the root cause of the problem. See [Transaction Snapshots](#) for more information.

Slow and Error Transactions Subtab

This subtab displays only snapshots for slow and error transactions.

Diagnostic Sessions Subtab

This subtab displays diagnostic sessions that have been started and enables you to start diagnostic sessions by clicking **Start Diagnostic Session**.

A diagnostic session captures detailed data about the processing of a transaction as transaction snapshots over a defined period of time. These snapshots include full call graphs. See [Diagnostic Sessions](#).

Periodic Collection Subtab

This subtab displays transaction snapshots that have been configured to be collected periodically, whether or not the transactions are slow or error. See [Configure Transaction Snapshots](#) for information about enabling and disabling periodic snapshot collection.

Transaction Analysis Tab

The Transaction Analysis tab shows the Transaction Analysis Tab histogram for the selected node.

Agents Tab

The Agents Tab provides information about the agent instrumenting the node. It contains these subtabs:

- App Server Agent
- Machine Agent
- Agent Diagnostic Stats

App Server Agent Subtab

This subtab describes the properties of the app agent.

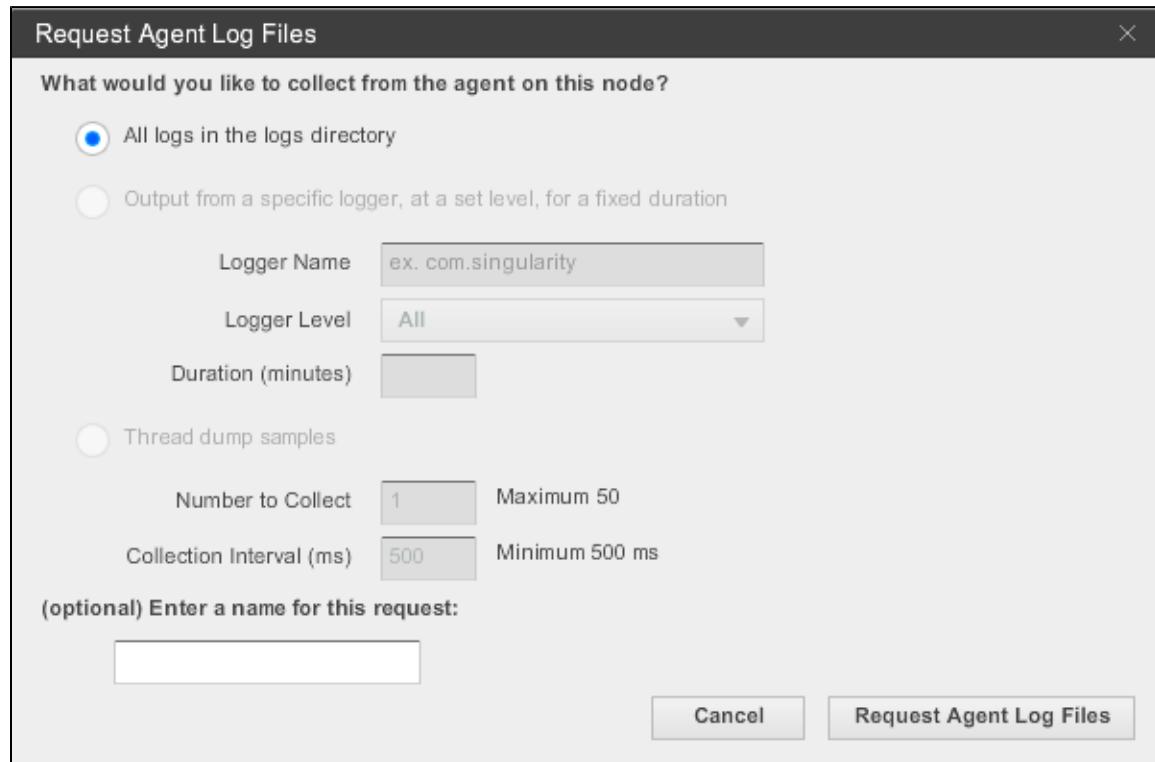
It also enables you to request the agent log files and to configure and start logging of specific types of requests for a specified duration. This gives you fine control of what to log.

To Request Agent Log Files

1. In the **Agents** tab of the node dashboard, scroll down to the Agent Logs panel.

2. Click **Request Agent Log Files**.

The Request Agent Log Files window opens.



3. Select one of the options:

- All logs in the logs directory: This option retrieves all the logs in the agent log directory for this node and creates a zip file. You can download the zip file for review.
- Output from a specific logger, at a set level, for a fixed duration: To use this option contact AppDynamics Support. (Java only)
- Thread dump samples: The agent takes a thread dump according to the specified collection interval and specified "number to collect". After the period elapses the thread dumps are uploaded by the agent as a zip file. (Java only)

4. (Optional) Enter a name for your request. The name is used to identify your request. It is not used to name the generated zip file.

5. Click **Request Agent Log Files**.

The AppDynamics controller processes your request. The status field changes as follows:

- PENDING - request submitted
- IN_PROGRESS - request being processed
- SUCCESSFUL - request is ready

When the status reads as SUCCESSFUL, you can click the icon at the end of the line to download the zip file to your local machine for review as shown in the following screen capture.

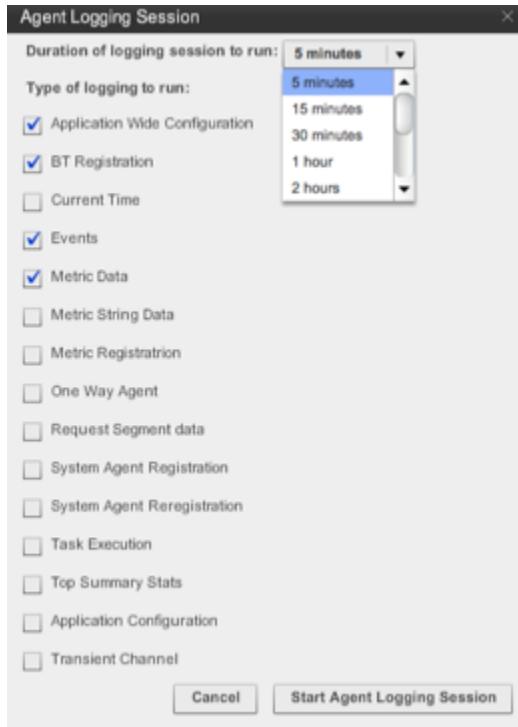
The screenshot shows the 'Agent Logs' section of the AppDynamics interface. A table lists a single entry: 'test' (Name), 'Zipped agent logs directory' (Description), '10/03/12 2:36:25 PM' (Date Log Request), and 'SUCCESSFUL' (Status). To the right of the status column is a small blue download icon with a white arrow. A callout box with a blue border and white text 'Click to download the zip file.' points to this icon. A red arrow also points from the text to the download icon.

To Configure and Start an Agent Logging Session (Java only)

1. In the **Agents** tab of the node dashboard, scroll down to the Agent Logs panel.

2. Click **Start Agent Logging Session**.

The Agent Logging Session window opens.



3. From the drop-down menu select the duration for which you want to log.

4. Check the check boxes for the types of requests that you want to log.

5. Click **Start Agent Logging Session**.

The selected logging sessions appear in the logging list.

The logged request and response output appears in the Controller and the agent log.

Machine Server Agent Subtab

This subtab describes the properties of the machine agent, if a machine agent is installed on the machine that hosts the node.

Agent Diagnostic Stats Subtab

This subtab displays agent diagnostic statistics, such as how many transactions successfully registered or how many metrics were uploaded. You can click a statistic to see it graphed in the Metric Browser.

See [App Agent for Java Diagnostic Data](#).

Learn More

- [App Agent Node Properties](#)
- [Filter and Analyze Events](#)

App Server List

- [Accessing the App Servers List](#)
- [Viewing the App Server List](#)
- [How the App Server List is Organized](#)
 - [Health Tab](#)
 - [Hardware Tab](#)
 - [Memory Tab](#)
- [Learn More](#)

This topic describes the application server list.

Accessing the App Servers List

AppDynamics displays all application servers in a business application on the application server list.

To access this list, in the left navigation pane, click **Server > App Servers**.

Viewing the App Server List

- To sort the servers based on the data in a particular column, click the column header.
- To filter the list, enter the app server to filter on in the filter field.
- To view the list in organized by app server, click the tree icon. To see it organized by node, click the grid icon.

How the App Server List is Organized

The app server list has three tabs: **Health**, **Hardware**, and **Memory**. Click the tab that matches the type of information that you are interested in.

Health Tab

The **Health** tab has the following columns:

- **# of Nodes**
This column shows the number of nodes (app servers) in the tier. This column is visible only in tree view.
- **Health**
The color of the Health icon describes the extent to which a server, or one of its nodes, is experiencing node-level health rule violations. Healthy servers green; servers with warning-level violations are yellow/orange; servers with critical-level violations are red.
For more information see See [Tutorial - Server Health](#) and [Troubleshoot Health Rule Violations](#).
- **App Agent Status**
A green up-arrow icon indicates that the app agent is reporting to the controller.
A red down-arrow icon indicates that the app agent is not reporting to the controller.
If the app agent is not reporting, see:
 - [Troubleshoot App Agent for Java](#)
 - [Troubleshoot App Agent for .NET Installation and Configuration](#)
- **App Agent Version**
This column shows the version of the app agent running on the node.

- **JVM** (Java only)
This column shows the name and version of the JVM .
- **Last JVM Restart** (Java only)
This column shows the timestamp of the last JVM restart.
- **CLR** (.NET only)
This column shows the name and version of the CLR.
- **Last CLR Restart** (.NET only)
This column shows the timestamp of the last CLR restart.
- **Machine Agent Status**
A green up-arrow icon indicates that the machine agent is reporting to the controller.
A red down-arrow icon indicates that the machine agent is not reporting to the controller.
If the machine agent is not reporting, see [Troubleshooting Machine Agent Installation](#).

Hardware Tab

If a machine agent is not installed on the physical device, no data will be available in the **Hardware** tab.

The **Hardware** tab has the following columns:

- **CPU % (current)**
- **CPU % (average)** - over the selected time range
- **Memory % (current)**
- **Memory % (average)** - over the selected time range
- **Disk IO KB reads/sec**
- **Disk IO KB writes/sec**
- **Network IO KB reads/sec**
- **Network IO KB writes/sec**

For more information see [Monitor Hardware](#).

Memory Tab

The **Memory** tab has the following columns for Java servers:

- **JVM % Heap**
- **Max Heap**
- **JVM CPU Burnt (ms/min)**
- **GC Time Spent (ms/min)**
- **Major Collections**
- **Major Col time min (ms)**
- **Minor Col time min (ms)**

For more information, see [Troubleshoot Java Memory Issues](#).

The **Memory** tab has the following columns for .NET servers:

- **Current Heap Utilization (MB)**
- **Average Heap Utilization (MB)** - over the selected time range
- **Committed Heap**
- **CPU Burnt (% proc time)**
- **Induced Collections**
- **Time Spent on Collections (%)**

Learn More

- [Monitor App Servers](#)
- [Logical Model](#)
- [Dashboards](#)

Databases List and Dashboard

- [The Database Server List](#)
 - To access the Database Server List
 - To delete databases from the list

- The Database Dashboard
 - To access a Databases Dashboard
- How the Database Dashboard is Organized
 - Action Menu
 - Database Tabs
 - Dashboard Tab
 - Compare Menu
 - Slowest Database Calls Tab
- Linking to AppDynamics for Databases
- Learn More

Each database has its own dashboard and is listed in the Database Servers List.

The Database Server List

AppDynamics displays all detected database servers on the database server list along with the key performance indicators of calls to them: response time, calls, calls per minute, errors, and errors per minute.

To access the Database Server List

In the left navigation pane, click **Servers -> Databases**.

- To sort the servers based on the data in a particular column, click the column header.
- To filter the list, enter the string to filter on in the filter field.

To delete databases from the list

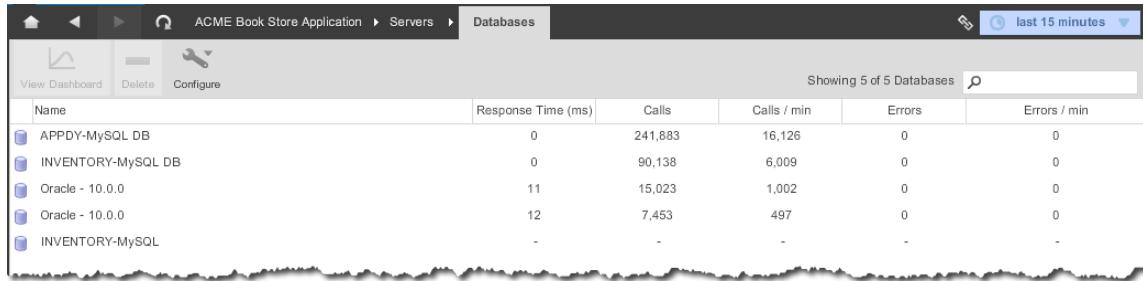
1. Select the database or databases that you want to delete from the list.

2. Click the delete icon.

The database or databases no longer appear in the UI.

The Database Dashboard

Access a dashboard from the Databases List.

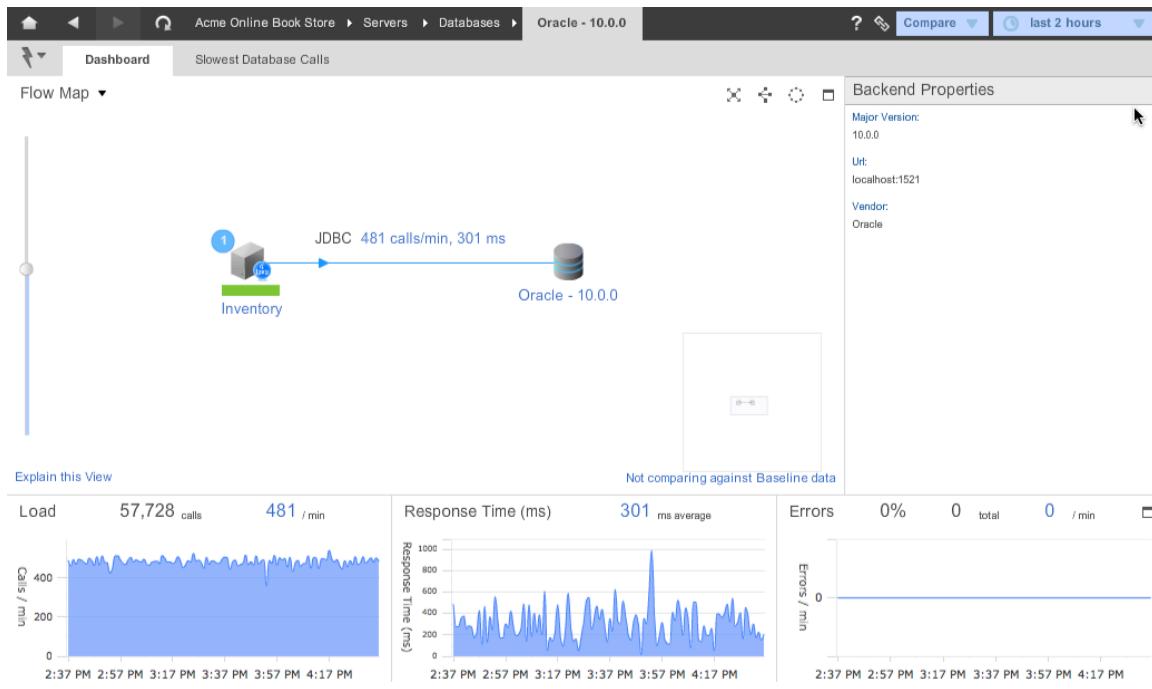


To access a Databases Dashboard

1. In the left navigation pane click **Servers -> Databases**.

2. From the Databases List select a database.

3. Click **View Dashboard**.



How the Database Dashboard is Organized

Action Menu

The Action menu provides these actions for a remote service:

- **Rename Backend:** Renames the database.
- **Resolve Backend to Tier:** Associates the database with the tier that you select so that the backend appears in the grid view of the tier and not as an independent component ("unresolved backend") on the application dashboard flow map. You can reverse this operation from the **Configure Backends resolving to this Tier** item in Actions menu in the tier dashboard.
- **Delete Backends:** Removes instances of the database from the controller and all agents. An agent can re-discover the database and register it with the controller.

Database Tabs

The dashboard has two tabs:

- Dashboard Tab
- Slowest Database Calls Tab

Dashboard Tab

The **Dashboard** tab displays information about calls to the database:

- Flow map: shows traffic from the calling tier to the database. See [Flow Maps](#).
- Backend Properties: varies according to database type.
- Key Performance Indicators: See [KPI Graphs](#).

Compare Menu

The **Compare** menu is visible when the Dashboard tab is selected.

By default the performance displayed in the dashboard is compared against the daily trend, which is the performance over the last 30 days. From the Compare menu you can direct AppDynamics to use a different baseline or no baseline. You can also configure baselines. For more information about baselines see [Behavior Learning](#) and [Anomaly Detection](#) and [Configure Baselines](#).

Slowest Database Calls Tab

The **Slowest Database Calls** tab lists up to ten calls to the database with the longest execution time, by tier and for all tiers.

Each call shows the SQL Query, Average Time, Number of Calls during the time range, and maximum execution time (Max Time). The Max Time is used to determine which calls are displayed in the Slowest Database Calls list.

If transaction snapshots are available for a slow call, you can click **View Snapshots** link. From there you can select a snapshot and click **View Transaction Snapshot** to drill down to the root cause of the slow database call.

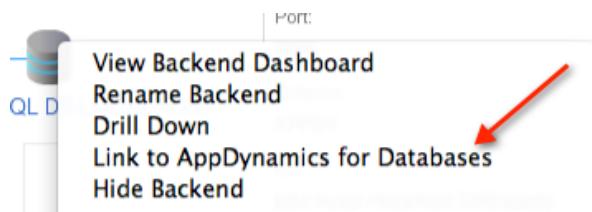
The screenshot shows the AppDynamics interface for the ACME Book Store Application. The top navigation bar includes 'Dashboard', 'Servers', 'Databases', 'Oracle - 10.0.0', and a search bar set to 'last 6 hours'. The main area is titled 'Slowest Database Calls' and displays a table of queries. One row is highlighted for the query 'INSERT INTO ORDERREQUEST (ITEM_ID, NOTES) VALUES ('3', 'A1 258.9')'. A green callout bubble points to the 'View snapshots' link in the 'Schemas' column of this row. Another green callout bubble points to the 'View Transaction Snapshot' link in the 'Correlated Snapshots' tab of the detailed view below. The detailed view shows a table of transaction snapshots with columns: Time, Exe Time (ms), URL, Business Transaction, Tier, and Node. Four rows are listed, all corresponding to the same business transaction and node.

Query	Average Time	Number of Calls	Max Time	Schemas
INSERT INTO ORDERREQUEST (ITEM_ID, NOTES) VALUES ('3', 'A1 258.9')	2130	10002	10002	View snapshots

Time	Exe Time (ms)	URL	Business Transaction	Tier	Node
09/12/12 1:05:32 PM	10006	/cart/services/C	ViewCart.sendItems	Inventory Serv...	Node_8002
09/12/12 1:05:42 PM	10007	/cart/services/C	ViewCart.sendItems	Inventory Serv...	Node_8002
09/12/12 1:05:52 PM	10007	/cart/services/C	ViewCart.sendItems	Inventory Serv...	Node_8002
09/12/12 1:06:02 PM	10006	/cart/services/C	ViewCart.sendItems	Inventory Serv...	Node_8002

Linking to AppDynamics for Databases

Users of AppDynamics for Databases can link to that product by right-clicking on a database from the database list or from the database icon on any flow map.



This linkage gives you deep visibility into the performance of an application's SQL queries.

Learn More

- Dashboards
- Flow Maps
- Transaction Snapshots
- Troubleshoot Slow Response Times
- Behavior Learning and Anomaly Detection
- Configure Baselines
- Integrate with AppDynamics for Databases

Remote Services Dashboard

- Accessing a Remote Services Dashboard
 - To access a Remote Service Dashboard
- How the Remote Services Dashboard is Organized
 - Action Menu
 - Remote Service Tabs
 - Dashboard Tab

- Compare Menu
- Slowest Remote Service Calls Tab
- To drill down into the Slowest Remote Services
- Learn More

Each remote service has its own dashboard.

Accessing a Remote Services Dashboard

Access a dashboard from the Remote Services List.

To access a Remote Service Dashboard

1. In the left navigation pane select **Servers -> Remote Services**.
2. From the Remote Services List select a remote service.

The screenshot shows a browser window with the URL 'ACME Book Store Application > Servers > Remote Services'. The main content area displays a table titled 'Showing 1 of 1 Remote Services'. The table has columns: Name, Response Time (ms), Calls, Calls / min, Errors, and Errors / min. One row is present: 'Active MQ-OrderQueue' with values 2, 15,068, 1,005, 0, and 0 respectively. At the top of the table are buttons for 'View Dashboard', 'Delete', and 'Configure'.

3. Click **View Dashboard**.

The screenshot shows a detailed dashboard for the 'Active MQ-OrderQueue' service. The top navigation bar includes 'ACME Book Store Application > Servers > Remote Services > Active MQ-OrderQueue'. The main area features a 'Flow Map' diagram showing the flow of calls between an 'Order Processing Server' and an 'ECommerce Server' through a 'Queue', with metrics like 'JMS 129 calls/min, - ms'. To the right is a 'Backend Properties' panel listing 'Destination Name: OrderQueue', 'Destination Type: QUEUE', and 'Vendor: Active MQ'. Below the flow map are three performance charts: 'Load' (15,496 calls, 129/min), 'Response Time (ms)' (4 ms average), and 'Errors' (0%). A note at the bottom states 'Not comparing against Baseline data'.

How the Remote Services Dashboard is Organized

Action Menu

The Action menu provides these actions for a remote service:

- **Rename Backend:** Renames the remote service
- **Resolve Backend to Tier:** Associates the service with the tier that you select so that the backend appears in the grid view of the tier and not as an independent component ("unresolved backend") on the application dashboard flow map. You can reverse this operation from the **Configure Backends resolving to this Tier** item in Actions menu in the tier dashboard.
- **Delete Backends:** Removes instances of the service from the controller and all agents. An agent can re-discover the service and register it with the controller.

Remote Service Tabs

The dashboard has these tabs:

- Dashboard Tab
- Slowest Remote Service Calls Tab

Dashboard Tab

The **Dashboard** tab displays information about calls to the service:

- Flow map: shows traffic to and from the service. See [Flow Maps](#).
- Backend Properties: varies according to service type
- Key Performance Indicators: See [KPI Graphs](#).

In the flow map, click the call from the app server to the remote service to view the key performance indicators (KPIs) and the business transaction breakdown for the calls to the service.

Compare Menu

The **Compare** menu is visible when the Dashboard tab is selected.

By default the performance displayed in the dashboard is compared against the daily trend, which is the performance over the last 30 days. From the Compare menu you can direct AppDynamics to use a different baseline or no baseline. You can also configure baselines. For more information about baselines see [Behavior Learning and Anomaly Detection](#) and [Configure Baselines](#).

Slowest Remote Service Calls Tab

The **Slowest Remote Service Calls** tab lists the ten calls to the service with the longest execution time, by tier.

Each call shows the Average Time, Number of Calls during the time range, and Max Time. The Max Time is used to determine which calls are in the list.

To drill down into the Slowest Remote Services

1. Select the call in the list.

The query and its key performance metrics display in the right panel.

2. If transaction snapshots are available for the selected call, click either **View Snapshots** at the end of the query row or the **Correlated Snapshots** tab in the lower panel.

3. From the snapshots list that appears select the snapshot that you want to examine and click **View Transaction Snapshot**.

4. In the snapshot flow map that displays, click **Drill Down** to view the transaction snapshot.

See [Transaction Snapshots](#).

Learn More

- Troubleshoot Slow Response Times
- Transaction Snapshots
- Dashboards
- Flow Maps
- Behavior Learning and Anomaly Detection
- Configure Baselines

EUM Dashboard

- Accessing the EUM Dashboard
- How the EUM Dashboard is Organized
- Learn More

Accessing the EUM Dashboard

If end user monitoring is not configured for your application, you will not see the EUM menu item or any EUM data. See [Configure End User Experience](#) for information about configuring EUM.

To access the End User Monitoring (EUM) Dashboard:

1. Select the business application for which EUM has been enabled..
2. In the left navigation pane, click **End User Experience**.
The EUM Dashboard opens.



How the EUM Dashboard is Organized

The EUM Dashboard displays end user experience information in two tabs:

- Geographic Distribution
- Browser Distribution

The Geographic Distribution view consists of three panels:

- A main panel in the upper left containing that displays geographic distribution on a map or a grid
- A panel on the right displaying summary information: total end user response time, page render time, network time and server time
- Trend graphs in the lower part of the dashboard that dynamically display data based on the level of information displayed in the other two panels

The geographic region for which the data is displayed throughout the dashboard is based on the region currently showed on the map or in the summary panel. For example, if you zoom down from global view to France in the map, the summary panel and the graphs display data for France.

The Browser Distribution view displays key performance indicators and graphs of average response time and by browser and browser version.

Learn More

- Monitor End User Experience (EUM)
- Configure End User Experience

Custom Dashboards

- Major Revision in 3.7.0
- Suggestions for Custom Dashboards
 - Key Performance Indicator Dashboard

- System Operations Dashboard
- DevOps Dashboard
- Learn More

Major Revision in 3.7.0

In release 3.7.0 custom dashboard functionality is completely revised, including:

- A contemporary look-and-feel
- Dashboards now render in HTML5 for viewing on many devices
- New widgets for image and iframe
- New editor supports advanced metrics including "Top 10" lists
- Full dashboards can be embedded in other dashboards using sharing, iframes, and URLs
- External data can be embedded in dashboards using iframes and URLs
- Support for metric wild-carding such as "all nodes whose names start with..."
- Supports anonymous sharing to provide access without requiring credentials
- Can copy dashboards on the same Controller
- Can include baseline data in charts

Pre-3.7 dashboards are still supported and can be edited using pre-3.7 functionality. However pre-3.7 dashboards cannot be imported to the new 3.7 editor.

Dashboard import/export is not supported.

The scatterplot widget is no longer provided.

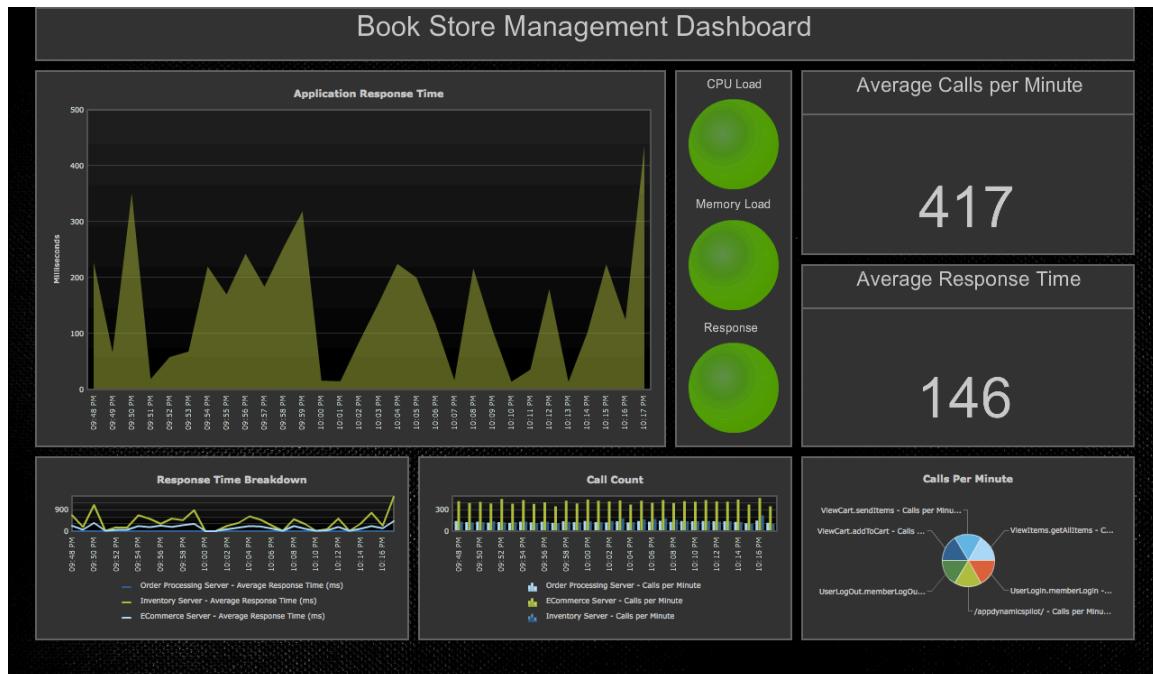
Suggestions for Custom Dashboards

A custom dashboard brings together the various metrics and policies that are important to your monitoring activities, all on one screen.

You can create a custom dashboard to:

- Provide a view of application performance customized to your needs
- Aggregate data from different applications
- Compare data from different applications
- Show a single view of both live and historical data
- Share with other users and stakeholders

Key Performance Indicator Dashboard



System Operations Dashboard

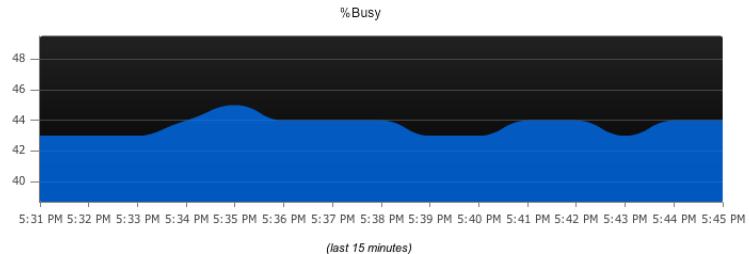
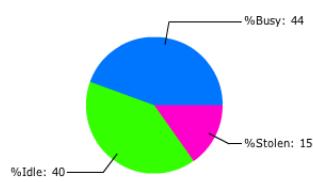
System Operations

ACME Book Store Application

CPU



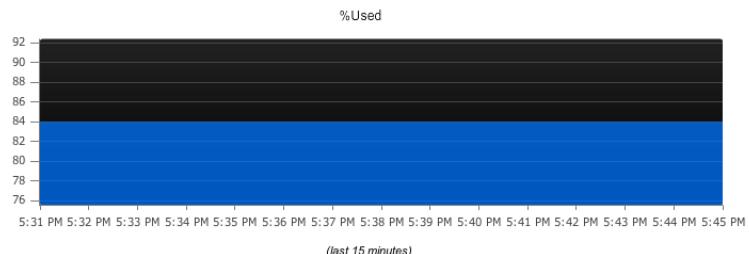
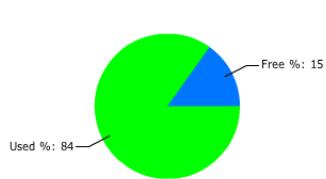
CPU Usage



Memory



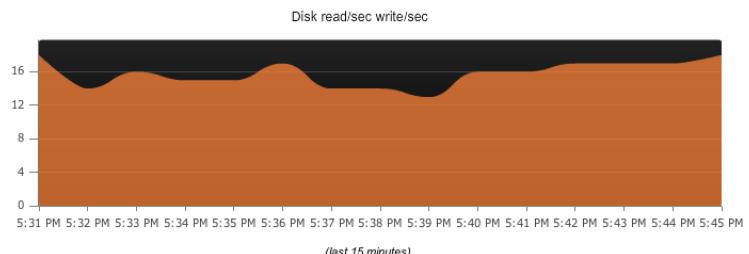
Memory Usage



Disk I/O



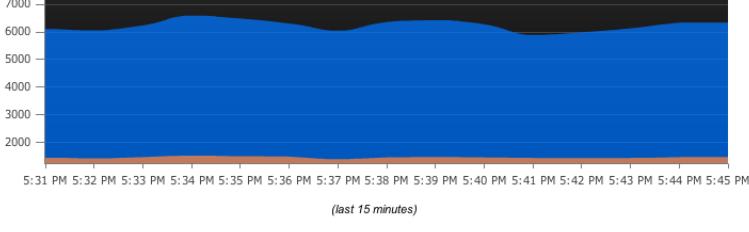
Disk Usage



Network Activity



Network Usage

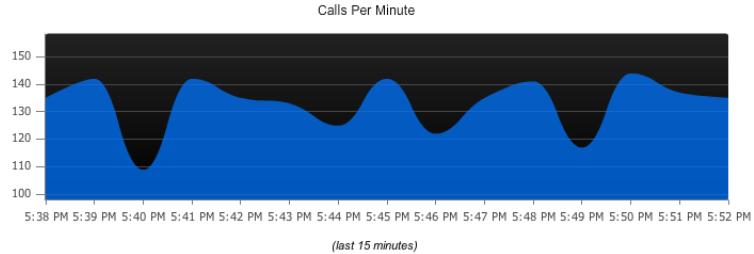


DevOps Dashboard

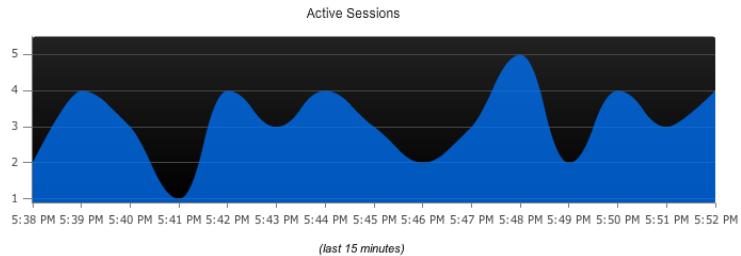
DevOps Dashboard

ACME Book Store Application

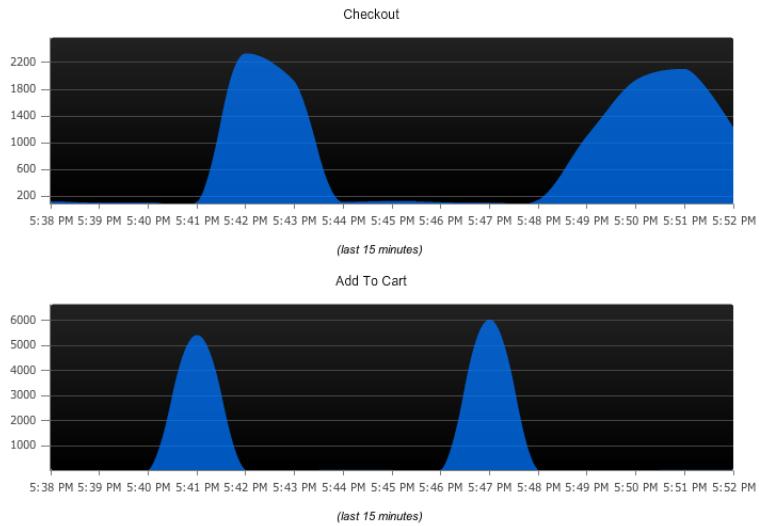
Calls Per Minute



Active Sessions



Business Transaction ART



Learn More

- Use a Custom Dashboard
- Create a Custom Dashboard
- Configure Roles
- To learn about the out-of-the-box dashboards see [Dashboards](#).

Create a Custom Dashboard

- Using Dashboard Widgets
- Creating a Custom Dashboard

- To create a custom dashboard
- To add a widget
- To specify a drilldown URL
- To specify a metric to display for a metric value widget
- To display metrics in a graph
- To display metrics in a pie chart
- To specify series to display for a graph or pie chart widget
- To display information from another website or dashboard
- To display health rule status indicators
- [Learn More](#)

This topic describes how to create custom dashboards. To use existing dashboards, see [Use a Custom Dashboard](#).

AppDynamics lists custom dashboards in the **Custom Dashboards** menu on the lower left navigation pane.

Using Dashboard Widgets

With dashboard widgets you can:

- Label your dashboard and status indicators using the label widget.
- Display a metric using a metric value, a graph, or pie chart widget.
- Add health status indicators for critical metrics.

Widgets have basic display properties such as position, size, font, font size, text color, background color, border and border color. You can arrange them on a dashboard as you like.

A widget can optionally have a drill-down URL. You can specify a drill-down URL as a shortcut to help find the root cause of a performance problem. For example, if you are monitoring slow requests for all the nodes of an application you may see an increase in the number of slow requests and want to further investigate the cause. You want to reach a particular node from your dashboard. Clicking on the drill-down URL will take you to the root cause or metric violation.

Each widget has specific characteristics. Widgets and their specific properties include:

- Label - Text used to annotate the dashboard
- Graph - One or more metrics displayed in a graph format
- Metric Value - A string that displays the value of a metric
- IFrame - URL to display content
- Image - URL to an image file
- Pie - Metrics in a pie chart format
- Status Light - Health rule violations

As you set the properties, the appearance of the widgets update.

Creating a Custom Dashboard

To create a custom dashboard

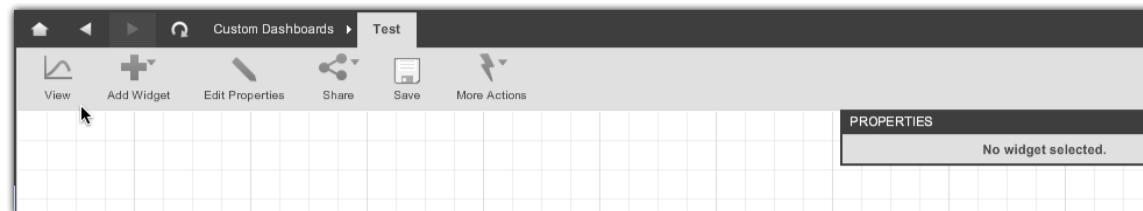
1. From the left navigation pane or from the top right menu bar click **Custom Dashboards**.

2. Click **Create Custom Dashboard**.

3. In the Create New Dashboard window:

- Provide a name for the new dashboard.
- If you want the widgets to refresh less frequently than every 120 seconds, increase the number of seconds.
- If you want the dashboard to be a different size than the defaults, change the width and height.
- If you prefer a background color other than white, use Background Color to pick a color or enter the hex code.
- Click **OK**.

The Dashboard Design window opens.



To add a widget

1. Drag a widget from the Add Widgets drop-down pane to the dashboard.
2. Select the widget to display its Properties pane.
3. In the Properties panel specify the values of its properties. Common properties include:

- **Position/Size:** These properties are the coordinates and dimension of the label on the grid. You can specify these numbers for precise alignment if needed.
- **Design:** Use these properties to set the font size and color, background color, and widget border.
- **Misc:** See [To specify a drill-down URL](#).
- **Time Range:** Select an appropriate time range from the pulldown. This could be the same or a different timeframe than what you set for the default dashboards.

Specific properties for each widget are described later in this topic.

4. When you are finished click Save.

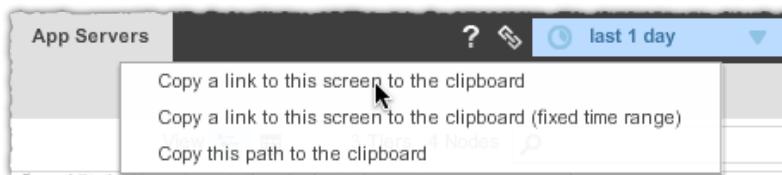
To specify a drilldown URL

You can specify a drilldown URL, a shortcut to an AppDynamics window, that will open when you click on the widget in the dashboard.

For example, if you are monitoring node health for all the tiers of an application you may see an increase in the number of slow requests and want to investigate the cause. You want to reach the App Servers List of all tiers from your dashboard by clicking the widget.

To get the URL:

1. Navigate to the window that you want to open when the widget is clicked.
2. Click the link icon and select **Copy a link to this screen to the clipboard**.



To set the URL:

1. Select the widget.
2. In the Properties window, select **Misc**.
3. Paste the URL.
4. Click "Save" to save the dashboard.

To specify a metric to display for a metric value widget

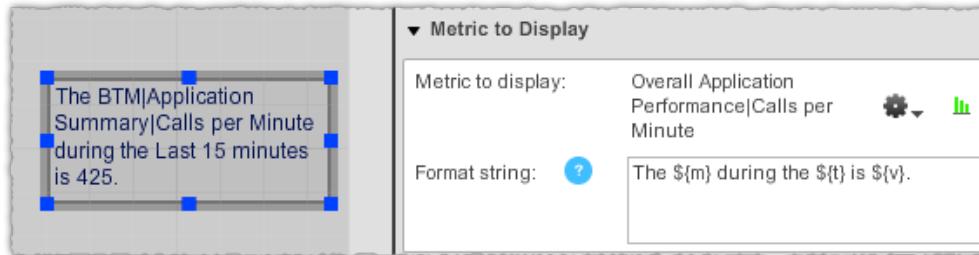
1. Click **Select a metric** and choose an application.
2. Browse the available metrics and select one
3. By default the widget shows the value of the selected metric. You can use the Configure dropdown (the cog icon) to change it to either the minimum or maximum value of the selected metric.
4. If you do not specify the **Format string**, the metric value widget will display the value of the selected metric. You may optionally specify the following format string or strings separated by a space and/or other characters.

Value	Format String
Value of the selected metric	<code> \${v}</code>
Time range as specified in this Properties pane	<code> \${t}</code>

Name of the selected metric	<code> \${m}</code>
-----------------------------	---------------------

For example:

The `${m}` during the `${t}` is `${v}`.



5. Click "Save" to save your updates.

To display metrics in a graph

1. In the Graph section of the Graph Properties pane, enter titles for the following:

- graph
- vertical axis
- horizontal axis

2. If you want the value of the time range to display in the horizontal axis label, check **Show Timerange**.

3. **Background colors** lets you specify two colors that blend into each other. If you only want one color, set the same color for both.

4. **Show Legend** displays the name of the metric.

5. See [To specify series to display for a graph or pie chart widget](#).

6. Click "Save" to save your updates.

To display metrics in a pie chart

1. In the Graph section of the Pie Chart Properties pane:

- Enter a title for the graph.
- If you want to display a legend check **Show Legend**.
- If you don't want to see the labels or values next to the pie chart, uncheck those options.
- If you want to show the values as percentages check **Show Values as %**.

You can use the label widget with a transparent background to further describe the pieces of the pie.

2. See [To specify series to display for a graph or pie chart widget](#). Add additional metrics that make sense for a pie chart; they will display in correct proportion to each other.

3. Click **Save** to save your updates.

[To specify series to display for a graph or pie chart widget](#)

1. Click **Add** (the + icon) to open the Edit Series window.

2. Each series has an internal name that is not displayed on the widget. You can use the default or enter a name more suited to your needs.

3. Select an application.

4. Select a metric category. You can choose from a variety of metrics in the system.

3 Select a Metric Category

Overall Application Performance (load, response time, num slow calls, etc)

4 Overall Application Performance (load, response time, num slow calls, etc)
 Business Transaction Performance (load, response time, slow calls, etc)
 Node Health - Transaction Performance (load, response time, slow calls, etc)
 Node Health - Hardware, JVM, CLR (cpu, heap, disk I/O, etc)
 Node Health - JMX (connection pools, thread pools, etc)
 Error Rates (exceptions, return codes, etc)
5 Custom (use any metrics)

Each category has its own properties to define the metric, such as from a particular tier or using names that meet certain match conditions.

Overall Application Performance: Use a pre-defined metric and refine it in Step 5.

Business Transaction Performance: Use the second pulldown to specify which business transaction or transactions to use. For example you can show metrics for a specific tier:

3 Select a Metric Category

Business Transaction Performance (load, response time, slow calls, etc)

Select which Business Transactions to include in this Series

Business Transactions within the specified Tiers:

Selected Tiers (1)		Other Tiers (2)	
Name	Type	Name	Type
ECommerce Server	Application Server	Inventory Server	Application Server
		Order Processing Server	Application Server

< ADD REMOVE >

Tip: You can drag items between these lists

Refresh List

Node Health - Transaction Performance and Node Health - Hardware, etc.: Use the options and pulldowns to specify tiers and nodes. For example to show the metrics by tier:

3 Select a Metric Category

Node Health - Hardware, JVM, CLR (cpu, heap, disk I/O, etc)

Select what to include in this Series

Tiers
 Nodes

Select which Tiers to include in this Series

All Tiers in the Application

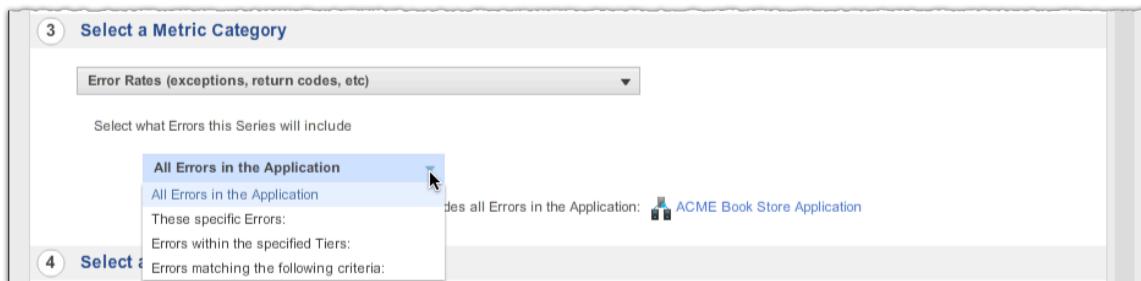
All Tiers in the Application
 These specific Tiers

Series includes all Tiers in the Application: ACME Book Store Application

Node Health - JMX: Use the options and metric browser to select specific JMX objects. For example to select all JDBC connection pools:

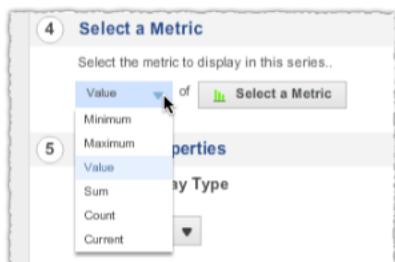


Error Rates: Use the pulldown to specify all errors, specific errors, errors in specific tiers, or errors that match naming conditions, including regular expressions.

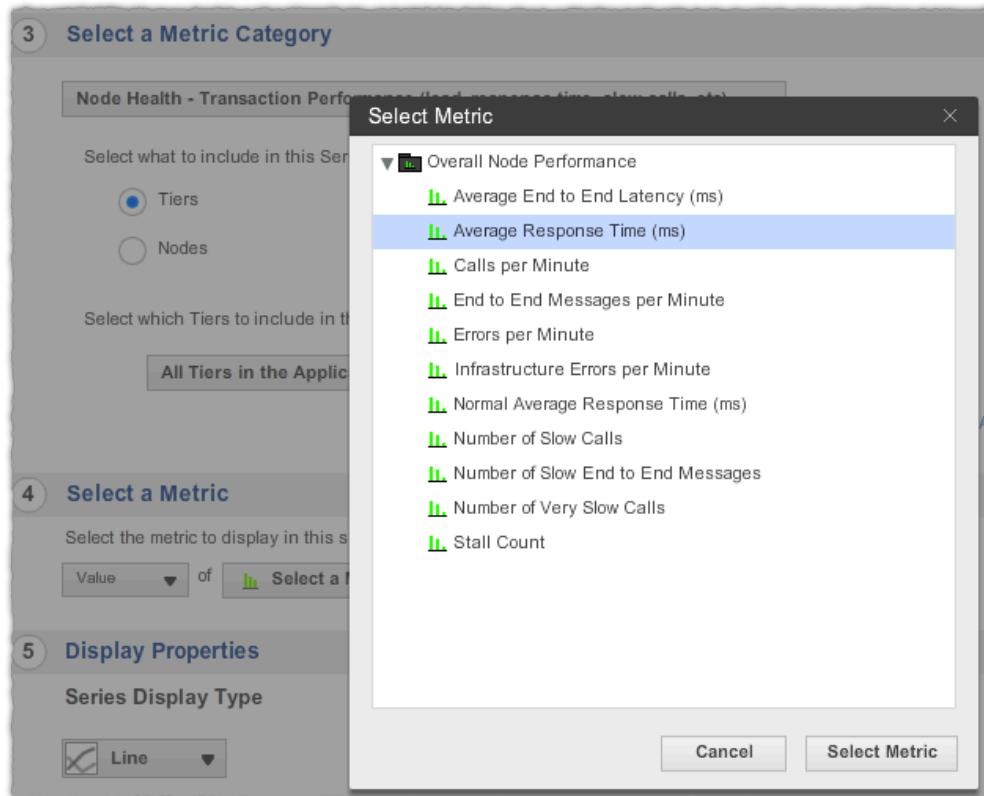


Custom: Use any metrics that are available in Step 5.

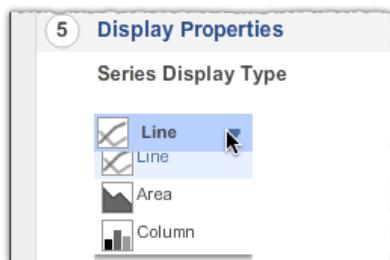
5. Select a value match condition and a metric from the category.



For example, if you chose a category of Node Health - Transaction Performance on all tiers, you can use the value of Average Response Time for all tiers:

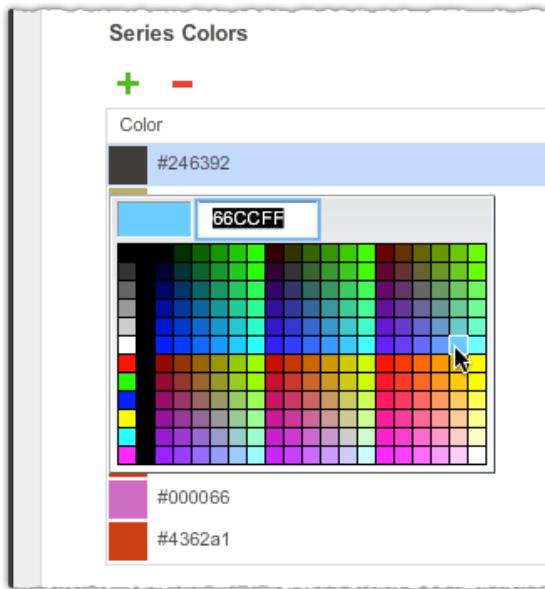


6. Select Line, Area, or Column from the **Display Type** pulldown.



7. Select a color from the **Series Colors** list. You can add your own colors:

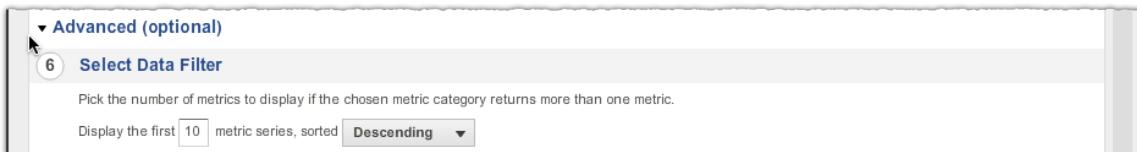
- Click an existing color square.
- Use the color picker to select a new color or type in your color hex code.



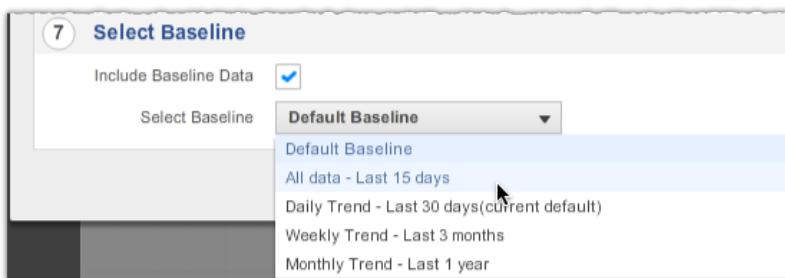
c. Click **Add** (the plus sign). The new color is added to the list.

8. By default AppDynamics formats the metric name. If you want to use the original metric name check **Use Raw Metric Name in Legend**.

9. Some metric categories return more than one metric. By default AppDynamics will display them all. If you want to display only some of them, click **Advanced (optional)** and enter the number of metrics and sort order to display. For example, to display the "Top 10":



10. It can be helpful to have baseline data in a graph to compare against current data. If you want to add baseline data, check **Include Baseline Data** and select a baseline to use. See [Configure Baselines](#).



11. Optionally add additional metrics. The metrics will display with each other.

i Note: Multiple metrics may expand the vertical axis such that some metrics may be difficult to see. You can resize the widget to make multiple metrics more visible.

12. Click Save to save the dashboard.

To display information from another website or dashboard

Use the iFrame widget to display another URL inside a dashboard in HTML mode.

1. In the IFrame text box, enter the URL to display.

2. Adjust the size of the IFrame so that it is larger than the website or dashboard. If using a dashboard, add space at the top for the embedded dashboard's header.
3. Click **Save** to save your updates.
4. Click **View** then **View in HTML** to see the URL in your dashboard. AppDynamics opens another browser tab or window to show you the HTML view.
5. Return to Edit mode to resize the widget as needed.
6. Click **Save** to save your updates.

To display health rule status indicators

1. Drag a status indicator widget from the **Add Widgets** dropdown to the dashboard.
2. In the Health Rules section of the properties pane, click **Select Health Rule**.
3. Select the application.
4. Select a health rule and click **Select Health Rule**.
5. Click **Save** to save your updates.

Learn More

- [Custom Dashboards](#)
- [Use a Custom Dashboard](#)

Use a Custom Dashboard

- [Sharing a Custom Dashboard](#)
 - To share a custom dashboard using a URL
 - To share a custom dashboard with other AppDynamics users
 - To share a custom dashboard using an image file
 - To grant role permission to view a custom dashboard
- [Copying a Custom Dashboard](#)
 - To copy a custom dashboard
- [Embedding a Dashboard in Another Dashboard](#)
 - To embed a dashboard in another dashboard
- [Learn More](#)

To learn how to create custom dashboards see [Create a Custom Dashboard](#).

Sharing a Custom Dashboard

To share a custom dashboard using a URL

You can share a URL of a custom dashboard for use on other computers as well as devices such as tablets and phones.

The Share menu is available in the Custom Dashboards List and in the dashboard Edit view.

- From the **Share** menu, click **Email Shared URL...** to open your default email program, or click **Copy Shared URL to Clipboard** and paste it into your email program.

To share a custom dashboard with other AppDynamics users

1. In the Custom Dashboard List, select a dashboard.

2. Check the box in the Shared column, or click **Share -> Share Dashboard**.

The dashboard is immediately shared and made visible to other AppDynamics users. They will be able to see it in their Custom Dashboards List.

The Share menu is also available in the Edit view.

Alternatively, you can share the link to your dashboard using the **Deep Link** option on the right side of the screen.

To share a custom dashboard using an image file

1. In the dashboard view mode, click **Take Snapshot**.

2. Save the .png file so that you can send it to your colleagues.

To grant role permission to view a custom dashboard

The custom dashboard viewer default role lets you show a "guest" user a selected set of metrics without giving the user access to the regular AppDynamics dashboards. See [Configure Roles](#).

Copying a Custom Dashboard

You can copy the work you've done in a dashboard as a basis for a new dashboard on the same Controller.

To copy a custom dashboard

1. In the Custom Dashboards List, select a dashboard.

2. Click **Copy**.

3. Enter the name of the copied dashboard.

4. Click **Copy Dashboard**.

Embedding a Dashboard in Another Dashboard

You can show one dashboard in another. This way you can reuse graphs and charts. You may want to do this when you have different operators who focus on particular dashboards and you to roll them all up into a "Master Dashboard" for management.

To embed a dashboard in another dashboard

1. Create a dashboard. See [Create a Custom Dashboard](#).

2. Share it via a URL. See [To share a custom dashboard using a URL](#).

3. Create another dashboard.

4. Use the IFrame widget to display the shared dashboard. See [Create a Custom Dashboard#To display information from another website or dashboard](#).

Learn More

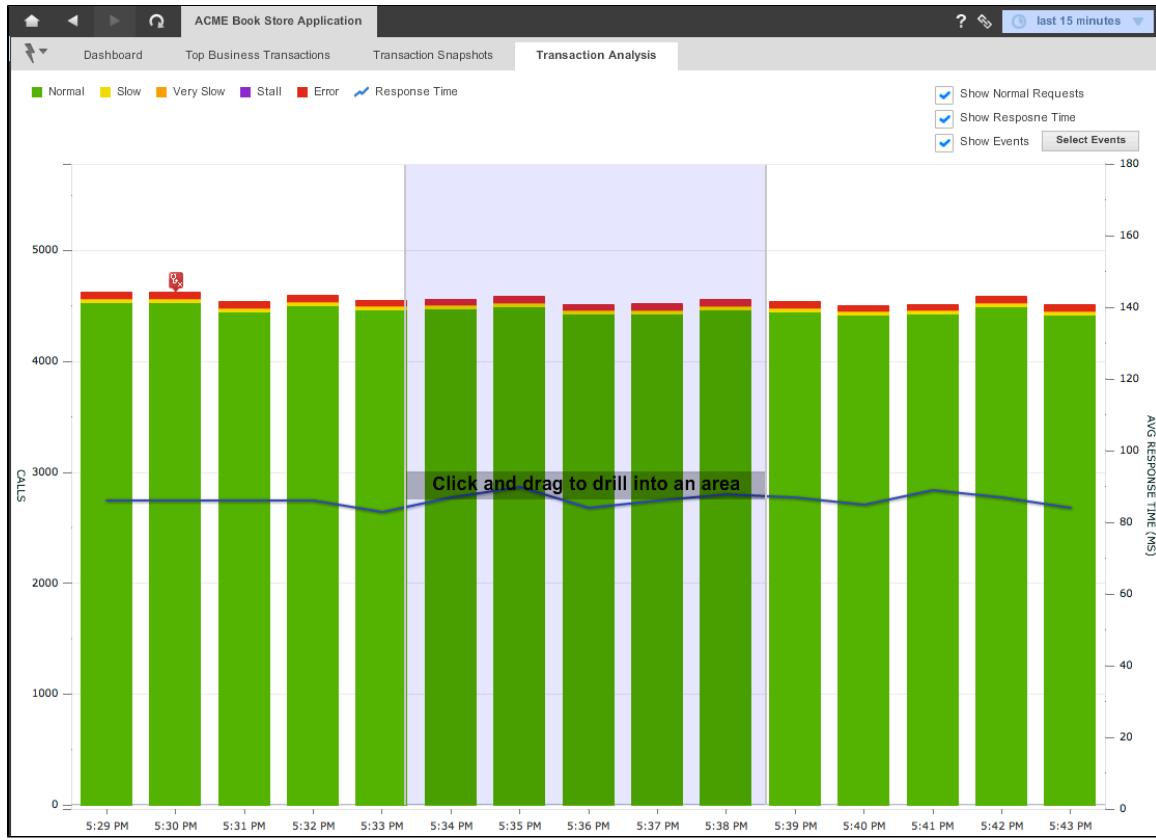
- [Create a Custom Dashboard](#)
- [Custom Dashboards](#)
- [Configure Roles](#)
- To learn about the out-of-the-box dashboards see [Dashboards](#).

Transaction Analysis Tab

- [Accessing the Transaction Analysis Tab](#)
- [How Transaction Analysis is Organized](#)
 - [Time Slices](#)
 - [Average Response Time](#)
- [Statistics for a Time Slice](#)

The Transaction Analysis tab represents the response time, which has a direct effect upon the user experience, of a business application as a histogram. AppDynamics classifies the response time based on the patterns of business transactions.

Accessing the Transaction Analysis Tab



How Transaction Analysis is Organized

Time Slices

The vertical bars represent a slice of time against which AppDynamics has gathered performance metrics such as response time. The bar metrics are stacked and color-coded according to the response times of the requests, as:

- Normal - green
- Slow - yellow
- Very Slow - red
- Stalls - grey

You can see details by hovering over different parts of the graph.

Use the **Time Range** menu in the upper right to change the time period for which data is displayed.

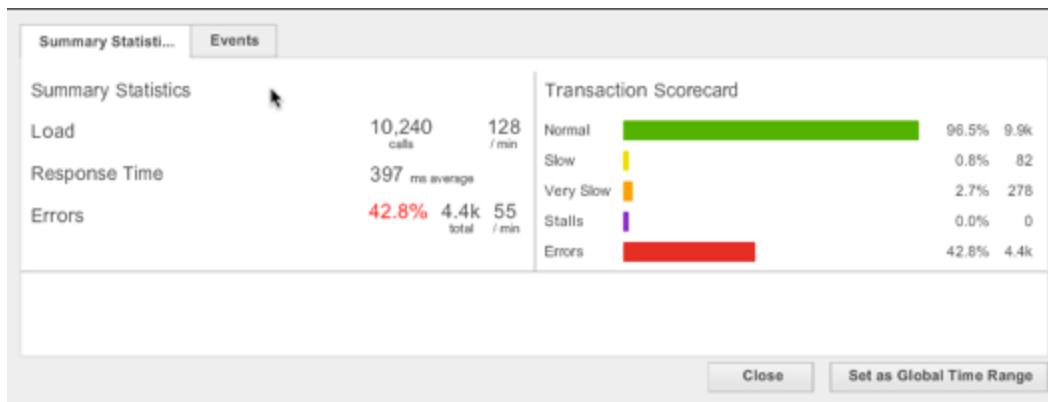
Average Response Time

The blue line represents the average response time. Spikes in response time indicate that the user experience is degrading.

The average response time dynamic baseline appears as a dotted line.

Statistics for a Time Slice

You can select a time slice by dragging your pointing device over the histogram. This displays a window that summarizes the key performance indicators and transaction scorecard over the selected time slice.



You can click **Select as Global Time Range** to reset the time range used in other screens in the AppDynamics UI to the time slice that you have set in the transaction analysis screen. This initiates a reload of the dashboards.

Click the **Events** tab to see the events generated over the selected time period. See [Events](#).

Flow Maps

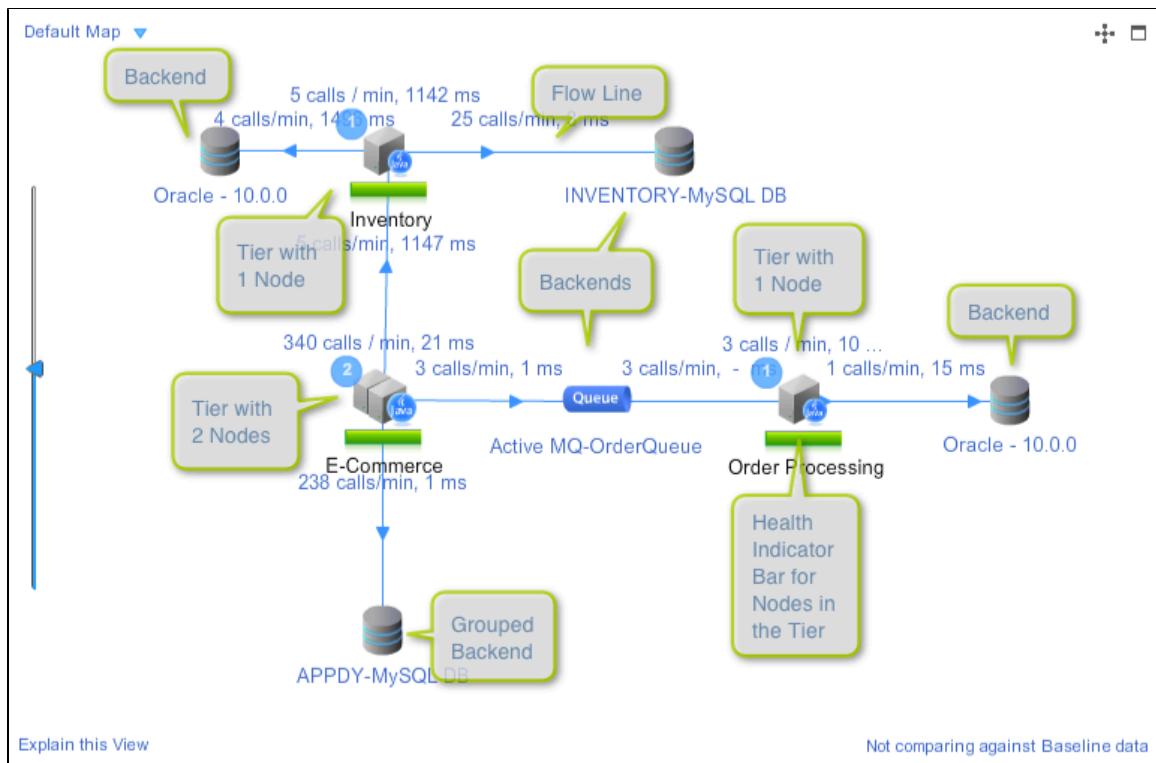
- How Flow Maps Visualize Business Transactions across an Application
- Interacting with Flow Maps
 - Comparing Against Baseline Data
- Getting the Most Out of Flow Maps
 - Individual Customized Flow Maps
 - Shared Flow Maps
- Flow Map Scope and Inheritance
- Changing the Layout of the Flow Map
 - To move the entire map
 - To automatically arrange the layout
 - To move the flow map icons
 - To maximize the size of the layout
 - To zoom in or out of the layout
 - To specify the region of the flow map to be displayed
 - To rename the tier, node, or backend icons
- Configuring a Custom Flow Map
 - To configure a custom flow map
- Editing Existing Flow Maps
 - To edit a flow map
 - To copy a flow map
 - To delete a flow map
- Learn More

This topic describes AppDynamics flow maps and how to use them to monitor applications.

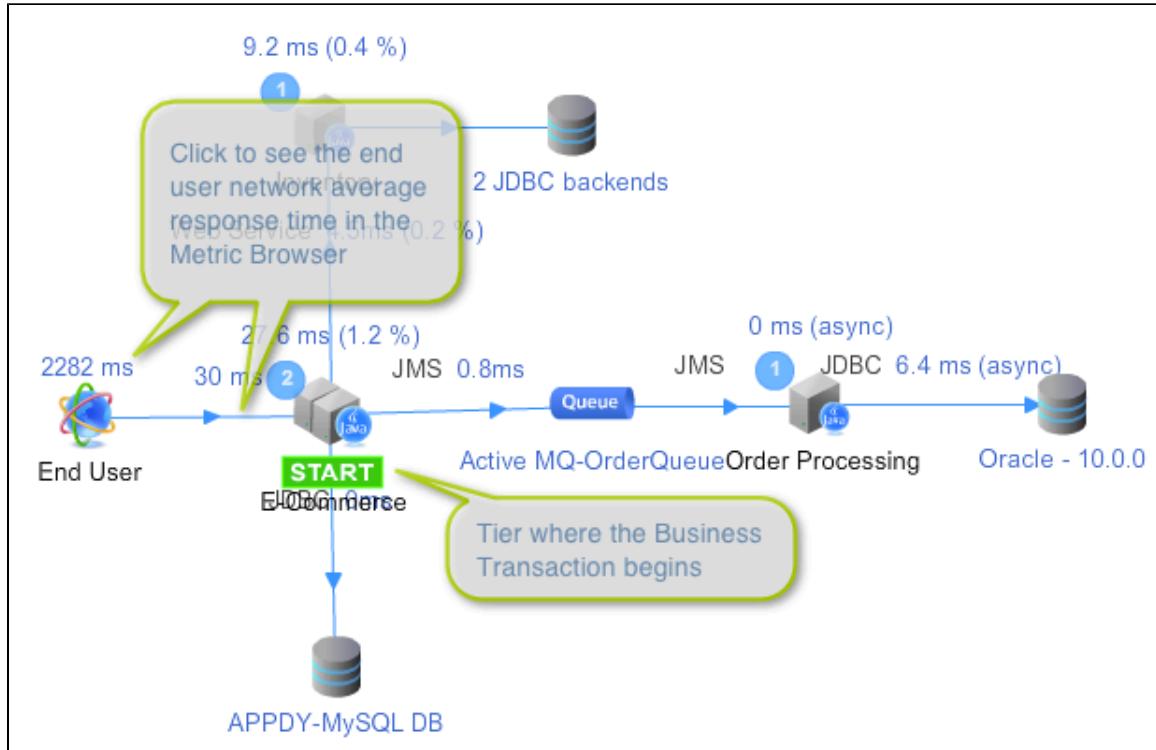
How Flow Maps Visualize Business Transactions across an Application

An AppDynamics flow map is a graphical representation of the tiers and backends in the application. Depending upon [flow map customizations](#), some tiers and backends may be grouped together or hidden from view.

Flow maps are an integral part of the application, tier, node, and business transaction dashboards. They have similar aspects and may differ slightly. For example, the Application Flow Map shows the topology of the application.

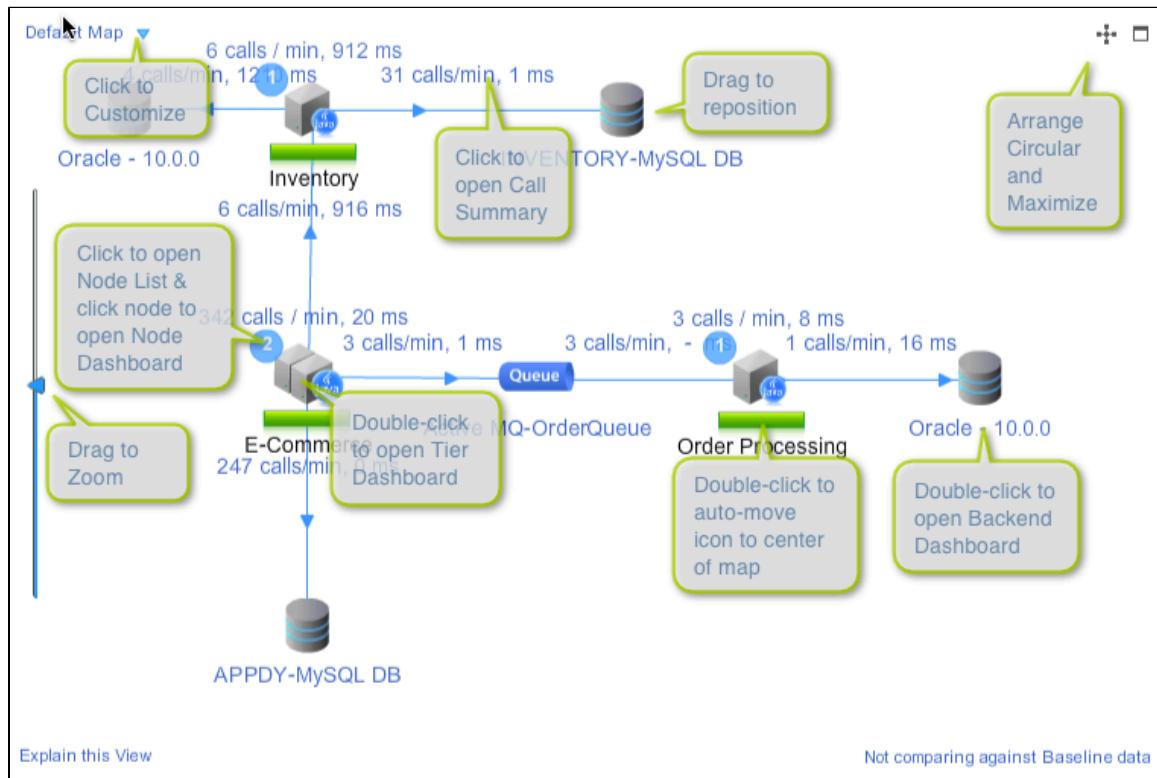


The Business Transaction Flow Map shows the activity of a single business transaction. It defines the node that starts the transaction, and includes the end user performance information. See [Business Transaction Monitoring](#).



Interacting with Flow Maps

You can click, drag, hover, and double-click items in the flow map to change how it looks, how much data is represented, or to drill down into more detailed information. You can right-click on an object and select actions from the context menu.



For all dashboards, the **Time Range** option determines how much data, using a given time range, is represented in the visualization.

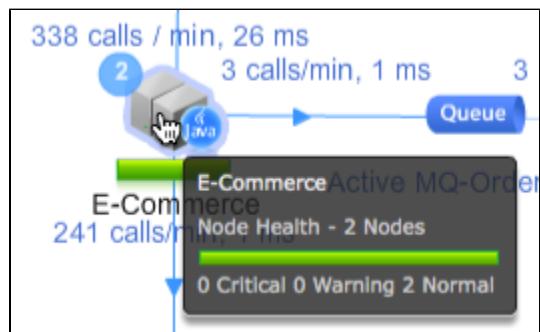
Activity in the current time range is represented by flow lines. Icons represent the tiers and backends, and flow lines represent the traffic between them.

You can hover the cursor over a tier icon to see its current node health indicators and to activate the animated arrows in the flow lines that show the direction of the traffic.

Annotated links show the current calls per minute and average call duration metrics. Click the flow line or its link to see more details about response time, call metrics, and errors. See [Measure Distributed Transaction Performance](#).

Comparing Against Baseline Data

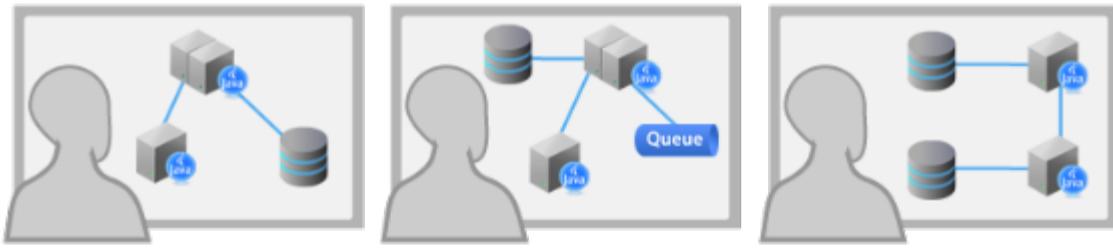
On the lower right of the flow map, an informational link tells you whether the flow map is using any baseline data. You can click on this link to find out more.



Getting the Most Out of Flow Maps

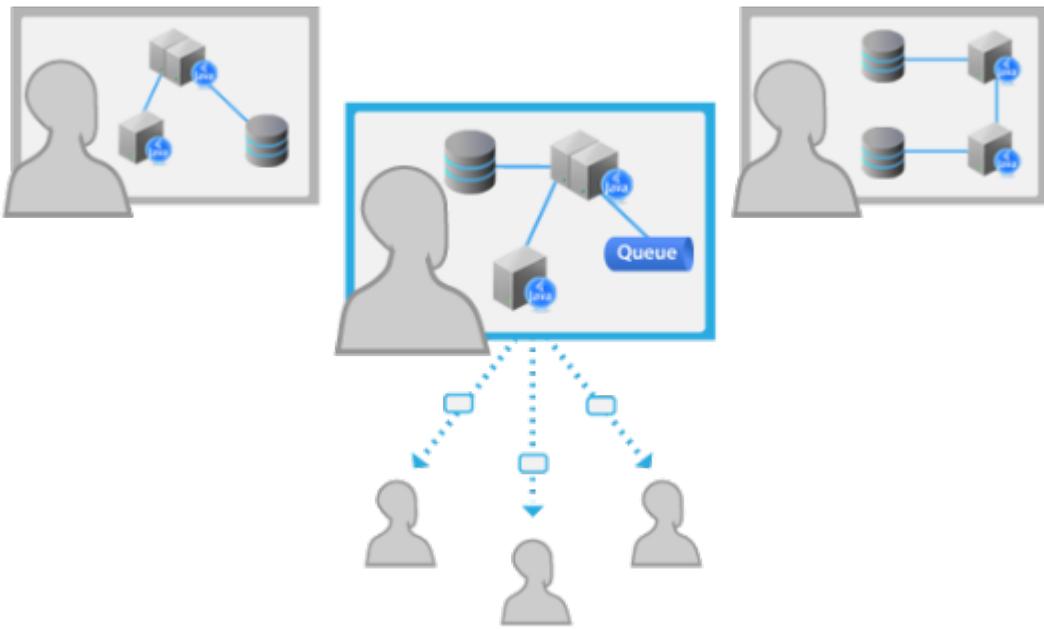
Individual Customized Flow Maps

When a team is monitoring an application environment, each team member can have their own view into the business application. Each user can create a custom flow map and save it for their own use.



Shared Flow Maps

You can share a single flow map with the entire team. When you share a custom flow map, it is listed in the Default Map menu for all users.



Flow Map Scope and Inheritance

By default the scope of a flow map depends on the context of the dashboard; that is, the application, business transaction, tier, or node. If there is not a flow map for the specific object, the UI will use the flow map of the parent object. For example if there is no custom flow map for a given node, the UI will first look for its tier flow map and if there is no tier flow map it will use the application flow map.

The application scope is the highest scope, so by default all flow maps use this scope. To better monitor at different levels of scoping, you can customize flow maps at all levels of the hierarchy and save them. See [Configuring a Custom Flow Map](#).

Changing the Layout of the Flow Map

To move the entire map

Click and hold in the background of the icons and flows, and move the mouse to reposition the map. To save this change for the next time you open the dashboard, click **Default Map -> Save Current Map**.

To automatically arrange the layout

The view option icons at the top right let you toggle the dashboard display modes between views of the flow map:

- Graphical flow map view
- Flow list grid view
- Infrastructure list (tiers and nodes)
- Auto-fit flow map contents to the screen
- Arrange Circular
- Maximize



Click **Arrange Circular** (the icon) to let AppDynamics rearrange the flow map in a circular pattern.

Tier, node, and backend dashboards also have an **Arrange Tree** (the icon) option.

To move the flow map icons

To move a flow map icon, click and drag it to a new position. The flow lines will follow and stay with the icon. When you move a tier or backend icon the flow map automatically saves the new location.

To maximize the size of the layout

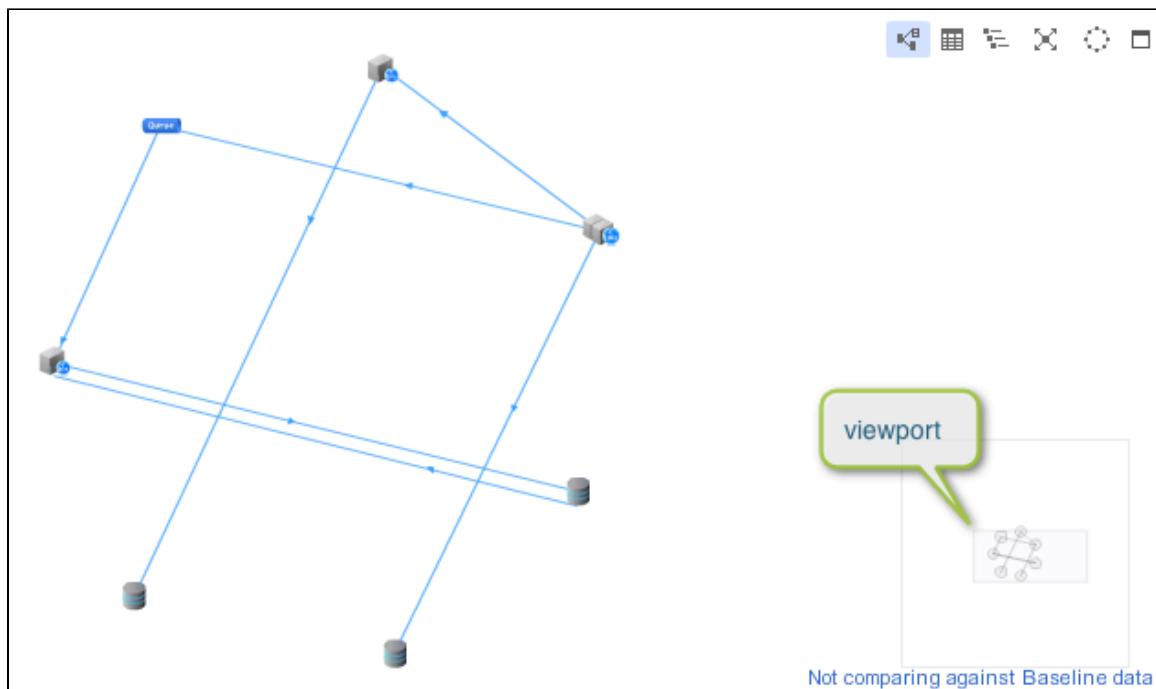
To have the flow map take up the entire AppDynamics window, click **Maximize** (the icon) to enlarge the flow map.

To zoom in or out of the layout

To zoom in or out of the image, drag the zoom slider at the left of the flow map. Alternatively you can use the mouse scroll wheel. To save this change for the next time you open the dashboard, click **Default Map -> Save Current Map**.

To specify the region of the flow map to be displayed

Use your pointing device to move the viewport in the lower right area of the flow map to specify which part of a large flow map to display.



To rename the tier, node, or backend icons

The names of the icons are based on the names of the tier or backend. To rename them use the **Tier, Node, Remote Services** and

Database dashboards.

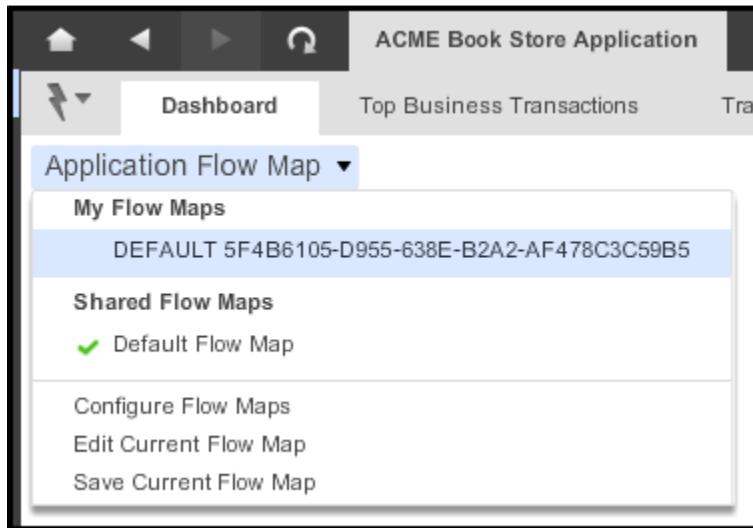
Configuring a Custom Flow Map

The Flow Map menu lists the Default Map and any other maps configured in the UI. You can change the configuration and save a custom flow map for future use.

To configure a custom flow map

1. In a flow map, click the Flow Map menu to see the available options.

The scope of the custom flow map depends on which dashboard you use. For example if you click the flow map menu from a tier dashboard, the scope of the new map is the tier.



2. In the options list, click Configure Flow Maps to see a list of the existing flow maps.

A screenshot of a modal dialog box titled "Custom Flow Maps". The title bar has a close button ("X"). Inside, there is a message: "The following Flow Maps are configured for the current dashboard. Using Flow Maps, you can configure how you want your Map to look, including the positions and visibility of Tiers and Backends." Below the message is a table with three columns: "Name", "Shared", and "Owner". The table contains two rows:

Name	Shared	Owner
Default Flow Map	✓	--
DEFAULT 5F4B6105-D955-638E-B2A2-AF478C3C59B5	✓	user1

At the bottom right of the dialog is a "Close" button.

3. Click Add (the icon).

4. In the Create Flow Map window **Overview** tab:

- Name the view.

- Confirm that the scope is at the correct level. If it is not, go back to Step 1 from the correct dashboard.
 - Check **Shared** to allow other users to see this flow map.
 - Use the **Colors** options to modify the look of the map.
 - Indicate if you want asynchronous activity shown with a dotted line
5. Click the **Tiers** tab and configure what tiers are visible.
- Check which types of tiers you want to see.
 - Hide any specific tiers, if needed.
6. Click the **Databases and Remote Services** tab and configure how you want the backends displayed.
- Group databases or remote services of the same type. By default AppDynamics groups 2 or more backends of the same type. If you want to see each database or remote service in the flow map, uncheck this option.
 - If there are specific databases or remote services that you do not want to see or group in the flow map, uncheck the corresponding **Visible** or **Groupable** check box. See [Grouping Databases and Remote Services on Flow Maps](#) for more details.
7. Click **Save**.
- When you create a custom flow map, the UI lists it in the:
- Default Map menu of each flow map
 - Custom Flow Maps window
- ## Editing Existing Flow Maps
- ### To edit a flow map
1. In a flow map, click the Flow Map menu to see the available options.
 2. In the options list, click Edit Current Flow Map.
 3. In the flow map editor you can change any of the following:
 - Shared - by default custom flow maps are shared; deselect this if you want to keep the flow map for your own individual use.
 - Text color
 - Link color
 - Background color treatment: solid or gradient
 - Background color
 Click **Save**.
 4. Click the **Tiers** tab to configure what tiers are visible.
 - Check which types of tiers you want to see.
 - Hide any specific tiers, if needed.
 Click **Save**.
 5. Click the **Databases and Remote Services** tab to configure how they are displayed.
 - Group databases and remote services of the same type. By default AppDynamics will group any 2 or more backends of the same type. Sometimes you want to see each database and remote service in the flow map. If so, uncheck this option.
 - If there are specific database or remote services that you do not want to see or group in the flow map, uncheck the Visible or Groupable check box. See [Grouping Databases and Remote Services on Flow Maps](#).
 6. Click **Save**.
- AppDynamics applies the changes to the flow map.
- ### To copy a flow map
1. In a dashboard Overview tab click **Default Map -> Configure Flow Maps**.
The Custom Flow Maps list appears.

2. In the Custom Flow Maps list, select a map.

3. Click **Copy** (the  icon).

To delete a flow map

1. In a dashboard Overview tab click **Default Map -> Configure Flow Maps**.

The Custom Flow Maps list appears.

2. In the Custom Flow Maps list, select a map.

3. Click **Delete** (the  icon).

Learn More

- Dashboards
- Application Flow Component Icons
- Logical Model
- Measure Distributed Transaction Performance
- Monitor End User Experience (EUM)

Grouping Databases and Remote Services on Flow Maps

- [Limitations on Grouping Backends on Flow Maps](#)

When you edit existing flow maps or create new customized flow maps, you can control how the databases and remote services appear. Some reasons to collapse similar databases or remote services into a single icon are to improve readability or focus on nodes of specific business interest. For example if individual databases or remote service instances are not important to the audience using this specific flow map, you can group into one icon.

Limitations on Grouping Backends on Flow Maps

These rules govern how databases and remote services can be grouped on flow maps.

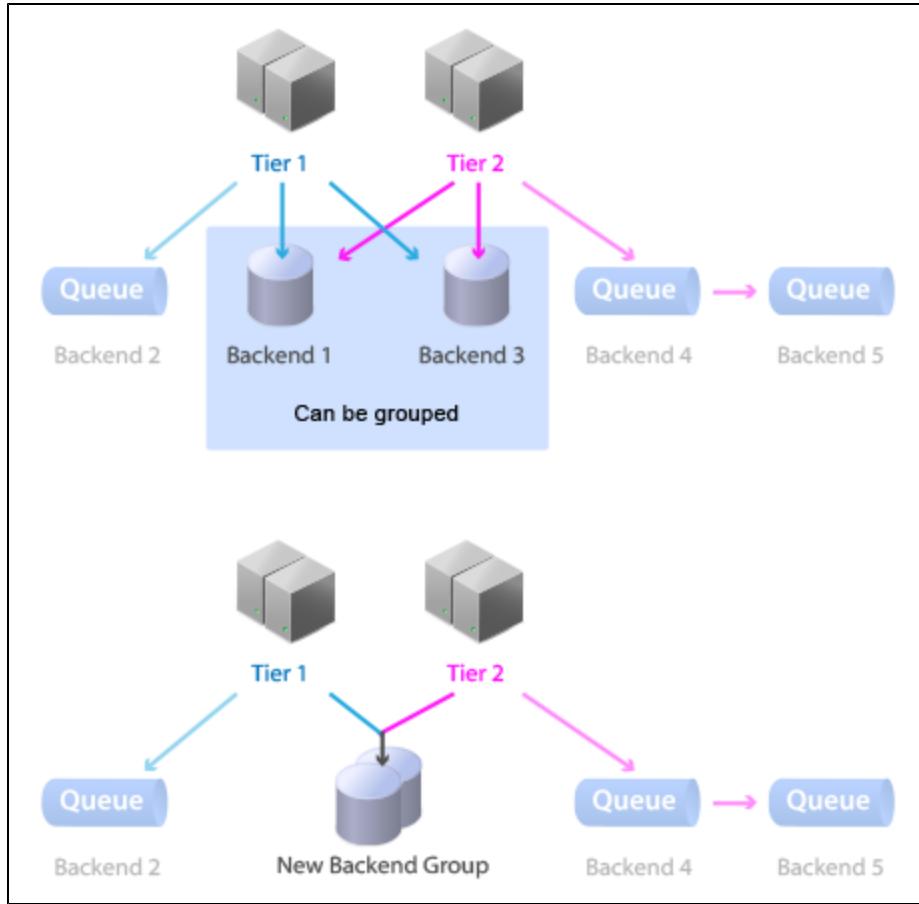
- A group consists of a minimum of two backends.
- A backend that calls other backends or calls into any other tiers cannot be part of a group.

A set of backends can be grouped under these conditions:

1. The backends are all of the same type.
2. The backends are called by the same tiers.

For example, consider the case shown in the top half of the following diagram where:

- Tier 1 calls Backend 1, Backend 2, Backend 3, and
- Tier 2 calls Backend 1, Backend 3, Backend 4, and
- Backend 4 calls Backend 5



The bottom half of the diagram shows that Backend 1 and Backend 3 can be grouped because they are called by Tier 1 and Tier 2. However, Backend 2 and Backend 5 can not be grouped because they are individual backends of a type within their tiers. Backend 4 cannot be grouped because it calls Backend 5.

Time Ranges

- The Time Range Dropdown
- Custom Time Ranges
 - To see performance data from a particular time range
 - To create a custom time range
 - To share a custom time range
- Learn More

This topic discusses the Time Range option.

The Time Range Dropdown

The Time Range dropdown menu is available at the top right of all dashboards and the Metric Browser. Use it to set the time range for the data displayed in the UI.



When you change a time range with the Time Range dropdown menu in one screen, the change is reflected in other screens in the AppDynamics UI. An exception is the Scalability Analysis screen where, by design, a change in the time range does not affect the time range in other screens in the UI.

Custom Time Ranges

You can see data from time ranges in the past as well as the present. If the time range you need is not predefined on the Time Range menu, you can specify the range.

You can create and share your own time ranges for greater control when monitoring trends in your managed environment.

For example, the ACME Online application experiences maximum load during a 3.00 p.m. to 5.00 p.m. every day. A user can monitor only this particular time duration using the custom time range "Maximum Load Duration".

You can share a custom time range with team members.

To see performance data from a particular time range

1. From the Time Range menu, click **Custom**. The date and time options appear.

A screenshot of a date and time selector interface. It shows two sets of input fields: "From" and "To". The "From" field is set to "11/06/12" and "12:59 PM". The "To" field is set to "11/09/12" and "2:59 PM". Between the fields are dropdown arrows for selecting dates and times. To the right are refresh and search icons.

2. Select the dates and times of the range. The UI updates with the data from that range.

To create a custom time range

1. From the Time Range menu, click **Manage Custom Time Range**.

2. Click **New**.
3. In the Manage Custom Time Ranges click the Add icon.
4. In the Create Time Range window name and describe the details of the time range.
5. Click **Create**

The custom time range appears as a new option in your time range menu.

To share a custom time range

When you create this time range, enable the option of "Share with everyone".

Learn More

- Business Transaction Monitoring

Key to the AppDynamics Icons

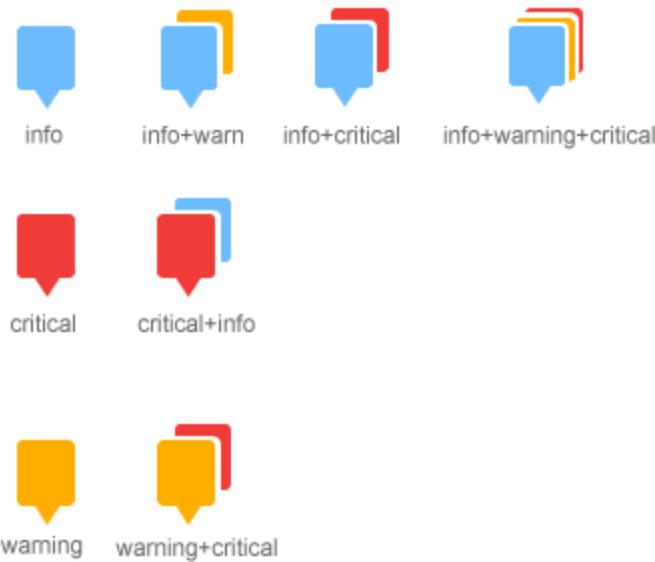
- Application Flow Component Icons
- Events
- Call Type Icons
- Transaction Entry Point Type Icons

This page contains the keys to the icons that appear in the AppDynamics user interface.

Application Flow Component Icons



Events



Call Type Icons

CALL TYPE (SNAPSHOT CALL GRAPH TREE)	
SERVLET,	/* DATA BASE CONNECTIONS ETC*/
SERVLET_FILTER,	DB_CONNECTION,
EJB,	
POJO,	STATEMENT,
SPRING_BEAN,	RESULT_SET,
SPRING_CONTROLLER,	JDBC_DATASOURCE,
SPRING_MVC_DISPATCHER,	
STRUTS_ACTION,	MESSAGE,
JMS_MESSAGE_LISTENER,	CONNECTION_FACTORY
WEB_SERVICE,	MESSAGE_PRODUCER,
JRUBY,	TOPIC_PUBLISHER,
THRIFT,	
ASPDOTNET,	QUEUE_SENDER,
POCO,	
WCF,	

Transaction Entry Point Type Icons

TRANSACTION ENTRY POINT TYPE

// Java Only

 BINARY_Remoting, // THRIFT, PROTOBUF etc.

  SERVLET,

  STRUTS_ACTION,

  SPRING_BEAN,

  EJB,

  POJO,

  JMS,

  WEB_SERVICE,

// .NET Only

 ASP_DOTNET, // similar to SERVLET

 DOTNET_Remoting, // RMI - similar to EJB

 ASP_DOTNET_WEB_SERVICE, // similar to WEB_SERVICE

 WCF, // similar to WEB_SERVICE

 DOTNET_JMS, // similar to JMS

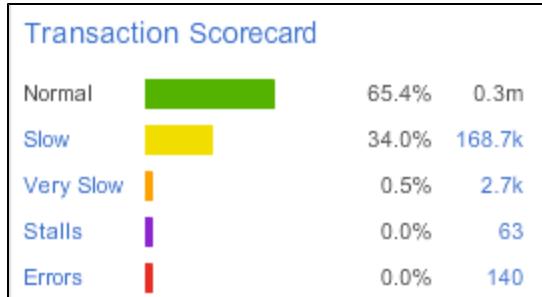
  POCO // similar to POJO

Scorecards

A scorecard summarizes the performance of a business transaction at the application, tier, or node level within a specified time range. It covers the number and percentage of transaction instances that are normal, slow, very slow, stalled, or errors based on the configured thresholds for these criteria.

Slow and very slow transactions have completed. Stalled transactions never completed or timed out. Configurable thresholds define the level of performance for the slow, very slow and stalled categories. See [Configure Thresholds](#).

Scorecards appear in many places in AppDynamics, including the application, tier, node, and business transaction dashboards.



Click one of the values for a slow, stalled or error transaction in the scorecard to see a graph of the metric in the Metric Browser.

In the application dashboard, the business transaction health scorecard aggregates the metrics for all the business transactions in the application, including the default business transactions. The percentages for those calls are based on the average for all the business transactions in the application. For example, if the scorecard displays 3% for very slow transactions, on average 3% of the calls in the application were slow.

In a [tier dashboard](#), the transaction scorecard aggregates the metrics for all the nodes in the tier.

In a [business transaction dashboard](#), the transaction scorecard displays the metrics for the tier that is the entry point to the business transaction.

In a [node dashboard](#), the transaction scorecard displays the metrics for the node.

Events

- [Event Types](#)
 - [Health Rule Violation Events](#)
 - [Slow Transactions](#)
 - [Errors](#)
 - [Code Problems](#)
 - [Application Changes](#)
 - To Register an Application Change Event
 - [AppDynamics Internal Diagnostics](#)
 - [Custom Events](#)
- [Notifications of Events](#)
- [Learn More](#)

An event is a change in application state that is of potential interest.

AppDynamics summarizes events generated for a specific AppDynamics entity - a business application, business transaction, tier or node- on the Dashboard tab of the entity's dashboard.



AppDynamics reports all events for an application in the application events list. You can filter the types of events displayed in the list by checking which events you are interested in. You can learn more about a specific event by selecting it from the list and viewing its details.

To access the events list, select **Events** from the left navigation pane.

The list of events that occurred within the period indicated by the time range dropdown menu appears in the right pane of the event viewer.

To view and set event filters, click **Show Filters**. To hide them, click **Hide Filters**.

AppDynamics also displays the list of events for the entity in the **Events** tab of the business transaction, tier or node dashboards.

Event Types

The event types are:

- Health Rule Violations
- Slow Transactions
- Errors
- Code Problems
- Application Changes
- AppDynamics Internal Diagnostics
- Custom

The event types include subtypes.

Health Rule Violation Events

Health Rule events include:

- Health Rule Warning Violation Started
- Health Rule Critical Violation Started
- Health Rule Violation changed from Warning to Critical
- Health Rule Violation changed from Critical to Warning
- Health Rule Violation Ended

You can filter the specific health rules by affected entity to specify which event data should be displayed.

For information about health rules and health rules violations see [Health Rules](#) and [Troubleshoot Health Rule Violations](#).

Slow Transactions

Slow transactions include:

- Slow
- Very Slow
- Stalled

AppDynamics provides default thresholds that define slow, very slow and stalled transactions, but you can reconfigure them for your environment. See [Configure Thresholds](#).

Errors

Error events include error snapshots and all application server exceptions.

You can restrict the errors for which events are displayed by selecting only errors caused by specific exceptions. If you do not select specific exceptions, all the error events are displayed.

Code Problems

Code problems include:

- Code Deadlock
See [Detect Code Deadlocks \(Java\)](#).
- Resource Pool Limit Reached
This event is generated when the maximum size for any resource pool, such as a thread pool or a connection pool, is reached.

Application Changes

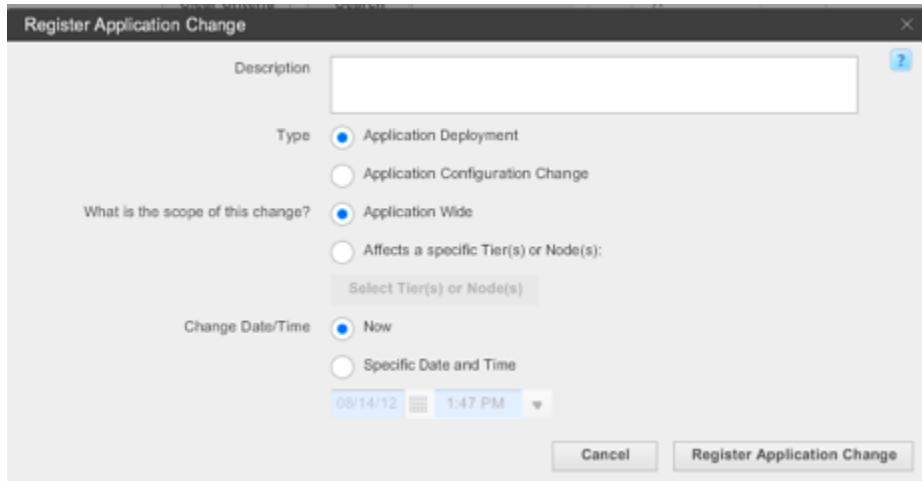
Application change events include:

- Application Deployment
This event is generated when an application is deployed.
- App Server Restarts
This event is generated when an app server is restarted.
- Application Configuration Change
This event is generated when application options are reconfigured.

You can register additional application changes of either the Application Deployment or Application Configuration Change type if you want them to be reported as application change events.

To Register an Application Change Event

1. In the event viewer click **Register Application Change**.



2. In the Register Application Change window enter a brief description of the application event.

3. Select the type, scope and date/time of the event.

4. Click **Register Application Change**.

For more information about application changes see [Monitor Application Change Events](#).

AppDynamics Internal Diagnostics

The AppDynamics Internal Diagnostics events include:

- Diagnostic Session
This event is generated when a diagnostic session is started. See [Diagnostic Sessions](#).
- Agent Enabled / Disabled
This event is generated when an agent is enabled or disabled. See [Manage App Agents](#).
- Bytecode Transformer Log
This event is generated when an entry is added to the bytecode transformer log. This log shows what AppD instruments and what classes are loaded in the JVM. See [bci-log-config](#) for information about bytecode transformer log.
- Agent Diagnostics Event
This event is generated when agent diagnostic data is created. You can access agent diagnostic data from the Action menu on the node dashboard. See [Action Menu](#) and [App Agent for Java Diagnostic Data](#).
- AppDynamics Internal Diagnostics
This event is used by AppDynamics for internal debugging.
- Internal UI Event
This event is used by AppDynamics for debugging data sent from the agent to the controller to the UI.

Custom Events

You can create a Custom event using the REST API.

See [Create a Custom Event](#).

Notifications of Events

You can create email digests to notify recipients when specific events occur.

Learn More

- Filter and Analyze Events
- Configure Email Digests
- Monitor Application Change Events
- Health Rules
- Email Digests

Monitor Events

An event is something that happens in the managed environment that may require attention.

Events are classified according to their event type.

See:

You can configure an email digest to send notification of certain events to a recipient list. See [Email Digests](#).

Filter and Analyze Events

- Filtering Events
 - To View the Event Filter
 - To Filter Events
 - To Filter Error Events
- Analyzing Events
 - To View Event Details

To view the events list, select **Events** from the left navigation pane.

The events list reports the most recent events in your application. If the Fetch More link is visible you can click it to retrieve older events.



You can get details about an event in the list. The content of the details depends on the type of event. If the event is an application change, such as an application restart, the details might be a static description of the event. If the event is a slow transaction or an error, the details are transaction snapshots from which you can drill down to the root cause of the problem.

Filtering Events

If all of the checkboxes in the entire event filter are clear, no events are filtered out and all events are reported.

If one or more of the checkboxes in the event filter are checked, all events are filtered out except for those event types that are checked.

You can filter events to display by event type and by object.

To View the Event Filter

- To display the event filter, click **Show Filter**.
- To hide the event filter, click **Hide Filter**.

The screenshot shows the 'FILTER BY EVENT TYPE' section of the filter panel. It includes a tree view of event types and subtypes, with checkboxes for selecting specific items. The visible categories include 'Policy Violations', 'Slow Transactions', 'Errors', 'Code Problems', 'Application Changes', and 'AppDynamics Internal Diagnostics'. Under 'Code Problems', 'Code Deadlock' and 'Resource Pool Limit Reached' are checked. Under 'Application Changes', 'Application Deployment', 'App Server Restart', and 'Application Configuration Change' are checked.

In the FILTER BY EVENT TYPE section of the filter panel, you can filter based on the event type (such as Slow Transactions or Code Problems) and subtype (such as Slow Transactions -> Very Slow Transactions or Code Problems -> Code Deadlock).

You can filter health rule violation events on the health rule type and on specific health rules. For example, you can specify events caused by business transaction performance health rules or only node health rule violations events caused by the "Memory utilization is too high" health rule.

When you filter by AppDynamics entity type in the FILTER BY OBJECT section of the filter panel, you can select specific business transactions or tiers and nodes for which to report events. If you filter by business transaction, filtering by tier and/or node is disabled. If you filter by tier and/or node, filtering by business transaction is disabled.

The screenshot shows the 'FILTER BY OBJECT' section of the filter panel. It includes a tree view of entities and a dropdown menu for selecting a business transaction. The 'Business Transactions' category is expanded, showing items like '/appdynamicspilot/' and 'UserLogOut.memberLogOut'. The 'Tiers / Nodes' category is also expanded, showing 'ECommerce Server', 'Inventory Server', and 'Order Processing Server', each with its own list of nodes. A dropdown menu is open over the 'Inventory Server' node, listing items such as 'Node_8003', 'Node_8002', and 'Node_8001'. The item 'Node_8003' is highlighted with a blue selection bar.

If you filter by event type but not by object (in other words, at least one of the event type filters is checked but all of the object filters are clear) the events of the selected type are displayed for all AppDynamics entities (business transactions, tiers, and nodes).

To Filter Events

1. In the left panel of the event viewer, check the check boxes for the types of events to display.
2. Clear the check boxes for the types of events not to display.

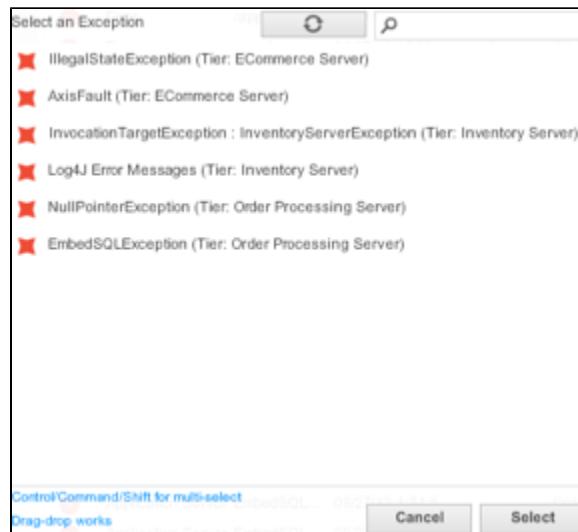
To clear all the check boxes, click **Clear Criteria**.

To Filter Error Events

If you are filtering Error events, checking the Errors checkbox causes all error snapshots and all errors caused by application server exceptions to be displayed. To reduce the number of error events, you can filter the list by specifying that only error snapshots caused by specific exceptions should be displayed. To do this:

1. In the Errors section of the Event filter panel click **Select Specific Exceptions**.

2. In the list of exceptions that appears, click the exception or exceptions for which you want to see error events.



3. Click **Select**.

If you select any exceptions, only error snapshots caused by the selected exceptions are displayed. If you do not select any exceptions, all error snapshots caused by all exceptions are displayed.

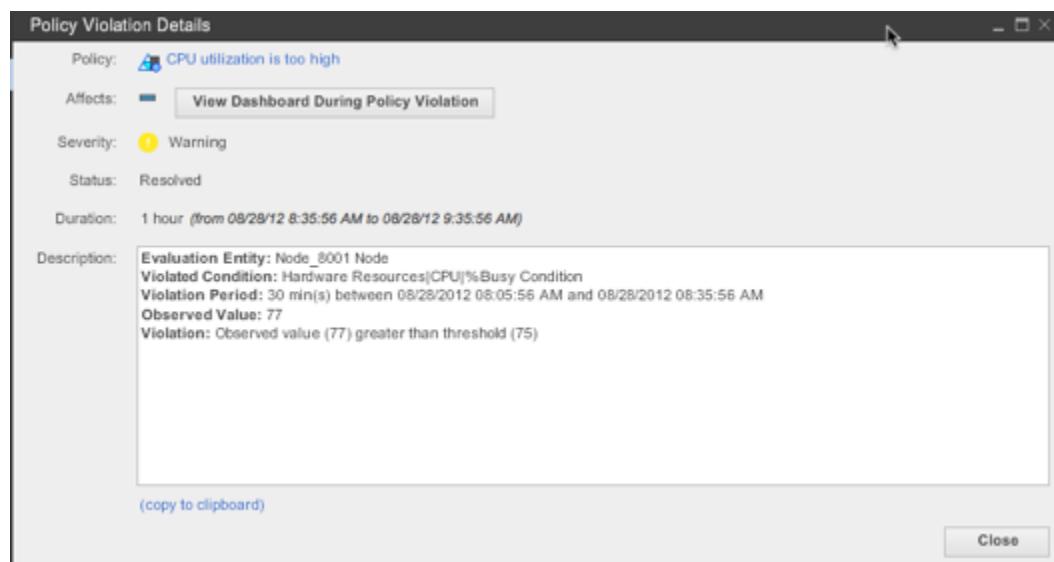
Analyzing Events

To View Event Details

1. In the events list select the event for which you want to see details.

2. Click **View Event Details**.

Usually AppDynamics displays a summary of the event in a new window.



If the event produced a transaction snapshot, as for example in the case of a slow transaction event, AppDynamics displays the transaction snapshot associated with the event. From there you can drill down to the root cause. See [Transaction Snapshots](#).

Monitor Application Change Events

- Application Change Monitoring
- Configuring Application Change Monitoring
- Correlating Application Events With Other Events
 - To monitor application changes with other events
- Learn More

You can monitor changes in your application, such as new code roll-outs or configuration changes, to application servers and correlate these changes with other monitoring data.

Application Change Monitoring

AppDynamics records application changes as events. You can monitor:

- Application Deployments
- Application Server Restarts
- Application Configuration Changes

The events appear in the events list. See [Events](#).

You can compare the system performance across all the application changes.

For example, you can compare between the old and the modified values of any application configuration event.

You can add these events from your release management systems so that an event is created in AppDynamics for every corresponding release in your product. For more information see [Use the AppDynamics REST API](#).

Configuring Application Change Monitoring

There are three ways in which an application change event is registered with AppDynamics.

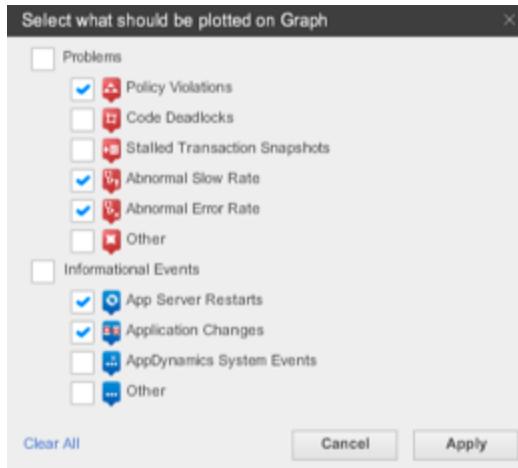
- Auto-discovery of application change events: Events such as machine restart, environment variable changes, and virtual machine system property changes are captured by default.
- Manually register from AppDynamics UI: See [To Register an Application Change Event](#).
- Auto-register from a third-party system using the [AppDynamics REST API](#).

Correlating Application Events With Other Events

You can add application changes to other events to be displayed together on a histogram.

To monitor application changes with other events

1. Open the Application Dashboard.
2. Click the **Transaction Analysis** tab.
3. In the histogram, click **Select Events to View**.
4. Select **Application Changes**.



5. Click **Apply**.

Learn More

- [Events](#)
- [Transaction Analysis Tab](#)
- [Use the AppDynamics REST API](#)

Configure Error Detection

- [Error Identification](#)
 - [Error Detection Using Logged Exceptions or Messages](#)
 - To configure error detection using logs for Java
 - To configure error detection using logs, system trace, and events for .NET
 - [Custom Logger Definitions](#)
 - To define a custom logger
 - [Configuring Exceptions to Ignore](#)
- [Configuring HTTP Response Code Errors](#)
- [Configuring Error Redirect Pages](#)
- [Learn More](#)

You can configure which exception or error codes should be ignored or mapped to a custom name.

Error Identification

You can configure:

- where AppDynamics detects errors
- exceptions and messages to ignore
- HTTP return codes to define as errors
- error detection using redirect pages
- custom loggers

The error detection configuration is applied to all tiers in the application.

Error Detection Using Logged Exceptions or Messages

By default AppDynamics instruments Java error and warning methods such as `logger.warn` and `logger.error`. AppDynamics captures the exception stack trace and automatically correlates it with the request.

To configure error detection using logs for Java

For details about the Java logging APIs see [Java Logging](#).

1. Click **Configure -> Instrumentation**.

2. Click the **Java - Error Detection** tab.

Error Detection Using Logged Exceptions or Messages

▼ Define where AppDynamics will look for log messages or exceptions

- Detect errors logged using java.util.logging (Java 1.6 is required)
- Detect errors logged using Log4j
- Detect errors by looking at messages logged with ERROR or higher
- Mark Business Transaction as error

3. Select where you want AppDynamics to look for log messages and exceptions.

If you want AppDynamics to count all business transactions with messages logged with ERROR or higher as error transactions, check the Mark Business Transaction as error checkbox. Clear the checkbox if you do not want any business transactions with these messages counted as errors. If you want to exclude only specific messages from causing the business transaction to be counted as an error transaction, check the checkbox and add the messages that you want to exclude in the Ignored Messages list below.

Messages logged as higher than ERROR would include more severe levels such as CRITICAL or FATAL.

4. Click **Save Error Configuration**.

To configure error detection using logs, system trace, and events for .NET

1. Click **Configure -> Instrumentation**.

2. Click the **.NET - Error Detection** tab.

Error Detection Using Logged Exceptions or Messages

▼ Define where AppDynamics will look for log messages or exceptions

- Detect errors logged using NLog
- Detect errors logged using Log4Net
- Disable System Trace
- Disable Event Log
- Detect errors by looking at messages logged with ERROR or higher
- Mark Business Transaction as error

3. Select where you want AppDynamics to look for log messages and exceptions.

- If you do not want system trace or event log messages to be monitored, check the appropriate box.
- If you want AppDynamics to count all business transactions with messages logged with ERROR or higher as error transactions, check the Mark Business Transaction as error checkbox. Clear the checkbox if you do not want any business transactions with these messages counted as errors. If you want to exclude only specific messages from causing the business transaction to be counted as an error transaction, check the checkbox and add the specific messages that you want not to cause error transactions in the Ignored Messages list.

6. Click **Save Error Configuration**.

Custom Logger Definitions

To configure AppDynamics to look for logged errors in a custom logger, create a custom logger definition.

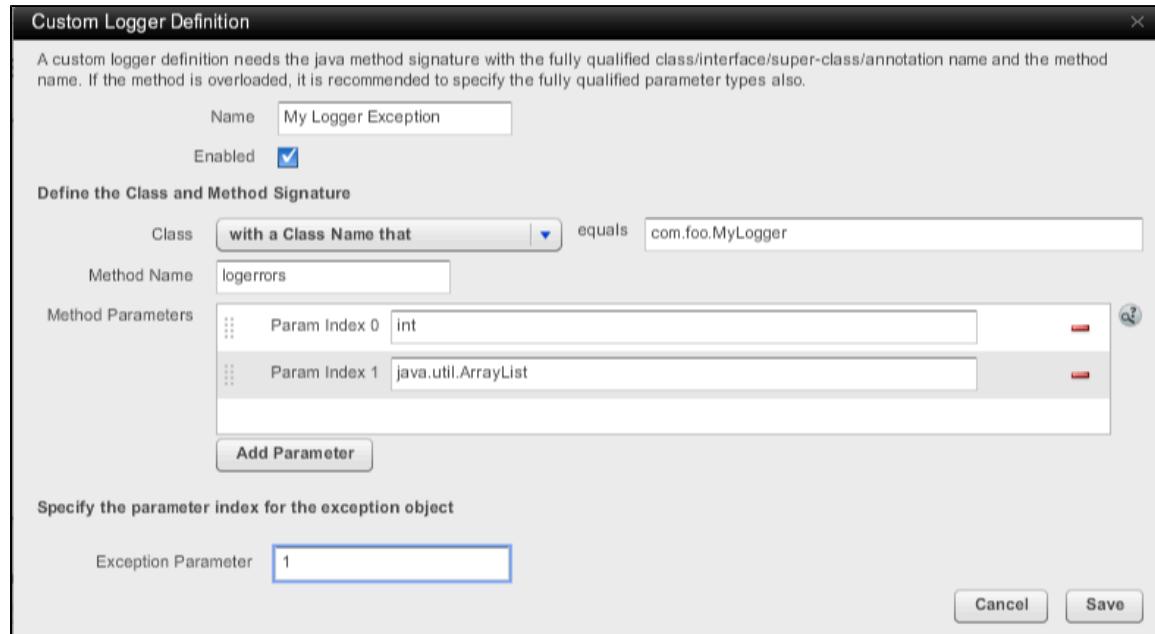
To define a custom logger

- In the error configuration screen, click **Add Custom Logger Definition**.

The Custom Logger Definition window opens.

- Enter a name for the custom logger.
- Check the Enabled check box to enable it.
- Use the drop-down menu and text field to define the logger's class.
- Enter the name of the method that implements the logger.
- Click **Add Parameter** to add a parameter to the logger.
- In the parameter field, enter the fully qualified class name for the parameter.
- Repeat steps 6 and 7 as necessary to define all of the logger method's parameters.
- In the Exception Parameter field, enter the zero-based index of the parameter to use to identify the exception. This index can be any type of Object, including Arrays. If the Object is not null, AppDynamics converts the Object to a string to report the error details.
- Click **Save**.

The following screenshot shows a logger definition for a method with two parameters. It uses the array parameter to describe the exception.



The next screenshot shows a logger definition for a method with one parameter. It uses a string parameter to describe the error details.

Custom Logger Definition

A custom logger definition needs the java method signature with the fully qualified class/interface/super-class/annotation name and the method name. If the method is overloaded, it is recommended to specify the fully qualified parameter types also.

Name	My Logger Error
Enabled	<input checked="" type="checkbox"/>
Define the Class and Method Signature	
Class	with a Class Name that
Method Name	equals com.foo.MyLogger
Method Parameters	Param Index 0 java.lang.String <input type="button" value="Add Parameter"/>
Specify the parameter index for the exception object	
Exception Parameter	0
<input type="button" value="Cancel"/> <input type="button" value="Save"/>	

Configuring Exceptions to Ignore

Configuring exceptions and log messages to ignore is useful for excluding a temporary known issue from the error count to avoid skewing error metrics and triggering unnecessary alerts from policies based on the error count. You can configure AppDynamics to ignore specified exceptions and logged messages.

When you configure an exception to be ignored, AppDynamics still detects the exception, logs the exception, increments the exception count, and displays the exception in the **Exceptions** subtab of the **Errors** tab of the tier and node dashboards and in the **Summary** and **Error Details** sections of any transaction snapshot that was in progress when the exception occurred. However, in the transaction snapshot the User Experience item in the transaction Summary would not be marked as ERROR. Ignoring an exception simply means that AppDynamics does not increment the business transaction error count for errors caused by exceptions that have been configured to be ignored. When an ignored exception occurs, AppDynamics does not count the business transaction in which the exception occurred as an error.

▼ Define exceptions or log messages to ignore when detecting error Transactions

Ignored Exceptions Ignore these exceptions when detecting errors <input type="button" value="Add New Exception to Ignore"/> <input type="button" value="Edit"/>	Ignored Messages Ignore these logged messages <input type="button" value="Add"/> <input type="button" value="Edit"/>
--	---

You can specify exception chain, using colons as separators.

Configure an Exception to Ignore when Detecting Errors

Exception, or Exception Chain

Enter fully qualified Class Names for Exceptions separated by :

[?](#)

You can direct AppDynamics to ignore an exception only when the exception.getMessage() call contains the string "format", or does not contain the string "format".

Configure an Exception to Ignore when Detecting Errors

Exception, or Exception Chain

Enter fully qualified Class Names for Exceptions separated by :

[?](#)

Match Condition for Exception Message

This condition will be used to match against exception.getMessage(). If it matches, for the exception configured above, the exception will not be detected as an Error

Exception Message

NOT Condition
 Selecting this will reverse the condition and return true if NOT (condition)

[Cancel](#) [Save](#)

The following configuration directs AppDynamics to ignore all java.lang.Runtime exceptions that wrap a javax.sql.SQLException. Other types of java.lang.Runtime exceptions will not be ignored.

Configure an Exception to Ignore when Detecting Errors

Exception, or Exception Chain

Enter fully qualified Class Names for Exceptions separated by : [?](#)

Match Condition for Exception Message

This condition will be used to match against exception.getMessage(). If it matches, for the exception configured above, the exception will not be detected as an Error

Exception Message

[Cancel](#) [Save](#)

The next configuration is defined the exceptions to ignore more narrowly. It directs AppDynamics to ignore all java.lang.RuntimeExceptions that wrap a javax.sql.SQLException only when the exception.getMessage() call contains the string "format".

Configure an Exception to Ignore when Detecting Errors

Exception, or Exception Chain

Enter fully qualified Class Names for Exceptions separated by : [?](#)

Match Condition for Exception Message

This condition will be used to match against exception.getMessage(). If it matches, for the exception configured above, the exception will not be detected as an Error

Exception Message

You can also specify Java logged messages to ignore based on a match condition.

Ignored Messages

Ignore these logged messages

Create Ignored Message Match Condition

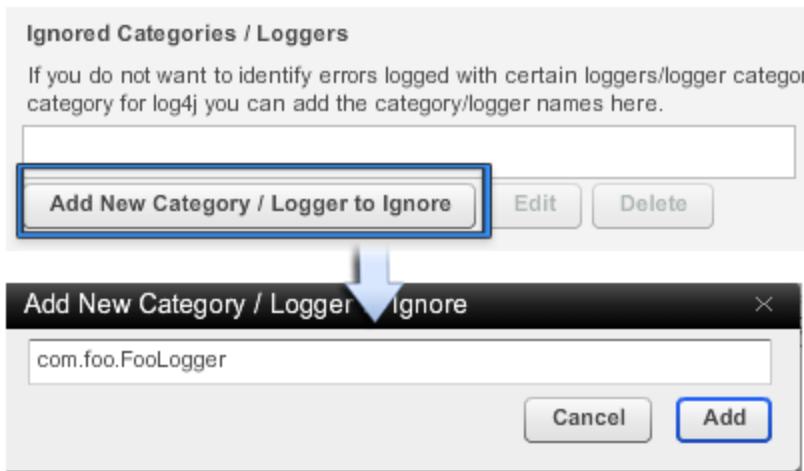
Logged message

Create Ignored Message Match Condition

Logged message

NOT Condition
Selecting this will reverse the condition and return true if NOT (condition)

You can specify that errors logged with certain loggers or logger categories be ignored.



Configuring HTTP Response Code Errors

HTTP response codes convey errors that occur while the application is serving user requests as well as business errors.

For example, a 404 error denotes a page not found where as a 430 error might represent an item out of stock situation.

By default, AppDynamics captures all error codes from 400 to 505 and marks them as errors. You need to configure error custom codes ranges and custom error codes with custom descriptions.

The following figure shows the error configuration for ACME Online store. This configuration causes AppDynamics to detect the Inventory Empty Error and Custom Error codes.

Detect Errors based on HTTP Return Codes			
Description	Lower Bound (inclusive)	Upper Bound (inclusive)	Enabled
Inventory Empty Error	512	520	<input checked="" type="checkbox"/>
Custom Error Code	522	522	<input checked="" type="checkbox"/>

To suppress an error code if you do not want it to appear, add it to the list by creating a custom error code range and then disable that error code by clearing its Enabled checkbox.

Configuring Error Redirect Pages

Applications can use custom error pages to redirect to identify application errors.

To specify a redirect page, click **Add Error Redirect Page**.

Detect Errors based on Redirect Pages [?](#)

If your application uses custom error pages which are redirected to to identify application errors, you can detect them here.

Name	Match Criteria	Match Pattern	Enabled
CustomErrorHandler	Equals	AcmeErrorHandler.jsp	<input checked="" type="checkbox"/>

[Add Error Redirect Page](#) [Delete](#)

In this example, whenever an error in the system redirects to "AcmeErrorHandler.jsp", AppDynamics logs an error.

Learn More

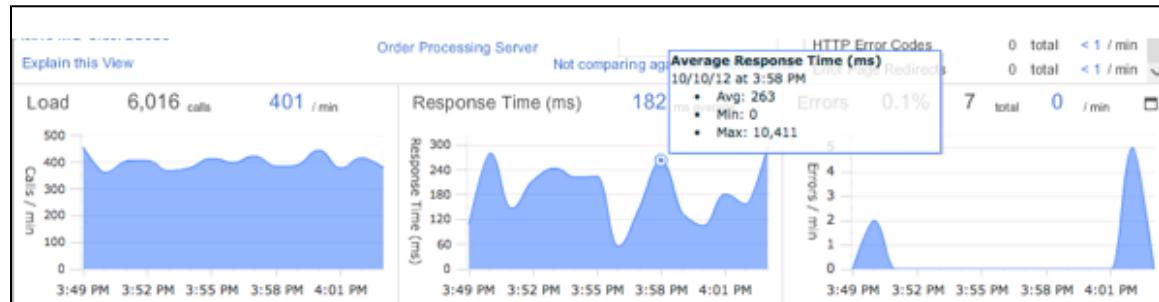
- [Troubleshoot Errors](#)

KPI Graphs

The Key Performance Indicator (KPI) graphs are displayed in the lower section of the various dashboards.

For most dashboards the KPIs are:

- load: number of calls and number of calls per minute
- average response time (ART)
- errors: number of errors and errors per minute



For the End User Monitoring (EUM) dashboard the KPIs are:

- load
- end user response time
- page render time
- network time
- server time

See [Monitor Overall End User Time](#) for information about these metrics.

The graphs show performance in terms of the KPIs for the selected time range.

You can hover with your pointing device at various points along the top of the graph to see the precise metric at a certain point in time.

You can click a link (value in blue) at the top of the graph to see the metric in the Metric Browser.

Metric Points in Graphs

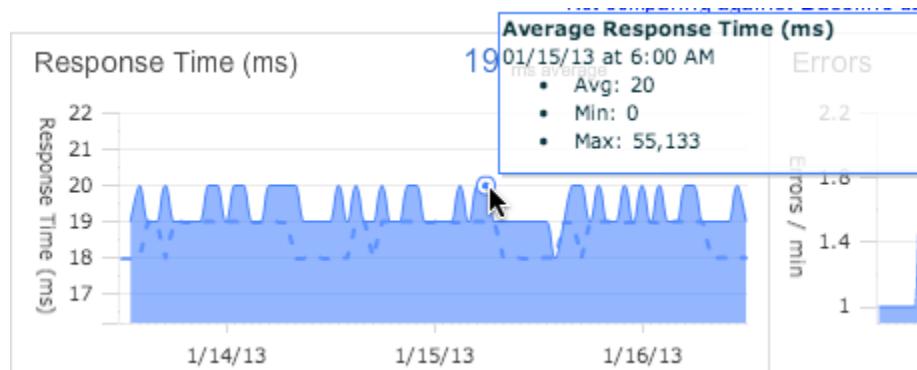
- [Learn More](#)

In graphs throughout the AppDynamics user interface, you can hover over a point on the graph to get the value of the metric at the precise time represented by the point.

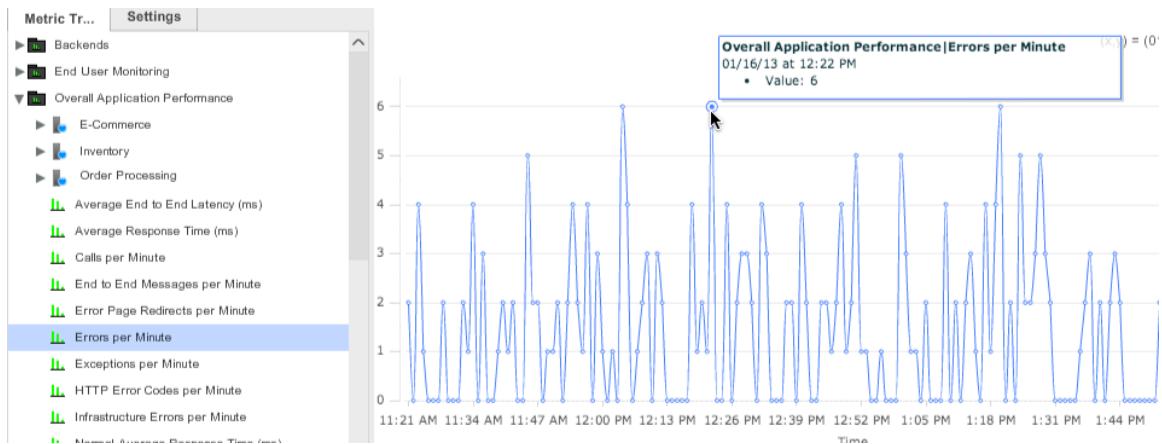
For metrics that represent an interval, the interval is the time that ends with the point. The length of the interval depends on the time range selected in the Time Range dropdown menu. For example, if the selected time range is the last 30 minutes, the interval is between points is one minute. If the selected time range is the last 6 hours the interval is 10 minutes. If the selected time range is the last one week, the interval is one hour.

- Last 5 Minutes (auto-refresh) (resolution: 1 min.)
- Last 15 Minutes (auto-refresh) (resolution: 1 min.)
- Last 30 Minutes (resolution: 1 min.)
- Last 1 Hour (resolution: 1 min.)
- Last 2 Hours (resolution: 1 min.)
- Last 3 Hours (resolution: 1 min.)
- Last 4 Hours (resolution: 1 min.)
- Last 6 Hours (resolution: 10 min.)
- Last 12 Hours (resolution: 10 min.)
- Last 1 Day (resolution: 10 min.)
- Last 3 Days (resolution: 1 hour)
- Last 1 Week (resolution: 1 hour)
- Last 2 Weeks (resolution: 1 hour)
- Last 1 Month (resolution: 1 hour)
- Last 3 Months (resolution: 1 hour)
- Last 6 Months (resolution: 1 hour)
- Last 1 Year (resolution: 1 hour)
- Custom

For example, the following graph is from an application dashboard in which the time range was set for the last three days, so the interval is one hour. The average response time reported at 6:00 AM represents metric recorded for the hour from 5:00 AM to 6:00 AM. It does not include any responses recorded after 6:00 AM.



In the following graph from the Metric Browser, the time range was set for the last three hours. The errors per minute reported for 12:22 PM represents the metric recorded for the minute from 12:21 PM to 12:22 PM. It does not include any errors recorded after 12:22 PM.



Learn More

- Metric Browser
- Time Ranges

Metric Browser

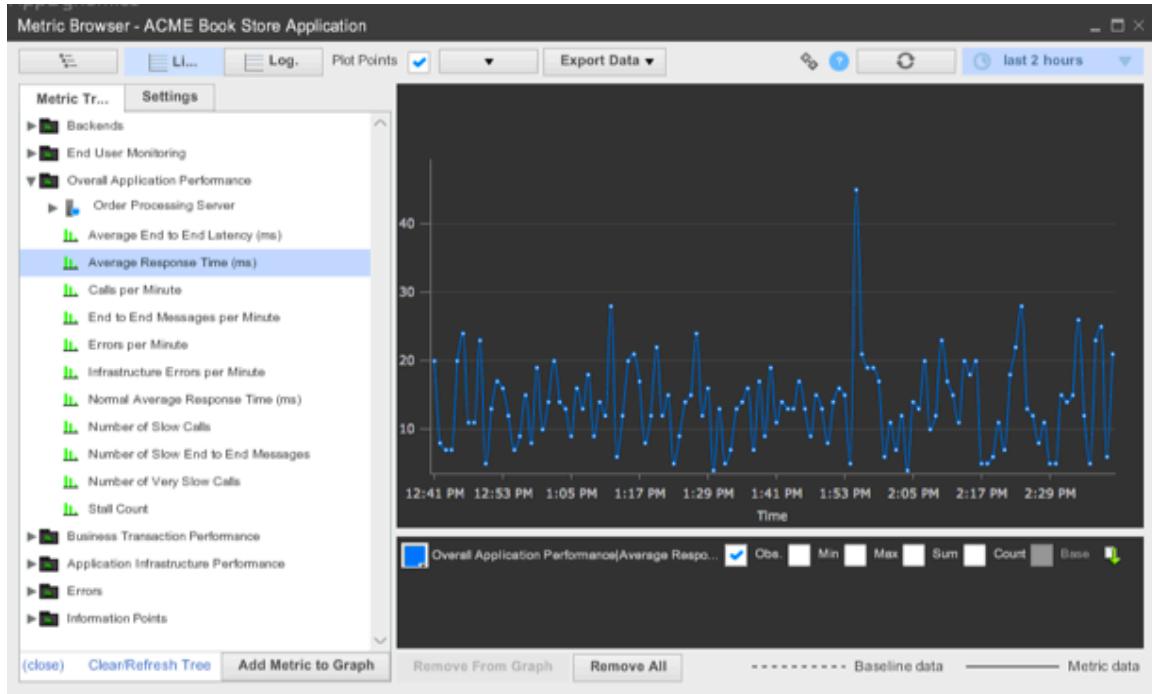
- Accessing the Metric Browser
- How the Metric Browser is Organized
 - End User Monitoring
 - Backends
 - Overall Application Performance
 - Business Transaction Performance
 - Application Infrastructure Performance
 - Errors
 - Information Points
- Options for Analyzing Metrics
 - To Set the Global Time Range
 - To Drill Down into a Specific Time Range
 - To Use Baseline Patterns
 - To Display Baseline Deviations
 - To Export Metrics Data

The Metric Browser displays metrics for your managed application over the time specified in the time range drop-down menu.

Accessing the Metric Browser

To access the Metric Browser:

- In the left navigation pane, click **Analyze -> Metric Browser**.



How the Metric Browser is Organized

AppDynamics metrics are organized into the following categories:

- End User Monitoring
- Backends
- Overall Application Performance
- Business Transaction Performance
- Application Infrastructure Performance
- Errors
- Information Points

To see the metrics in each category, access the Metric Browser and expand the categories in the left pane of the browser until you get to the leaves. AppDynamics frequently adds new metrics with new releases. The metrics that are displayed may not be identical on all platforms.

To graph a metric, drag it from the left pane to the graph.

To remove a metric from the graph select it in the graph and click **Remove from Graph** or to remove all the metrics from the graph click **Remove All**.

End User Monitoring

End user monitoring (EUM) provides key performance metrics about your end users' experience starting from the users' Web browsers instead of at the application server. See [Monitor End User Experience \(EUM\)](#) for general information.

EUM metrics in the metric browser appear only if EUM is enabled for your application. See [Enabling and Disabling EUM](#).

Backends

Backend monitoring provides key performance metrics for calls to databases and remote services.

Overall Application Performance

Overall application performance metrics are displayed at application, tier, and node levels. All node-level metrics are shown under individual nodes for each tier.

Overall performance metrics include metrics for the external calls between two tiers or between a tier and a database or remote service.

Business Transaction Performance

Business transaction performance metrics are displayed for the business transaction groups and for individual business transactions.

Metrics for individual business transactions are grouped under each tier where the business transaction is invoked. The metric browser also displays metrics for external calls from the tiers.

Application Infrastructure Performance

Application infrastructure performance metrics are displayed for the entire application and for individual nodes. Expand the individual nodes item to see metrics at the node level. Infrastructure metrics cover:

- Agent performance
- Hardware resources
- JMX
- JVM and CLR

Errors

Error metrics for the specific errors are shown by tier and then individual node

Information Points

Information points metrics are provided for each configured information point. See [Business Metrics](#).

Options for Analyzing Metrics

You can use following options to analyze performance data for the selected metrics:

- Set Global Time Range
- Analyze performance for a specific time range
- Baseline patterns
- Display baseline deviations
- Export metrics data

To Set the Global Time Range

- From the time range drop-down menu in the upper right corner for the metric browser, select the time range for the metrics to display.

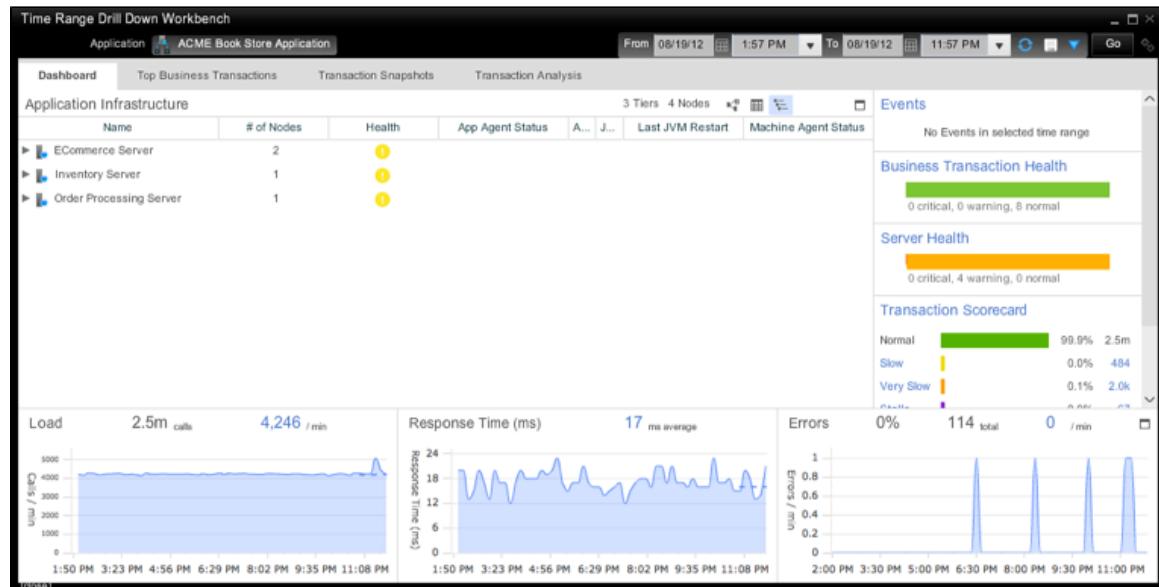
To Drill Down into a Specific Time Range

1. Drag and drop one or metrics onto the graph.
2. Drag your pointing device across the times on the graph to delineate the timer range that you want to analyze.



3. From the sub-menu on the left, click **Drill Down**.

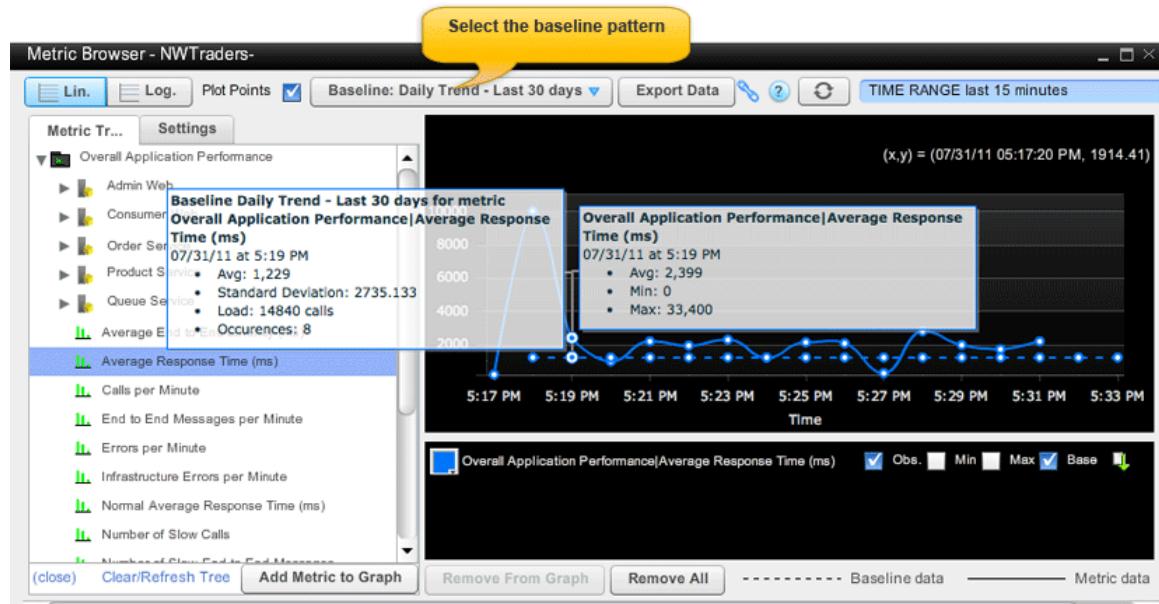
A time range workbench opens for the time range that you specified. From here you can get transaction snapshots for the selected time range using the **Transaction Snapshots** tab.



To Use Baseline Patterns

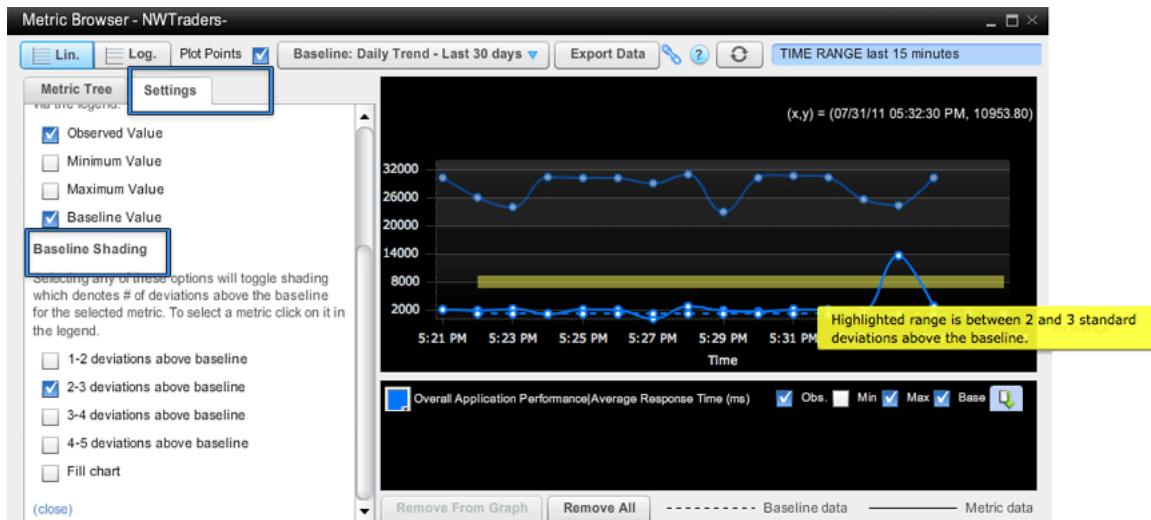
1. Drag and drop the metrics on the graph.
2. Select the baseline pattern to use from the drop-down menu.

For information about the automatic calculations for the baseline patterns see Behavior Learning and Anomaly Detection.



To Display Baseline Deviations

1. Drag and drop the metrics that you want to monitor on the graph.
2. Click the **Settings** tab in the left panel.
3. In the Baseline Shading section, select the range of deviations to shade for the trend for the selected metrics.



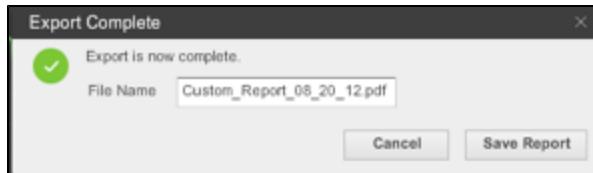
Monitoring baseline deviation is a good way to be aware of performance metrics that might be violating your Service Level Agreements (SLAs).

To Export Metrics Data

1. Select the metrics data that you want to export.
 2. From the Export Data menu choose the format: CVS or PDF
 3. Do one of the following:
- * For a CVS export, in the Metric Data Export window, check the check boxes for the data to be exported and click **Export to File**.

The screenshot shows the 'Metric Data Export' window. In the 'Configure Export' section, the metric name is set to 'BTM|Application Diagnostic Data|Error:97|Errors per Minute'. Under 'Exported Columns', several checkboxes are checked: 'Value', 'Minimum', 'Maximum', 'Sum', and 'Count'. The 'Export Format' is set to 'Horizontal Rows'. In the 'Export Preview' section, the preview shows the following data structure: 'BTM|Application Diagnostic Data|Error:97|Errors per Minute', 'Date,Value,Min,Max,Sum,Count,Notes'. At the bottom, there are buttons for '(close)', 'Export to File', and 'Copy to Clipboard'.

* For a PDF export, in the Export Complete window, click **Save Report**.



Behavior Learning and Anomaly Detection

- Anomalies in Application Performance
 - How Anomaly Detection Works
- Baselines and Periodic Trends
- Baseline Patterns
 - Time Periods
 - Trends
 - Baselines and Deviations
 - Out-of-the-Box Baseline Patterns
 - Daily Trend - Last 30 Days
 - Weekly Trend - Last 90 Days
 - Monthly Trend - Last 365 Days
 - All Data - no trend
- Selecting and Configuring the Best Baselines for Your Application
 - The Default Baseline
 - Changing Existing or Creating New Baseline Patterns
- Using Baselines, Policies, and Alerts to Proactively Identify Performance Problems
- Learn More

This topic discusses dynamic baseline metrics and how AppDynamics learns to detect anomalies in those metrics if they occur.

Anomalies in Application Performance

A complex distributed application has a large number of performance metrics and each metric is important in one or more contexts. In such environments, it is difficult to:

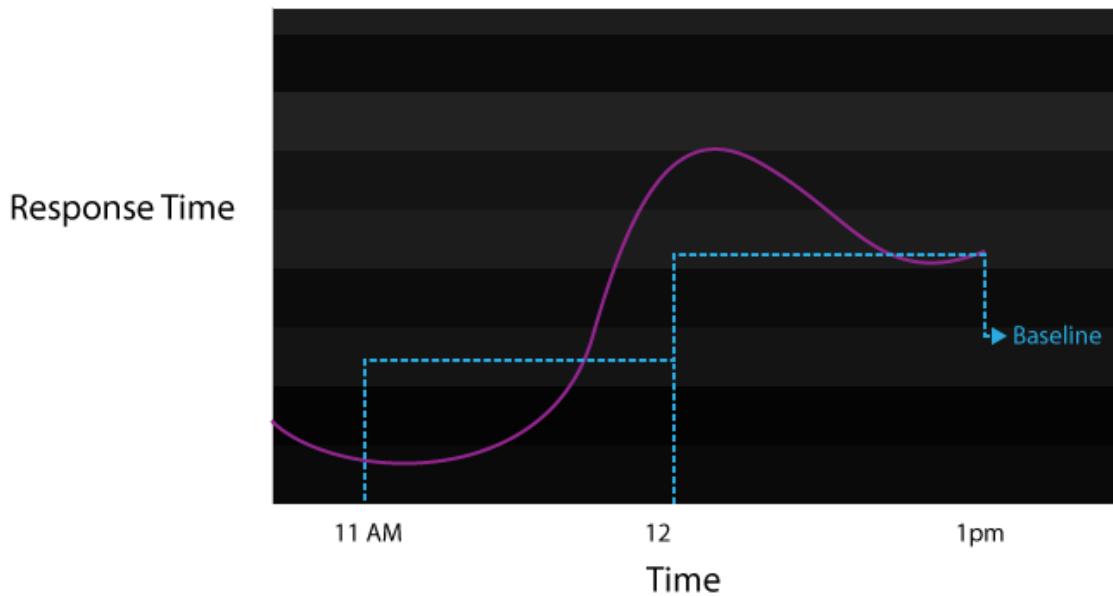
- Manually determine the values or ranges that can be considered normal for a particular metric
- Set meaningful thresholds in order to receive relevant alerts
- Maintain the normal metric when the application or infrastructure undergoes change

For these reasons **anomaly detection** is one of the most important features in AppDynamics.

Anomaly detection means that AppDynamics automatically identifies those metrics whose values are out of the normal range, based on dynamic baselines for these metrics. This eliminates the need for static thresholds that are tedious to manage and error-prone in rapidly changing application environments. Anomaly detection can be applied to all metrics including Business Transaction health and infrastructure performance.

How Anomaly Detection Works

To detect an anomaly, first the accepted baseline value (what is "normal") is defined, either out-of-the-box or according to your configuration. A dynamic baseline for a metric is a rolled-up value for every hour during a rolling time window.



You can configure anomaly detection for a metric by choosing either percentages or standard deviations for comparison against the baseline value. For example, if the actual value of the performance metric is higher than the corresponding baseline value by 3 standard deviations, the value is considered to be an anomaly.

Baselines and Periodic Trends

Most applications have periodic load patterns. For example:

- A retail application typically experiences heavy traffic on the weekend compared to rest of the week.
- A typical payroll application experiences higher load at the beginning and end of the month compared than the rest of the month.
- A Customer Relationship Management (CRM) application experiences heavy load during business hours, Monday - Friday and relatively light traffic over the weekend.

In these example situations, if all of the captured performance data is used to calculate the baseline, then the captured data from the time period with low application load will skew the baseline. Therefore AppDynamics recommends that you use a periodic trend based on the type of load pattern that is applicable for your business application. The trend ensures that AppDynamics determines the relevant baseline.

You do not have to use periodic trends for baselines. You can choose to use a single baseline value that calculates the baseline values every hour.

You can define one or more baselines depending on the trends that you want to monitor. You can configure any one of your baselines to be the default baseline for defining performance health rules.

Baseline Patterns

Baseline patterns are the definitions of how a baseline is calculated. The following factors create a pattern which determine the calculation for a baseline:

- **Time period** baselines use a fixed range of time for calculations, such as "1/1/2011 - 1/31/2011".
- **Trend** baselines use a rolling range of time for calculations, such as "Last 3 months, "Last 1 month", etc.

Time Periods

In some application environments you need monitoring information during specific time periods. For example, if you had a release cycle at a specific time you can monitor the performance during that time.

You can configure AppDynamics to use a baseline calculated from data gathered at a specific hour of each day or week, or a specific

hour of a specific day of each month.

Monthly

Calculates based on each hour of the month.



Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3

- e.g. 5pm - 6pm, 4th Day of the Month.

Weekly

Calculates based on each hour of the week.



Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3

- e.g. 5pm - 6pm, Wednesday.

Daily

Calculates based on each hour of the day.



- e.g. 5pm - 6pm

You can use data from a specific time as a baseline. For instructions see [To use data from a specific time as a baseline](#).

Name	All data - Last 15 days
Trend	<input checked="" type="radio"/> None - Average data for the whole time period. ? <input type="radio"/> Daily <input type="radio"/> Weekly <input type="radio"/> Monthly
Time Period	Fixed - Use a specific time range From 02/10/2011 10:55 PM To 07/10/2011 10:55 PM (all data from this hour will be included in the baseline) Duration: 5 months.

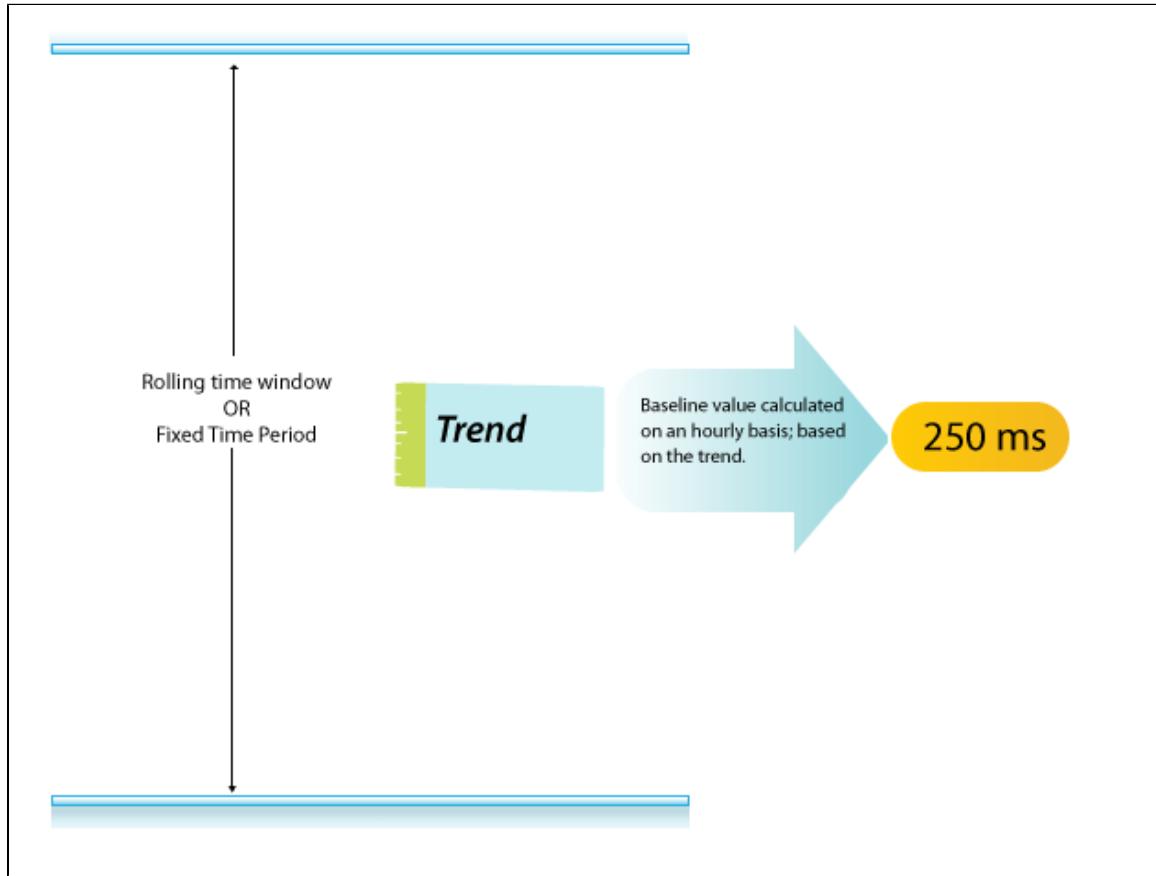
Trends

Trends determine the periodicity used in calculating the baseline. The following options configure trends:

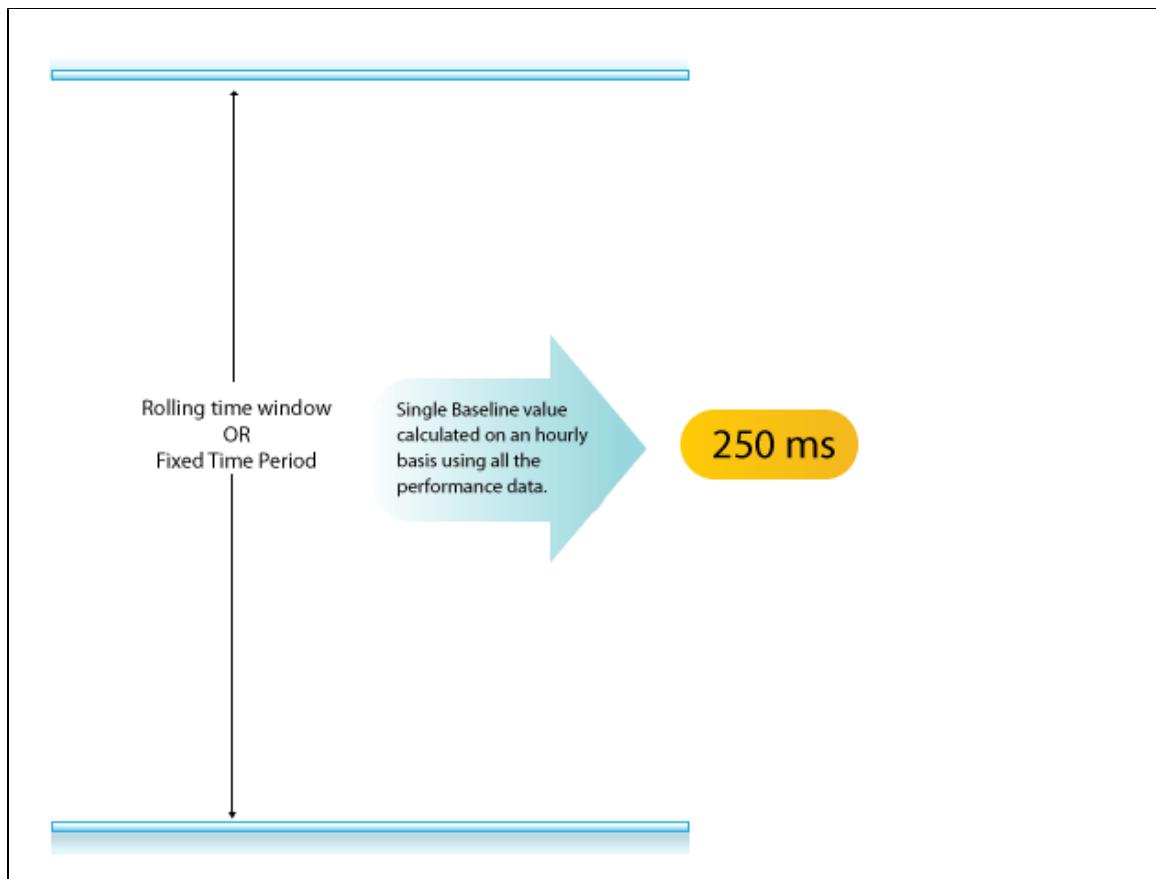
- **Periodic** calculates different values based on monthly/daily/weekly trends.

- **No trend** calculates the same value for the whole time period.

The following diagram describes the baseline when a trend is used:



The following diagram describes the baseline when no trend is applied:



Baselines and Deviations

Baseline deviations specify the degree of deviation from the baseline at any given point of time and are represented by a "number of" deviations. You can use deviations to create policies that trigger alerts when metric values deviate from the baseline. For example:

- A baseline for a "warning" level performance issue is 2 deviations for a period greater than 15 minutes.
- A baseline for a "critical" level performance issue is 4 deviations for a period greater than 15 minutes.

Out-of-the-Box Baseline Patterns

AppDynamics provides out-of-the-box baseline patterns that you can access in the **Configure** section in the AppDynamics UI.

The Baselines view shows the default baseline patterns:

New Delete			
Name	Trend	Time Period	Default
All data - Last 15 days	None	Last 15 days	
Daily Trend - Last 30 days	Daily	Last 30 days	✓
Weekly Trend - Last 3 months	Weekly	Last 90 days	
Monthly Trend - Last 1 year	Monthly	Last 365 days	

AppDynamics provides following preconfigured baseline patterns:

- **Daily Trend - Last 30 Days** compares the value of a metric at a particular hour of the day using the data captured during that hour for last thirty days. This baseline pattern is the default pattern for comparison.
- **Weekly Trend - Last 90 Days** compares the value of a metric at a particular hour of the day for a particular day in the week

using the data captured during that hour and that day of week for last 180 days.

- **Monthly Trend - Last 365 Days** - Last 365 Days compares the value of a metric at a particular hour of the day for a particular day in the month using the data captured over the last 365 days.
- **All Data - no trend** - no trend calculates the baseline using all performance data from the last 15 days.

Daily Trend - Last 30 Days

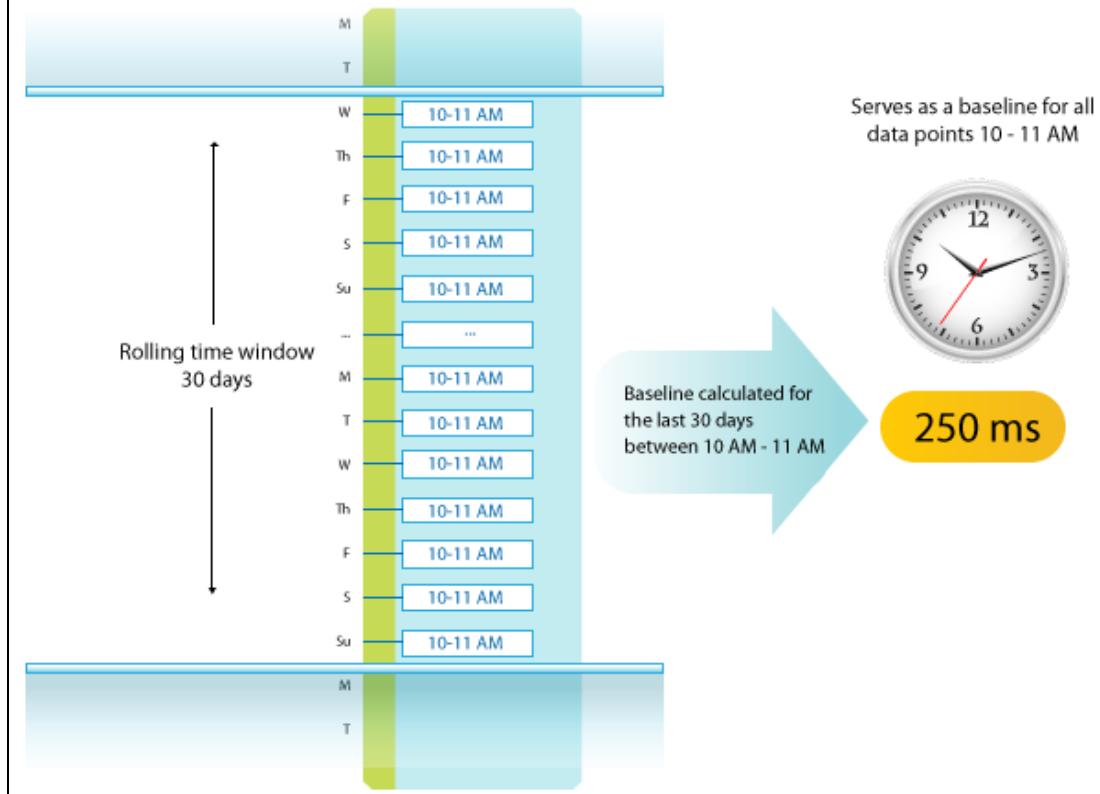
Use this baseline pattern to compare the value for a metric (at a particular hour of the day) using the data captured for last thirty days during that hour.

The dialog box shows the following settings:

- Name: Daily Trend - Last 30 days
- Trend: Daily (selected)
- Time Period: Dynamic - Use a rolling time window
 - Use the last 30 days (selected)
- Buttons: Delete, Set as Default, Save

For example, to compare the metric value at 10:30 AM, the baseline data for this time is calculated based on the captured performance data for that metric during the last thirty days:

'Hour of the Day' - Load patterns similar on hours of the day



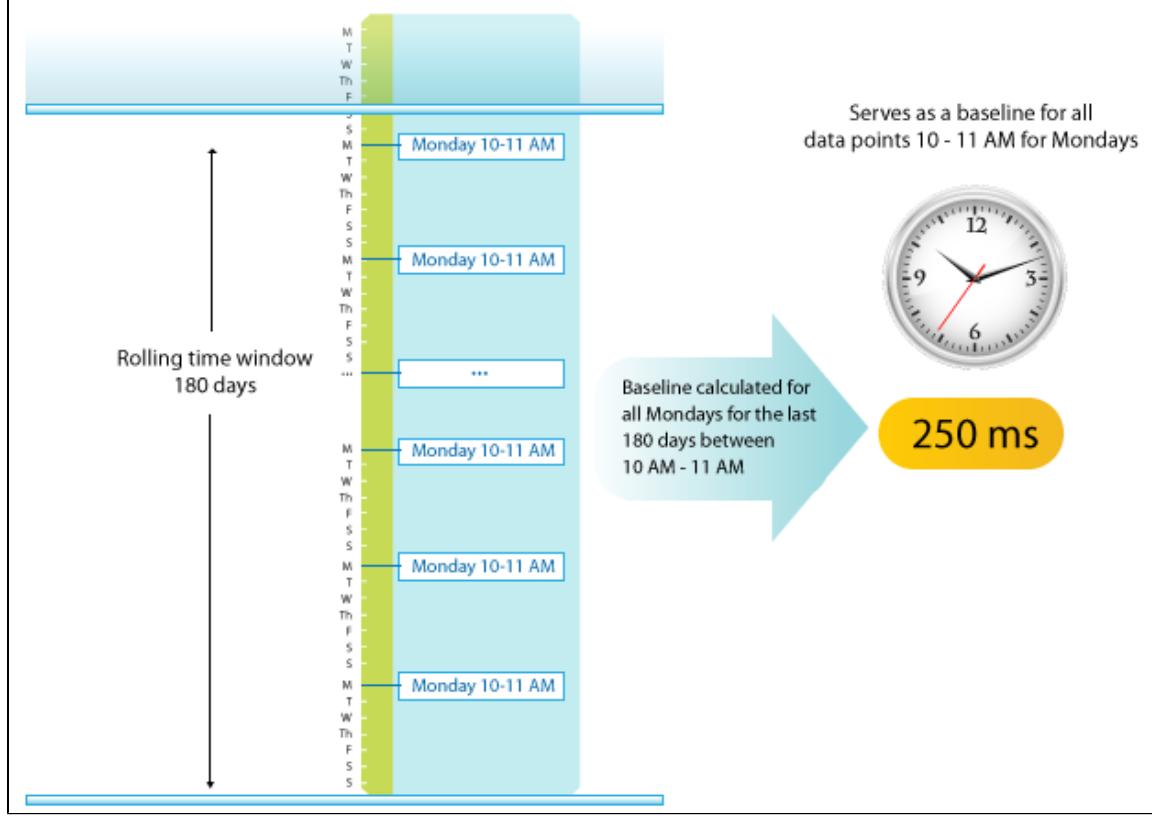
Weekly Trend - Last 90 Days

Use this baseline pattern to compare the value for a metric (at a particular hour of the day for a particular day in the week) using the data captured for last 180 days during that hour and that day of week.

Name	<input type="text" value="Weekly Trend - Last 6 months"/>
Trend	<input type="radio"/> None - Average data for the whole time period. ? <input type="radio"/> Daily <input checked="" type="radio"/> Weekly <input type="radio"/> Monthly
Time Period	<input type="button" value="Dynamic - Use a rolling time window"/> <input type="radio"/> Use all available data <input checked="" type="radio"/> Use the last <input type="text" value="180"/> days
<input type="button" value="Delete"/> <input type="button" value="Set as Default"/> <input type="button" value="Save"/>	

For example, to compare the metric value at 10:30 AM on a Monday, the baseline pattern calculates the performance data for the metric using the data captured for that hour and that day in the last 180 days:

'Hour of the Day of the week' - Load patterns similar on days of the week



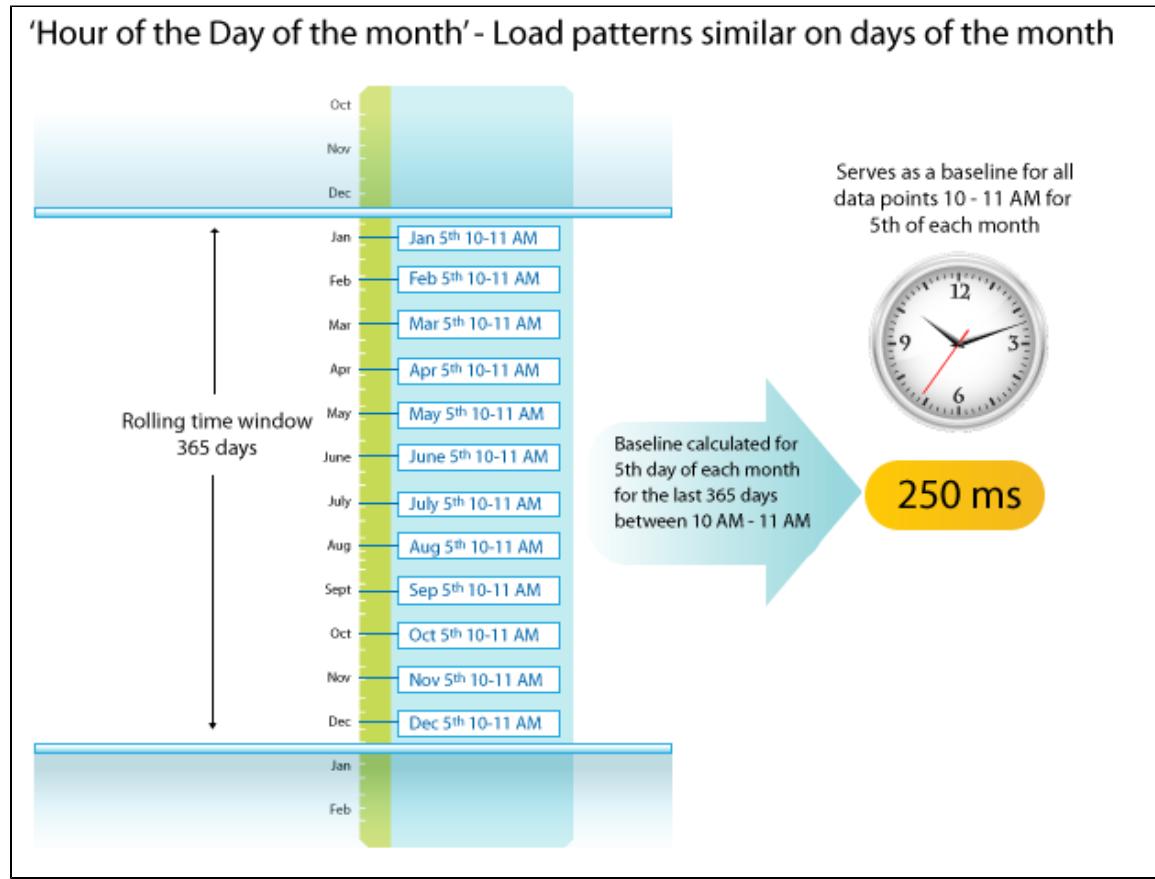
Monthly Trend - Last 365 Days

Use this baseline pattern to compare the value for a metric (at a particular hour of the day for a particular day in the month) with the performance data for this metric over 365 days.

Name	<input type="text" value="Monthly Trend - Last 1 year"/>
Trend	<input type="radio"/> None - Average data for the whole time period. ? <input type="radio"/> Daily <input type="radio"/> Weekly <input checked="" type="radio"/> Monthly
Time Period	Dynamic - Use a rolling time window <ul style="list-style-type: none"> <input type="radio"/> Use all available data <input checked="" type="radio"/> Use the last <input type="text" value="365"/> days
<input type="button" value="Delete"/> <input type="button" value="Set as Default"/> <input type="button" value="Save"/>	

For example, to compare the metric value at 10:30 AM on the 5th of a month, the baseline pattern calculates the performance data for

the metric using the data captured for that hour and that day of each month in the last 365 days:



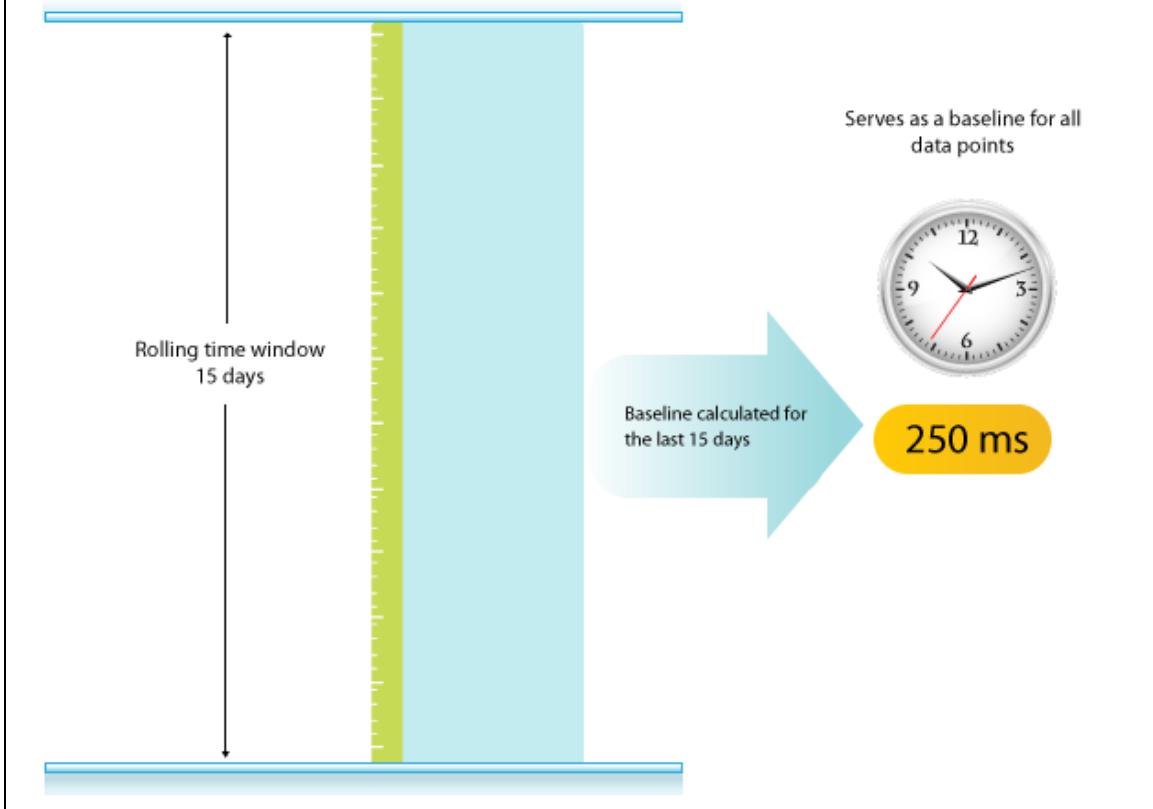
All Data - no trend

This baseline pattern calculates the baseline using the average of the performance data for the last 15 days.

Name	All data - Last 15 days
Trend	<input checked="" type="radio"/> None - Average data for the whole time period. ? <input type="radio"/> Daily <input type="radio"/> Weekly <input type="radio"/> Monthly
Time Period	Dynamic - Use a rolling time window <input type="radio"/> Use all available data <input checked="" type="radio"/> Use the last 15 days
Delete Set as Default Save	

The following illustration shows how the baseline is calculated for comparison of the metric at any time.

'No Trend' - all data is used without any trend



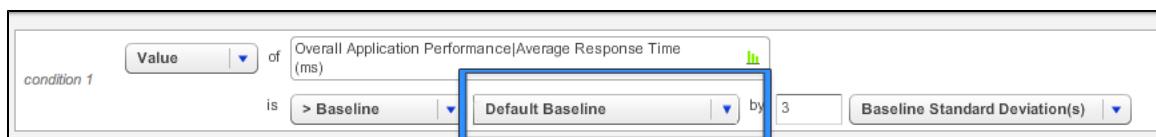
Selecting and Configuring the Best Baselines for Your Application

Use the following options to configure baseline patterns in your environment:

- [The Default Baseline](#)
- [Changing the Values of the Baseline Patterns](#)
- [Using Data from a Specific Time as a Baseline](#)

The Default Baseline

If your application load pattern matches one of the out-of-the-box patterns you can choose that it as your default baseline pattern. When you choose a default pattern it automatically results in changes for the performance health rules. All health rules using the default baseline will immediately start using the new baseline pattern.



For instructions to configure a baseline pattern as the default baseline, see [Configure Baselines#To set a baseline as the default baseline](#).

Changing Existing or Creating New Baseline Patterns

In order to match the load pattern of your application you may need to change the values of an existing baseline or create a completely new baseline. For instructions see [Configure Baselines](#).

Using Baselines, Policies, and Alerts to Proactively Identify Performance Problems

After configuring meaningful baselines for your application, you can define health rules, policies and alerts to let you know when performance problems occur or may be about to occur. For more information see [Policies](#).

Learn More

- [Health Rules](#)
- [Policies](#)

Configure Baselines

- [Configuring Baselines](#)
 - To configure an existing baseline pattern
 - To set a baseline as the default baseline
 - To create a new baseline pattern
 - To use data from a specific time as a baseline
- [Learn More](#)

This topic discusses how to configure baselines against which you monitor the performance of your application. For an overview of baselines see [Behavior Learning and Anomaly Detection](#).

Configuring Baselines

You can define one or more baselines depending on the trends that you want to monitor. You can configure any one of your baselines to be the default baseline for defining health rules.

To configure an existing baseline pattern

1. Click **Configure-> Baselines**.

AppDynamics lists all available baseline patterns:

Name	Trend	Time Period	Default
All data - Last 15 days	None	Last 15 days	
Daily Trend - Last 30 days	Daily	Last 30 days	<input checked="" type="checkbox"/>
Weekly Trend - Last 3 months	Weekly	Last 90 days	
Monthly Trend - Last 1 year	Monthly	Last 365 days	

Daily Trend - Last 30 days

Name: Daily Trend - Last 30 days

Trend: None - Average data for the whole time period. [?](#)
 Daily
 Weekly
 Monthly

Time Period: Dynamic - Use a rolling time window

Use all available data
 Use the last days (Min: 1 Max: 31)

[Delete](#) [Set as Default](#) [Save](#)

2. Select the baseline pattern that you want to modify.

AppDynamics displays the configuration details.

3. Select the time period you want to use for the baseline pattern.

4. Select the data that you want to use for baseline calculation. You can also specify the data to be selected from "number of days".

5. Click **Save**.

To set a baseline as the default baseline

 The default baseline is used by all existing and future health rules. Be aware of your existing baselines and health rule definitions before you select this option.

1. Click **Configure-> Baselines**.

2. Select the baseline pattern that you want to be the new default baseline.

3. Click **Set as Default**.

4. Click **Save**.

To create a new baseline pattern

1. Click **Configure-> Baselines**.

2. Click **New**.

3. Enter a **Name**.
4. Select a **Trend**.
5. Select the **Time Period** you want to use for the baseline pattern.
6. Select the data that you want to use for baseline calculation. You can also specify the data to be selected from "number of days".
7. Click **Save**.

To use data from a specific time as a baseline

1. Click **Configure-> Baselines**.
2. Click **New**.
3. Enter a **Name**.
4. The **Trend** can be either periodic or none.
5. Set the **Time Period** for your baseline as "Fixed - Use a specific time range".
6. Set the date ranges.
7. Click **Save**.

Learn More

- Behavior Learning and Anomaly Detection
- Configure Health Rules

Reports

- Using the Report Templates
 - Business Transaction Summary
 - Business Transaction Trend
 - Hardware Utilization
 - Memory Utilization
 - Node Performance Summary
 - Node Trend
- Other Reports

Reporting enables you to create reports based on metrics that AppDynamics collects and then export them to PDF documents. Reports provide information to use for long-term analysis and planning.

AppDynamics provides report templates that you can configure for your purposes. You can also create a report from the data displayed in the Metric Browser.

Using the Report Templates

To access and use any of the report templates:

1. From the left navigation pane, click **Analyze -> Reporting**.
The Report Browser is displayed.
2. Click the **Generate Report** button on the report template that you want to use.
The report template is displayed.
3. Configure the report in the template.
4. Click the **Export PDF** button to export the report to a PDF document. In the dialog that appears, name the report and click **Save**.

Business Transaction Summary

The Business Transaction Summary Report summarizes the following indicators for the Business Transactions that you select:

- Service Levels

- Time in Milliseconds
- Calls
- Calls / Minute
- Errors
- Error %
- Slow Requests
- Very Slow Requests
- Stalled Requests,
- Tier
- Type

In the template, configure time range covered by the report, the Business Transactions to be summarized, and the sorting parameters.

You can start entering the name of the Business Transaction in the search field to locate the Business Transactions.

Business Transaction Trend

The Business Transaction Trend Report shows the following key performance indicators for the specified Business Transaction.

Problems

- Policy Violations
- Stalls
- Abnormal Slow Rate
- Abnormal Error Rate
- Other Problems

Summary Statistics

- Load in Calls
- Load in Calls / Minute (includes graph)
- Average Response Time in Milliseconds (includes graph)
- Error %
- Total Errors
- Errors / Minute (includes graph)

Response Time Breakdown - absolute and percentage

- Normal
- Slow (includes graph)
- Very Slow (includes graph)
- Stalls (includes graph)

In the template, configure the time range covered by the report and the Business Transaction to be covered in the report. You can start entering the name of the Business Transaction in the search field to locate the Business Transaction.

Hardware Utilization

The Hardware Utilization Report shows the following hardware utilization metrics for the selected tiers:

- Number of Nodes
- Current CPU %
- Average CPU %
- Current Memory %
- Average Memory %
- Disk IO Reads / Second in KB
- Disk IO Writes / Second in KB
- Network IO Writes / Second in KB

For hardware utilization per node, see [Node Trend](#).

In the template, configure the time range covered by the report, the tier or tiers to report on, the sorting parameters, and the format of the report (tree or grid).

Memory Utilization

The Memory Utilization Report shows the following memory utilization metrics for the selected tiers and nodes:

- Max Heap
- JVM CPU Burnt in Milliseconds per Minute

- GC Time Spent in Milliseconds per Minute
- Major Collections
- Major Collection times
- Minor Collections
- Minor Collection Times

In the template, configure the time range covered by the report, the tier or tiers to report on, the sorting parameters, and the format of the report (tree or grid).

Node Performance Summary

The Node Performance Summary Report summarizes the following performance indicators for the selected tiers and nodes:

- Number of Nodes in the Tier (if using tree format)
- Response Time in Milliseconds
- Number of Calls
- Number of Errors
- Number of Slow Requests
- Number of Very Slow Requests
- Number of Stalled Requests

In the template, configure the time range covered by the report, the tiers and nodes to report on, the sorting parameters, and the format of the report (tree or grid).

Node Trend

The Node Trend Report shows the following information for the selected node:

Problems

- Policy Violations
- Stalls
- Abnormal Slow Rate
- Abnormal Error Rate
- Other Problems

Summary Statistics

- Load in Calls
- Load in Calls / Min (includes graph)
- Average Response Time in Milliseconds (includes graph)
- Number of Errors
- Errors Per Minute (includes graph)

Response Time Breakdown

- Normal
- Slow - Number and Percentage
- Very Slow - Number and Percentage
- Stalls - Number and Percentage

If there is a machine agent installed on the node, the report includes the graphs of the following hardware metrics on the node if the data is available:

- Network-Outgoing KB / Sec
- Network Incoming KB / Sec
- Disks KB Written / Sec
- Disks KB Read / Sec
- Memory % Used
- CPU % Busy

In the template, configure the time range covered by the report and the node to report on.

Other Reports

You can also create PDF-formatted reports of the data displayed in the following components:

- Application Dashboard: In the Actions drop-down menu, click **Export as PDF Report**.
- Tier Dashboard: In the Actions drop-down menu, click **Export as PDF Report**.

- Node Dashboard: In the Actions drop-down menu, click **Export as PDF Report**.
- Snapshot Viewer: In the Call Drill Down window, click **Export to PDF**.
- Metric Browser: In the Export Data drop-down menu, click **Export as PDF Report**.

Business Transaction Monitoring

- Business Transactions
 - Business Transactions and Application Health
 - Business Transactions in Distributed Environments
- How AppDynamics Discovers Business Transactions
- Managing Multiple Business Transactions
- Optimizing Business Transaction Monitoring
- Learn More

Business Transactions

The business transaction is the mechanism by which AppDynamics orders and aligns load traffic (response time, throughput, and so on) with the business perspective.

AppDynamics uses business transactions to:

- Organize all user requests to represent accurately the load on your business application.
- Provide diagnostic deep-dive data into requests that experience errors or perform slower than the usual trend.

Business transactions represent either a category or a type of user request. They depict the primary functions of your application. Examples of business transactions include:

- In an **e-commerce application**, user actions such as logging in, searching for items, adding items to the cart, etc.
- In a **content portal**, the content sections that users navigate such as sports, world news, entertainment, etc.
- In a **stock trading application**, operations such as receiving a stock quote, buying and selling stocks, etc.



Example:
The “Checkout” Business Transaction for a Shopping Cart application monitors performance of every individual Checkout request



The Checkout Business Transaction includes all individual user checkouts



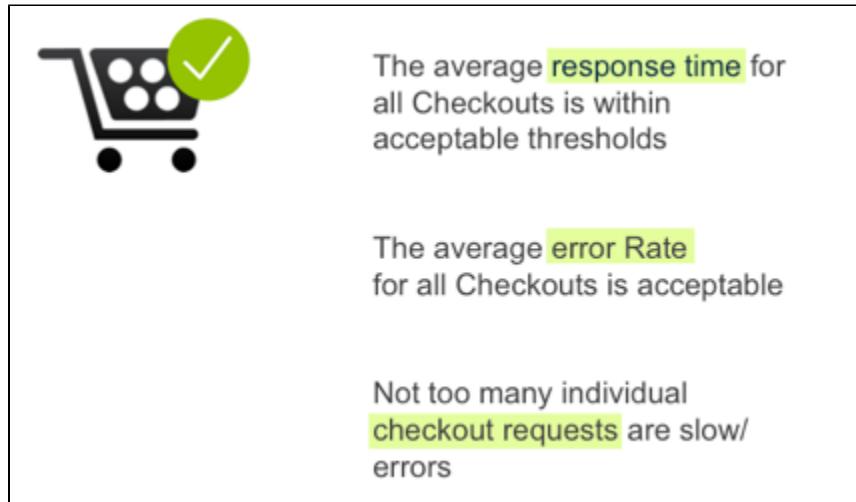
Monitoring the Checkout Business Transaction would tell us how the application is serving these requests

Business Transactions and Application Health

How well a business transaction performs is the most important factor when monitoring the health of your application. The performance data for a particular business transaction in your system provides information such as:

- **Availability:** Is the functionality currently available? For example, are the users currently able to log in to the site?

- **Performance:** How is this task currently performing? For example, how much time, on average, does it take to complete a check-out request? How is the business transaction performing compared to what is considered acceptable performance? The picture below is a summary view of what it means to say that "Checkout is OK".

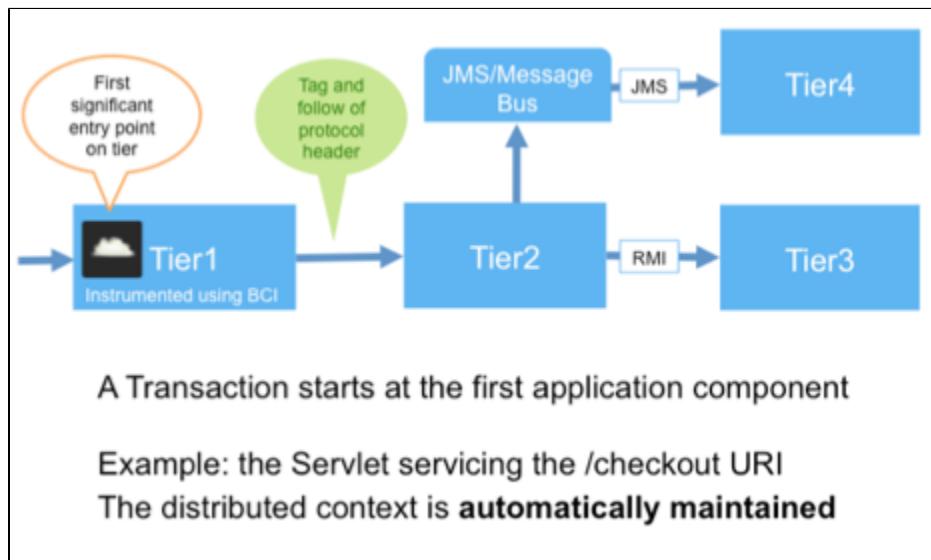


- **User Experience:** How many users performing a particular task experienced slow response times or stalls? Are there performance spikes or lulls due to end user patterns such as holidays?
- **Historical Comparisons:** How does historic data for particular time ranges compare to current data?

Business Transactions in Distributed Environments

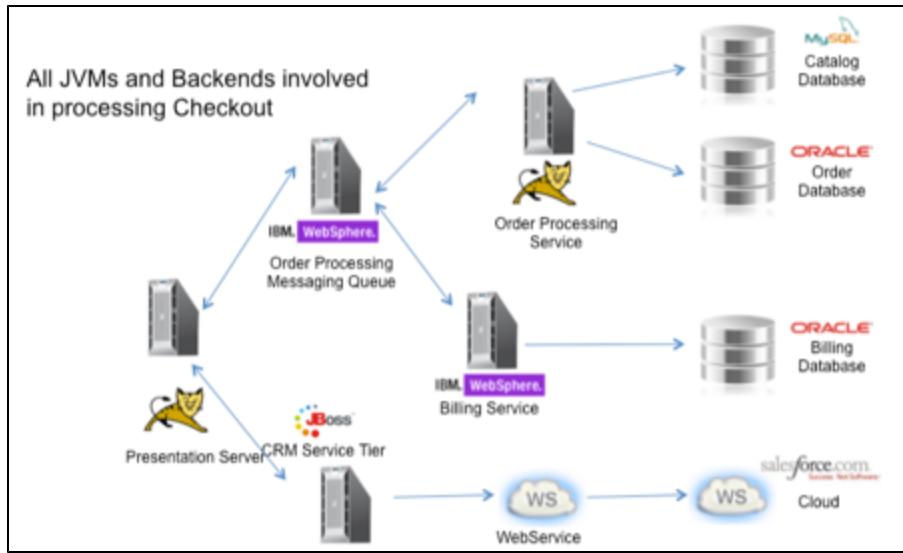
A distributed environment typically has multiple tiers deployed with multiple app servers, databases, remote services, etc. A business transaction may span multiple nodes through remote calls to external services, collecting data from all of these components and measuring the processing time required to collect both synchronous and asynchronous data across all tiers.

The method call that starts the business transaction is called the entry point.



For example, when an app server invokes the business logic associated with an eCommerce checkout operation, it makes a database call plus multiple calls to services in other tiers. AppDynamics discovers and groups these activities as a single "checkout" business transaction. This business transaction captures processing time, which relates to the response time typically experienced by a user.

The transaction also collects all code paths invoked by the business logic when diagnostic data is collected.



How AppDynamics Discovers Business Transactions

When application requests are received by transaction entry points, AppDynamics automatically discovers and identifies the requests as business transactions. AppDynamics uses information from the transaction entry point to name the transaction.

Auto Assignment of User Requests



Multiple user requests, as long as they follow the same code path, are categorized instances of a single business transaction. For example, one user-click results in the Checkout business transaction being identified as a single request. Then, when another user performs the same action, AppDynamics associates the two requests with the same Checkout business transaction.

You can configure how AppDynamics discovers business transactions by creating custom match rules that map precisely to your business processes. You can also exclude auto-discovery of business transactions that you do not want to monitor by creating custom exclude rules.

Managing Multiple Business Transactions

For both performance and practical reasons, it is not optimal to monitor a large number of business transactions. AppDynamics has a limit of 50 business transactions per agent and 200 business transactions per business application.



If traffic exceeds the 50 business transaction per agent limit or the 200 business transaction per application limit, the traffic from what would be 51st detected business transaction and greater for the agent or the 201st and greater detected business transaction for the business application is collected into a default business transaction called "All Other Traffic". There is one for each tier. You can view them in the business transaction List.

Optimizing Business Transaction Monitoring

AppDynamics recommends that you configure your system to monitor the business transactions and other operations that are key to your business. There are multiple ways to approach this. For example, you can exclude certain operations from detection. You can also group several operations into a single business transaction to reduce the total number of business transactions. See [Configure Business Transaction Detection](#).

Learn More

- [A Look at Business Transactions](#)
- [Configure Business Transaction Detection](#)
- [Business Transactions List](#)
- [Business Transaction Dashboard](#)

Organizing Traffic as Business Transactions

- [Reasons for Fine-Tuning Your Business Transactions](#)
- [The Number of Business Transactions](#)
 - [The Right Set of Business Transactions](#)
 - [Determining What User Requests Should be Business Transactions](#)
 - [The Business Transaction Limit](#)
 - [Business Transaction Limits for the Controller](#)
 - [Business Transaction Limits for Agents](#)
 - [Changing Business Transaction Limits](#)
- [Organizing Business Transactions into Groups](#)
 - [To group Business Transactions](#)
- [Learn More](#)

If you organize your monitored application as the set of your web site users' most important activities (i.e. business transactions), you will get the maximum benefits and the shortest MTTR from AppDynamics.

A normal user doesn't complain that the server seems to be experiencing memory leaks or that CPU is maxing out. Instead, he says that he cannot log into the application or that checkout is slow.

That is why we use business transactions, such as login and checkout, to identify and troubleshoot real-world problems in production.

Reasons for Fine-Tuning Your Business Transactions

You can help AppDynamics organize your application traffic in terms of business transactions by configuring how AppDynamics identifies business transactions in your managed environment. There are many reasons why you might want to do this. The most common are:

- You are getting more monitoring information than you want about requests that are not essential to understanding your business, such as administrative console requests, heartbeat pings, and so on.
- You are exceeding or close to exceeding the business transaction limit of 50 transactions per agent or 200 transactions per

application.

- You want to exclude whole classes of transactions: for example, URIs that match or do not match a certain pattern.
- The automatic detection mechanism is not identifying the appropriate entry points for your transactions. For example, you may want the agent to discover transactions based on a class/method combination instead of at a higher component level.
- The automatic detection mechanism is not identifying any business transactions in your application. This is typically the case if your application is not built with a framework that AppDynamics can auto-detect. You can configure AppDynamics to discover business transactions by creating custom match rules for determining the correct entry points.
- You want to split a discovered transaction into multiple transactions. For example, a login request may be detected as a single transaction, but you want to split into two transactions based on whether the request branches to a new-user or existing-user operation.
- By reviewing traffic in the All Other Traffic business transaction you can determine the best strategy for configuring business transaction discovery to reflect your application's actual traffic patterns. For example, if an All Other Traffic transaction is called frequently, you could configure a new business transaction for that traffic. You could "trade off" and keep within the limits by excluding another business transaction, such as one that has little or no load.

To get optimal value from AppDynamics, make sure that your business transactions are set up to describe the traffic that you need to monitor and that they are not set up to monitor extraneous traffic that is not important to your success. Although you will derive some value from AppDynamics auto-discovery, you will significantly increase the utility of the product if your business transactions are tuned to reflect your organization's needs.

AppDynamics automatically detects the business transactions in your application by looking for the business transactions' entry points. See [Web Entry Points](#) and [Configure Business Transaction Detection](#) for more information.

The Number of Business Transactions

There are two factors that influence the number of business transactions in a business application:

- [The Right Set of Business Transactions](#)
- [The Business Transaction Limit](#)

The Right Set of Business Transactions

Before configuring the business transactions that are most helpful for your team, identify the **optimal or right set of business transactions** to monitor.

The easiest way to determine the optimal number of business transactions is to consider the following:

- The number of unique types of requests present in the application
- The number of service levels per business transaction

AppDynamics recommends that you have only as many service levels as are necessary for a particular transaction. For example, a Login transaction represents the performance characteristics of all users who login to a system. The service level for a Login operation for any given time indicates the performance of the transaction. This data translates directly to what end users experience while logging in.

For web traffic, every unique URL is not a business transaction. Multiple URLs will map to the same type of request and therefore map to the same business transaction.

AppDynamics customers typically have only 100 -150 business transactions per business application.

Determining What User Requests Should be Business Transactions

The most important factor for identifying business transactions is functionality. When you analyze user requests, you can produce a manageable list of functionality that is related to service levels.

For example, in an e-commerce application the individual requests that "add items to the shopping cart" can be tracked as the "Add To Cart" business transaction.

For a web portal or content site, you could define business transactions by the category of the content, such as "Politics", "Sports", etc. You could also create more granular business transactions, such as sub-transactions of "Sports" based on the type of sports.

The Business Transaction Limit

There are limits on the number of Business Transactions.

- An App Server Agent is restricted to **50** Business Transactions.
- A Controller is limited to **200** Business Transactions per Business Application.

AppDynamics collects and aggregates data about:

- Nodes
- Business Transactions

The following sections explain why a limit on number of Business Transactions is imposed for Agents and the Controller.

Business Transaction Limits for the Controller

After every minute, the Controller aggregates and monitors service levels of each Business Transaction and accepts data from multiple agents processing the same Business Transaction. The Controller stores data about those agents which identify the Business Transaction or the Entry Points and the data about other nodes (where the business transaction context is maintained).

The Controller I/O processing ability is affected by both the number of Business Transactions and the number of nodes in the application.

To learn more about what type of hardware can efficiently manage how many nodes and Business Transactions see [Controller System Requirements](#).

Business Transaction Limits for Agents

The transaction limit for agents is very important because AppDynamics is designed for production environments. (It can also run in environments where the available free memory on a JVM is less than 50 MB.) AppDynamics Agents do not simply collect data and report it to the Controller, they also:

- Observe service levels for each Business Transaction.
- Observe metrics, such as the number of slow requests, and start diagnostic data collection when thresholds are reached.

Imposing limits on the number of transactions is one way to guarantee the memory requirements of the agent in any type of application environment.

Changing Business Transaction Limits

Before changing the limits, take into account the following factors:

- Controller hardware. For details see [Controller System Requirements](#).
- Memory available for the agent on the managed JVM or CLR. To analyze the memory allowed for the agent: ** After the agent discovers 50 transactions, compare the current memory footprint of the JVM or CLR with that of the agent.
 - Then, compare this data with the memory available for the agent.

To change the limits, contact [AppDynamics Support](#).

Organizing Business Transactions into Groups

When multiple business transactions are similar and you want to roll up their metrics, you can put multiple business transactions into a group.

To group Business Transactions

1. In the left navigation panel, click **Monitor -> Business Transactions**.

AppDynamics lists the Business Transactions for the application.

2. Select the number of Business Transactions which you want to include in a group.

3. Right-click and select **Create Group**.

4. Provide a name for the group.

5. Save the information.

The group is listed on the Business Transactions List.

Learn More

- Business Transaction Dashboard
- Business Transactions List
- Configure Business Transaction Detection
- Monitor All Other Traffic

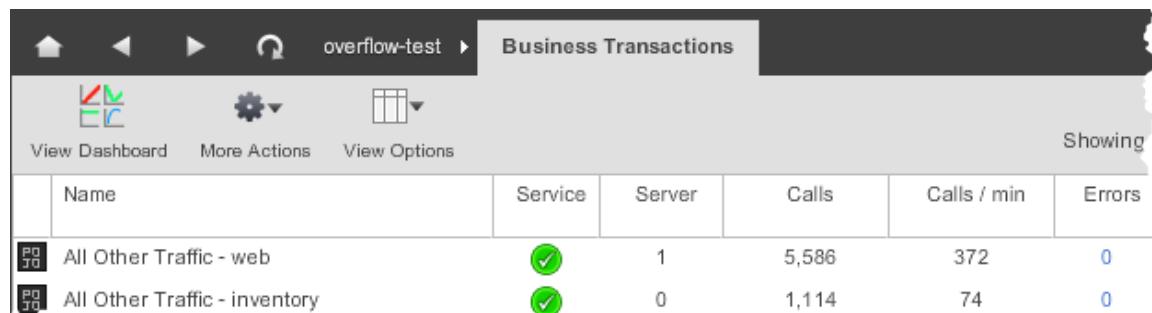
Monitor All Other Traffic

- Retrieving Calls for Unclassified Traffic
 - To Retrieve Counts for Unclassified Traffic
 - Learn More

You can select an All Other Traffic business transaction in the Business Transaction List, view its dashboard, and see its key performance metrics in the Metric Browser.

Retrieving Calls for Unclassified Traffic

You can retrieve the number of calls for a default business transaction by selecting it in the business transaction list and clicking **View Dashboard** to view its dashboard.



	Name	Service	Server	Calls	Calls / min	Errors
PO	All Other Traffic - web	✓	1	5,586	372	0
PO	All Other Traffic - inventory	✓	0	1,114	74	0

To Retrieve Counts for Unclassified Traffic

1. In the default business transaction dashboard click **View Traffic Details**.



The traffic details window displays the count of calls for the unclassified traffic in the tier for the selected time range. If you modify the time range, the query resets.

The Business Transaction name column is a generated name for the auto-detected operation that the application called.

Business Transaction Name	Count	Type
/two/37	27	Servlet
/two/9	22	Servlet
/two/16	24	Servlet
/one/14	26	Servlet
/one/4	19	Servlet
/two/24	29	Servlet

2. If the **Fetch more** link is displayed, click it to see more calls.

The default business transaction viewer retrieves 600 events per query. If the **Fetch more** link is not displayed, there are no more calls to retrieve for the selected time range.

Learn More

- [Configure Business Transaction Detection](#)

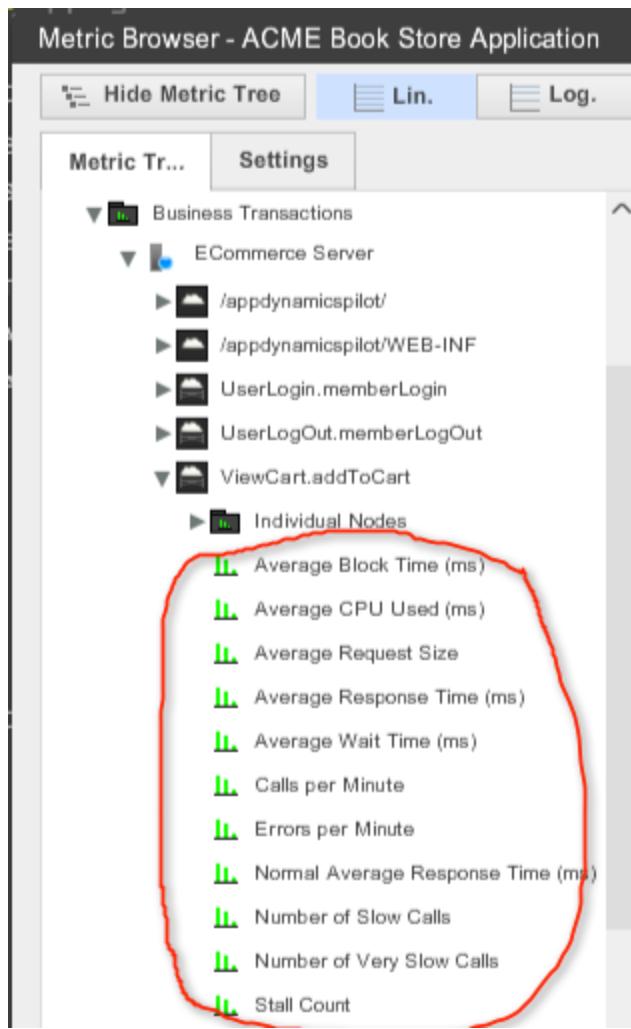
Measure Business Transaction Performance

The key performance indicators (KPIs) that AppDynamics uses to measure business transaction performance are:

- Load (calls per minute)
- ART (average response time in milliseconds)
- Errors (total number and errors per minute)

For information about where to observe these metrics see [KPI Graphs](#), [Business Transaction Dashboard](#) and [Business Transactions List](#).

You can see all the performance metrics that AppDynamics collects for business transactions (not just the key metrics) in the Business Transactions tree of the Metric Browser, where you can plot them on a graph. See [Metric Browser](#).



Apparent Business Transaction Flow Anomalies

- [Learn More](#)

The application flow map displays calls per minute and average response time for calls from a tier to backends, such as databases and remote services, as well as to other tiers.

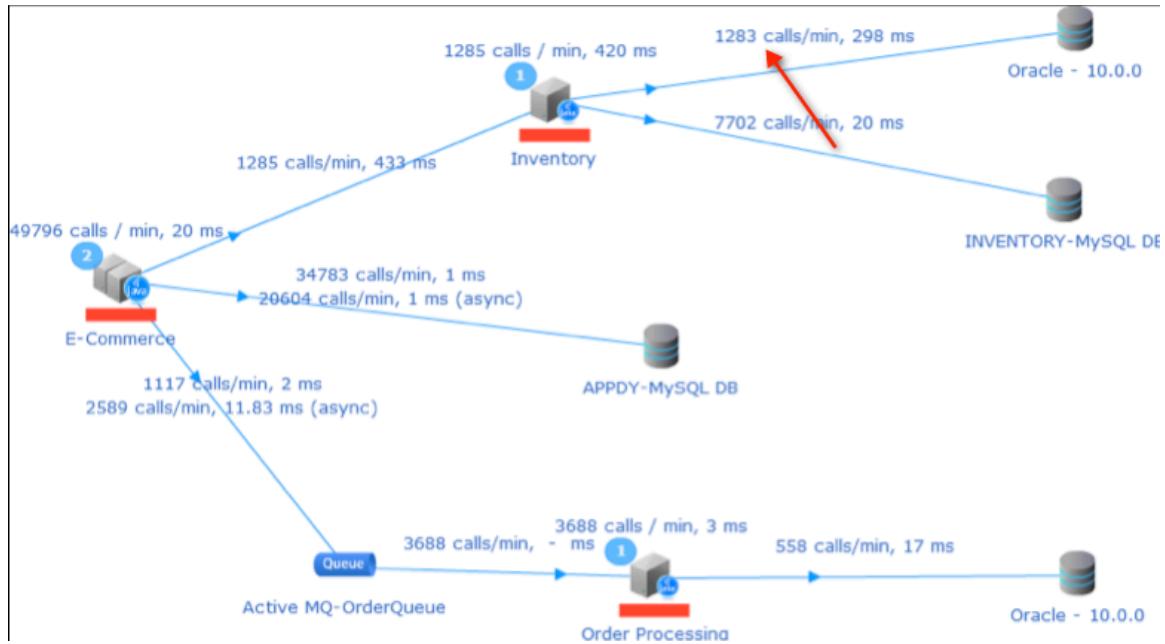
These metrics, as seen along an application flow line, represent the total of all calls made from a tier to another server or backend across *all business transactions* in the application. Similarly, the tier and node flow maps display these metrics across all business transactions that pass through the tier or node.

The business transaction flow map is different from these infrastructure flow maps. Metrics in a business transaction flow map report calls per minute and average response time for calls *per business transaction*.

Therefore, metrics in business transaction flow maps typically do not match metrics in application flow maps.

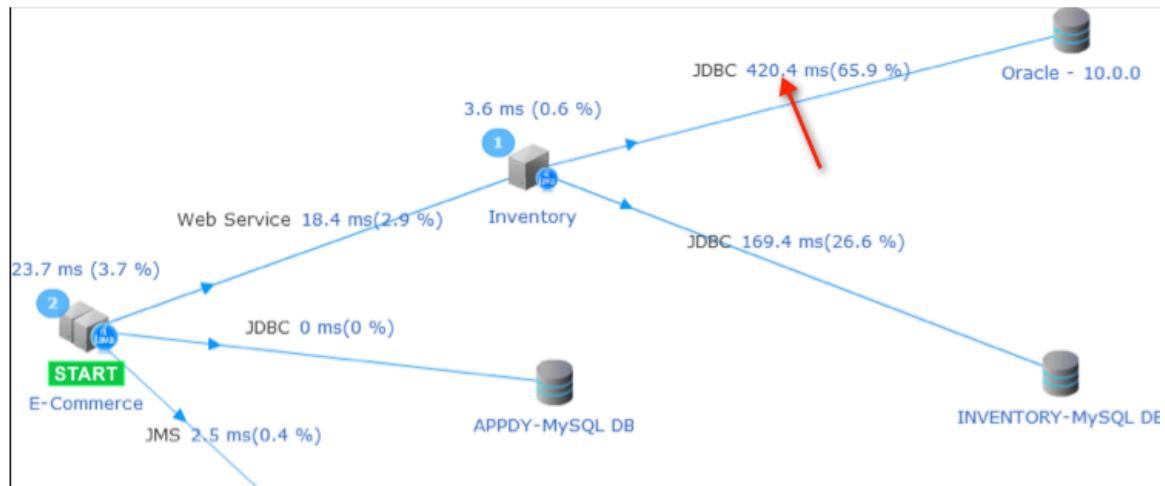
For example, the following application flow map shows that for the selected time range, 1283 calls per minute were made from the Inventory tier to the Oracle-10.0.0 database. The average response time for these calls was 298 ms.

Application Flow Map

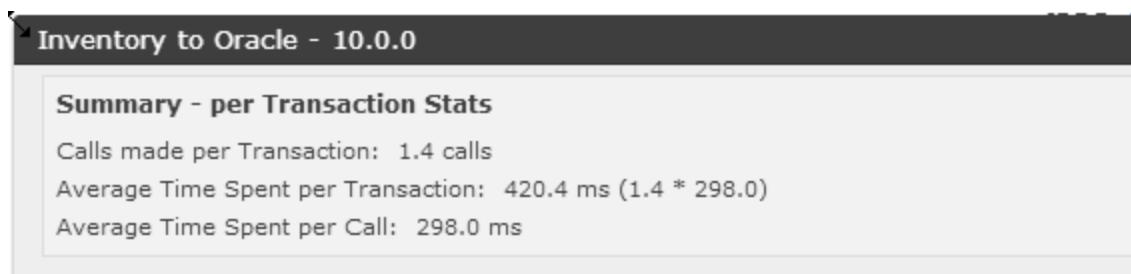


The following business transaction flow map shows the Checkout business transaction during approximately the same time range as the application flow map above. It also shows the calls from the Inventory tier to the Oracle-10.0.0 database for this single business transaction. Because multiple database calls may be executed by a single business transaction, this metric is not likely to match the average response time per call shown between these the tier and the database on the application flow map. This business transaction flow map for the Checkout business transaction shows that the average time per transaction spent on calls from the Inventory tier to the 10.0.0 database was 420.4 ms and that time represents 65.9% of the time used for the entire Checkout business transaction.

Business Transaction Flow Map



You can click the 420.4ms(65.9%) metric to see the summary statistics for the transaction:



The Checkout BT makes 1.4 database calls per transaction. Since the average time per call is 298 ms, the average time spent on database calls per transaction instance for the Checkout transaction is 420.4 ms. This is different from the average response time for all calls from all business transactions from the Inventory tier to the Oracle-10.0.0 database reported in the application flow map.

Learn More

- [Dashboards](#)
- [Flow Maps](#)

Measure Distributed Transaction Performance

- [Distributed Transactions](#)
- [Transaction Tracing](#)
- [Measuring Distributed Communication Performance using Flow Maps](#)
- [Multi-Threaded Transactions](#)
- [Learn More](#)

Distributed Transactions

In a service-oriented environment, when multiple teams are responsible for different services, the performance indicators of an overall transaction (such as average response time) are insufficient when you want to do both performance characterization and troubleshooting. The distributed nature of the transaction is an integral aspect of characterizing transaction performance.

For example, when a transaction flows through multiple tiers, you may ask these questions about the distributed activity:

- How much time is spent in each of the tiers on an average?
- How much time is spent over the network between tiers on an average?
- How many times is each tier being called on an average?
- For a messaging tier, what is the average incoming message rate?
- Are the correct infrastructure components being used (such as staging vs. production)?
- What are the characteristics of an individual request?

AppDynamics applies distributed transaction tracing to answer these types of questions for all business transactions as well as for the entire application.

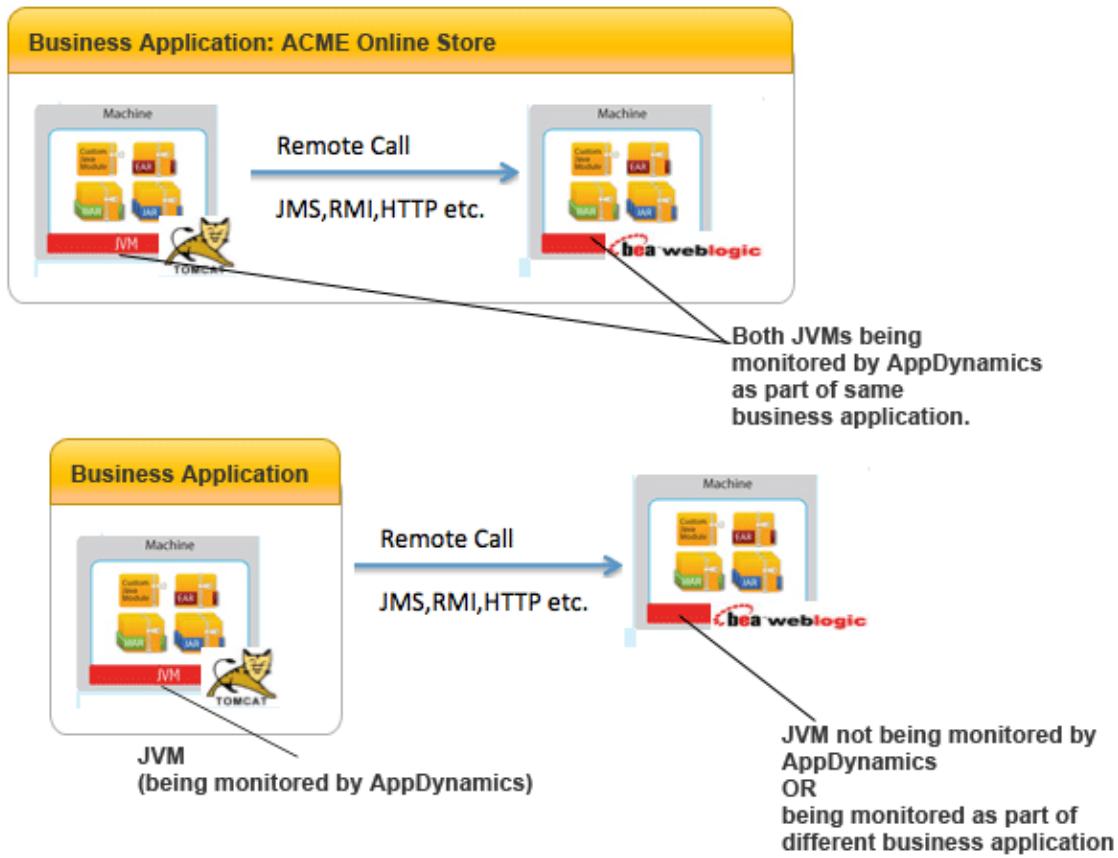
Using a "tag and follow" approach, AppDynamics traces the transaction context across all tiers, including JVMs, CLRs, and calls to HTTP, JMS, SOAP, databases, third party web services, etc.

AppDynamics displays distributed communication performance in the business application and business transaction flow maps.

Transaction Tracing

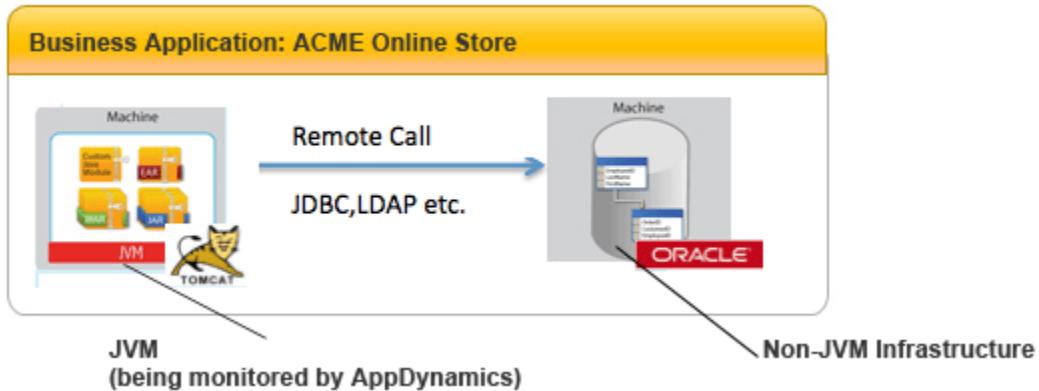
When two tiers communicate use a protocol supported by AppDynamics and both the tiers are instrumented as part of the same business application, the application flow map automatically connects the two corresponding tiers. By default, the flow map displays all measurements per tier. You can drill down to see the performance data for individual nodes in the tier. For details see [Application Dashboard](#) and [Flow Maps](#).

- Inter-JVM or inter-CLR communication



When a JVM or CLR communicates with the external component that is not being monitored as part of the same business application, such as a database or a message queue, you can monitor the performance data for the communication. For more information see [Monitor Databases](#) and [Monitor Remote Services](#).

- CLR or JVM-Backend communication



Measuring Distributed Communication Performance using Flow Maps

The Application Dashboard **flow map** characterizes performance across the business application.

The application flow map provides information about the following performance indicators:

- The time spent on an average for a tier
- The time spent over the network between tiers
- The time spent over the network between a tier and a database or remote service such as a messaging service (backends).

See [Monitor Databases](#) and [Monitor Remote Services](#).

The Business Transaction Dashboard [flow map](#) displays the performance characteristics of all requests for a particular business transaction.

The business transaction flow map provides information about the following performance indicators:

- Rate of distributed activity specific to each tier
- The percentage of time spent in each tier

In addition to providing performance data and showing bottlenecks, you can use flow maps to compare against baselines. Comparisons against baselines help identify anomalies in distributed communication performance. For more information see [Behavior Learning and Anomaly Detection](#).

Multi-Threaded Transactions

If your application spawns multiple threads to perform its task, you can use AppDynamics to monitor individual threads.

See [Trace Multi-Threaded Transactions \(Java\)](#) or [Enable Thread Correlation \(.NET\)](#) depending on your platform.

Learn More

- [Behavior Learning and Anomaly Detection](#)
- [Monitor Databases](#)
- [Monitor Remote Services](#)

Transaction Snapshots

- Deep Code-Level Visibility in Production
 -  See how AppDynamics monitored performance in real time during elections
- Overview of Transaction Snapshots
 - To View Transaction Snapshots
 - Understanding the Flow Map
 - How Transaction Snapshots are Organized
 - Diagnostic Data Captured by a Transaction Snapshot
- Sorting and Searching for Specific Transaction Snapshots
 - To Filter Transaction Snapshots using Search Criteria
 - To Filter Transaction Snapshots by Refining the Results List
 - To Compare Snapshots
 - To Analyze the Most Expensive SQL Call
- Learn More

A transaction snapshot depicts a set of diagnostic data, taken at a certain point in time, for an individual transaction across all app servers though which the transaction has passed.

Transaction snapshots are the vehicle for troubleshooting the root causes of performance problems.

Deep Code-Level Visibility in Production

Code level visibility is essential for troubleshooting performance problems in production. You need details about the exact code path taken by a particular transaction and the time spent in the methods that were executed. A distributed environment presents a challenge because multiple code paths are executed across multiple application servers.

AppMan Advice



See how AppDynamics monitored performance in real time during elections

Overview of Transaction Snapshots

AppDynamics generates a transaction snapshot to capture the code paths executed on instrumented application servers involved in a distributed transaction.

Transaction snapshots contain considerable amounts of data and are processed and sent from the app server on which the application is running. Therefore, they are captured selectively under following conditions:

- **When a diagnostic session is triggered**
AppDynamics starts diagnostic sessions when it detects a pattern of performance problems. In addition you can manually start a diagnostic session from the Business Transaction Dashboard. For details see [Diagnostic Sessions](#).
- **When slow, stalled, or error transactions are identified**
These snapshots may have partial call graph information, starting at the time when the transaction slowed or experienced an error.
- **Based on the Periodic Collection setting**
By default AppDynamics captures one snapshot every 10 minutes. For details see [Transaction Snapshots](#).

To View Transaction Snapshots

You can get a list of transaction snapshots for the selected time range:

- From the **Transaction Snapshots** tab of the application, tier, node, or business transaction dashboards
- From the links in the transaction scorecards in the application, tier, node, or business transaction dashboards
- From the **Troubleshoot-> Slow Response Time** or **Troubleshoot-> Errors** in the left navigation pane

1. Navigate to an application, tier, node, or business transaction dashboard.
2. Click the **Transaction Snapshots** tab.
3. From the list of transaction snapshots that displays, select the snapshot that you want to view and click **View Transaction Snapshot**.
4. In the transaction flow map click **Drill Down**.

Or

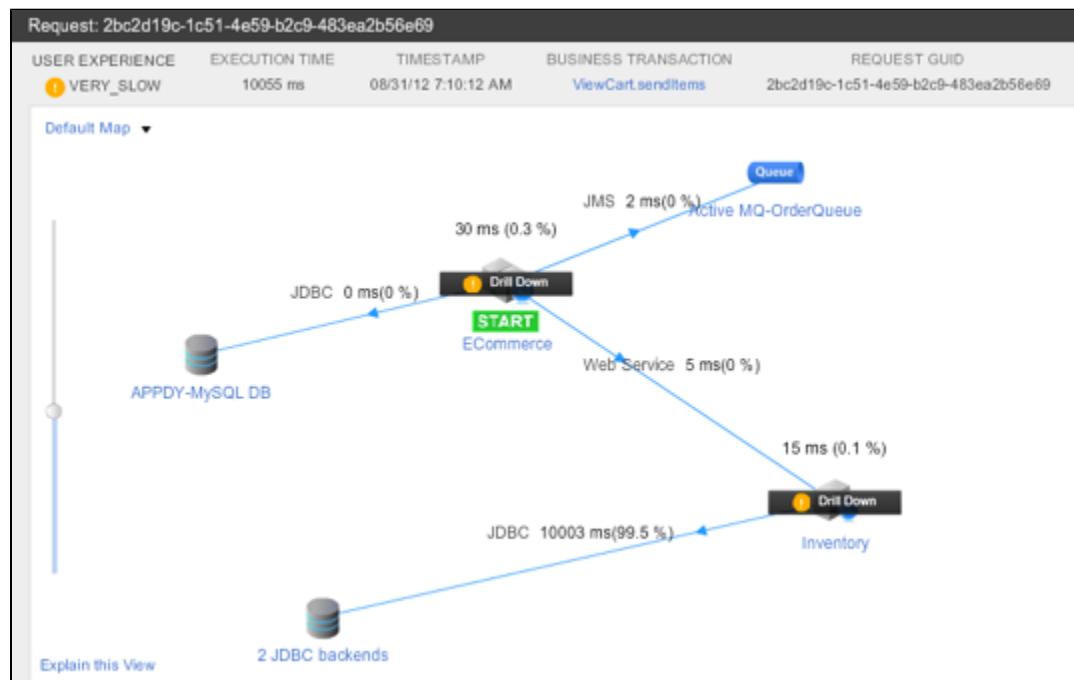
1. Navigate to an application, tier, node, or business transaction dashboard.
2. Click the individual slow, very slow, stalls or errors links in the Transaction Scorecard section of the dashboard.
3. From the list of transaction snapshots that displays, select the snapshot that you want to view and click **View Transaction Snapshot**.
4. In the transaction flow map click **Drill Down**.

Or

1. Click **Troubleshoot-> Slow Response Time** or **Troubleshoot-> Errors** in the left navigation pane.
2. Select a slow or error transaction from the list.
3. Click **View Transaction Snapshot**.
4. In the transaction flow map click **Drill Down**.

Understanding the Flow Map

When you double-click on any item in the list of transaction snapshots, a flow map displays. You can see the user experience, execution time, and timestamp of the transaction. The flow map also provides details of the overall time that is spent in a particular tier and in database and remote service calls



How Transaction Snapshots are Organized

A transaction snapshot contains diagnostic information for individual business transaction instances. It provides information about why a particular transaction or series of transactions are slow or have errors. You get this information when you click **Drill Down** on a tier in the flow map.

Diagnostic Data Captured by a Transaction Snapshot

The following details are captured in a transaction snapshot:

- **Summary:** Problem summary, execution time, CPU, timestamps tier, node process ID, thread name, etc.
- **Call Graphs:** Call graphs for every tier involved in a transaction. You can drill down to the method call that is causing the problem. AppDynamics automatically filters non-application classes such as core Java classes, third party libraries, application server and database driver implementation classes. You can configure call graph settings to control which classes should be included or excluded from the call graph. To configure, see [Configure Call Graphs](#).
- **Hot Spots:** Call graphs for the most expensive calls
- **SQL Calls:** All SQL queries fired during a request. AppDynamics normalizes the queries and by default does not display raw/bind values. You can use SQL data collectors to limit the type of queries captured by AppDynamics. For information see [Configure Data Collectors](#). You can also configure SQL capture settings to monitor raw SQL data in the queries. To configure, see [Configure Call Graphs](#).
- **HTTP Params:** HTTP payloads contain basic data such as the URL and session ID, and additional data for Servlet entry points, Struts, JSF, Web Services, etc. You can use HTTP data collectors to specify which query parameter or cookie values should be captured in the transaction snapshot. To configure, see [Configure Data Collectors](#).
- **Cookies:** The snapshot can use cookie values to help identify the user who initiated the slow or error transaction. To configure, see [Configure Data Collectors](#).
- **User Data:** User data from any method executed during a transaction, including parameter values and return values, to add context to the transaction. You can use method invocation data collectors to specify the method and parameter index. To configure, see [Configure Data Collectors](#).
- **Error Details:** Exception stack traces and HTTP error codes.

- **Hardware/Mem:** Graphs for hardware (CPU Memory, Disk IO, Network IO), Memory (Heap, Garbage Collection, Memory Pools), JMX, etc.
- **Node Problems:** Viewer to analyze node problems. See [Troubleshoot Node Problems](#).
- **Additional Data**

Transaction snapshots include distributed call graphs and response time distribution details only when a series of bad transactions or a performance policy violation trigger a diagnostic session on a node.

Sorting and Searching for Specific Transaction Snapshots

To Filter Transaction Snapshots using Search Criteria

1. In the **Transaction Snapshots** tab click the **All Snapshots** subtab.
2. Click **Show Filters** if filters are not showing.
3. Click **Search Criteria**. Check the following criteria to sort the list:
 - **User Experience**
 - Slow
 - Very Slow
 - Stall
 - **Execution Time** in milliseconds
 - **Business Transaction**
 - Click **Add** (the + icon) to select a Business Transaction as a search criteria
 - **Errors**
 - Click **Error Occurred** to list all errors
 - Click **Add** (the + icon) to select a particular error as a search criteria
 - **HTTP Request Data**
 - URL
 - Session ID
 - User Principal
 - **Data Collector Data**
 - Collector Type: Any, HTTP Parameter, Business Data, Cookie
 - Name
 - Value
 - **Archived**
 - Return Only Archived Snapshots
 - **Advanced**
 - Request GUIDs

3. Click **Search**.

To Filter Transaction Snapshots by Refining the Results List

You can refine the results list of a previous query. AppDynamics instantly updates the list as you select or deselect the **Refine Results** filters.

1. In the **Transaction Snapshots** tab of a dashboard click the **All Snapshots** subtab.
2. Click **Show Filters**.
3. Click **Refine Results**. Use the following criteria to sort the list:
 - **User Experience**
 - Normal
 - Slow
 - Very Slow
 - Stall

- **Business Transactions**
- Business Transactions by name
AppDynamics also shows the number of snapshots.
- **Tiers**
- Tiers by name
AppDynamics also shows the number of snapshots.
- **Nodes**
Nodes by name
AppDynamics also shows the number of snapshots.
- **Errors**
Snapshots where an error occurred

To Compare Snapshots

1. In the transaction snapshot list, select two snapshots that you want to compare.

2. Click **Analyze** -> **Compare Snapshots**.

AppDynamics displays a comparison of the selected snapshots.

Class and Method	Time in S1	Time in S2	Change (ms)	Call Count
com.appdynamics.inventory.OrderServiceSOAP11BindingStub:createOrder:158	10008	10010	2	1
Spring Bean - proucerJmsTemplate:doSend:513	3	4	1	4

To Analyze the Most Expensive SQL Call

1. Select a single or a group of snapshots that you want to analyze.

2. Click **Analyze** -> Identify the most expensive SQL calls in a group of snapshots.

AppDynamics displays the most expensive call in the snapshots and also isolates the expensive SQL calls.

Learn More

- Configure Transaction Snapshots
- Call Graphs
- Diagnostic Sessions
- Configure Data Collectors
- Configure Call Graphs

Configure Transaction Snapshots

- To Configure Periodic Snapshot Collection
- To Disable Periodic Snapshots

To Configure Periodic Snapshot Collection

1. In the left navigation pane click **Configure** -> **Slow Transaction Thresholds**.
2. In the view navigation, select **Default Thresholds**.
3. Modify the settings for periodic snapshots in the Configure Periodic Snapshot Collection section.

Important: AppDynamics recommends that you do not use low values if you have a high load production environment. When there are thousands or millions of requests per minute, collecting snapshots frequently could flood the system with many snapshots that may not be of high value. Either turn OFF the periodic snapshots and apply to all Business Transactions, or choose a very conservative (high) rate depending on the expected load. For example, if you have high load on the application, choose every 1000th executions or every 20 minutes, depending on the load pattern.

If your SLA-violation-based policies are enabled, you can safely disable the periodic snapshots. See [Policies](#).

To Disable Periodic Snapshots

1. In the left navigation pane click **Configure** -> **Slow Transaction Thresholds**.
2. In the view navigation, select **Default Thresholds**.
3. Clear the following check boxes: Take one Snapshot every xxx executions*...* and Take one Snapshot every xxx minutes.



4. Click **Save Default Diagnostic Session Settings**.

Alerting for Business Transaction Health Problems

- Default Health Rules for Business Transactions
- Alerting with Notification Actions
- Creating Policies to Match Health Rule Violations with Alerts
- Learn More

Business transaction health refers to the extent to which a business transaction is experiencing critical and warning health rule violations.

Use health rules, notification actions and policies to alert staff of business transaction performance problems.

Default Health Rules for Business Transactions

AppDynamics provides the following default health rules for business transaction performance:

- **Business Transaction response time is much higher than normal**

This rule defines a critical condition as the combination of an average response time greater than the default baseline by 3 standard deviations and a load greater than 50 calls per minute.

This rule defines a warning condition as the combination of an average response time greater than the default baseline by 2 standard deviations and a load greater than 100 calls per minute.

- **Business Transaction error rate is much higher than normal**

This rule defines a critical condition as the combination of an error rate greater than the default baseline by 3 standard deviations and an error rate greater than 10 errors per minute and a load greater than 50 calls per minute.

This rule defines a warning condition as the combination of an error rate greater than the default baseline by 2 standard deviations and an error rate greater than 5 errors per minute and a load greater than 50 calls per minute.

You can use these health rules as they are or you can modify them.

For information on how AppDynamics determines normal performance, see [Behavior Learning and Anomaly Detection](#) and [Configure Baselines](#).

For information about modifying health rules see [Health Rules](#) and [Configure Health Rules](#).

Alerting with Notification Actions

You can create notification actions to set up email and SMS addresses to receive notifications when the health rules are violated. See [Notification Actions](#).

You can also create email digests that are sent on a predefined schedule to email recipients summaries of these (and other) health rule violations. See [Email Digests](#).

Alerts in email notifications and digests provide a deep link to help the recipient start analyzing the root cause of the problem.

Creating Policies to Match Health Rule Violations with Alerts

Create one or more policies to assign a notification action to a specific health rule violation. You can optionally create different policies for warning and critical health rule violations. See [Policies](#) and [Configure Policies](#).

If you want simply to send an email notification to a single email address whenever any health rule violation is started, you can use the Alerting Wizard.

Learn More

- [Behavior Learning and Anomaly Detection](#)
- [Configure Baselines](#)
- [Health Rules](#)
- [Configure Health Rules](#)
- [Policies](#)
- [Configure Policies](#)
- [Notification Actions](#)
- [Email Digests](#)
- [Alerting Wizard](#)

Data Collectors

- [Collecting Diagnostic Data from Application Payload](#)
 - To view the data collectors for a particular business transaction
 - To apply a data collector to a business transaction
- [Learn More](#)

Data collectors help you determine whether data that a transaction passed into an application is causing problems. When you configure data collectors AppDynamics accesses information in application code arguments, return values, and variables and displays the information in Call Drill Down panels.

Collecting Diagnostic Data from Application Payload

Some application performance problems are related to the data processed in a request.

For example, consider a transaction that searches for a specific category. If a particular category has a problem, it can be difficult to diagnose the problem. You would need to:

- Isolate the name of the category that experienced a slow or failed search.

- Identify all the categories that experienced slow or failed searches.
- Other details such as which users experienced problems accessing the search functionality.

AppDynamics can collect contextual business data either to correlate between different performance problems or to correlate a particular performance problem with the business data. The contextual data can be collected either from the HTTP payload data or from the methods that are invoked as part of the transaction.

You can capture the following diagnostic data on business context attached to the business transaction:

- The basic details such as the type of business transaction, URI, and the time stamp
- The HTTP parameters for the transaction
- The cookies set for the transaction
- The SQL queries executed as part of the transaction
- User data gathered from any POJO or POCO method executed during the transaction

AppDynamics provides data collectors to collect the business context for each incoming business transaction. These data collectors capture the diagnostic data from all transactions that are slow, very slow, stalled or that experience any error during execution. If data collectors are configured, the data appears in various panels, such as the HTTP Params, Cookies, and User Data sections, of a transaction snapshot. To access these panels see [To view call graphs](#).

The screenshot shows a sidebar menu for a transaction with the ID 45318781. The menu items are: SUMMARY, CALL GRAPH, HOT SPOTS, SQL CALLS, HTTP PARAMS (highlighted with a blue background and a red arrow), COOKIES (highlighted with a blue background and a red arrow), USER DATA (highlighted with a blue background and a red arrow), ERROR DETAILS, HARDWARE / MEM, NODE PROBLEMS, and ADDITIONAL DATA.

These are three types of data collectors

- HTTP Diagnostic Data Collectors
- SQL Data Collectors
- Method Invocation Data Collectors

Collectors can be configured for a particular business transaction or for all business transactions in a business application.

Data collectors are different than information points, which gather data outside the context of a business transaction. See [Data Collectors Versus Information Points](#).

To view the data collectors for a particular business transaction

1. In the business transactions list, select a business transaction.
2. Click the **More Actions** drop-down menu.
3. Click **Configure Data Collectors** to view the data collectors configured for that transaction.

To apply a data collector to a business transaction

1. Follow the instructions above in [To View the Data Collectors for a Particular Business Transaction](#).
2. Move the data collector from the Available Data Collectors list to the Data Collectors list.
3. Click **Save**.

Learn More

- [Configure Data Collectors](#)
- [Data Collectors Versus Information Points](#)

Configure Data Collectors

- [Configuring Method Invocation Data Collectors](#)
 - [To configure Method Invocation Data Collectors](#)
 - To use a getter chain to specify the data collection on method invocation
- [Configuring HTTP Data Collectors](#)
 - [To Configure HTTP Data Collectors](#)
 - To capture all the parameters for HTTP Request data
 - To use the HTTP content-length header to monitor the correlation between the size of response and overall throughput for an application:
- [Configuring SQL Data Collectors](#)
 - [To Configure SQL Data Collectors](#)
- [Learn More](#)

This topic provides instructions on configuring data collectors.

To access data collector configuration:

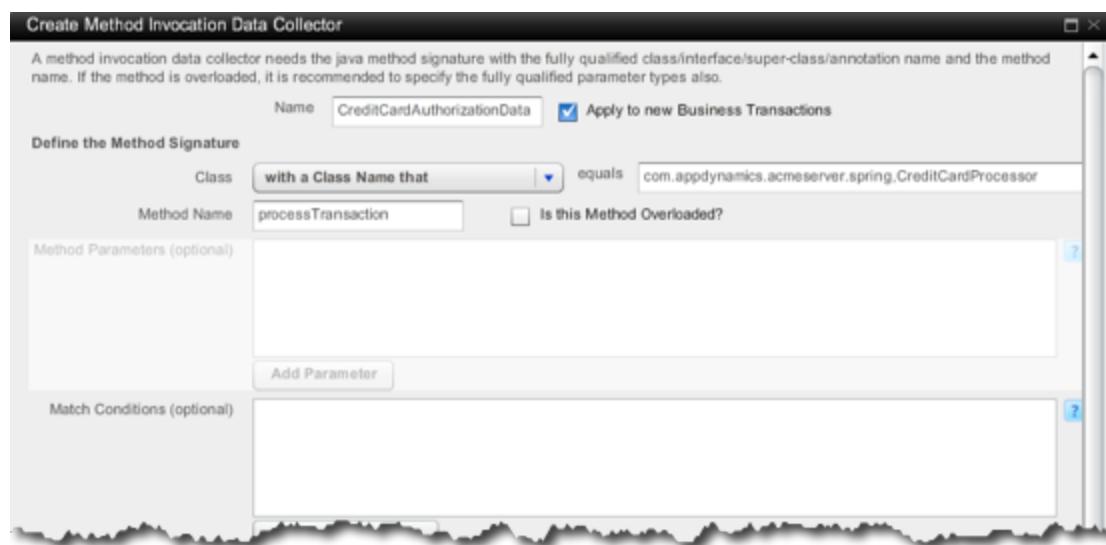
1. In the left navigation panel, click **Configure -> Instrumentation**.
2. Click the **Diagnostic Data Collectors** tab.

Configuring Method Invocation Data Collectors

Configure custom method invocation data collectors to capture parameters or return value for a particular method. The captured data appears in panels of the snapshot viewer.

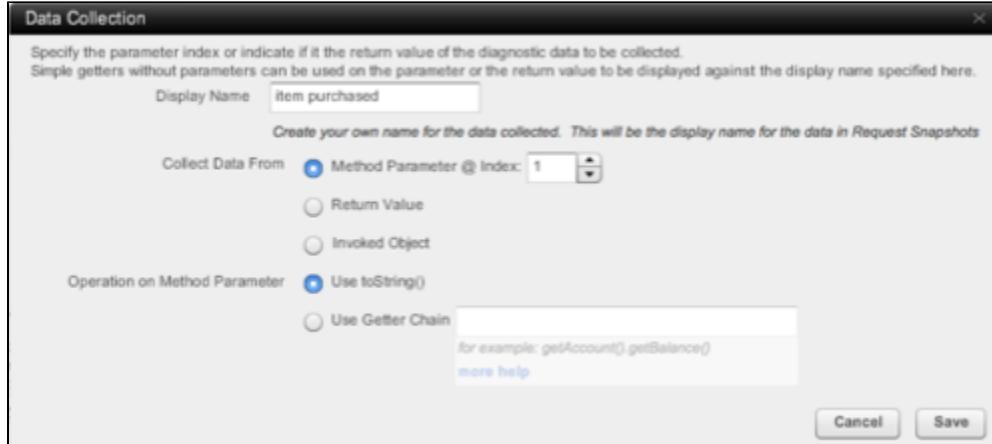
To configure Method Invocation Data Collectors

1. In the Method Invocation Data Collectors panel, click **Add**.
2. Define the method signature.
Provide the class, method, and parameters if the method is overloaded.



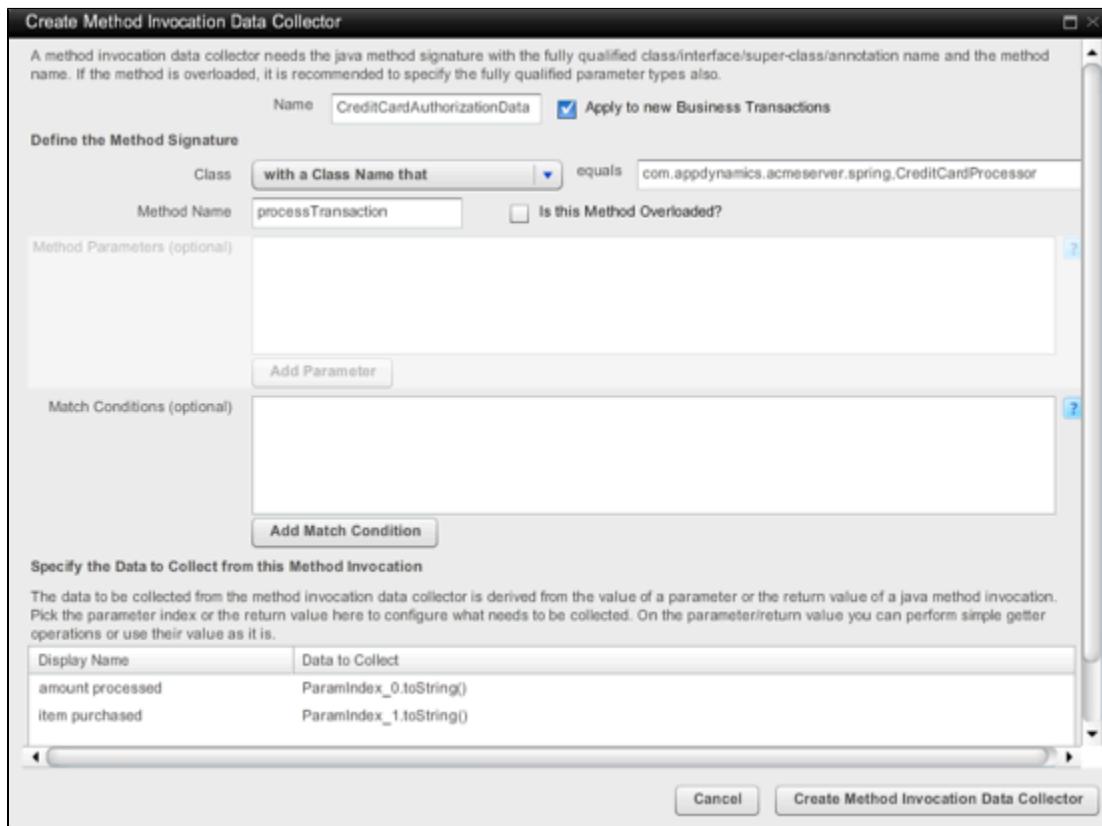
3. To specify the data to collect from this method invocation, click **Add** in the Specify the Data to Collect from this Method Invocation panel.

4. In the Data Collection screen, specify where to collect the data and which operation to use to collect it.



5. To enable this data collector for newly discovered transactions, check **Apply to new Business Transactions**.

6. Click **Create Method Invocation Data Collector**.



To use a getter chain to specify the data collection on method invocation

Specify the getter chain in the Data Collection screen.

Data Collection

Specify the parameter index or indicate if it the return value of the diagnostic data to be collected. Simple getters without parameters can be used on the parameter or the return value to be displayed against the data collector.

Display Name

Book Name

Create your own name for the data collected. This will be the display name for the data collector.

Collect Data From

Method Parameter @ Index:

1	▲	▼
---	---	---

Return Value

Invoked Object

Use `toString()`

Use Getter Chain

`getTitle().getName()`

for example: `getAccount().getBalance()`

Operation on Method Parameter



Specify the Data to Collect from this Method Invocation

The data to be collected from the method invocation data collector is derived from the value of a parameter or the return value of the method. Pick the parameter index or the return value here to configure what needs to be collected. On the right, you can see the operations or use their value as it is.

Display Name

Data to Collect

Book Name

ParamIndex_1.getTitle().getName()

See:

[37BKUP: Getter Chains in Java Configurations]

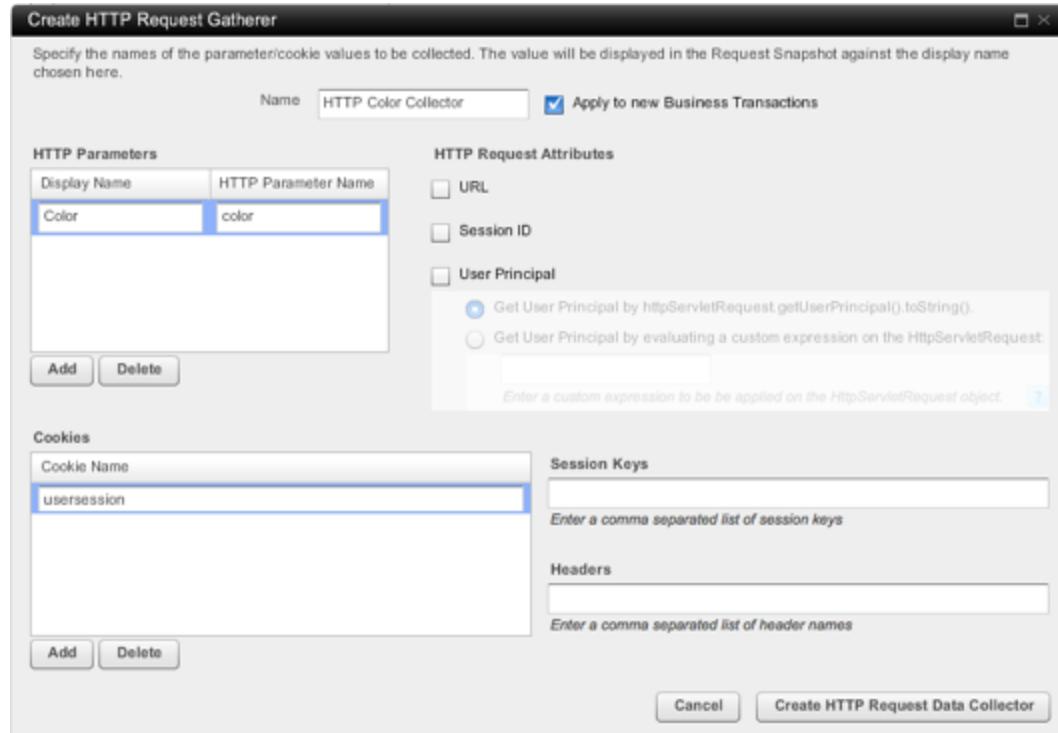
Getter Chains in .NET Configurations

Configuring HTTP Data Collectors

Configure custom HTTP data collectors to collect HTTP payload data for diagnosis in the transaction snapshot.

To Configure HTTP Data Collectors

1. In the HTTP Request Data Collectors panel, click **Add**.
2. Click **Add** in the HTTP Parameters section to set the parameter values that you want to collect.
3. Click **Add** in the Cookies section to set the cookie values that you want to collect.
4. If you want to enable this data collector for newly discovered transactions, check **Apply to new Business Transactions**.
5. Click **Create HTTP Request Data Collector**.



You can also configure the HTTP data collector to capture following data:

- URL
- Session ID
- User Principal
- Session Keys
- Headers

To capture all the parameters for HTTP Request data

Specify "*" in the HTTP Parameter Name section.



To use the HTTP content-length header to monitor the correlation between the size of response and overall throughput for an application:

Configure the Content-Length in the Headers section.



The content length header appears in the HTTP Params section of the transaction snapshot.

A screenshot of the AppDynamics transaction snapshot interface. The top bar shows 'Call Drill Down (Request: b66ea32f-8d57-4e12-8731-6cbfd7bcf28)'. On the left, there's a sidebar with tabs: 'SUMMARY' (selected), 'CALL GRAPH', 'HOT SPOTS', 'SQL CALLS', and 'HTTP PARAMS' (highlighted with a red arrow). The main area displays a table titled 'The HTTP parameter values/cookies for a particular parameters/cookies to collect.' with one row: 'Name' (Header-content-length) and 'Value' (56).

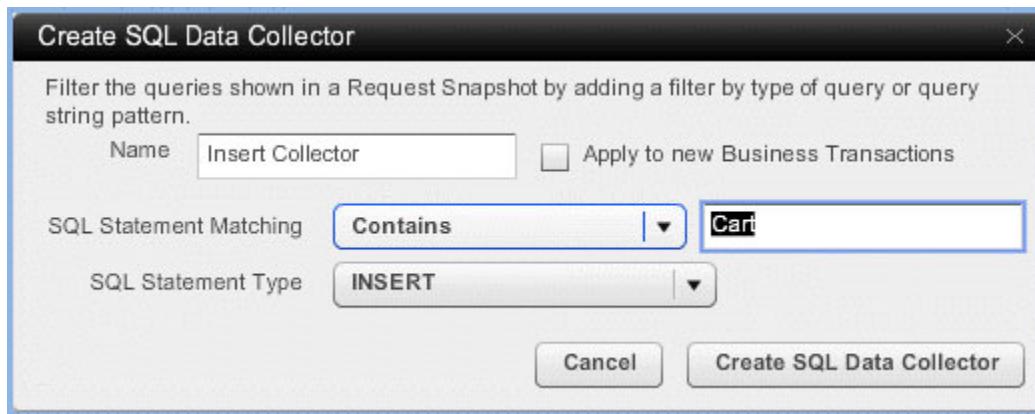
Configuring SQL Data Collectors

Use SQL data collectors to restrict diagnostic data captured in transaction snapshots.

To Configure SQL Data Collectors

1. In the SQL Data Collectors panel click **Add**.
2. In the Create SQL Data Collector screen name and configure the SQL filters.
You can filter the diagnostic data for SQL queries using either:
 - SQL statement matching
 - SQL statement type (whether this statement is of type INSERT, UPDATE, etc.).
3. To apply this data collector to new business transactions, check **Apply to new Business Transactions**.
4. Click **Create SQL Data Collector**.

The following example shows a SQL data collector to collect only INSERTs that contain the string, "Cart".



Learn More

- [Data Collectors](#)

Data Collectors Versus Information Points

- Differences Between Data Collectors and Information Points
 - [Data Collectors](#)
 - [Information Points](#)
- [Learn More](#)

Differences Between Data Collectors and Information Points

Sometimes there is confusion between data collectors and information points because both enhance performance monitoring with information about data passed to an application and both must be explicitly configured. It is also possible to create data collectors and information points on the same method, but for different purposes.

A major difference between data collectors and information point metrics is that:

- Data collectors exist in the context of a business transaction
- Information points exist outside of business transactions

Data Collectors

Data collectors exist only in the context of a business transaction. They add information to a business transaction snapshot, where they can highlight poor performance that may be associated with data passed in by the request. Data from data collectors appear in the panels of the transaction snapshot call drill down - HTTP PARAMS, COOKIES, or USER DATA - depending on the type of the data collector. Data collectors do not appear in the Metric Browser and cannot be used in health rules. You can filter transaction snapshots based on the value of a data collector in the transaction snapshot list or using the AppDynamics REST API.

Here are some use cases for data collectors:

- You want to identify which users are experiencing problems with a business transaction. Often the userid is carried in the request as an HTTP query string or in the cookie. Create a data collector on the userid so you can identify the users in transaction snapshots for slow transactions. You can then filter transaction snapshots for those specific user ids.
- You can identify a business context for slow transactions. Use a method invocation data collector to extract some business information, such as the country or the orderid, to better understand the execution context in which users are experiencing poor performance.
- You can identify a technical context for slow transactions. For example, you may suspect that a cache is performing poorly for a specific set of keys, so you create a data collector on the cache retrieve method. Then the transaction snapshots can help determine if there is correlation between the suspect keys and poor performance.
- You can take advantage of the ability to filter transaction snapshots on a data collector to tag snapshots. For example, create a data collector based on a header value or path pattern to tag transaction snapshots initiated by synthetic transactions.

Information Points

Information points aggregate numeric data about invocations of a method outside the context of any business transaction. Use information points to get metrics about method invocations across multiple business transactions. Any time the method configured for the information point is invoked, no matter from where, the information point is triggered.

Here are some use cases for information points:

- You can use a code metric information point to track the number of times a servlet is invoked when the servlet is not part of a business transaction.
- You have a slow XML parser used by many business transactions. Use an information point to report when the parser was called and how many lines it parsed.
- The development team is trying to optimize common application code used by many business transactions. Creating information points on the important methods can give them KPIs to assess the success of their optimization efforts.
- You can use two information points to track the number of concurrent users of an application, regardless of the business transaction. Create an information point on the login method, another on the logout method. The difference between them is the concurrent users logged into the application in the specified time range.
- Create a custom business metric based to report how many of a certain item were sold or returned or how many credit cards were being rejected.

Learn More

- Configure Code Metric Information Points
- Code Metrics
- Business Metrics

Configure Business Transaction Detection

- Planning Business Transaction Detection Configuration
- Basic Configurations
 - To access business transaction detection configuration
- Entry Points
- Exclude Rules
 - Default Exclude Rules
 - To Create Custom Exclude Rules
 - To View Default Exclude Rules
 - Default Exclude Rules for Java Platforms
 - Default Exclude Rules for .NET Platforms
- Custom Match Rules
 - To Create Custom Match Rules
 - Sample Custom Match Rule

- Transaction Splitting
 - Sample Custom Match Rule with Split
- Sequence and Precedence for Auto-Naming and Custom Match Rules
 - The Priority Parameter
- Transaction Detection Concepts
- Learn More

Planning Business Transaction Detection Configuration

Before you configure transaction discovery, select and name the operations you want to monitor. For example, you could:

- Interview the team responsible for each tier to identify the key 5 or 10 or 20 operations that they feel are important to monitor. Identify which key operations must work well for the tier to be a successful part of the site.
- Map the operations that you have identified to the appropriate business transaction entry points. The entry point to a business transaction is an operation that begins and ends every time the user request is invoked.
- Name the business transactions so that it is recognizable to everyone who monitors your application.

You can keep fine tuning your business transaction configuration, excluding some transactions that you currently detect and adding others that you have not been detecting, as you learn more about which transactions give you the most useful information about your application.

Basic Configurations

There are three areas you can configure:

- Entry Points
- Exclude Rules
- Custom Match Rules

To access business transaction detection configuration

1. From the left navigation pane select **Configure -> Instrumentation**.
2. Click the **Transaction Detection** tab if it is not already selected.
3. Click the detection subtab that corresponds to your environment.
4. Select the tier for which you want to configure the transactions or select the application if you want to configure at the application level.
5. To configure a custom configuration for the tier, select Use Custom Configuration for this tier. For information about inheritance for transaction detection, see [Hierarchical Configuration Model](#).

Entry Points

See [Web Entry Points](#) and [Messaging Entry Points](#).

Exclude Rules

Create exclude rules to prevent detection of transactions that match certain criteria.

You might want to do this because AppDynamics is detecting business transactions that you are not interested in monitoring or to stay under the agent and controller limits for detected transactions or to exclude a default entry point so you can substitute a more appropriate entry using a custom match rule. See [Entry Points](#) for information about the default auto-detected entry points and [Custom Match Rules](#) for information about creating your own match rules for transaction detection.

For example, the Weblogic Jax RPC Servlets rule excludes any business transaction with a servlet-type entry point based on a class with a name that starts with "weblogic.wsee.server.servlet".

Exclude Business Transaction Match Rule - Servlet

Name	Weblogic JAX-RPC Servlets																																																							
Enabled	<input checked="" type="checkbox"/>																																																							
<input type="button" value="Transaction Match Criteria"/> <input type="button" value="Split Transactions Using Request Data"/> <input type="button" value="Split Transactions Using Payload"/>																																																								
<table border="1"> <tr> <td><input type="checkbox"/></td> <td>Method</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/></td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>URI</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/></td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>HTTP Parameter (Both GET query parameters and POST parameters can be used)</td> <td><input type="button" value="Check for parameter value"/></td> <td><input type="button" value="Parameter Name"/></td> <td><input type="text"/></td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td></td> <td><input type="button" value="Value"/></td> <td><input type="button" value="Equals"/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>Header</td> <td><input type="button" value="Check for parameter value"/></td> <td><input type="button" value="Parameter Name"/></td> <td><input type="text"/></td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td></td> <td><input type="button" value="Value"/></td> <td><input type="button" value="Equals"/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>Hostname</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/></td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>Port</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/></td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input checked="" type="checkbox"/></td> <td>Class Name</td> <td><input type="button" value="Starts With"/></td> <td><input type="text"/> weblogic.wsee.server.servlet.</td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>Servlet Name</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/></td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>Cookie</td> <td><input type="button" value="Check for cookie existence"/></td> <td><input type="button" value="Cookie Name"/></td> <td><input type="text"/></td> </tr> </table>		<input type="checkbox"/>	Method	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>	<input type="checkbox"/>	URI	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>	<input type="checkbox"/>	HTTP Parameter (Both GET query parameters and POST parameters can be used)	<input type="button" value="Check for parameter value"/>	<input type="button" value="Parameter Name"/>	<input type="text"/>	<input type="checkbox"/>			<input type="button" value="Value"/>	<input type="button" value="Equals"/>	<input type="checkbox"/>	Header	<input type="button" value="Check for parameter value"/>	<input type="button" value="Parameter Name"/>	<input type="text"/>	<input type="checkbox"/>			<input type="button" value="Value"/>	<input type="button" value="Equals"/>	<input type="checkbox"/>	Hostname	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>	<input type="checkbox"/>	Port	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>	<input checked="" type="checkbox"/>	Class Name	<input type="button" value="Starts With"/>	<input type="text"/> weblogic.wsee.server.servlet.	<input type="button" value="..."/>	<input type="checkbox"/>	Servlet Name	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>	<input type="checkbox"/>	Cookie	<input type="button" value="Check for cookie existence"/>	<input type="button" value="Cookie Name"/>	<input type="text"/>
<input type="checkbox"/>	Method	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>																																																				
<input type="checkbox"/>	URI	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>																																																				
<input type="checkbox"/>	HTTP Parameter (Both GET query parameters and POST parameters can be used)	<input type="button" value="Check for parameter value"/>	<input type="button" value="Parameter Name"/>	<input type="text"/>																																																				
<input type="checkbox"/>			<input type="button" value="Value"/>	<input type="button" value="Equals"/>																																																				
<input type="checkbox"/>	Header	<input type="button" value="Check for parameter value"/>	<input type="button" value="Parameter Name"/>	<input type="text"/>																																																				
<input type="checkbox"/>			<input type="button" value="Value"/>	<input type="button" value="Equals"/>																																																				
<input type="checkbox"/>	Hostname	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>																																																				
<input type="checkbox"/>	Port	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>																																																				
<input checked="" type="checkbox"/>	Class Name	<input type="button" value="Starts With"/>	<input type="text"/> weblogic.wsee.server.servlet.	<input type="button" value="..."/>																																																				
<input type="checkbox"/>	Servlet Name	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>																																																				
<input type="checkbox"/>	Cookie	<input type="button" value="Check for cookie existence"/>	<input type="button" value="Cookie Name"/>	<input type="text"/>																																																				
<input type="button" value="Cancel"/> <input type="button" value="Save"/>																																																								

The ASP.NET WebService Session Handler rule excludes any business transaction with an ASP.NET-type entry point based on the System.Web.Services.Protocols.SyncSessionlessHandler class.

Exclude Business Transaction Match Rule - ASP .NET

Name	Service Session Handler																																																		
Enabled	<input checked="" type="checkbox"/>																																																		
<input type="button" value="Transaction Match Criteria"/> <input type="button" value="Split Transactions Using Request Data"/>																																																			
<table border="1"> <tr> <td><input type="checkbox"/></td> <td>Method</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/></td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>URI</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/></td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>HTTP Parameter (Both GET query parameters and POST parameters can be used)</td> <td><input type="button" value="Check for parameter value"/></td> <td><input type="button" value="Parameter Name"/></td> <td><input type="text"/></td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td></td> <td><input type="button" value="Value"/></td> <td><input type="button" value="Equals"/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>Header</td> <td><input type="button" value="Check for parameter value"/></td> <td><input type="button" value="Parameter Name"/></td> <td><input type="text"/></td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td></td> <td><input type="button" value="Value"/></td> <td><input type="button" value="Equals"/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>Hostname</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/></td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>Port</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/></td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input checked="" type="checkbox"/></td> <td>Class Name</td> <td><input type="button" value="Equals"/></td> <td><input type="text"/> System.Web.Services.Protocols.</td> <td><input type="button" value="..."/></td> </tr> <tr> <td><input type="checkbox"/></td> <td>Cookie</td> <td><input type="button" value="Check for cookie existence"/></td> <td><input type="button" value="Cookie Name"/></td> <td><input type="text"/></td> </tr> </table>		<input type="checkbox"/>	Method	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>	<input type="checkbox"/>	URI	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>	<input type="checkbox"/>	HTTP Parameter (Both GET query parameters and POST parameters can be used)	<input type="button" value="Check for parameter value"/>	<input type="button" value="Parameter Name"/>	<input type="text"/>	<input type="checkbox"/>			<input type="button" value="Value"/>	<input type="button" value="Equals"/>	<input type="checkbox"/>	Header	<input type="button" value="Check for parameter value"/>	<input type="button" value="Parameter Name"/>	<input type="text"/>	<input type="checkbox"/>			<input type="button" value="Value"/>	<input type="button" value="Equals"/>	<input type="checkbox"/>	Hostname	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>	<input type="checkbox"/>	Port	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>	<input checked="" type="checkbox"/>	Class Name	<input type="button" value="Equals"/>	<input type="text"/> System.Web.Services.Protocols.	<input type="button" value="..."/>	<input type="checkbox"/>	Cookie	<input type="button" value="Check for cookie existence"/>	<input type="button" value="Cookie Name"/>	<input type="text"/>
<input type="checkbox"/>	Method	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>																																															
<input type="checkbox"/>	URI	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>																																															
<input type="checkbox"/>	HTTP Parameter (Both GET query parameters and POST parameters can be used)	<input type="button" value="Check for parameter value"/>	<input type="button" value="Parameter Name"/>	<input type="text"/>																																															
<input type="checkbox"/>			<input type="button" value="Value"/>	<input type="button" value="Equals"/>																																															
<input type="checkbox"/>	Header	<input type="button" value="Check for parameter value"/>	<input type="button" value="Parameter Name"/>	<input type="text"/>																																															
<input type="checkbox"/>			<input type="button" value="Value"/>	<input type="button" value="Equals"/>																																															
<input type="checkbox"/>	Hostname	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>																																															
<input type="checkbox"/>	Port	<input type="button" value="Equals"/>	<input type="text"/>	<input type="button" value="..."/>																																															
<input checked="" type="checkbox"/>	Class Name	<input type="button" value="Equals"/>	<input type="text"/> System.Web.Services.Protocols.	<input type="button" value="..."/>																																															
<input type="checkbox"/>	Cookie	<input type="button" value="Check for cookie existence"/>	<input type="button" value="Cookie Name"/>	<input type="text"/>																																															
<input type="button" value="Cancel"/> <input type="button" value="Save"/>																																																			

You can view the other default exclude rules for more examples. See [To View Default Exclude Rules](#).

If you create an exclude rule after transactions based on the exclude criteria have already been created, you need to remove the transactions manually using the Exclude operation from the Actions menu in the business transaction list. See [Business Transactions List Operations](#).

Default Exclude Rules

AppDynamics provides and enables default exclude rules for Servlets and for ASP.NET.

You can view, modify, disable, or remove these rules.

To Create Custom Exclude Rules

1. In the Exclude Rules section of the **Transaction Detection** tab, click the Add icon.
2. From the drop-down menu select the Entry Point type for which you want to create the exclude rule and click Next. The New Exclude Business Transaction Match Rule window appears for the selected entry point type.
3. Enter the match criteria for business transactions to be excluded from discovery. The match criteria on which you can configure the rule vary with the entry point type.
4. Click **Create Exclude Rule**.

To View Default Exclude Rules

1. In the Exclude Rules section of the **Transaction Detection** tab, select an existing exclude rule.

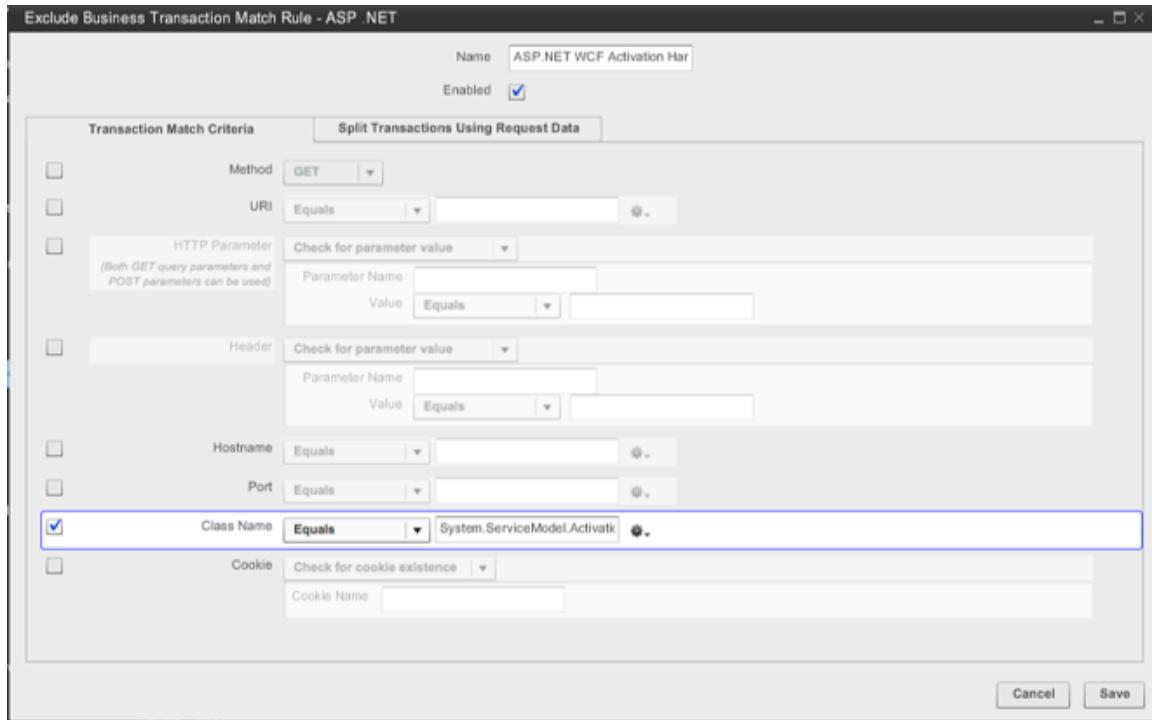
Default Exclude Rules for Java Platforms

Exclude Rules		
If a Transaction matches any of the following rules, it will not be discovered.		
Type	Name	Enabled
Servlet	Apache Axis Servlet	✓
Servlet	Apache Axis2 Servlet	✓
Servlet	Apache Axis2 Admin Servlet	✓
Servlet	Struts Action Servlet	✓
Servlet	Websphere web-services Servlet	✓
Servlet	Websphere web-services axis Servlet	✓
Servlet	JBoss web-services servlet	✓
Servlet	XFire web-services servlet	✓

Default Exclude Rules for .NET Platforms

Exclude Rules		
If a Transaction matches any of the following rules, it will not be discovered.		
Type	Name	Enabled
ASP .NET	ASP.NET WCF Activation Handler	✓
ASP .NET	ASP.NET WebService Script Handler	✓
ASP .NET	ASP.NET WebService Session Hand	✓

2. Click the Edit (pencil) icon.
3. To exit the exclude rule configuration screen without changing anything, click **Cancel**.



Custom Match Rules

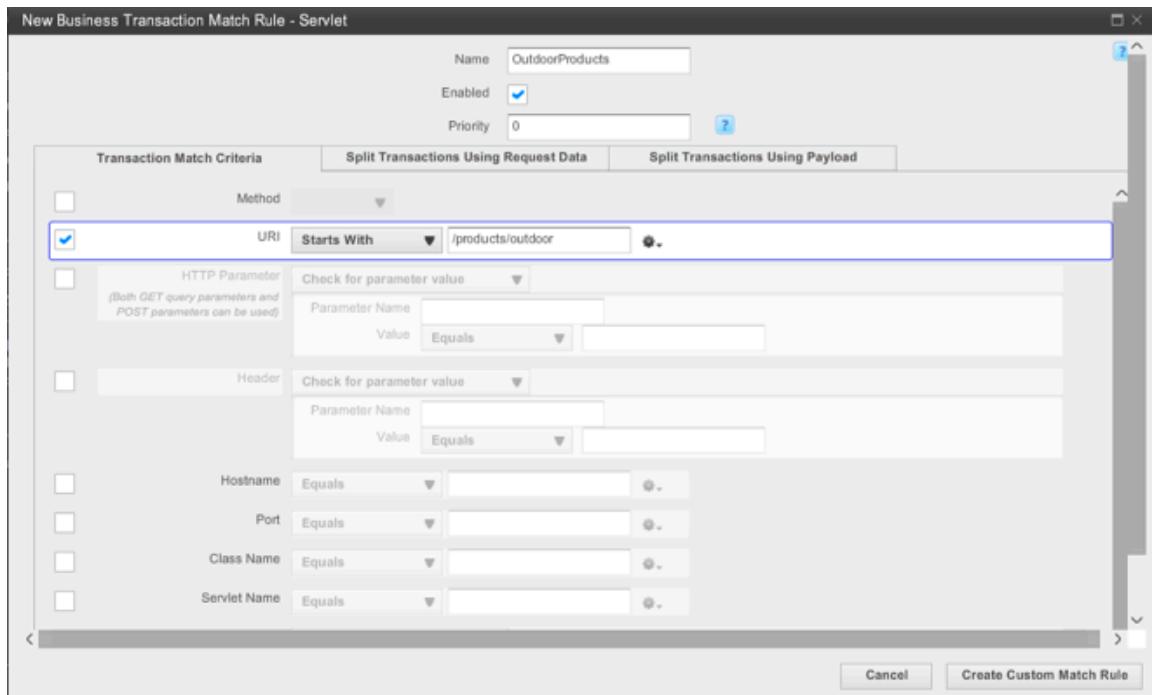
To gain better visibility for selected types of user requests, you can customize how AppDynamics detects business transactions at the tier or application level. You may want to do this if the auto-discovered entry points for certain types do not map precisely to the business perspective or if AppDynamics is detecting too many or too few business transactions.

To Create Custom Match Rules

1. In the Custom Match Rules section of the **Transaction Detection** tab, click the Add icon.
2. From the drop-down menu select the Entry Point type for which you want to create the custom match rule and click Next. The New Business Transaction Match Rule window appears for the selected entry point type.
3. Check the check boxes and enter the match criteria for AppDynamics to use to detect business transactions of this type. The match criteria on which you can configure the rule vary with the entry point type.
4. Click **Create Custom Match Rule**.

Sample Custom Match Rule

The following rule creates a custom match rule for a business transaction for servlet-based requests in which the URI begins with "/products/outdoor". This will also be the name of the business transaction as it appears in the business transaction list, unless you rename it. See [Business Transactions List Operations](#) for information about renaming.



Transaction Splitting

Configuration for some types of custom match and exclude rules allow for transaction splitting. Transaction splitting allows you to fine-tune transaction detection or exclusion based on a parameter or user data.

For example, the following URL represents a Checkout transaction when a user checks out an item from the electronics section:

```
http://nwtrader.com/checkout?category=electronics
```

You can configure the transaction auto-detection mechanism to identify these types of user requests as a Checkout.electronics business transaction by specifying the user request data to use to define the transaction.

With this configuration the business transaction is created dynamically based on the user request. If the request is

```
http://nwtrader.com/checkout?category=clothing
```

the transaction would be Checkout.clothing and so on.

Transaction splitting is optional for the entry point types for which it is offered.

If a splitting part is configured, the transaction is named as:

<custom match rule name> + <name derived from the split configuration>

Sample Custom Match Rule with Split

This example splits a custom match rule similar to the one created in [Sample Custom Match Rule](#) into separate business transactions, one for each color value in the request.

New Business Transaction Match Rule - Servlet

Name	OutdoorProductsByColor
Enabled	<input checked="" type="checkbox"/>
Priority	10
<input type="button" value="Transaction Match Criteria"/> <input type="button" value="Split Transactions Using Request Data"/> <input type="button" value="Split Transactions Using Payload"/>	
<input checked="" type="checkbox"/> Split Transactions using request data ? <ul style="list-style-type: none"> <input type="radio"/> Use the first <input type="text"/> segments in Transaction names <input type="radio"/> Use the last <input type="text"/> segments in Transaction names <input type="radio"/> Use URI segment(s) in Transaction names <ul style="list-style-type: none"> Segment Numbers <input type="text"/> <small>Enter a comma separated list of parameter numbers (e.g. 1,3,4)</small> <input checked="" type="radio"/> Use a parameter value in Transaction names <ul style="list-style-type: none"> Parameter Name <input type="text" value="color"/> <input type="radio"/> Use a header value in Transaction names <ul style="list-style-type: none"> Header Name <input type="text"/> <input type="radio"/> Use a cookie value in Transaction names <ul style="list-style-type: none"> Cookie Name <input type="text"/> <input type="radio"/> Use a session attribute value in Transaction names <ul style="list-style-type: none"> Session Attribute Key <input type="text"/> 	

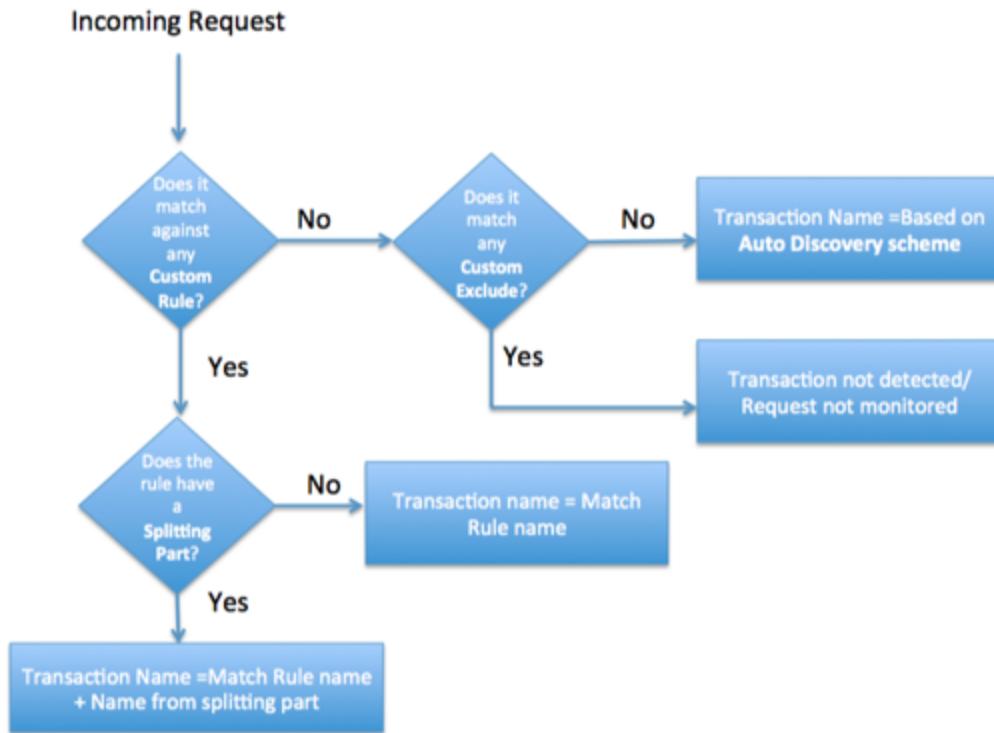
After this configuration is applied, requests that were previously detected as the "/product/outdoor" business transaction are now detected as "/product/outdoor/blue", "/product/outdoor/red" and so on. Now that the "/product/outdoor" business transaction is split, metrics are no longer collected for the old "/product/outdoor" business transaction. Now that the "/product/outdoor" business transaction is split, metrics are no longer collected for the old "/product/outdoor" business transaction.

Sequence and Precedence for Auto-Naming and Custom Match Rules

AppDynamics transaction detection process uses following sequence for automatically naming the transactions:

1. As soon as the entry point is discovered, AppDynamics checks for a custom match rule. If a custom match rule exists, the business transaction is named based on that rule.
2. If no custom match rule exists, AppDynamics checks the custom exclude rules. If the custom exclude rule exists, the transaction is excluded from discovery.
3. If no custom exclude rule exists, AppDynamics applies its auto-detection naming scheme.

The following illustration shows the sequence for the transaction discovery process:



The Priority Parameter

You can use the Priority parameter on a custom rule to specify which rule to apply to the detection of a business transaction if it could be detected by more than one rule.

For example, you can set the priority of a custom rule to greater than 0 (zero) if you want the custom rule to take priority over the default rule.

Transaction Detection Concepts

The following concepts are useful for configuring transaction detection.

- Match Rule Conditions
- Getter Chains in .NET Configurations
- POJOs and POCOs

Learn More

- Hierarchical Configuration Model
- Business Transactions List
- Web Entry Points
- Messaging Entry Points

Match Rule Conditions

- Example Match Criteria for a Servlet-based Request
- Learn more

AppDynamics uses match conditions in rules that specify entities to be monitored or excluded from monitoring. You configure match conditions to fine-tune transaction detection, backend detection, data collectors, EUM injection, health rules, etc.

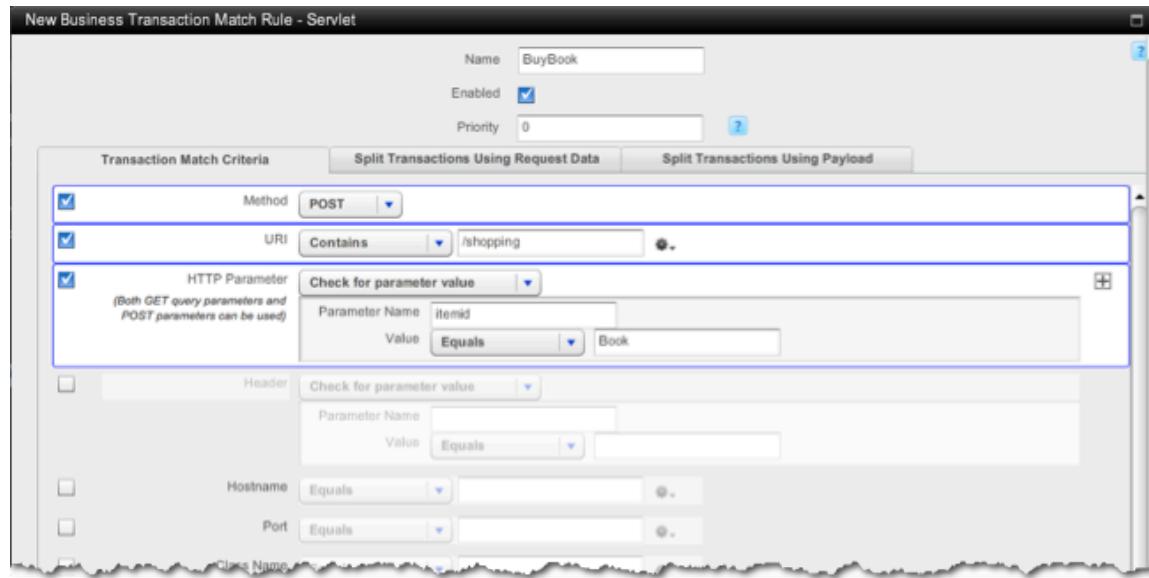
A match condition is a comparison consisting of:

- A match criterion (such as a method name, servlet name, URI, parameter, hostname, etc.)
- A comparison operator typically selected from a drop-down list
- A value

The entities and values being compared and the list of appropriate comparison operators vary depending on the type of configuration.

Example Match Criteria for a Servlet-based Request

The following example is from a custom match rule named BuyBook used for detecting the business transaction. Detection is based on the discovery of the string "/shopping" in the URI and of a POST parameter with the value "Book" for the itemid parameter. When AppDynamics receives a servlet-based request matching these conditions, it monitors the business transaction.



Learn more

- Configure Business Transaction Detection

Regular Expressions In Match Conditions

In AppDynamics you can use regular expressions (sometimes referred to as "RegEx") as an alternative to simple pattern matching. Regular expressions let you combine similar transactions, rather than having to record and identify every variation. You can use regular expressions in custom rules, custom exit points, and other configurations.

AppDynamics uses Java libraries for regular expressions. For more information see:

- Tutorial: <http://download.oracle.com/javase/tutorial/essential/regex/index.html>
- Javadoc: <http://download.oracle.com/javase/1.5.0/docs/api/java/util/regex/Pattern.html>

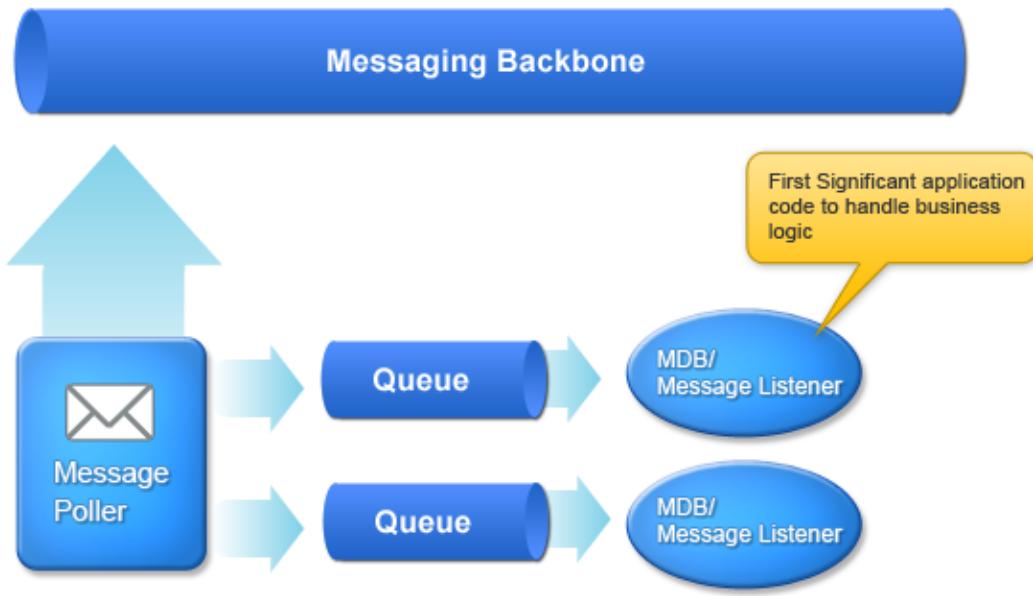
Messaging Entry Points

- Messaging Entry Points
 - Default Naming Conventions
 - To Access the Default Configuration for Messaging Entry Points
 - Custom Match Rules for Messaging Entry Points
 - Grouping Example
 - Message Payload Example
- Learn More

This topic discusses messaging entry points configuration.

Messaging Entry Points

When an application uses asynchronous message listeners or message driven beans, such as JMS or equivalent MQ providers as the primary trigger for business processing on the entry point tier AppDynamics can intercept the message listener invocations and track them as business transactions. This is relevant only for the entry point tier.



You can define messaging entry points for queue listeners to monitor Service Level Agreements (SLAs). An SLA is often reflected in the rate of processing by a queue listener. When you monitor a messaging entry point, you can track the rate of processing by a particular queue listener.

Default Naming Conventions

AppDynamics automatically detects and names the messaging entry points. When a message listener is invoked, the transaction is named after the destination name (the queue name) or the listener class name if the destination name is not available.

To Access the Default Configuration for Messaging Entry Points

1. Access the business transaction configuration page.
See [To Access Business Transaction Detection Configuration](#).
2. Scroll down to message entry point entry for your framework.

Custom Match Rules for Messaging Entry Points

If you want finer control over naming messaging requests, you can use custom match rules either to specify names for your messaging entry points or to group multiple queue invocations into a single business transaction.

See [To Create Custom Match Rules](#) for general information about configuring custom match rules.

Grouping Example

The following custom match rule groups all transactions to queues starting with the word "Event"

New Business Transaction Match Rule - JMS

Name	EventQueues
Enabled	<input checked="" type="checkbox"/>
Business Transaction Match Criteria	
<input checked="" type="checkbox"/> Message Destination	Queue Starts With Event
<input type="checkbox"/> Message Property	Check for property existence Property Type: BOOLEAN Property Name:
<input type="checkbox"/> Message Content	Equals Applies to text messages only
<input type="button" value="Cancel"/> <input type="button" value="Create Custom Match Rule"/>	

Message Payload Example

The following custom rule uses message properties (headers) to define the custom match rule. You can also use the message content. You can also use message properties or message content to define custom exclude rules to exclude certain messaging entry points from detection.

New Business Transaction Match Rule - JMS

Name	EventQueues
Enabled	<input checked="" type="checkbox"/>
Business Transaction Match Criteria	
<input checked="" type="checkbox"/> Message Destination	Queue Starts With Event
<input checked="" type="checkbox"/> Message Property	Check for property value Property Type: BOOLEAN Property Name: customerType Property Value: Priority
<input type="checkbox"/> Message Content	Equals Applies to text messages only
<input type="button" value="Cancel"/> <input type="button" value="Create Custom Match Rule"/>	

Learn More

- Configure Business Transaction Detection

Web Entry Points

- Auto-Detected Entry Points
 - Enable or Disable Transaction Monitoring
 - Enable or Disable Automatic Transaction Detection

- Business Transaction Names
- Learn More

Entry points define where a business transaction begins.

Auto-Detected Entry Points

AppDynamics classifies entry points by the type of the app server platform and maintains a default detection scheme for each type. The following screenshots show the configurable Java and .NET entry point types and summaries of their default automatic detection and naming rules.

Auto-Detected Java Entry Points

Java - Transaction Detection			.NET - Transaction Detection																																				
Copy	Configure all Tiers to use this Configuration																																						
▼ Entry Points <table border="1"> <thead> <tr> <th>Type</th> <th>Transaction Monitoring</th> <th colspan="2">Automatic Transaction Detection</th> </tr> </thead> <tbody> <tr> <td> Servlet</td> <td><input checked="" type="checkbox"/> Enabled</td> <td><input checked="" type="checkbox"/> Discover Transactions automatically for all Servlet requests Configure Naming</td> <td><input type="checkbox"/> Enable Servlet Filter Detection ?</td> </tr> <tr> <td> Struts Action</td> <td><input checked="" type="checkbox"/> Enabled</td> <td><input checked="" type="checkbox"/> Discover Transactions automatically for all Struts Action invocations Transactions will be named: ActionName.MethodName</td> <td></td> </tr> <tr> <td> Web Service</td> <td><input checked="" type="checkbox"/> Enabled</td> <td><input checked="" type="checkbox"/> Discover Transactions automatically for all Web Service requests Transactions will be named: ServiceName.OperationName</td> <td></td> </tr> <tr> <td> POJO</td> <td><input checked="" type="checkbox"/> Enabled</td> <td colspan="2">Any Java method can be the entry point for a Business Transaction. The class to which the method belongs to can be picked using different parameters like its name, its super class name, the interfaces it implements, or the annotations it has.</td> </tr> <tr> <td> Spring Bean</td> <td><input checked="" type="checkbox"/> Enabled</td> <td><input type="checkbox"/> Discover Transactions automatically for all Spring Bean invocations Transactions will be named: BeanName.MethodName</td> <td></td> </tr> <tr> <td> EJB</td> <td><input checked="" type="checkbox"/> Enabled</td> <td><input type="checkbox"/> Discover Transactions automatically for all EJB invocations Transactions will be named: EJBNName.MethodName</td> <td></td> </tr> <tr> <td> JMS</td> <td><input checked="" type="checkbox"/> Enabled</td> <td><input checked="" type="checkbox"/> Discover Transactions automatically for all incoming JMS Messages Transactions will be named: Destination Name or Listener Class Name (If Destination Name not available)</td> <td></td> </tr> <tr> <td> Binary Remoting</td> <td><input checked="" type="checkbox"/> Enabled</td> <td><input checked="" type="checkbox"/> Discover Transactions automatically for all Binary Remoting requests (Thrift) Transactions will be named: RemoteInterfaceClassName.methodName</td> <td></td> </tr> </tbody> </table>				Type	Transaction Monitoring	Automatic Transaction Detection		Servlet	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Servlet requests Configure Naming	<input type="checkbox"/> Enable Servlet Filter Detection ?	Struts Action	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Struts Action invocations Transactions will be named: ActionName.MethodName		Web Service	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Web Service requests Transactions will be named: ServiceName.OperationName		POJO	<input checked="" type="checkbox"/> Enabled	Any Java method can be the entry point for a Business Transaction. The class to which the method belongs to can be picked using different parameters like its name, its super class name, the interfaces it implements, or the annotations it has.		Spring Bean	<input checked="" type="checkbox"/> Enabled	<input type="checkbox"/> Discover Transactions automatically for all Spring Bean invocations Transactions will be named: BeanName.MethodName		EJB	<input checked="" type="checkbox"/> Enabled	<input type="checkbox"/> Discover Transactions automatically for all EJB invocations Transactions will be named: EJBNName.MethodName		JMS	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all incoming JMS Messages Transactions will be named: Destination Name or Listener Class Name (If Destination Name not available)		Binary Remoting	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Binary Remoting requests (Thrift) Transactions will be named: RemoteInterfaceClassName.methodName	
Type	Transaction Monitoring	Automatic Transaction Detection																																					
Servlet	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Servlet requests Configure Naming	<input type="checkbox"/> Enable Servlet Filter Detection ?																																				
Struts Action	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Struts Action invocations Transactions will be named: ActionName.MethodName																																					
Web Service	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Web Service requests Transactions will be named: ServiceName.OperationName																																					
POJO	<input checked="" type="checkbox"/> Enabled	Any Java method can be the entry point for a Business Transaction. The class to which the method belongs to can be picked using different parameters like its name, its super class name, the interfaces it implements, or the annotations it has.																																					
Spring Bean	<input checked="" type="checkbox"/> Enabled	<input type="checkbox"/> Discover Transactions automatically for all Spring Bean invocations Transactions will be named: BeanName.MethodName																																					
EJB	<input checked="" type="checkbox"/> Enabled	<input type="checkbox"/> Discover Transactions automatically for all EJB invocations Transactions will be named: EJBNName.MethodName																																					
JMS	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all incoming JMS Messages Transactions will be named: Destination Name or Listener Class Name (If Destination Name not available)																																					
Binary Remoting	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Binary Remoting requests (Thrift) Transactions will be named: RemoteInterfaceClassName.methodName																																					

Auto-Detected .NET Entry Points

[Java - Transaction Detection](#) [.NET - Transaction Detection](#)

[Copy](#) [Configure all Tiers to use this Configuration](#)

▼ Entry Points

Type	Transaction Monitoring	Automatic Transaction Detection
ASP .NET	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for ASP .NET requests Configure Naming
Web Service	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Web Service requests Transactions will be named: ServiceName.OperationName
WCF	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for WCF requests Transactions will be named: ServiceName.OperationName
.NET Class / Method	<input checked="" type="checkbox"/> Enabled	Any .NET method can be the entry point for a Business Transaction. The class to which the method belongs to can be picked using different parameters like its name, its super class name, the interfaces it implements, or the annotations it has.
Message Queues	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for message listeners (IBM XMS, Tibco EMS, Tibco RV, Apache MQ) Transactions will be named: Destination Name or Listener Class Name (If Destination Name not available)

Enable or Disable Transaction Monitoring

For each entry point type, you can enable and disable transaction monitoring. When monitoring is disabled, the agent stops counting, measuring, recording, etc. all activity on servers of that entry type throughout the application (if detection is being configured at the application level) or for specific tiers (if transaction is being configured at the tier level). Transactions discovered by automatic detection and those discovered by custom rules are equally affected.

For each type you can also enable and disable automatic transaction detection. After you disable detection for an entry point type, no new transactions based on auto detection for that type are discovered. Custom rules are still active and all transactions that are based on custom rules for the entry point type report data. If a transaction was detected earlier when automatic detection was enabled and then you disable it, the agent stops reporting performance metrics for that transaction. However, the transaction is not deleted or excluded.

Enable or Disable Automatic Transaction Detection

For each entry point type, you can also enable and disable automatic transaction detection.

If automatic detection was enabled and then you disable it, the agent stops reporting metrics for the transactions previously detected by the now disabled entry point and detects only transactions based on custom rules. Calls to methods and operations of the disabled entry point type are no longer auto-detected. Custom rules for the entry point type remain active and the agent reports metrics for them.

Business Transaction Names

For Java Servlet and ASP .NET transactions you can configure whether to use the entire URI or just part of the URI as the transaction name. You can also name transactions dynamically using a specific part of the request, such as a specific parameter value, cookie value, session attribute value, etc. as the transaction name.

Servlet Naming Configuration

Servlet Transaction Naming Configuration

What part of the URI should be used in the Transaction Name?

Use the full URI
 Use a part of the URI (for example, if you have dynamic URIs)
 Use the first ▾ 2 segments of the URI in Transaction Names [What does this do?](#)

Name Transactions dynamically using part of the request

Use URI segment(s) in Transaction names
 Segment Numbers Enter a comma separated list of parameter numbers (e.g. 1,3,4)

Use a parameter value in Transaction names
 Parameter Name itemid

Use a header value in Transaction names
 Header Name

Use a cookie value in Transaction names
 Cookie Name

Use a session attribute value in Transaction names
 Session Attribute Key

Use the request method (GET/POST/PUT) in Transaction names

Use the request host Transaction in names

Use the request originating address in Transaction names

Apply a custom expression on HttpServletRequest and use the result in Transaction Names [Explain This](#)

[Cancel](#) [Save](#)

ASP .NET Naming Configuration

ASP .Net Transaction Naming Configuration

What part of the URI should be used in the Transaction Name?

Use the full URI
 Use a part of the URI (for example, if you have dynamic URIs)
 Use the first ▾ 4 segments of the URI in Transaction Names [What does this do?](#)

Name Transactions dynamically using part of the request

Use URI segment(s) in Transaction names
 Segment Numbers Enter a comma separated list of parameter numbers (e.g. 1,3,4)

Use a parameter value in Transaction names
 Parameter Name

Use a header value in Transaction names
 Header Name

Use a cookie value in Transaction names
 Cookie Name

Use a session attribute value in Transaction names
 Session Attribute Key

Use the request method (GET/POST/PUT) in Transaction names

Use the request host Transaction in names

Use the request originating address in Transaction names

[Cancel](#) [Save](#)

Learn More

- Hierarchical Configuration Model
- Business Transactions List
- Configure Business Transaction Detection
- Messaging Entry Points

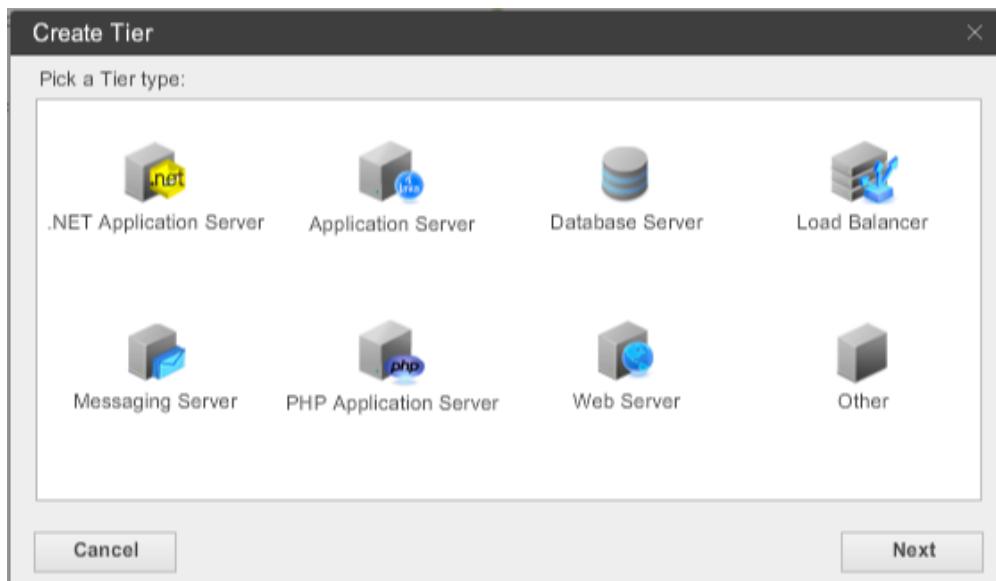
Creating New Tiers

- Creating a New Tier in an Application
 - To Create a Tier
- Viewing Detailed Information about a Tier or Node
 - To View Detailed Information about a Tier or Node
- Learn More

Creating a New Tier in an Application

To Create a Tier

1. Click **Create Tier**.
2. In the Create Tier window click the tier type.



3. Click **Next**.
4. Enter the name and description of the tier.
5. Click **Finish**.

The tier appears in the left navigation panel under the assigned type. If the menu item for the type does not exist, it is created. For example, if you create a tier of type Other, the tier appears under Other Servers in the left navigation pane.

See [Mapping Application Services to the AppDynamics Model](#) for information about tiers.

Viewing Detailed Information about a Tier or Node

From the App Server List you can access the Tier and Node Dashboards for specific tiers and nodes.

To View Detailed Information about a Tier or Node

1. In the App Server List click the Tree View icon. Alternatively use the left navigation panel and expand the listings.
2. In the listing, select the tier or node that you want to view and click **View Dashboard**.
The dashboard appears. From there you can select the various tabs for details about performance of the tier or node. See [Tier Dashboard](#) or [Node Dashboard](#).

Learn More

- Logical Model

- Dashboards

Thresholds

- Applying Thresholds
- Static and Dynamic Thresholds
 - Percentage Deviation
 - Standard Deviation
- Learn More

This topic describes how AppDynamics thresholds help you to maintain the service level agreements (SLAs) and ensure optimum performance levels for your system.

Thresholds provide a flexible way to associate the right business context with a slow request to isolate the root cause.

Applying Thresholds

You can apply request thresholds at following levels:

- Thresholds for a business application
- Thresholds for individual business transactions
- Thresholds for slow and stalled transactions and background tasks
- Thresholds for diagnostic sessions

Static and Dynamic Thresholds

A static threshold is a static value, which when exceeded, violates the threshold.

A dynamic threshold is based on a changing value. You can specify dynamic thresholds using either Percentages or Standard Deviation measures.

Percentage Deviation

Percentage deviation defines a threshold based on the moving average of the requests over a certain interval. Default time interval is 2 hours. If the average response time for the last two hours is X milliseconds, and if the request takes the percentage deviation over X ms, the transaction violates threshold.

For example, a transaction occurs at 10:00:01 AM. The average of the requests between 8:00:00 and 10:00:00 is 100 ms and the slow threshold is defined as 20 % over threshold. If the transaction takes 120 ms, it will be considered a slow transaction.

Standard Deviation

Standard deviation defines a threshold based on the moving average over a certain interval. Default time interval is 2 hours. This means if the average response time for the last two hours is X milliseconds, and if the request takes the percentage deviation over X ms, it violates threshold.

For example, a transaction begins at 10:00:01 AM and the slow threshold is defined as 3 times the standard deviation (variation from the mean). If the transaction takes more than 3 standard deviations, it will be considered a slow transaction.

Learn More

- Configure Thresholds

Configure Thresholds

- Configuring Business Transaction Thresholds
 - To Configure Slow Transaction Thresholds
- Configuring Background Task Thresholds
- Learn More

A threshold is a boundary of acceptable or normal business transaction or background task performance. See [Thresholds](#).

AppDynamics provides default thresholds and you can configure them for your own environment.

To access threshold configuration:

1. Click **Configure -> Slow Transaction Thresholds** in the left navigation pane.
2. Click the **User Transaction Thresholds** or **Background Transactions Thresholds** tab depending on the type of entity for which you want to configure thresholds.

Thresholds for End User Monitoring (EUM) are configured separately. For information about EUM thresholds see [Configuring EUM Thresholds](#).

Configuring Business Transaction Thresholds

You can configure the thresholds for slow and very slow requests and for stalls. When a transaction or task exceeds a threshold, AppDynamics starts capturing snapshots of it. See [Transaction Snapshots](#). Because the snapshots are not normally captured while performance is within normal range, the snapshots often do not contain the full call graph for the transaction.

You can configure dynamic thresholds using either percentage of standard deviation measurements.

A percentage deviation threshold is based on the moving average of the transaction instances over a certain interval. In the following example, if the average response time is x milliseconds, a slow transaction would be one which is 50% over x milliseconds over the last two hours.

User Transaction Thresholds

This section lets you configure Slow Transaction Thresholds, and when to trigger Diagnostic Sessions.

Slow Transactions Thresholds

Every Transaction that is processed by the Application will be categorized as normal, slow, very slow, or stalled based on these thresholds.

Slow Transaction Threshold

- More than % slower than the average of the last hours
- Greater than Milliseconds
- Greater than Standard Deviations for the last hours

Very Slow Transaction Threshold

- More than % slower than the average of the last hours
- Greater than Milliseconds
- Greater than Standard Deviations for the last hours

Stall Threshold

- Disable Stall detection
- Stall occurs when a request takes more than seconds.
- Stall occurs when a request's response time is deviations above the average for the last 2 hours.

Apply to all Existing Business Transactions

A standard deviation threshold is also based on a moving average of the transaction instances over a certain interval, but it uses a multiple of the standard deviation rather than a percentage. The following configuration defines a transaction as very slow if it exceeds 4 times the standard deviation over the last three hours.

Very Slow Transaction Threshold

- More than % slower than the average of the last hours
- Greater than Milliseconds
- Greater than Standard Deviations for the last hours

You can also configure a static threshold using static values.

To Configure Slow Transaction Thresholds

1. In the left navigation pane, click **Configure -> Slow Transaction Thresholds**.
2. Click the **User Transaction Thresholds** tab to configure transaction thresholds.
3. In the thresholds tree list, select the scope of the threshold, either:
 - Default Thresholds for all business transactions
or
 - Individual Business Transaction
4. In the thresholds section, configure the thresholds by selecting the radio button for the type of threshold (dynamic based on percentage, dynamic based on standard deviation or static) and entering the threshold values in the text fields. You can also disable stall detection if desired.
5. If you want these thresholds to apply to all Business Transactions already discovered in your application, click **Apply to all Existing Business Transactions**.
6. Click **Save Default Slow Thresholds** (for default thresholds) or **Save Slow Transaction Thresholds** (for individual thresholds).

AppDynamics displays the resulting performance data in the Transaction Scorecard section of the application and business transaction dashboards.

Configuring Background Task Thresholds

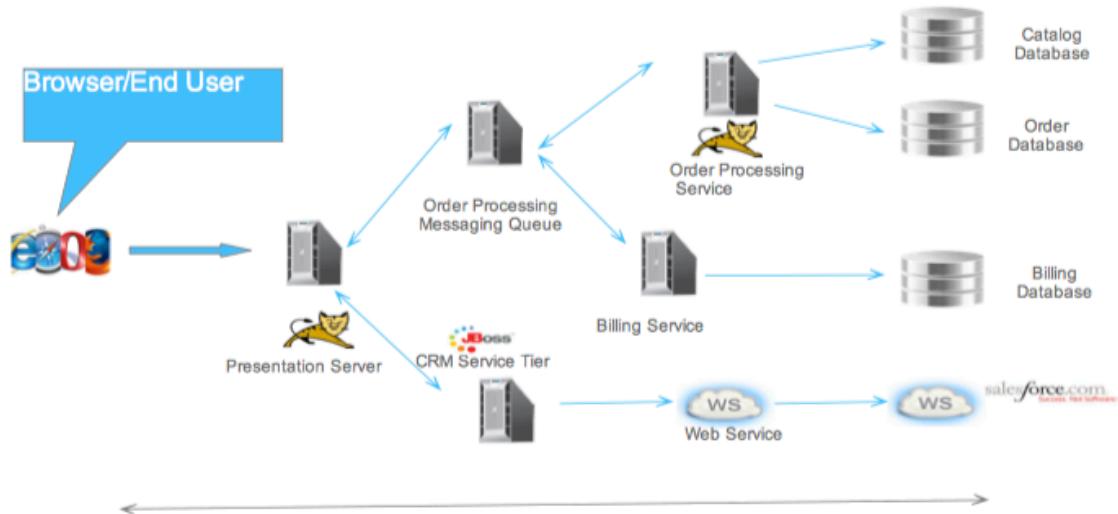
To configure thresholds for background tasks see [Configure Background Tasks](#).

Learn More

- [Scorecards](#)
- [Thresholds](#)

End User Experience

EUM User Experience Monitoring (EUM) provides information about your users' experience starting from their Web browsers. This is different from other types of AppDynamics monitoring, which typically begin at the application server.



EUM helps you determine the extent to which poor user experience may be caused by problems in the browser or in the network by showing how much of the total end-user time is spent getting and rendering a Web page. EUM gives you visibility into the users' browsers on a global basis, showing you:

- where your heaviest loads originate
- where your slowest end-user response times occur
- how performance varies by location
- how performance varies by Web browser

- how performance varies by device
- your slowest Web pages and what is causing the slowdown
- your slowest Ajax requests and what is causing the slowdown

EUM produces its own data set that is separate from data reported by AppDynamics app agents. This EUM data is visible in various EUM dashboards as well as in the Metric Browser. In addition, EUM can link with server-side business transaction information collected by the Java and .NET app agents to give you a complete view of your end users' experience from the browser all the way to the backend.

To learn more about setting up EUM, see:

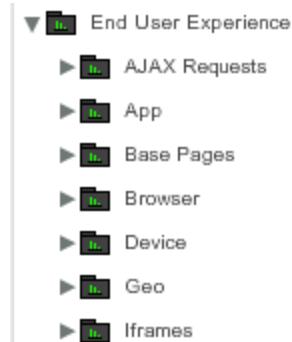
- Additional topics
-  [End User Monitoring](#)

Since EUM produces additional metrics, if you have an on-premise controller you may need to evaluate your current configuration's ability to handle the additional load. See [Additional Sizing Considerations](#).

EUM Metrics

- [EUM Metrics Defined](#)
- [EUM Metrics Availability](#)
 - Apparent Anomalies Resulting from Variations in Browser Reporting
- [Learn More](#)

AppDynamics reports key EUM metrics on the Geo and Page dashboards, on the page list, and in the Metric Browser. In the Metric Browser, EUM metrics are aggregated by Ajax request, application, base page, browser, device, and geographic location and IFrame.



You can build custom health rules that access EUM metrics in the embedded metric browser in the health rule builder.

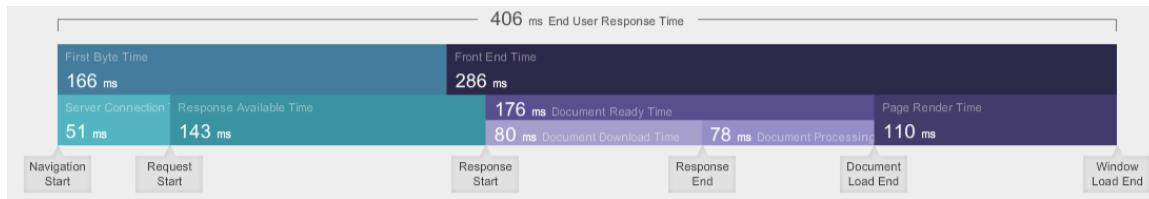
EUM Metrics Defined

Time metrics are the average times, in milliseconds, over the time range selected in the AppDynamics UI or REST API call.

- **End User Requests per minute:** average number of end user requests per minute
- **Total Number of End User Requests:** derived from adding all the end user requests per minute for the selected time range
- **Pageviews with JavaScript Errors per minute**
- **End User Response Time** is the average interval between the time that a user initiates a request and the completion of the page load of the response in the user's browser. In the context of an Ajax request, it ends when the response has been completely processed.
A user initiates a request by actions such as clicking a hyperlink, submitting a form, typing a URL in a browser, etc.
- **First Byte Time** is the interval between the time that a user initiates a request and the time that the browser receives the first response byte. In the context of an Ajax request, First Byte Time is the interval between the Ajax request dispatch and the time that the browser receives the first response byte.
- **Server Connection Time** is the interval between the time that a user initiates a request and the start of fetching the response document from the server or application task. Includes the time spent on redirects, domain lookups, TCP connects and SSL

handshakes.

- **Response Available Time** is the interval between the beginning of the processing of the request on the browser to the time that the browser receives the response. Includes time in the network from the user's browser to the server.
- **Front End Time** is the interval between the arrival of the first byte of text response and the completion of the response page rendering by the browser. Includes Document Download Time, Document Ready Time, Document Processing Time and Page Render Time.
- **Document Download Time** is the time for the browser to download the complete HTML document content. In the context of an Ajax request, Document Download Time is the time for the browser to download the complete Ajax response.
- **Document Processing Time** is the time for the browser to build the Document Object Model (DOM) and make it available for JavaScripts to apply rendering logic. In the context of an Ajax request, Document Processing Time is the time for the browser to process the Ajax response; this typically includes the time to apply the response data to the DOM.
- **Document Ready Time** is the time to make the complete HTML document (DOM) available for JavaScript to apply rendering logic. Includes the Document Download Time and the Document Processing Time.
- **Page Render Time** is the time for the browser to complete the download of remaining resources, including images, and finish rendering the page.



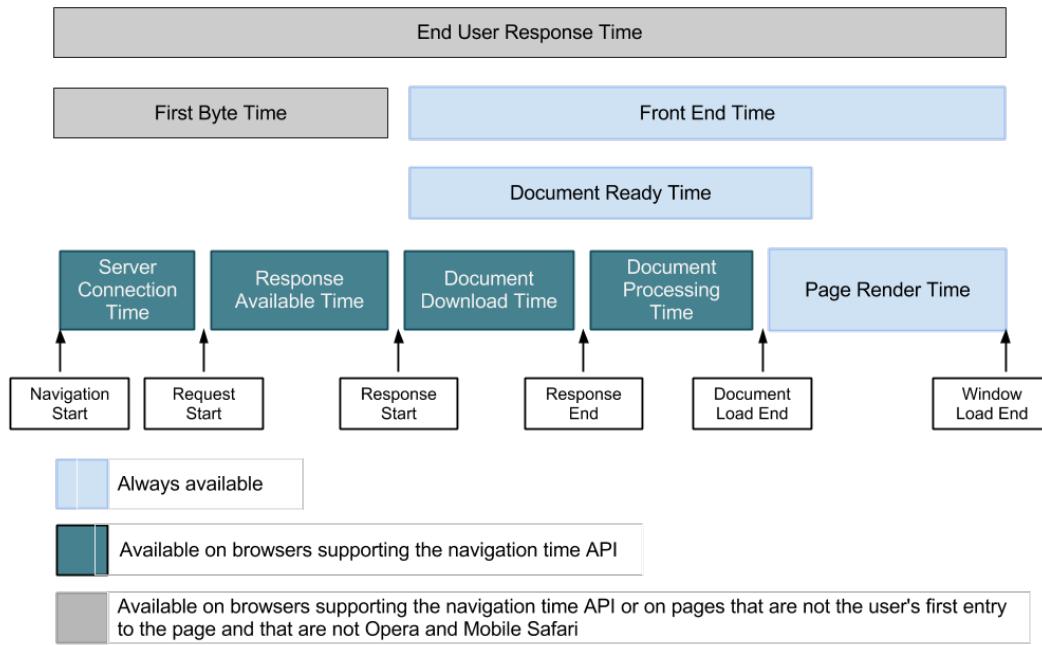
EUM Metrics Availability

The availability of some EUM metrics depends on the capability of the end-user's Web browser. This can lead to a value of Unknown for unavailable metrics in some browser snapshots as well as to aggregated data in summary graphs that does not seem to add up. See [Apparent Anomalies Resulting from Variations in Browser Reporting](#) below.

Some metrics are always available.

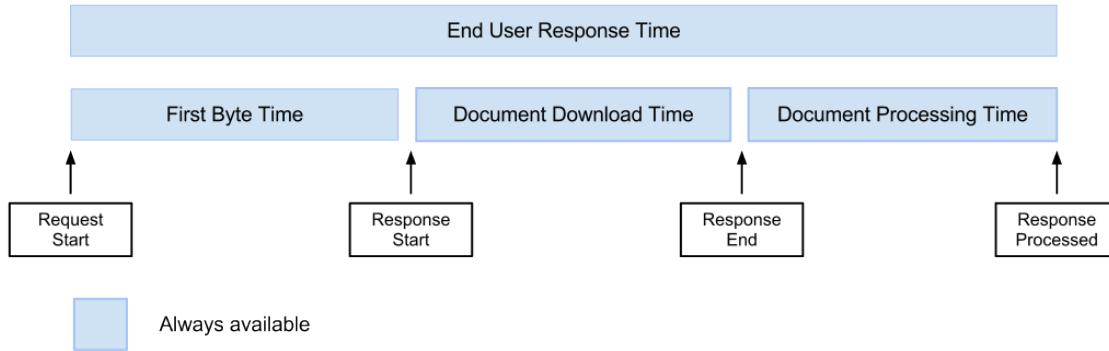
Some metrics for pages and IFrames are available only if the browser uses the Navigation Timing API. Some metrics are available for all pages, except the end-user's initial entry to the application, even if the browser does not use the Navigation API.

Page and IFrame Core Metrics



For Ajax requests, the four core metrics are always available.

Ajax Core Metrics



Apparent Anomalies Resulting from Variations in Browser Reporting

When all EUM metrics are reported, component metrics add up to the summary metrics. For example, for a base page or IFrame:

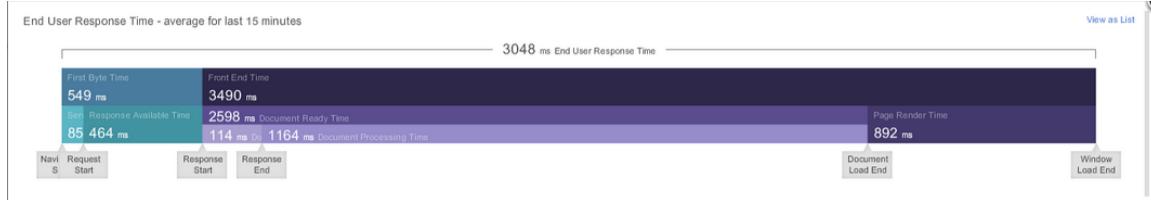
$$\text{First Byte Time} + \text{Front End Time} = \text{End User Response Time}$$

In this case, End User Response Time captures all of the time spent processing and receiving the response.

However, as illustrated in the Page and IFrame Core Metrics diagram above, not all web browsers report First Byte Time all the time. For example, when the reported metrics from all of your end users' browsers are averaged, and some of those browser are not reporting First Byte Time, the segments do not add up to the total. In this case you would see:

$$\text{AVG(First Byte Time)} + \text{AVG(Front End Time)} \neq \text{AVG(End User Response Time)}$$

The following summary graph illustrates this behavior:



From this data, one might expect total End User Response Time to be

$$549\text{ms} + 3490\text{ms} = 4009\text{ms}$$

but in fact AppDynamics reports it as 3048 ms. This occurs when some of the traffic being averaged into the summary metrics comes from browsers that do not report First Byte Time, Document Download Time or Document Processing Time.

In the browser snapshots, the metrics add up because they show a single snapshot, not a summary based on an aggregate. If the browser did not report a metric, the unreported metric as well as the summary in which it is contained is displayed in the browser snapshot as Unknown:



Learn More

- Metric Browser
- Use the AppDynamics REST API
- Page List
- Page Dashboard
- Geo Dashboard
- Browser Snapshots
- WC3 Navigation Timing API Overview

Monitor End User Experience (EUM)

- Enabling EUM
- EUM Views
- Learn More

To access EUM, click **End User Monitoring** in the left navigation panel.

You see the Geo Dashboard, which displays key EUM performance metrics by geographic location. See [Geo Dashboard](#) for details about this dashboard.

Enabling EUM

For EUM to be accessible, EUM must be enabled at the application level. By default it is disabled and does not appear in the AppDynamics menu.

You need to provide your EUM License Key to enable EUM. See [EUM License](#) for information about getting the license key.

For information on enabling or disabling EUM and entering the license key, see [Configure End User Experience](#).

EUM Views

AppDynamics offers three views into EUM data.

- Geographic view reports EUM data by geographic location. Monitor the geographic view to learn which regions have the highest loads, the longest response times, the most errors, etc. See [Geo Dashboard](#).
- Pages and Ajax Requests view reports EUM data by web page. There are three kinds of pages. A base page represents what an end user sees in a single browser window. An IFrame is a page embedded in another page. An Ajax request is a request for data sent from a page using Ajax. Each page has a unique name and dashboard. Monitor the page view to learn which pages and child pages (IFrame and Ajax requests) are experiencing performance problems. See [Page List](#) and [Page Dashboard](#). From the page dashboard, you can also access snapshots captured for individual pages to drill down into root cause of problems. See [Browser Snapshots](#). For general information about Ajax see http://en.wikipedia.org/wiki/Ajax_%28programming%29.
- Browser view and Devices view report EUM data by browser, browser version, and device. Monitor browser and device view to learn which browsers and devices are experiencing performance problems. See [Monitor End User Experience by Browser](#) and [Monitor End User Experience by Device](#).

Performance problems that appear in one view may be masked in other views, so to get a complete picture of end-user experience it is a good idea to monitor using all three views. For example, one page may be experiencing extremely poor performance, but if that page is not accessed frequently, overall performance may not be anomalous and the summary key performance indicators (KPIs) displayed in geographic view may show that locations are performing normally.

 Note: EUM is not available with AppDynamics for BMC Software Solutions. For information about integrating AppDynamics with BMC's End User Experience Management solution, see [Integrate AppDynamics with BMC End User Experience Management](#).

Learn More

- [EUM Metrics](#)
- [Geo Dashboard](#)
- [Configure End User Experience](#)
- [Pages](#)
- [Page List](#)
- [Page Dashboard](#)
- [Browser Snapshots](#)
- [Monitor End User Experience by Browser](#)
- [Monitor End User Experience by Device](#)
- [EUM License](#)

Monitor End User Experience by Browser

- [Accessing EUM Data by Browser](#)
 - To access performance by browser:
- [Overall Browser Distribution](#)
- [Performance by Browser](#)
- [Distribution by Country](#)
- [Learn More](#)

You can monitor end user experience by browser and by browser version.

The browser dashboard helps you discover:

- the slowest browsers in terms of total end-user response time
- the slowest browsers to render the response page
- the browsers that most of your end users use
- the browsers that most of your end users use in a particular country or region

Accessing EUM Data by Browser

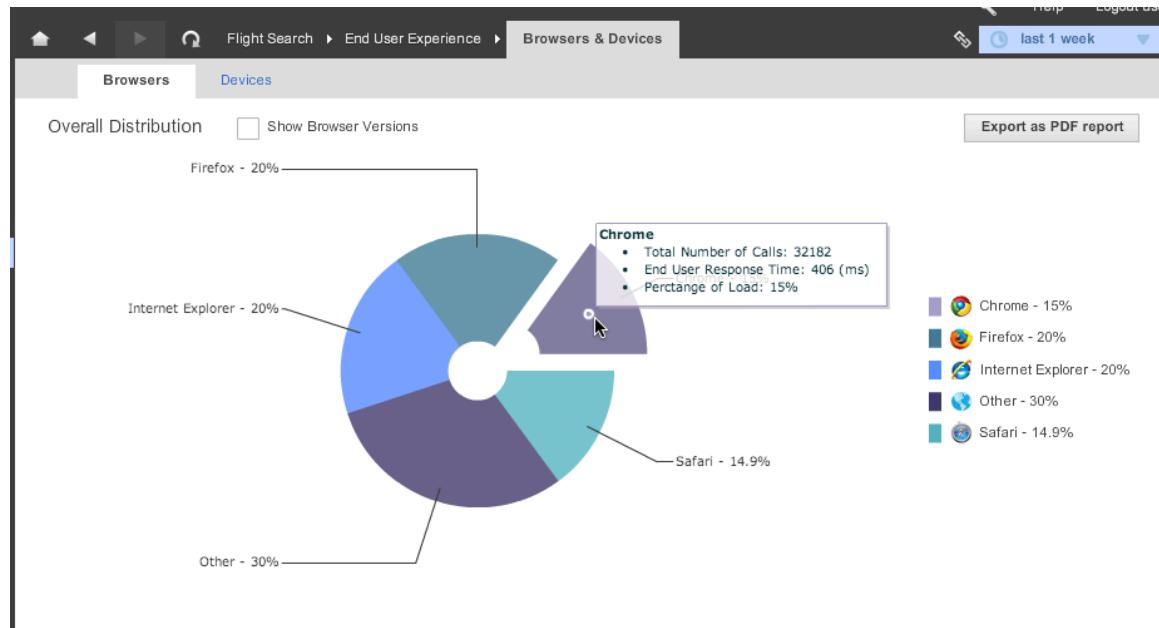
To access performance by browser:

1. In the left navigation pane click: **End User Monitoring -> Browsers & Devices**.
2. Click the Browsers tab if it is not already selected.

Overall Browser Distribution

The Overall Distribution chart shows the percentages of your end users using different Web browsers.

To see total number of calls, average end user response time, and percentage of the total load for a particular browser, hover over the browser section in the chart.



To see the distribution by browser version, check Show Browser Versions.

Performance by Browser

The Performance by Browser list below the chart displays a row for each browser or each browser version if Show Browser Versions is checked.

Performance by Browser		<input type="checkbox"/> Show Browser Versions															
	Name	Requests per	Total Number of End	End User Response Time (ms)	Front End Time	Page Render Time	Document Ready	Document Download	Document Processing	First Byte Time	Response Available	Server Connection	Page Views with	AJAX Requests			
	Chrome	159	9681	408	287	111	177	80	79	166	144	52	1600	411			
	Firefox	208	12875	405	286	110	176	80	78	164	141	51	2174	532			
	Internet Explorer	199	12763	407	286	109	177	81	78	166	142	52	2191	552			
	Other	299	19116	406	286	110	176	80	78	166	143	51	3240	822			
	Safari	154	9559	404	284	108	176	80	78	166	143	51	1535	441			

The columns contain EUM metrics by browser. See [EUM Metrics](#). Click a column to sort the browsers based on the column's metric. For example, if you want to sort the slowest browsers in terms of Download Time with the slowest browsers at the top of the list, click the Download Time column. You can toggle the column header to switch between ascending and descending order.

To filter the list to see only rows for a particular browser, enter the browser in the filter field.

It is a good idea to monitor this list even if AppDynamics reports that overall EUM performance is normal. Information about poor performance on one browser or browser version can be masked by overall metrics if that browser is not used by a significant percentage of your users.

Distribution by Country

The Distribution by Country list, below the Distribution by Browser list, breaks out the browser distribution of your end users by country. You can sort by any browser by clicking the browser's column header.

Distribution by Country



Name	Other	Chrome	Internet Explorer	Safari	Firefox
France	31.0 %	12.8 %	18.6 %	15.5 %	22.2 %
Malaysia	30.2 %	15.7 %	19.6 %	14.5 %	20.0 %
New Zealand	31.3 %	16.0 %	18.8 %	15.0 %	19.0 %
Austria	29.0 %	15.5 %	19.6 %	15.0 %	20.9 %
China	30.3 %	15.3 %	19.7 %	14.7 %	20.0 %
Netherlands	29.0 %	15.2 %	20.8 %	15.1 %	20.0 %
Taiwan	28.7 %	14.4 %	20.3 %	17.2 %	19.4 %
Denmark	28.0 %	16.2 %	19.1 %	15.6 %	21.1 %
United Kingdom	31.1 %	14.7 %	19.3 %	15.0 %	20.0 %
Hong Kong	29.0 %	15.3 %	18.9 %	15.6 %	21.2 %
Japan	29.7 %	15.1 %	20.1 %	15.0 %	20.2 %
United States	30.0 %	15.2 %	19.9 %	14.9 %	20.0 %
Poland	28.3 %	13.2 %	21.5 %	15.1 %	21.8 %
Canada	30.3 %	14.8 %	20.8 %	13.5 %	20.6 %
Australia	29.0 %	15.0 %	21.9 %	15.2 %	18.9 %

This list is particularly useful when viewed in conjunction with the worst performing regions panel in the geo dashboard. If a particular country is experiencing poor performance, it is possible that a significant percentage of your users in that country use a poorly-performing browser. This list can help you to determine whether the browser is a contributing factor.

Learn More

- [EUM Metrics](#)
- [Time Ranges](#)
- [Geo Dashboard](#)
- [Monitor End User Experience \(EUM\)](#)
- [Monitor End User Experience by Device](#)

Monitor End User Experience by Device

- [Accessing EUM Data by Device](#)
 - To access EUM performance by device:
- [Overall Device Distribution](#)
- [Performance by Device](#)
- [Distribution by Country](#)
- [Learn More](#)

You can monitor end user experience by the devices that your end users use to access your application.

The device dashboard helps you discover:

- the slowest devices in terms of total end-user response time
- the slowest devices to connect to the server
- the devices that most of your end users use
- the devices that most of your end users use in a particular country or region

Accessing EUM Data by Device

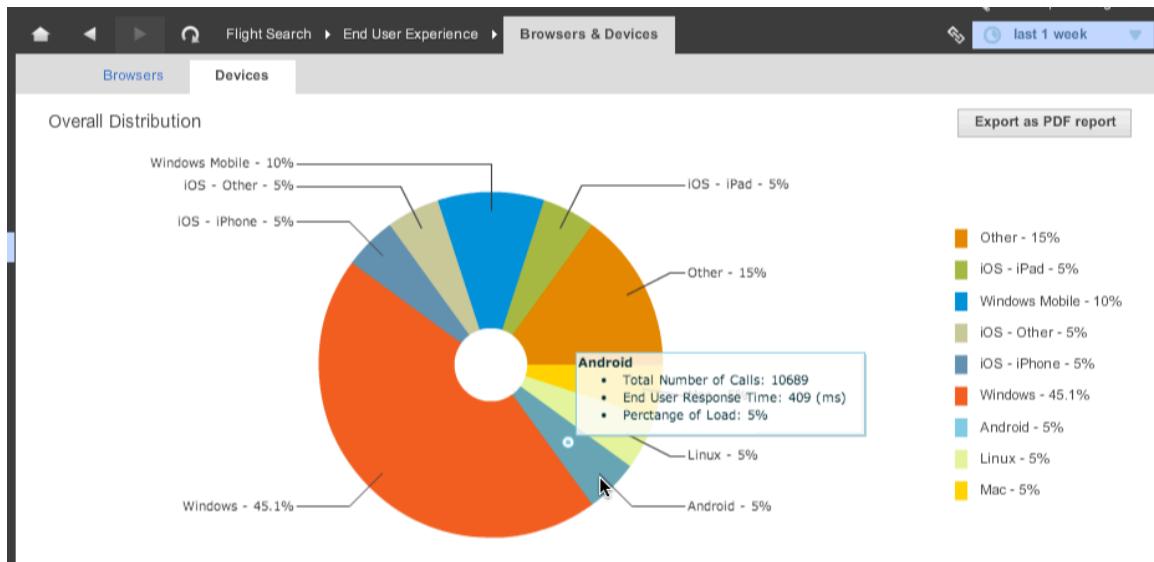
To access EUM performance by device:

1. In the left navigation pane click: **End User Monitoring -> Browsers & Devices**.
2. Click the Devices tab if it is not already selected.

Overall Device Distribution

The Overall Distribution chart shows the percentages of your end users using different devices.

To see total number of calls, average end user response time, and percentage of the total load for a particular device, hover over the device in the chart.



Performance by Device

The Performance by Device list below the chart displays a row for each device.

Name	Requests per Minute	Total Number of End	End User Response Time (ms)	Front End Time	Page Render Time	Document Ready	Document Download	Document Processing	First Byte Time	Response Available	Server Connection	Page Views with	AJAX Request
Other	165	10419	402	285	110	176	79	78	164	142	51	1807	464
Windows Mobile	110	7018	409	287	110	177	81	78	167	144	52	1249	277
iOS - iPad	55	3508	415	286	109	177	81	79	172	148	51	579	144
iOS - Other	55	3503	399	283	108	175	79	77	163	141	51	548	178
iOS - iPhone	55	3522	398	284	109	176	80	77	161	140	51	549	169
Windows	493	32049	405	286	110	176	80	78	165	142	52	5312	1368
Android	56	3473	408	286	109	177	81	78	166	141	51	587	144
Linux	55	3485	408	284	109	175	80	78	168	142	53	567	139
Mac	55	3538	412	288	110	178	81	78	169	146	51	594	148

The columns show the various EUM metrics by device. See [EUM Metrics](#).

Click a column to sort the devices based on the column's metric. For example, if you want to sort the slowest devices in terms of Download Time with the slowest devices at the top of the list, click the Download Time column. You can toggle the column header to switch between ascending and descending order.

To filter the list to see only rows for a particular device, enter the device in the filter field.

It is a good idea to monitor this list even if AppDynamics reports that overall EUM performance is normal. Information about poor performance on one device can be masked by overall metrics if that device is not used by a significant percentage of your users.

Distribution by Country

The Distribution by Country list breaks out the device distribution of your end users by country.

Distribution by Country

Name	Windows Mobile	Windows	iOS - iPad	iOS - Other	Mac	Android	iOS - iPhone	Linux
France	10.2 %	53.0 %	6.1 %	5.2 %	6.5 %	5.9 %	7.3 %	5.9 %
Hong Kong	12.5 %	53.9 %	6.4 %	6.3 %	5.0 %	5.7 %	5.9 %	4.4 %
Australia	12.7 %	51.7 %	6.8 %	5.0 %	6.9 %	5.7 %	5.6 %	5.5 %
Taiwan	10.6 %	52.5 %	5.3 %	7.8 %	5.0 %	6.1 %	6.8 %	5.9 %
Malaysia	11.4 %	53.8 %	5.5 %	6.0 %	5.9 %	6.1 %	5.8 %	5.5 %
Netherlands	12.0 %	53.0 %	5.7 %	5.6 %	5.9 %	5.9 %	6.2 %	5.6 %
Austria	11.2 %	54.4 %	6.7 %	4.2 %	6.1 %	5.5 %	5.4 %	6.4 %
United States	11.7 %	53.3 %	5.9 %	5.6 %	5.7 %	6.0 %	5.9 %	5.9 %
Poland	12.8 %	53.3 %	6.1 %	5.8 %	6.4 %	5.2 %	5.8 %	4.6 %
Canada	12.3 %	53.9 %	4.9 %	5.7 %	5.9 %	5.8 %	5.4 %	6.1 %
Denmark	11.3 %	54.0 %	5.5 %	6.0 %	5.9 %	6.0 %	6.1 %	5.1 %
Japan	11.2 %	53.8 %	5.8 %	6.4 %	6.0 %	5.4 %	5.5 %	5.9 %
United Kingdom	11.7 %	52.2 %	5.6 %	6.9 %	6.6 %	5.9 %	5.3 %	5.9 %
China	11.5 %	53.4 %	5.8 %	5.8 %	6.2 %	5.5 %	5.8 %	5.9 %
New Zealand	12.4 %	52.2 %	6.2 %	5.9 %	6.1 %	5.4 %	5.7 %	6.0 %

This list is particularly useful in conjunction with the worst performing regions panel in the Geo Dashboard. If a particular country is experiencing poor performance, it is possible that a significant percentage of your users in that country use a poorly performing device. This list can help you to determine whether the device is a contributing factor.

Learn More

- [EUM Metrics](#)
- [Time Ranges](#)
- [Geo Dashboard](#)
- [Monitor End User Experience \(EUM\)](#)
- [Monitor End User Experience by Browser](#)

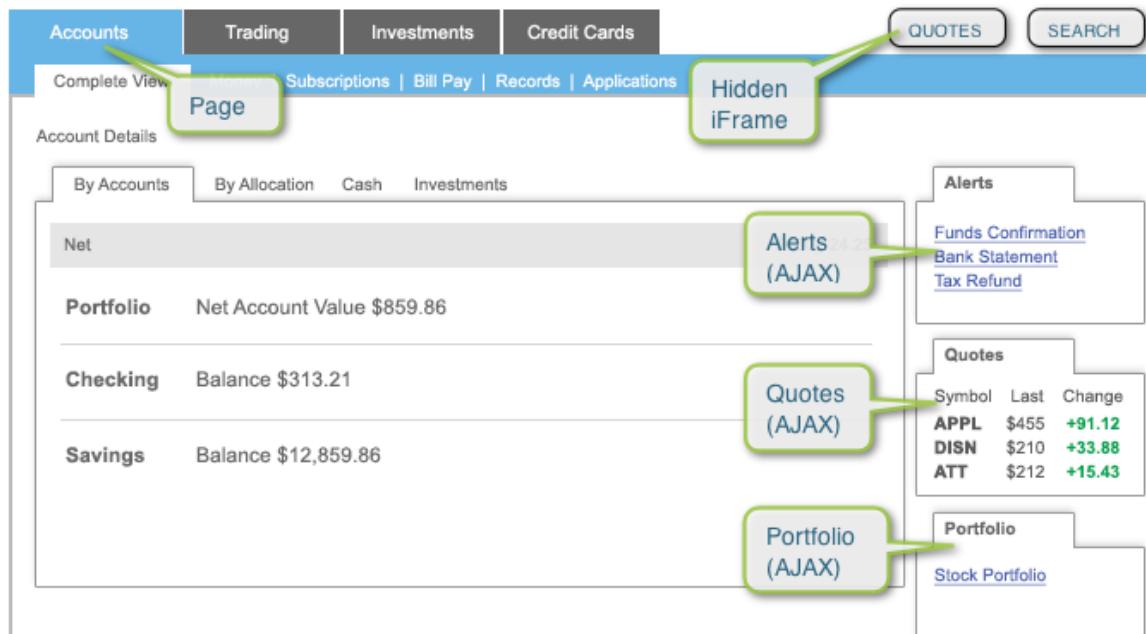
Pages

In AppDynamics a page represents what an end-user sees in a single browser window.

A page may include one or more hidden IFrame, which can be nested.

A page or an iframe can make Ajax requests to fetch data to display on the Web page.

You can monitor EUM metrics for pages, IFrames and Ajax requests.



Each page, IFrame and Ajax request has a unique name and page dashboard from which you can monitor its metrics and get browser snapshots that report information about the page at a certain point in time.

Learn More

- Page List
- Page Dashboard
- Browser Snapshots
- Monitor End User Experience (EUM)

Page List

- Accessing the Page List
- How the Page List is Organized
- Top Pages
- More Actions Menu
- Learn More

The page list shows all the monitored base pages, Iframes, and Ajax requests in the application, along with their key performance indicators.

Accessing the Page List

To access the page list:

In the left navigation pane, click **End User Experience -> Pages & Ajax Requests**.

How the Page List is Organized

The page list has a row for each base page, Iframe and Ajax request used to send requests to your application.

Pages & Ajax Requests										
	Name	Type	Requests per Minute	Total Number of End User Requests	End User Response Time (ms)	Front End Time (ms)	Page Render Time (ms)	First Byte Time (ms)	Server Connection Time (ms)	AJAX Request Errors per Minute
1	/index.html	IFrame	2	198	495	295	112	200	57	-
2	/postOnLoadError.html	IFrame	2	126	488	284	100	205	55	-
3	/played.jsp	Page	6	554	488	284	106	203	54	-
4	/cookieNoNav.html	Page	27	2400	484	288	111	196	-	-
5	/visual.html	Page	14	1204	482	288	112	194	51	-
6	/checkout.html	IFrame	3	220	482	286	107	196	51	-
7	/cookieNoNav.html	IFrame	9	818	481	285	109	196	-	-
8	/fred.html	Page	533	48006	480	285	109	194	51	-
9	/index.html	Page	7	620	479	288	111	191	52	-
10	/login.html	Page	7	587	478	288	112	191	59	-

The Name column shows the name of the page as it is configured, but always in lower-case. See [Configure Page Identification and Naming](#) for information on how to configure page names.

The Type column indicates whether the row represents a base page, IFrame or Ajax request.

The remaining columns report EUM metrics for the pages. See [EUM Metrics](#). You can specify which metrics to display by clicking **View Options**.

Click a column header to sort the pages based on the column's metric. For example, if you want to sort the slowest pages in terms of Page Render Time with the slowest pages at the top of the list, click the Page Render Time column header. You can toggle the column to switch between ascending and descending order.

To view the page dashboard for a specific page, select the page and click **View Dashboard** or just double-click the page.

To filter the types of pages displayed in the list, select the type at the top of the list. For example, to see only Ajax requests, select **Ajax Requests** and clear **Pages** and **IFrames**. You can also specify not to display pages that have no load in the selected time frame.

Top Pages

Click the Top Pages tab as a shortcut to troubleshooting the ten worst performing pages in terms of various metrics.

This tab displays multiple page list panels in which the pages are sorted by these metrics:

- Requests Per Minute
- End User Response Time
- Page Render Time
- Document Ready Time
- First Byte Time
- Page download time
- Server Connection Time

Click on an item in one of the lists to display the page dashboard for the page. Click **View All** to return to the unified list of all the pages.

More Actions Menu

In the More Actions menu in the All Pages tab, you can select a page in the list and perform the following actions on that page

- **Exclude** Use this option to direct AppDynamics to ignore this page and stop reporting metrics for it. You can use the **View Excluded Pages** option to see pages that have been excluded and then you can "unexclude" them.
- **Rename** Use this option to rename the page in the AppDynamics console.
- **Delete Item** Use this option to remove the page from the list. If AppDynamics discovers the page again it will reappear in the list. To prevent it from re-appearing use **Exclude**.

Learn More

- [Pages](#)
- [Page Dashboard](#)
- [Configure Page Identification and Naming](#)

- Dashboards
- Monitor End User Experience (EUM)

Page Dashboard

- Accessing a Page Dashboard
- How the Page Dashboard is Organized
 - The Dashboard Tab
 - Dashboard Information for a Base Page or IFrame
 - Summary Graphical View
 - Summary List View
 - End User Response Time Trend Graph
 - Load Trend Graph
 - Page Views with JavaScript Errors Trend Graph
 - Dashboard Information for an Ajax Request
 - Summary Graphical View for Ajax
 - Ajax End User Response Time Trend Graph
 - Ajax Request Load Trend Graph
 - The Browser Snapshots Tab
 - Learn More

The Page Dashboard provides single-click access to End User Monitoring (EUM) metrics and browser snapshots for pages, IFrames and Ajax requests.

Each page, IFrame and Ajax request has its own dashboard.

For Page Dashboards to be visible, EUM must be enabled at the application level. By default it is disabled. For information on disabling EUM either entirely or partially for specific business transactions, see [Configure End User Experience](#).

Accessing a Page Dashboard

To view a dashboard for a page, IFrame or Ajax request:

1. Select the business application.
2. In the left navigation pane, click **End User Experience -> Pages & Ajax Requests**. AppDynamics displays the page list.
3. From the list select the page, IFrame or Ajax request for which you want to see the dashboard.
4. Either double-click on the item or click **View Dashboard**.

How the Page Dashboard is Organized

The Page Dashboard shares the same components as other [Dashboards](#), including [Time Ranges](#).

The Page Dashboard contains two tabs: Dashboard and Browser Snapshots.

The Dashboard Tab

The top section in the Dashboard tab depicts key EUM performance indicators for the page, IFrame, or Ajax request showing the total End User Response Time for the selected time range and the breakdown of the End User Response Time into its component metrics. This gives a clear picture of which tasks are taking the most time. See [EUM Metrics](#) for details about the metrics.

The information reported for base pages and IFrames is identical. Different information is reported for Ajax requests.

You can view this information as a graph or as a list. Click **View as List** or **View Graphical** as you prefer.

Below the summary graph or list are three trend graphs:

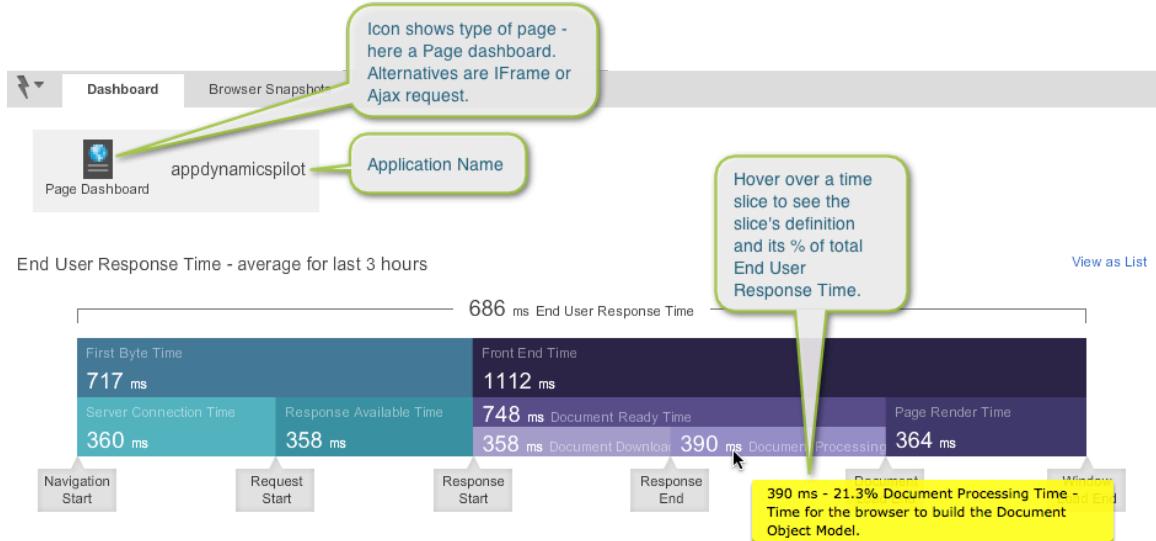
- The End User Response Time trend graph shows the same key EUM performance indicators as the summary graph, but over time.
- The Requests per Minute trend graph shows when the highest loads occurred.
- The Page Views with JavaScript Errors (for a page or IFrame) or AJAX Request Errors per minute (for an Ajax request) trend graph shows when most of the errors occurred.

It is possible that the aggregated component metrics do not always add up to the displayed total End User Response Time because the

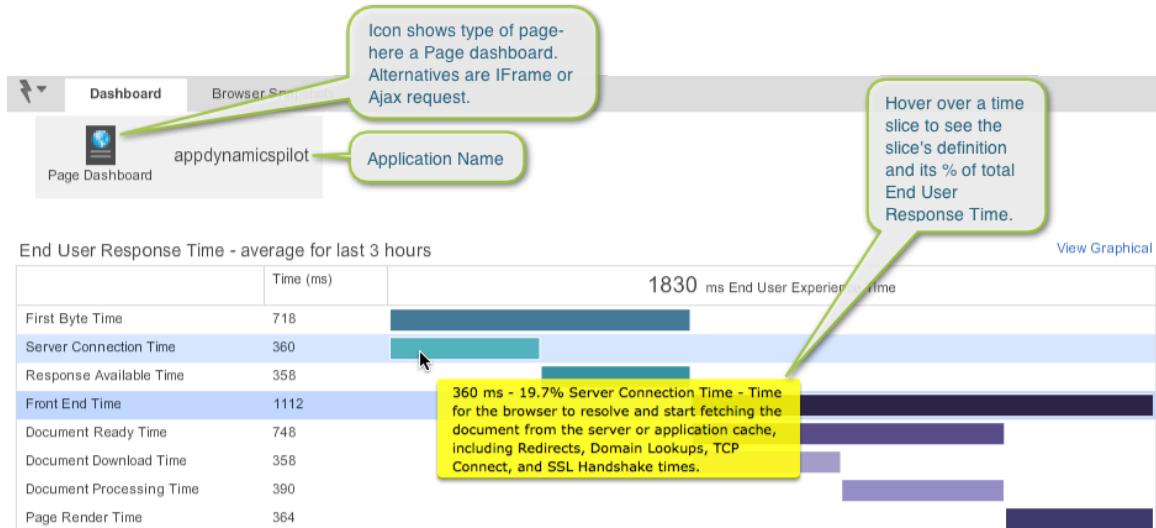
availability of some EUM metrics depends on the capability of the end-user's Web browser. See [Apparent Anomalies Resulting from Variations in Browser Reporting](#).

Dashboard Information for a Base Page or IFrame

Summary Graphical View

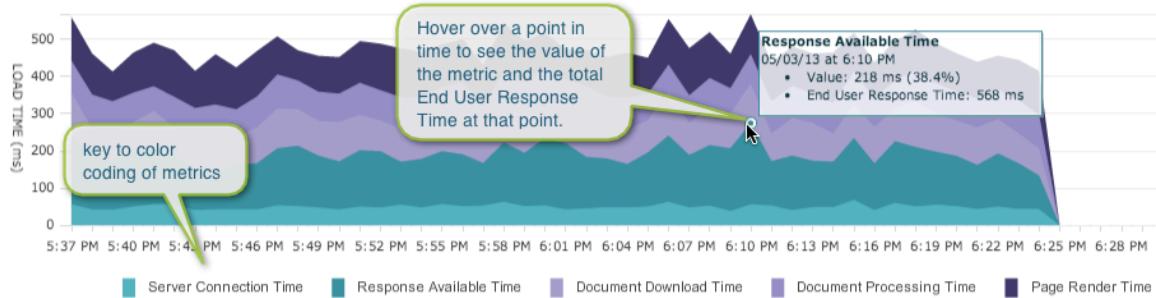


Summary List View

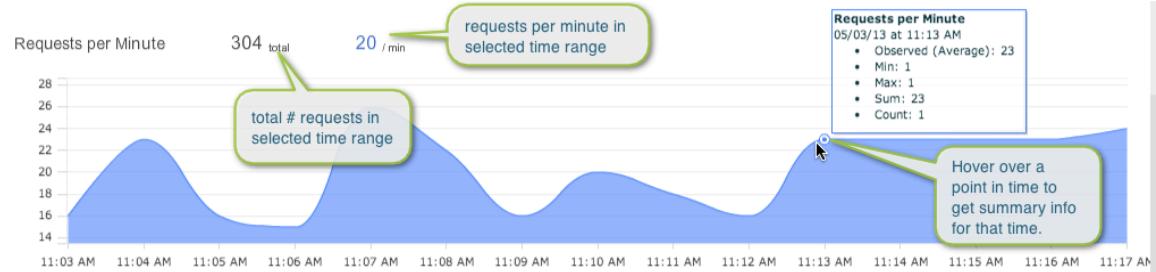


End User Response Time Trend Graph

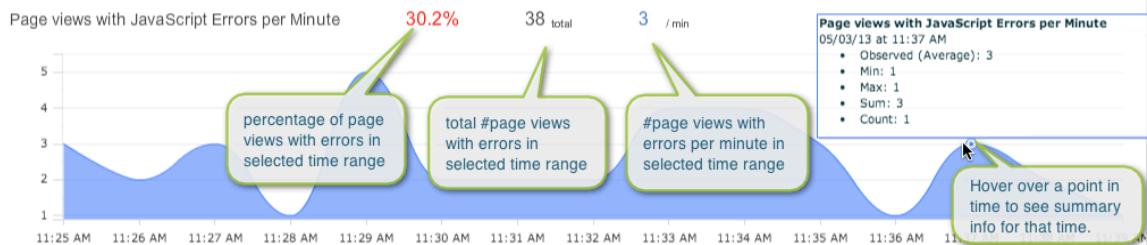
End User Response Time - Trend



Load Trend Graph

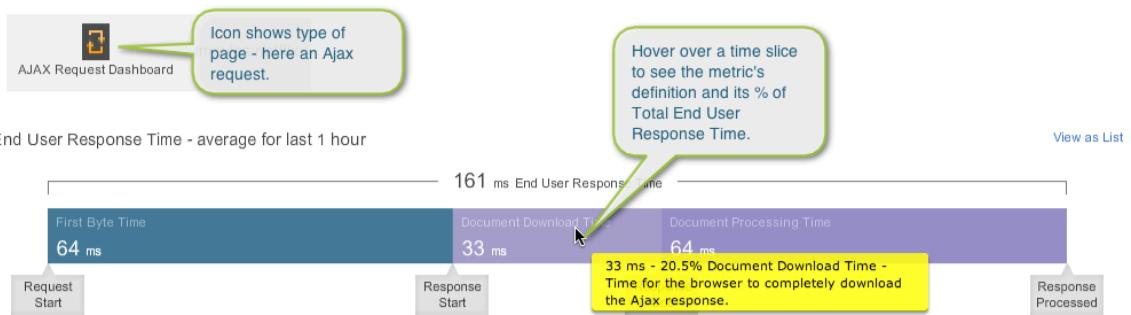


Page Views with JavaScript Errors Trend Graph



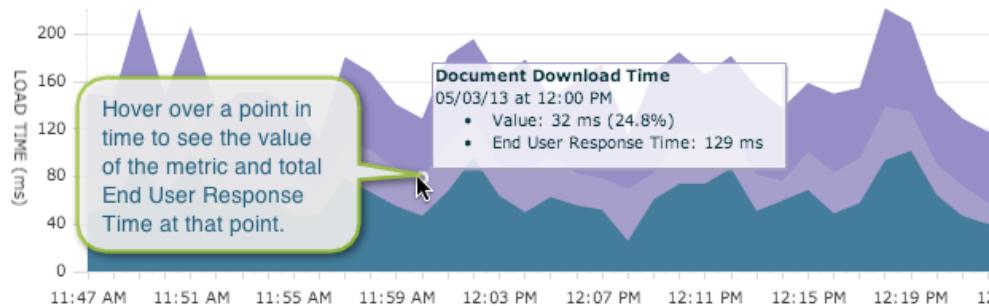
Dashboard Information for an Ajax Request

Summary Graphical View for Ajax



Ajax End User Response Time Trend Graph

End User Response Time - Trend



Ajax Request Load Trend Graph



The Browser Snapshots Tab

The Browser Snapshots tab shows a list of page snapshots for the base page, IFrame or Ajax request. See [Browser Snapshots](#).

Learn More

- Monitor End User Experience (EUM)
- Dashboards
- Pages
- EUM Metrics
- Page List
- Browser Snapshots

Browser Snapshots

- Accessing a Browser Snapshot
 - To access a browser snapshot
 - Configuring the Browser Snapshot List
- Browser Snapshot Data
 - Business Transactions in Browser Snapshots
 - To locate transaction snapshots associated with a browser snapshot
 - Data Not Captured in Browser Snapshots
 - JavaScript and Ajax Errors in Browser Snapshots
- Learn More

Browser snapshots capture metrics from an instance of an actual user experience. They also give you the ability to drill into errors and server-side transaction snapshots associated with the experience when the app servers are instrumented with AppDynamics app agents.

A browser snapshot presents diagnostic data, taken at a certain point in time, for an individual base page, Iframe or Ajax request. You access the browser snapshot list for a page from the Browser Snapshots tab of the page dashboard.

When EUM is enabled, AppDynamics collects browser snapshots for:

- every base page, IFrame an Ajax request; serves as a heartbeat snapshot
- the slowest page in every region, device and browser
- unique JavaScript errors; identified by script name and line number
- unique Ajax errors; identified by the HTTP error code in the Ajax response

See [Configure Browser Snapshot Collection](#) for more information about browser snapshot collection.

Accessing a Browser Snapshot

To access a browser snapshot

1. Click the Browser Snapshots tab in the page dashboard.

A list of snapshots for the page, with their summary data, taken over the time range selected in the Time Range dropdown menu appears.

	Time	End User Response Time (ms)	URL	Browser	Device	OS	Country	State / Region	City	
✓	03/21/13 9:53:58 AM	432	http://mycompany.com/marketplace.html	Other	Other		United States	California	Los Angeles	▲
✓	03/21/13 9:53:56 AM	455	http://mycompany.com/marketplace.html	Firefox 3	Computer	Windows	Japan	Osaka	Osaka	
✓	03/21/13 9:53:01 AM	421	http://mycompany.com/marketplace.html	Other	Other		Netherlands	Overijssel	Hengelo	
✓	03/21/13 9:52:53 AM	551	http://mycompany.com/marketplace.html	Chrome 16	Computer	Windows	United States	North Carolina	Skyland	
✓	03/21/13 9:51:58 AM	306	http://mycompany.com/marketplace.html	Other	Computer	Mac	Netherlands	Utrecht	Utrecht	
✓	03/21/13 9:51:52 AM	435	http://mycompany.com/marketplace.html	Other	Other		Japan			
✓	03/21/13 9:51:50 AM	447	http://mycompany.com/marketplace.html	Internet Explorer	Computer	Windows	China	Yunnan	Zhaotong	
!	03/21/13 9:51:16 AM	902	http://mycompany.com/marketplace.html	Safari 4	Mobile & Tablets	iOS - iPad	Australia	Western Australia	Perth	
!	03/21/13 9:51:12 AM	878	http://mycompany.com/marketplace.html	Firefox 16	Computer	Windows	Canada	Ontario	Toronto	
✓	03/21/13 9:50:57 AM	611	http://mycompany.com/marketplace.html	Other	Other		New Zealand	Auckland	Auckland	
✓	03/21/13 9:49:52 AM	808	http://mycompany.com/marketplace.html	Chrome 19	Computer	Windows	Japan	Miyagi	Natori	
✓	03/21/13 9:49:51 AM	322	http://mycompany.com/marketplace.html	Internet Explorer	Mobile & Tablets	Windows 8	Netherlands	Noord-Holland	Amstelveen	
✓	03/21/13 9:48:51 AM	420	http://mycompany.com/marketplace.html	Other	Computer	Mac	Netherlands	Noord-Holland	Amstelveen	
✓	03/21/13 9:48:50 AM	437	http://mycompany.com/marketplace.html	Internet Explorer	Computer	Windows	Poland	Lubelskie	Bielsko-Biala	
!	03/21/13 9:47:55 AM	1226	http://mycompany.com/marketplace.html	Other	Other		United States	California	Concord	
✓	03/21/13 9:47:55 AM	535	http://mycompany.com/marketplace.html	Other	Computer	Mac	China	Guangdong	Meizhou	

You can sort the snapshots by clicking a column header. For example, click the End User Response Time (ms) column header to sort the snapshots in descending order, with the highest response times at the top of the list.

2. Either

- Double-click the snapshot that you want to examine
- or
- Select the snapshot that you want to examine and click **View Browser Snapshot**.

Configuring the Browser Snapshot List

Click **View Options** to configure the columns to display in the browser snapshot list.

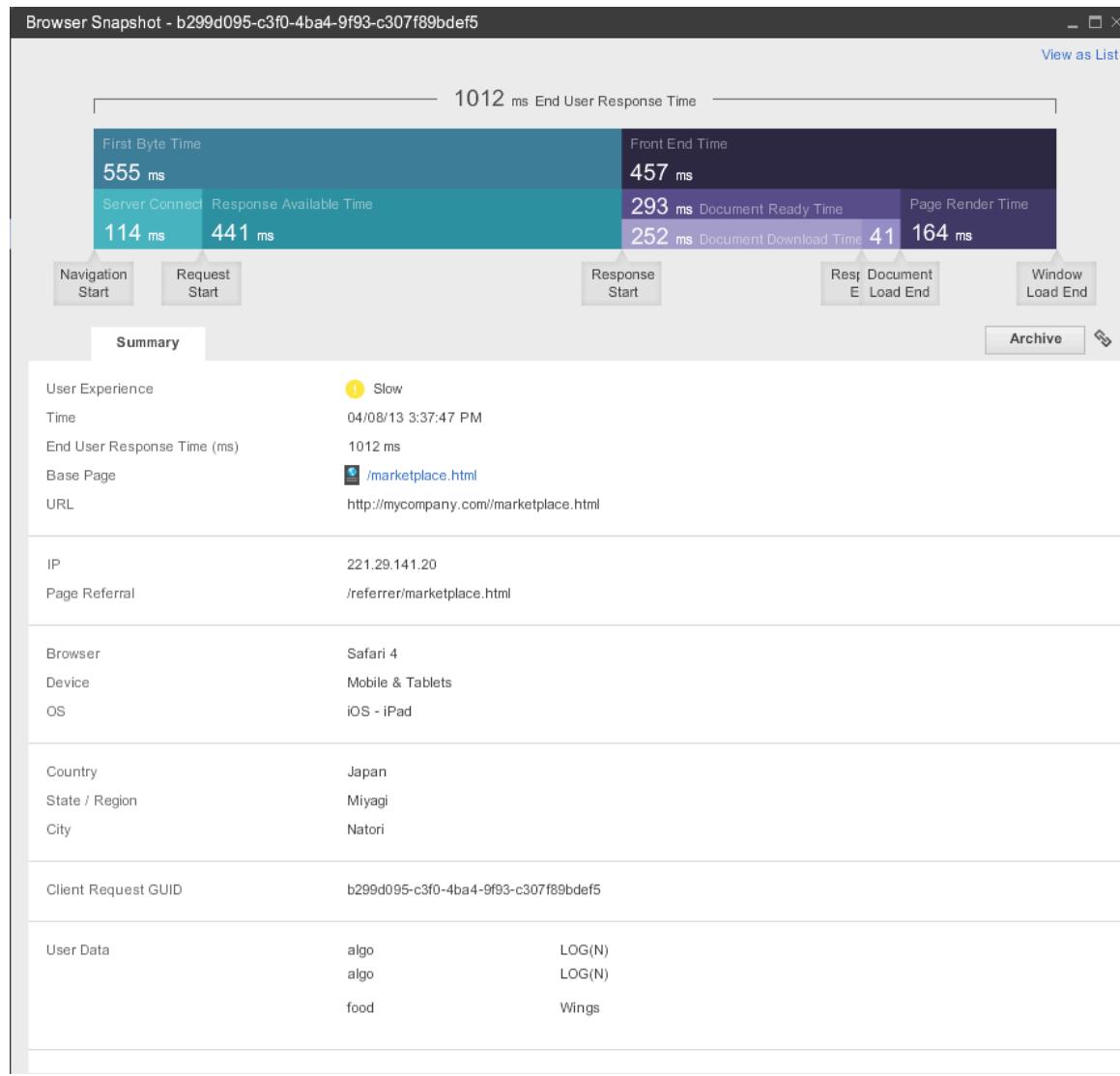
Configure what columns to show		Total Minute	Number of End
<input checked="" type="checkbox"/>	Type		
<input checked="" type="checkbox"/>	Requests per Minute	28557	
<input checked="" type="checkbox"/>	Total Number of End User Requests	11573	
<input checked="" type="checkbox"/>	End User Response Time (ms)	9562	
<input checked="" type="checkbox"/>	Front End Time (ms)	1458	
<input type="checkbox"/>	Page Render Time (ms)	736	
<input type="checkbox"/>	Document Ready Time (ms)	723	
<input type="checkbox"/>	Document Download Time (ms)	483	
<input type="checkbox"/>	Document Processing Time (ms)	387	
<input type="checkbox"/>	First Byte Time (ms)	370	
<input type="checkbox"/>	Response Available Time (ms)	370	
<input type="checkbox"/>	Server Connection Time (ms)	368	
<input checked="" type="checkbox"/>	Page views with JavaScript Errors per Minute	356	
<input checked="" type="checkbox"/>	AJAX Request Errors per Minute	356	

You can filter the list to display only browser snapshots that meet certain criteria. For example, the following configuration restricts browser snapshots to JavaScript and AJAX errors that occurred on Internet Explorer.

The screenshot shows the 'Browser Snapshots' filter interface. At the top, there are tabs for 'Dashboard' and 'Browser Snapshots'. Below the tabs are filter controls: a funnel icon, a search icon, a date range selector, and a 'View Options' button. Under these are buttons for 'Normal', 'Slow', 'Very Slow', and 'Stall' performance levels. A 'Hide Filters' link is also present. The main area contains three expandable sections: 'Errors' (with 'JavaScript error occurred' and 'Error during AJAX call' checked), 'Execution Time' (with dropdowns for various times set to '>= 500 ms'), and 'Browsers' (with 'Internet Explorer' selected). A '+' button is available to add more browser filters.

Browser Snapshot Data

Every browser snapshot displays EUM metrics for the time of the snapshot and a summary tab describing the user experience. You can view this information as a graph or as a list. Click **View as List** or **View Graphical** as you prefer.

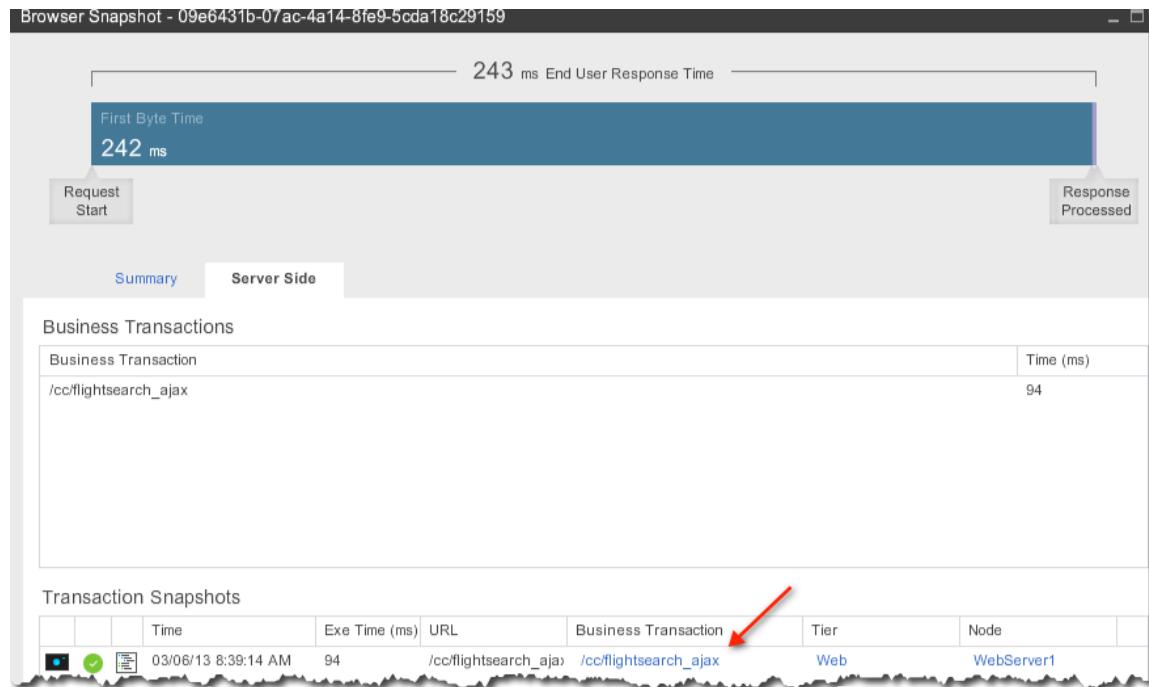


The IP address is the machine from which the page was accessed. The Page Referral is the page that the end-user visited prior to the page that generated the snapshot; this is the page that referred the end-user to the current page.

The User Data data appears only if you have configured AppDynamics to collect user data from the browser. See [Add Information to a Browser Snapshot](#).

Business Transactions in Browser Snapshots

When a Web page or Ajax request generates a business transaction on an instrumented server, the browser snapshot displays a Server Side tab with the list of business transactions associated with the browser snapshot. If transaction snapshots for the associated business transaction were captured at the same time as the browser snapshot, they are linked in the Transaction Snapshots section of the Server Side tab of the browser snapshot. Transaction snapshots are triggered when slow or stalled transactions are identified, when a diagnostic session is started, or periodically based on a configured interval. In general, slow, very slow and stalled transactions are more likely to trigger a transaction snapshot on the server than transactions operating within normal range. For more information about when server-side transaction snapshots are captured see [Transaction Snapshots](#) and [Configure Transaction Snapshots](#).



This capability lets you drill down from the original end-user experience in the browser all the way to the associated transaction snapshot on the app server with its call graphs and other server-side diagnostics.

On the server side, if there are browser snapshots associated with a transaction snapshot, the ADDITIONAL DATA tab in the transaction snapshot may have a EUM GUID that you can use to access the browser snapshot.

To locate transaction snapshots associated with a browser snapshot

1. In the Server Side tab of the browser snapshot, scroll down to the Transaction Snapshots section.
2. Click the business transaction that you want to investigate.
This takes you to the transaction snapshot list for the associated business transaction.
3. In the transaction snapshot list, select a transaction snapshot that was taken approximately at the time of the browser snapshot.
4. Click **View Transaction Snapshot**.

See [Transaction Snapshots](#) for information on how to troubleshoot root cause using transaction snapshots.

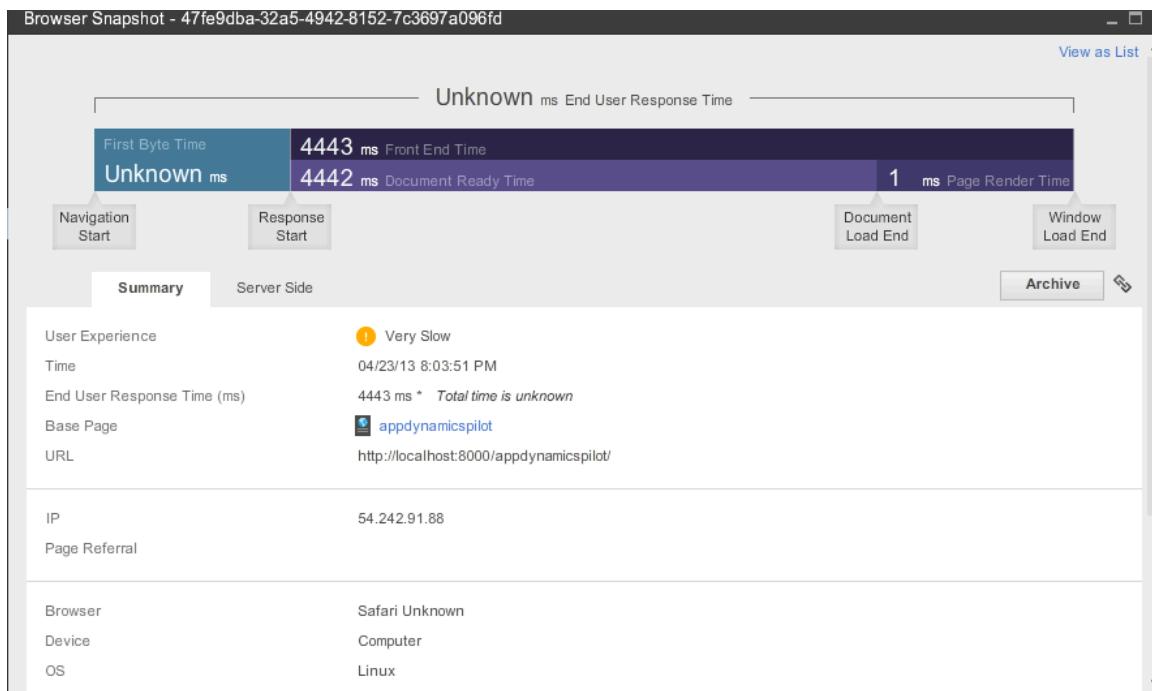
Data Not Captured in Browser Snapshots

AppDynamics captures the metrics reported by your end-users' web browsers.

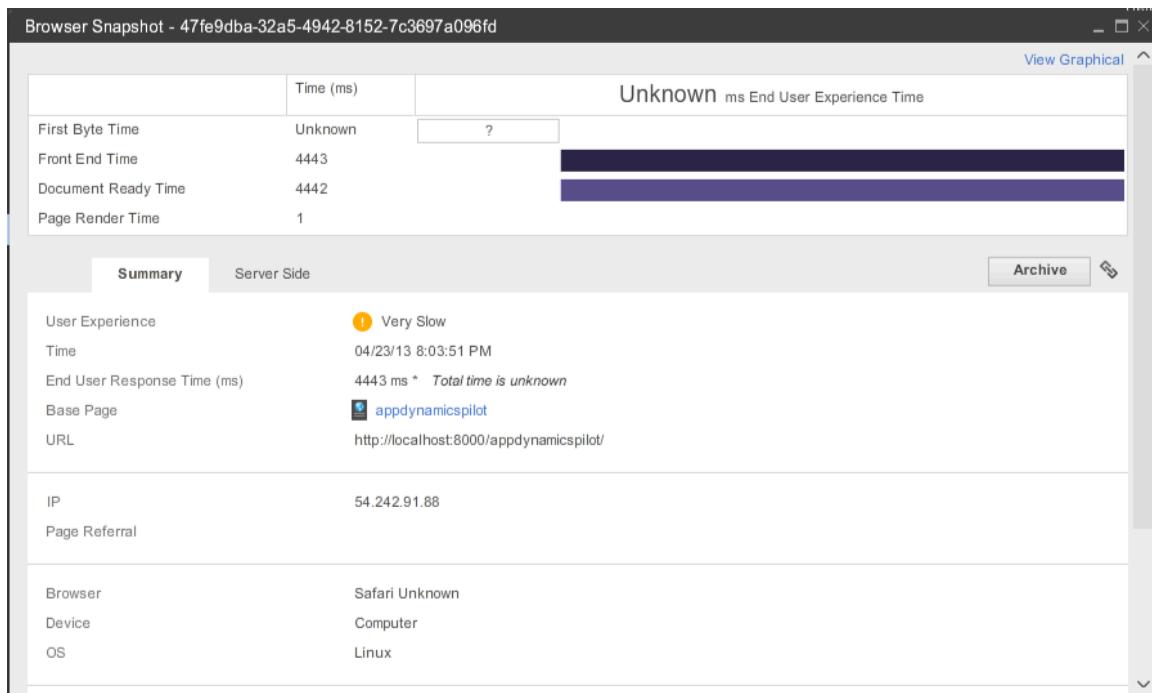
Occasionally you may see Unknown data reported for one or metrics in a browser snapshot. This occurs when a user's browser does not report all metrics, probably because the browser does not support the navigation timing API. See [EUM Metrics Availability](#) for details about which metrics may not be captured based on browser capabilities.

For example, the following browser snapshots do not have data for First Byte Time, so they cannot display First Byte Time or End User Response Time.

Unknown Metrics in Graphical View

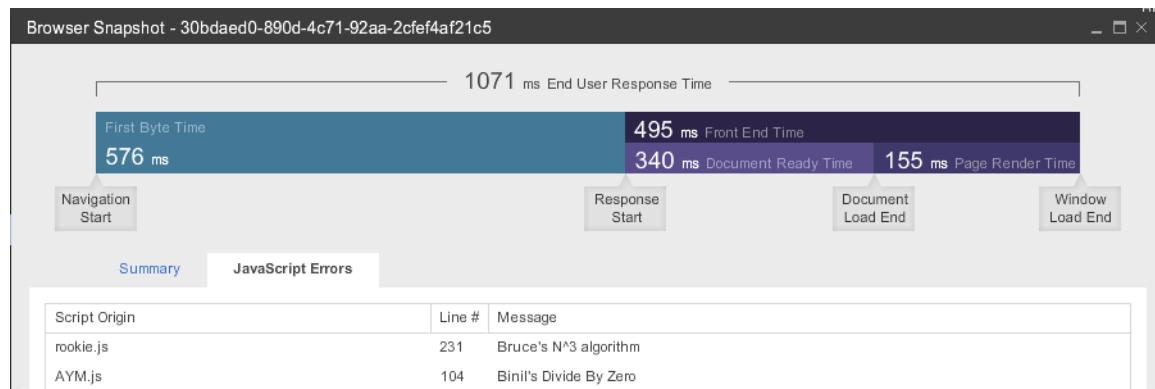


Unknown Metrics in List View



JavaScript and Ajax Errors in Browser Snapshots

When there are JavaScript or Ajax errors, the browser snapshot includes a tab with details about the error. For example, the following browser snapshot shows a line in the script that was the source of the error.



The next browser snapshot shows the HTTP response code from an Ajax error.

User Experience	✓ Normal
Time	04/15/13 3:48:59 PM
End User Response Time (ms)	96 ms
AJAX	! [redacted]
URL	[redacted]
AJAX Error	Status: 408

You can configure errors to ignore if you are seeing too many errors that are not of interest. See [Configure JavaScript and Ajax Error Detection](#).

Learn More

- [Page Dashboard](#)
- [Transaction Snapshots](#)
- [Configure Browser Snapshot Collection](#)
- [EUM Metrics](#)
- [Configure EUM Thresholds](#)
- [Configure End User Experience](#)
- [Inject the JavaScript Agent for EUM](#)
- [Add Information to a Browser Snapshot](#)
- [Configure JavaScript and Ajax Error Detection](#)

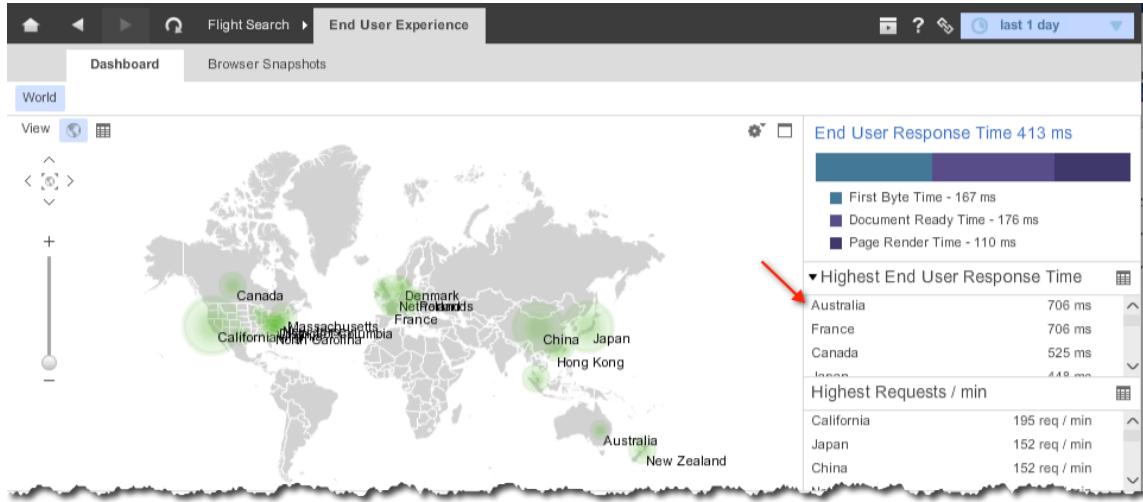
Geo Dashboard

- [Accessing the Geo Dashboard](#)
- [How the Geo Dashboard is Organized](#)
- [Using Map View](#)
 - [Geographic Drill-Down](#)
 - [Map Actions](#)
 - [Possible Effects of Unsupported Regions on Map Display](#)
- [Configuring the Map View Options](#)
 - [To Configure the Map](#)
- [Learn More](#)

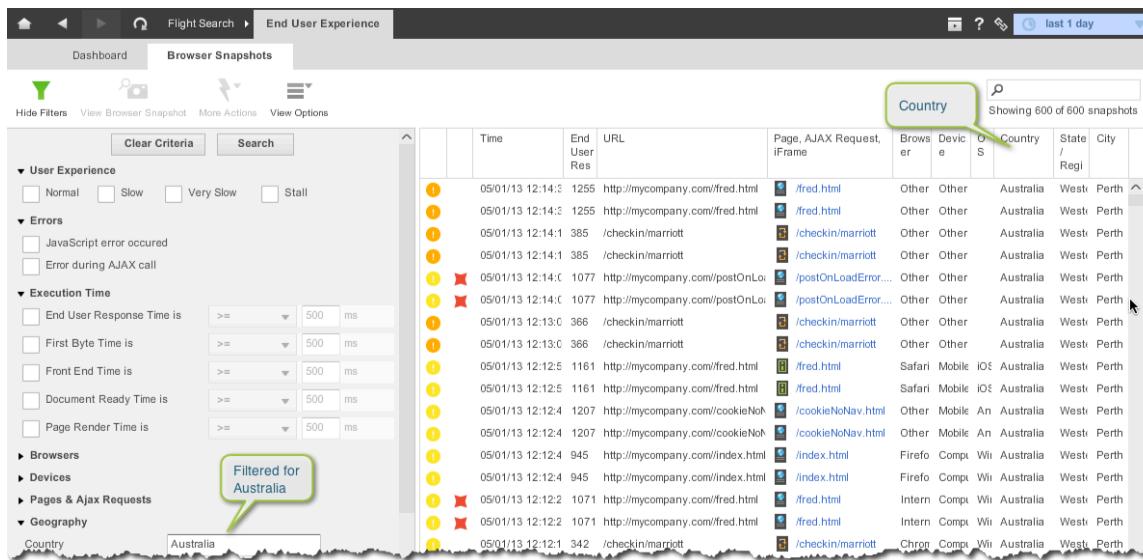
The Geo Dashboard displays key EUM performance metrics by geographic location. It shows you which regions have the highest loads, the longest response times, and the most page views with errors.

The Geo Dashboard lets you see at a glance which locations are active and which of those are slow. Then you can use this information to drill down into browser snapshots just for the slowest regions.

For example, in the Dashboard tab below you can see that the slowest end-user experience is currently in Australia.



Then you can click the Browser Snapshots tab and filter the snapshots to get just the list of snapshots for Australia.



For the Geo Dashboards to be visible, EUM must be enabled at the application level. By default it is disabled. For information on enabling or disabling EUM, see [Configure End User Experience](#).

Accessing the Geo Dashboard

To access the Geo Dashboard:

1. Select the business application.
2. In the left navigation pane, click **End User Experience**.

How the Geo Dashboard is Organized

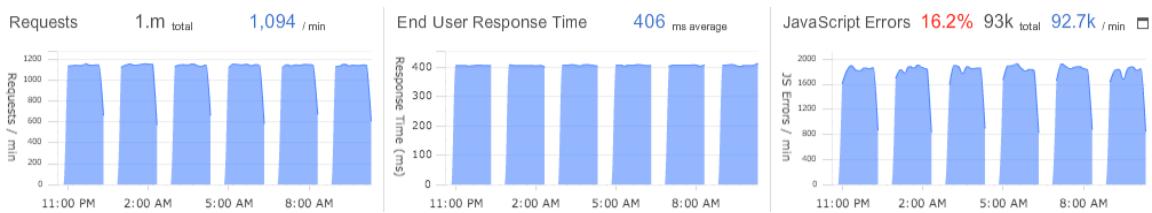
The dashboard is divided into three panels:

- a main panel in the upper left that displays geographic distribution of end users on a map, if you selected clicked the map view icon, or on a grid if you clicked the grid view icon
 - You can switch between map view and grid view by clicking these icons.
 - You can expand the map panel or the grid panel to fill the entire EUM Dashboard by toggling the expand icon in the upper right corner of the panel.
- a panel on the right displaying:

- Summary metrics for the selected time range: End User Response Time, First Byte Time, Document Ready Time and Page Render Time. Click the **End User Response Time** link to see more EUM metrics.
- Regions with highest response times, based on the metric you select from Highest End User Experience Time, Highest First Byte Time or Highest Front End Time.
To specify which metric will be used to determine and sort the regions with the highest response times, click the dropdown menu in the Highest End User Response Time panel.
- Ten regions with the highest load



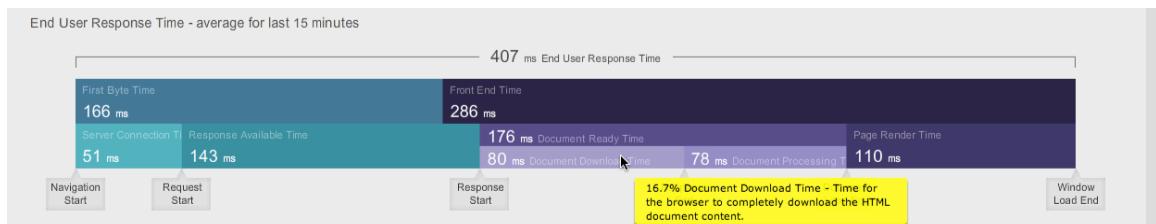
- Trend graphs in the lower part of the dashboard that dynamically display the number and rate of end user requests (load), average end user response time, and number and rate of page views with JavaScript errors.



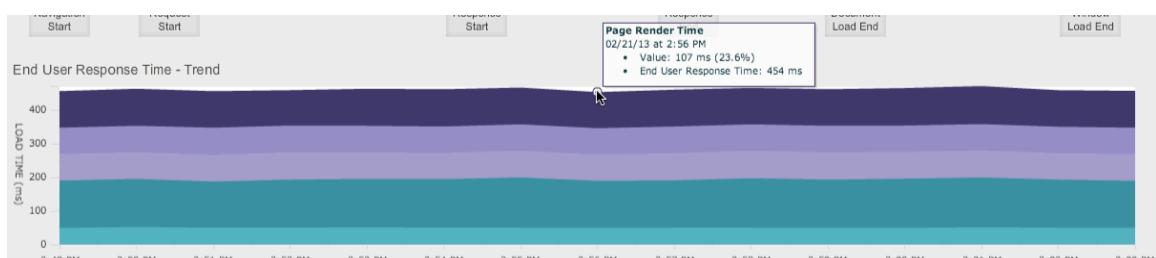
The metrics displayed throughout the dashboard are for the region currently selected on the map or in the grid. For example, if you zoom down from world view to France in the map, the summary panel and the trend graphs display data for France.

See [EUM Metrics](#) for definitions of the metrics.

You can hover over a metric time value in a chart to see a description of the metric and the percentage of the entire end user response time consumed by that time slice.



You can hover over a point in time in a trend graph to get the precise values for the individual metric you are hovering over, the percentage of the end user response time that the metric represents, and the average end user response time at the point in time.



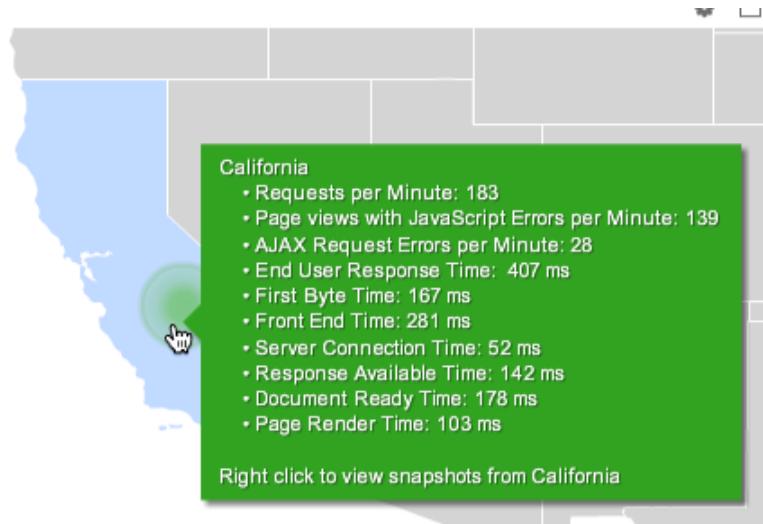
Using Map View

The main panel in map view displays a map superimposed with circles that represent average end user experience by region.

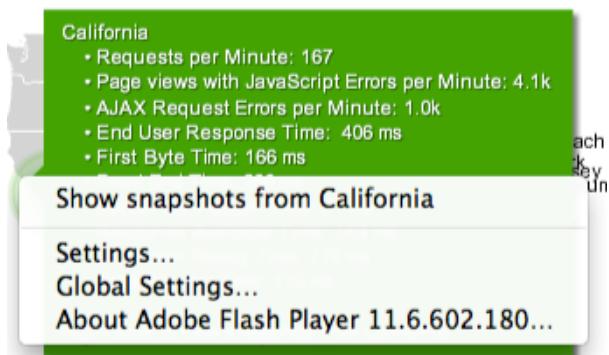
The size of a circle indicates the relative amount of traffic in a region: the larger the circle the higher the load. The color of a circle represents the relative response time experienced by users in a region: green for a fast response time, yellow for a medium response time, red for a slow response time. By default the colors are based on end user response time but you can configure them to be based on first byte time or front end time. In any case, large red circles represent regions of most concern. See [To configure map appearance](#) to adjust circle color and size ranges.

Geographic Drill-Down

You can hover over any region on the map to get summary metrics for that region.



Right-clicking lets you access browser snapshots for the region. See [Browser Snapshots](#).

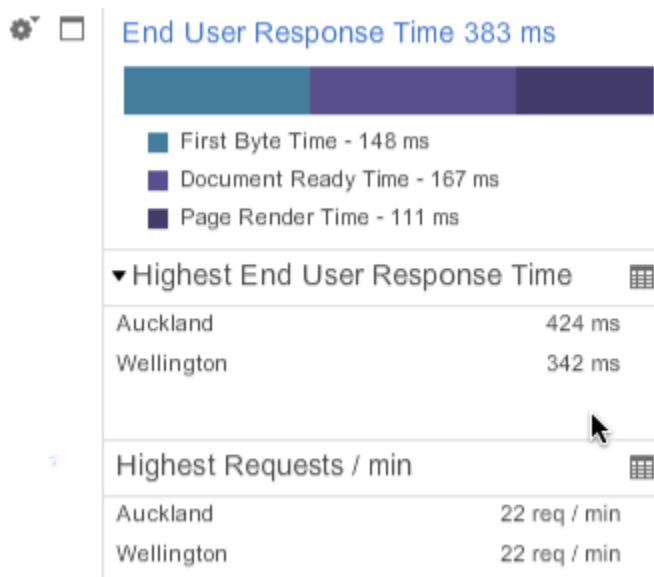


You can double-click any region on the map to drill down into metrics for that region.

When you drill down into the largest (by GDP) countries, a detailed map with the country's subregions is displayed. You can then drill down further into the subregions.



For locations for which detailed maps are not available, the country or region is colored blue to indicate that it is drilled down. Even when detailed maps are not available for subregions, EUM metrics are still collected and reported for the supported subregions in the summary panels and trend graphs.



For a complete list of the supported regions by country see [EUM Supported Regions](#).

Map Actions

You can perform the following actions directly in the map:

- Zoom into and drill down to a subregion in the map by clicking on the subregion. To zoom out to restore the currently selected region to the world or country, click the link in the location control in the upper left corner of the map. For example, if you have drilled down to India and then to West Bengal and now want to return to global view, select "World" in World > India > West Bengal.
- View summary statistics for a region by hovering over its circle.
- Zoom the entire map using the slider on the left. You can also use your mouse wheel to increase or decrease the map's zoom level.
- Reposition the map by clicking and dragging it or by clicking the directional arrows in the map control widget.

Possible Effects of Unsupported Regions on Map Display

It is possible for a country to display as slow (red circles) on the global map, but when you drill down to the country all its regions appear normal (green circles). Or a country may display as normal on the global map, but some subregions may display as slow when you drill down.

This can occur when a large number of slow requests are reported from a region that is unknown to your EUM Geo Database relative to the number of normal requests from known regions in the same country. Or conversely, the country may display as normal and some subregions display as slow, when some of those subregions are unknown regions. The metrics for the unknown region are displayed to the right of the map and on the dashboard in grid view, but not in the regional maps.

An unknown region is one that neither among the supported regions listed in EUM Supported Regions nor mapped through a private IP map for installations that host their own geo servers as described in [Alternate Geo Server Location](#).

Configuring the Map View Options

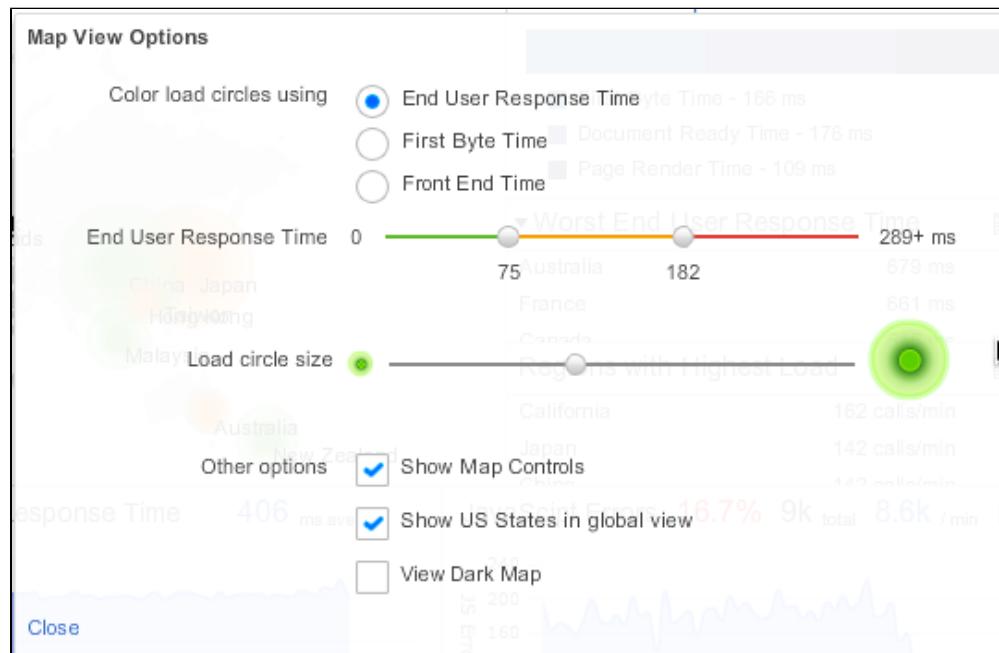
You can configure the dashboard display in a variety of ways:

- The ranges of the colors that indicate normal, warning and critical performance on the map
- Metric to use for evaluating performance for the colored circles: end user response time, first byte time, front end time
- The ranges of the circle sizes that indicate relative load on the map
- Whether to display the zoom slider and Home button on the map
- Whether to display a dark or light colored map
- The key metric on which to sort performance

All of these configurations are saved for the next time you log into AppDynamics.

To Configure the Map

1. In the Geo Dashboard, click the gear icon in the upper right corner of the map or grid panel to get the configuration window.



2. To configure circle color ranges representing performance thresholds, adjust the End User Response Time slider. For example, if you want circles to be red whenever the end user response time is 182 milliseconds or greater, set the maximum value of the yellow slider value to 182.

3. To configure circle sizes representing load (number of end user requests per minute), adjust the Load circle size slider to make the circles ranges larger or smaller.

4. To display the map control widget, check Show Map Controls. To hide them clear this check box. The map controls let you reposition the map using arrows and zoom the map using + and - buttons. After moving or zooming the map, if you want to return to the default zoomed out home view, click the globe icon in the center of the map control widget.

5. To show the individual state boundaries in the United States in global view, check Show US States in global View. To hide the state boundaries clear this check box.

The individual states in the United States are treated as countries, in that you can display an individual state as if it were a country. Click the state in the United States map to display EUM data for an individual state.

6. To view a dark colored map check View Dark Map. To view a light colored map clear this check box.

Learn More

- Dashboards
- EUM Metrics
- Browser Snapshots
- Configure End User Experience
- Customize Your EUM Deployment
- Hosting a Geo Server

Hosting a Geo Server

AppDynamics hosts a geo server to resolve the geographic information associated with end users. Some customers prefer to host their own geo server. This is most common:

- for intranet applications that do not want to reference a URL outside the organization's network
- when the accessing script comes from many locations through a single gateway, in which case a single IP address masks the actual locations of the end-users

See [Alternate Geo Server Location](#) for information on configuring your own geo server and private IP mapping file.

Configure End User Experience

- [Accessing EUM Configuration](#)
 - [To access EUM Configuration](#)
- [EUM Prerequisites](#)
- [Enable and Disable EUM](#)
 - If you are prompted for the license key
- [Inject the JavaScript Agent for EUM into Your Application](#)
- [Additional EUM Configurations](#)
- [Integration with BMC](#)
- [Learn More](#)

You must explicitly enable End User Monitoring (EUM) at the application level. By default it is disabled. See [Enable and Disable EUM](#).

A special EUM license key is required. See [EUM License](#).

AppDynamics collects metrics from your end users' experience in their Web browsers using a special JavaScript for agent for EUM. This agent must be injected into the Web pages to be monitored. See [Inject the JavaScript Agent for EUM](#).

Accessing EUM Configuration

To access EUM Configuration

1. In the left navigation menu, click **Configure -> Instrumentation**.
2. Click the End User Experience tab.

When you are configuring EUM settings, click **Save** in the EUM configuration screens whenever you make a change.

EUM Prerequisites

To turn on EUM functionality you need to:

- Enable End User Monitoring
- Enter the EUM License Key
- Inject the JavaScript Agent for EUM into your application

Enable and Disable EUM

In the Javascript Instrumentation tab in the EUM configuration screen:

- To enable EUM for the application, check the Enable End User Experience Monitoring check box.
- To disable EUM for the application, clear the Enable End User Experience Monitoring check box.

Click **Save** in the EUM configuration screen after you have enabled or disabled EUM.

If you first enable then disable EUM, you may see the EUM screen but there will be no EUM data.

The screenshot shows the AppDynamics EUM configuration interface. At the top, there's a navigation bar with icons for Home, Back, Forward, Flight Search, Configure, and Instrumentation (which is the active tab). Below the navigation bar are several tabs: Transaction Detection, Backend Detection, End User Experience (selected), Error Detection, and Diagnostic Data Collectors. Under the End User Experience tab, there are three sub-tabs: Javascript Instrumentation (selected), Page Naming, Error Detection, Thresholds, etc., and License Key. A large 'Save' button is located at the bottom left of the main content area. In the main content area, there's a section titled 'Enable End User Experience Monitoring' with a checked checkbox. Below this, there's a section titled 'Instrument your HTML pages with the AppDynamics JavaScript Agent' containing two numbered steps: '1 Download the JavaScript Agent' and '2 Include this line in your pages immediately after the <head> tag:' followed by a code snippet: <script>window["adrum-start-time"] = new Date().getTime();</script><script src="/adrum.js"></script>. There's also an 'Advanced' link and another 'Save' button at the bottom.

If you are prompted for the license key

1. Click the **License Key** tab.
2. Enter the EUM license key that you received by email from AppDynamics.
If you do not have your license key, see [EUM License](#) for information about how to obtain it.
3. Click **Save**.

On .NET only, the .NET app agent must also be enabled for EUM for full functionality. See [Configure the .NET App Agent for EUM](#).

Inject the JavaScript Agent for EUM into Your Application

The JavaScript Agent for EUM collects EUM metrics. See [EUM Metrics](#).

The JavaScript Agent for EUM must be injected into the headers of the pages for which you want to see these metrics. There are several ways to accomplish this. See [Inject the JavaScript Agent for EUM](#).

Additional EUM Configurations

You can also configure:

- Page Identification and Naming
- JavaScript and Ajax Error Detection
- EUM Thresholds
- EUM Snapshot Collection
- EUM Deployment

Integration with BMC

AppDynamics End User Experience is not available with AppDynamics for BMC Software Solutions. For information about integrating AppDynamics with BMC's EUM solution, see [Integrate AppDynamics with BMC End User Experience Management](#)

Learn More

- EUM License
- Inject the JavaScript Agent for EUM
- Configure the .NET App Agent for EUM
- Browser Snapshots

Configure the .NET App Agent for EUM

On .NET only, the .NET app agent must also be enabled for EUM for full functionality.

If, when you set up your .NET app agent, you chose to configure the tiers manually, in the app agent configuration screen you will see an End User Monitoring check box for every tier that you have assigned to the application. Check the check box to enable EUM in every tier for which you want to capture EUM metrics.

The screenshot shows the 'Assign IIS applications to tiers' screen. On the left, there's a list of 'Tiers' including Machine Agent, OrderSvc, Portal, QueueSvc, and Weblog. A tooltip for 'Assign tier' says: 'Select the tier and then the application to assign the tier to that application. Select application to unassign a tier.' A tooltip for 'Delete Tier' says: 'To delete a tier and all its assignments: select the tier name on the left and press Delete. You cannot delete tiers that are defined on the Controller.' Below this is an 'Add Tier' input field with a placeholder 'Please type the tier name and press Add Tier to add a new tier.' On the right, there's a table titled 'Local machine IIS applications' with columns: IIS Application, AppDynamics App, Tier, and End User Monitoring. The table lists several IIS sites under the 'Consumer' node, each assigned to a tier (NwTraders, OrderSvc, QueueSvc, Weblog) and having the 'End User Monitoring' checkbox checked. An arrow points to the 'End User Monitoring' checkbox for the 'Portal' tier.

If you chose to let AppDynamics configure the tiers automatically, by default all the tiers are configured to enable EUM.

See [Configure the App Agent for .NET](#).

The following features are available when you enable EUM in the .NET configuration:

- More options for injecting the JavaScript Agent for EUM. These include automatic injection (on ASP.NET and ASPX) and assisted injection-using attribute injection. Without this functionality you can use only manual injection.
See [Inject the JavaScript Agent for EUM](#).
- Server-side information linked to browser snapshots including name and timing for business transactions associated with browser snapshots and linking browser and transaction snapshots when transaction snapshots are available.
See [Business Transactions in Browser Snapshots](#).

Configure Page Identification and Naming

- Logic of Page Naming Rule Evaluation
- Default Page Naming Rules
- Custom Page Naming Rules
- Custom Page Exclude Rules
- [Learn More](#)

You can configure how pages, Ajax requests, and Iframes are named in the page list and page dashboards.

- You can use AppDynamics default naming rule, which you can leave as is or modify.

- You can create custom naming rules to override the default convention.
- You can disable the default naming rule and use only your own custom naming rules.
- You can create custom exclude rules to exclude from monitoring pages that meet certain criteria.

In this topic, the term "pages" includes Iframes and Ajax requests as well as base pages.

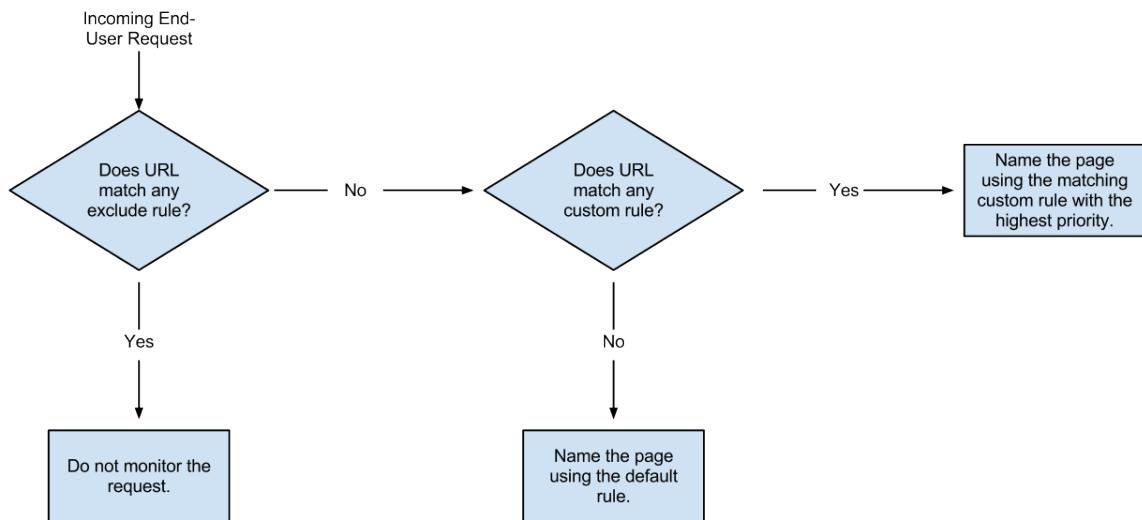
No matter how the page is named, AppDynamics always reports the page name in lower-case.

1. Access the EUM configuration screen if you are not already there. See [To access EUM Configuration](#).
2. Click the Page Naming, Error Detection, Thresholds, etc tab.
3. Expand **Configure how Pages, AJAX Requests, and Iframes will be named**.

Whenever you configure page naming, click **Save** to save the configuration.

Logic of Page Naming Rule Evaluation

This is the order in which AppDynamics evaluates the page naming rules.



Default Page Naming Rules

If you enable the default naming configuration and do not modify it, AppDynamics identifies and names your pages using the first 2 segments of the page URL.

You can modify the default configuration in the Default Naming Configuration section of the Page Naming, Error Detection, Thresholds, etc tab. For example, you can include the protocol or domain in the name, or use different segments of the URL, or run a regular expression on the URL, or include query parameters in the name.

Default Naming Configuration

Pages will be named using these configurations.

Enabled	<input checked="" type="checkbox"/>
Show Protocol	<input type="checkbox"/>
Show Domain	<input type="checkbox"/> <input type="radio"/> Show Full Domain <input type="radio"/> Show Subdomain
What part of the URL should be used in Page Names	<input checked="" type="radio"/> Use first <input type="text" value="2"/> segments <input type="radio"/> Use last <input type="text" value="1"/> segments <input type="radio"/> Use segment numbers <input type="text"/> (comma separated list, starts at 1) Example
	<input type="radio"/> Run regex on URI <input type="text"/> Pick indices from regex output <input type="text"/> (comma separated list of indices from regex output, starting at 0.) Example
Query String Parameters to use in Page Name	<input type="text"/> Comma separated list of parameter names. The values will be used in Page Names.

If you do not want to use the default convention at all, disable it by clearing the Enabled check box. In this case you must configure at least one custom page naming rule so that AppDynamics can identify and name pages.

Custom Page Naming Rules

You can create custom rules for identifying and naming pages.

To create a custom page naming rule, click the add icon in the Custom Naming Rules section. Then configure the custom rule for AppDynamics to use to identify and name the page.

This configuration screen is similar to the default configuration screen but it includes a priority field. The priority specifies which rule to apply to the naming of a page if it could be identified by more than one rule. For example, if CustomRuleA specifies **Use the first 3 segments of the URL** and has a priority of 9 and CustomRuleB specifies **Use the last 3 segments of the URL** and has a priority of 8, a page in which the URI has more than 3 segments will be named by CustomRuleA because it has a higher priority.

The default rule, if enabled, has a priority of 0.

For example, AppDynamics would use the TicketRule configured below to name the page in AppDynamics when the URL contains "getAllTickets". Otherwise it would use the default rule, or another rule that matched the url with a priority greater than 4 if such a rule exists.

Name	<input type="text" value="TicketRule"/>
Enabled	<input checked="" type="checkbox"/>
Priority	4
URL	Contains <input type="text" value="getAllTickets"/>
Show Protocol	<input type="checkbox"/>
Show Domain	<input type="checkbox"/> Show Full Domain <input checked="" type="radio"/> Show Subdomain
What part of the URL should be used in Page Names	<input type="radio"/> Use first <input type="text" value="2"/> segments <input type="radio"/> Use last <input type="text" value="1"/> segments <input checked="" type="radio"/> Use segment numbers <input type="text" value="1,3,4"/> <small>(comma separated list, starts at 1) Example</small>
	<input type="radio"/> Run regex on URI <input type="text"/>
	Pick indices from regex output <input type="text"/> <small>(comma separated list of indices from regex output, starting at 0.) Example</small>
Query String Parameters to use in Page Name	<input type="text"/>
Comma separated list of parameter names. The values will be used in Page Names.	
<input type="button" value="Cancel"/> <input type="button" value="OK"/>	

Custom Page Exclude Rules

You can configure custom exclude rules for pages. Any page with a URL matching the configuration is excluded from monitoring.

Custom Exclude Rule	
Name	<input type="text" value="ExcludeRule1"/>
Enabled	<input checked="" type="checkbox"/>
Priority	5
URL	Contains <input type="text" value="appdynamics"/>
<input type="button" value="Cancel"/> <input type="button" value="OK"/>	

Learn More

- [Configure End User Experience](#)
- [Page List](#)
- [Page Dashboard](#)
- [Configure Page Names of Any Format](#)

Configure JavaScript and Ajax Error Detection

- To access JavaScript and Ajax error detection configuration
- [Enabling and Disabling EUM Error Detection](#)
- [Configuring Rules to Ignore Errors by Script or Error Message](#)
 - To configure Ignore JavaScript Error Rules by Script
- [Configuring Rules to Ignore Errors by Page](#)
 - To configure Ignore Error Rules by Page Name
- [Configuring Rules to Ignore Errors by URL](#)
 - To configure Ignore Error Rules by URL
- [Learn More](#)

You can enable and disable reporting of JavaScript and AJAX request errors.

You can configure which errors are included in the error count by specifying which error to "ignore". AppDynamics does not really

ignore "ignored errors". It still reports them, but does not increment the error count for them where error totals are reported.

When enabled, JavaScript and AJAX request errors are reported throughout the EUM UI: in the geo page, browser and device dashboards, in the page list, and in the browser snapshots.

You can specify errors to ignore:

- by script and / or error message
- by page
- by URL

To access JavaScript and Ajax error detection configuration

1. Access the EUM configuration screen if you are not already there. See [To access EUM Configuration](#).
2. Click the Page Naming, Error Detection, Thresholds, etc tab.
3. Expand **Configure Detection of JavaScript and AJAX Errors**.

Whenever you configure error detection, click **Save** to save the configuration.

Enabling and Disabling EUM Error Detection

In the EUM error detection configuration screen:

- To enable reporting of JavaScript errors, check the Enable JavaScript Error Capture check box.
- To stop reporting of JavaScript errors, clear the Enable JavaScript Error Capture check box.
- To enable reporting of Ajax errors, check the Enable Ajax Request Error Capture check box.
- To stop reporting of Ajax errors, clear the Enable Ajax Request Error Capture check box.

If both check boxes are clear, AppDynamics will not capture any JavaScript or Ajax request errors from the end-users' Web browsers.

If capture is enabled, you can configure certain errors to be ignored so that they are not counted in the error totals.

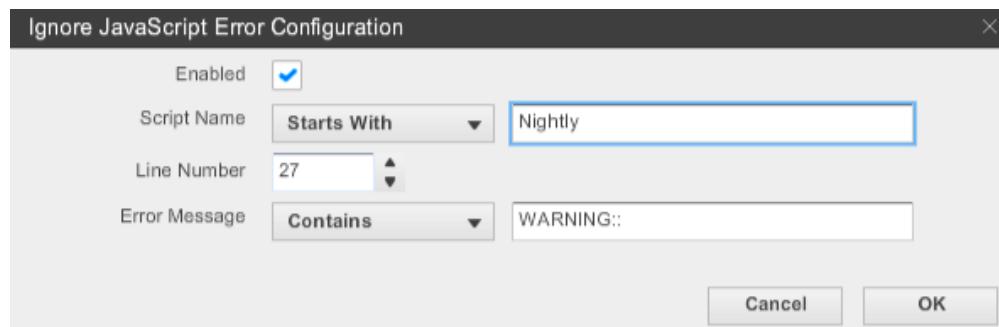
Configuring Rules to Ignore Errors by Script or Error Message

You can configure AppDynamics to ignore JavaScript errors that are identified by:

- a matching string pattern in the name of the script that generated the error
- line number in the script
- a matching string pattern in the error message

You can specify one, two or all three of these criteria. The more criteria you configure, the more specific is the error to ignore.

For example, the following configuration, in which all three fields are specified, means "Ignore all errors generated by line 27 of a script whose name starts with "Nightly" and whose error message contains the string "WARNING::".



If the line number were not specified (e.g. set to 0), the configuration would mean "Ignore all errors generated any line of a script whose name starts with "Nightly" and whose error message contains the string "WARNING::".

If neither the line number nor the error message field were specified, the configuration would mean "Ignore all errors generated by any line of a script whose name starts with "Nightly".

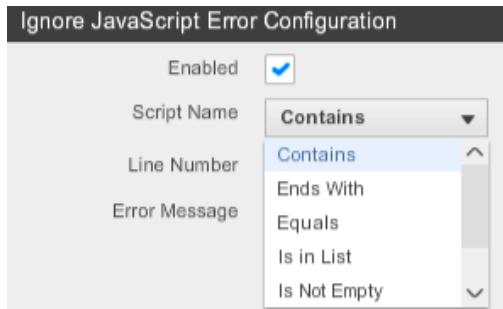
If the error message were the only field specified, the configuration would mean "Ignore all errors generated by any script when the error message contains the string "WARNING::".



To configure Ignore JavaScript Error Rules by Script

1. Access the configuration screen using the instructions in [To access JavaScript and Ajax error detection configuration](#).
2. In the Ignore JavaScript Error Rules section, click the add icon.
3. Check the Enabled check box to enable the ignore rule.
4. Do one or more of the following depending on how narrowly you want to define the rule:

- Use the dropdown menu to specify the script name of the script generating the error to ignore.



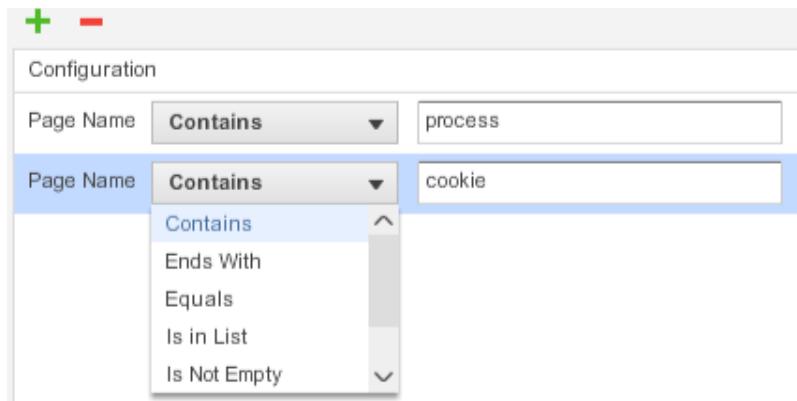
- Specify the line number in the script that generates the error to ignore.
- Specify a string in the error message of the error to ignore, using the dropdown menu.

5. Click **OK**.

To modify an existing ignore rule, select the rule in the list and click the edit icon.
To remove an ignore rule, select the rule in the list and click the delete icon.

Configuring Rules to Ignore Errors by Page

You can configure AppDynamics to ignore all errors generated by a specific page, Iframe, or Ajax request. Configure one rule for every page for which you want to ignore all errors.



To configure Ignore Error Rules by Page Name

1. Access the configuration screen using the instructions in [To access JavaScript and Ajax error detection configuration](#).
2. In the Ignore Page Name Rules section, click the add icon.
3. Use the dropdown menu to specify the page name of the page generating errors that you want to ignore.

Errors from the specified pages will not be counted in the error totals.

To remove an ignore rule, select it in the list and click the delete icon.

Configuring Rules to Ignore Errors by URL

You can configure AppDynamics to ignore all errors generated by a specific URL.
Configure one rule for every URL for which you want to ignore all errors.

To configure Ignore Error Rules by URL

1. Access the configuration screen using the instructions in [To access JavaScript and Ajax error detection configuration](#).
2. In the Ignore URL Rules section, click the add icon.
3. Use the dropdown menu to specify the URL of the page generating errors that you want to ignore.

Errors from the specified URLs will not be counted in the error totals.

To remove an ignore rule, select it in the list and click the delete icon.

Learn More

- [Configure End User Experience](#)
- [Browser Snapshots](#)

Configure EUM Thresholds

- To access the EUM threshold settings:
- [Learn More](#)

You can configure the thresholds that define slow, very slow, and stalled end-user requests for browser snapshots.

You can define EUM thresholds either

- as a multiple of the standard deviation; for example, "Experience is slow if end user response time is slower than 3 X the standard deviation."
- as a static value; for example, "Experience is stalled if end user response time is slower than 30000 ms."

The default thresholds are:

- Slow = 3 x standard deviation
- Very Slow = 4 x standard deviation

- Stalled = 300000 ms

To access the EUM threshold settings:

1. Access the EUM configuration screen if you are not already there. See [To access EUM Configuration](#).
2. Click the Page Naming, Error Detection, Thresholds etc tab.
3. Expand **Thresholds for Slow End User Experience**.

▼ **Thresholds for Slow End User Experience**

Configure User Experience Thresholds for Browser Snapshots.

Slow Threshold	Standard Deviation	3
Very Slow Threshold	Standard Deviation	4
Stall Threshold	Static (ms)	30000

4. Do one or more of the following:

- Set the Slow Threshold.
- Set the Very Slow Threshold.
- Set the Stalled threshold.

Use the dropdown menu to indicate whether the threshold is based on the standard deviation or a static value.

Type the values in the fields or select them using the scrollbars.

5. Click **Save**.

Learn More

- [Browser Snapshots](#)

Configure Browser Snapshot Collection

- [Learn More](#)

By default, when EUM is enabled, the JavaScript agent for EUM captures browser snapshots every 60 seconds.

You can:

- Enable and disable periodic snapshot collection.
- Enable snapshots to be captured whenever JavaScript or Ajax requests receive HTTP error responses. These are responses with an HTTP code equal to or greater than 400.
- Enable and disable slow snapshot collection. When slow snapshot collection is enabled, AppDynamics starts capturing browser snapshots whenever the End User Response Time is higher than the configured threshold.



Slow snapshot collection is useful for troubleshooting problems in development and quality assurance but not recommended for monitoring applications in production.

If periodic collection and error collection and slow collection are all disabled, the agent does not collect any browser snapshots.

To access the snapshot collection settings:

1. Access the EUM configuration screen if you are not already there. See [To access EUM Configuration](#).
2. Click the Page Naming, Error Detection, Thresholds etc tab.
3. Expand **Event Policy Configuration**.

▼ Event Policy Configuration

Enable Slow Snapshot Collection

Collect Snapshots when End User Response Time is higher than the threshold

Enable Periodic Snapshot Collection

Collect a normal Snapshot every minute

Enable Error Snapshot Collection

Collect Snapshots for Pages and IFrames that have JavaScript Errors and AJAX Requests that have HTTP Errors

4. Do one or more of the following:

- To enable periodic collection check the Enable Periodic Snapshot Collection check box.
- To disable periodic collection clear the Enable Periodic Snapshot Collection check box.
- To enable error collection check the Enable Error Snapshot Collection check box.
- To disable error collection clear the Enable Error Snapshot Collection check box.
- To enable slow collection check the Enable Slow Snapshot Collection check box.
- To disable slow collection clear the Slow Periodic Snapshot Collection check box.

5. Click **Save**.

Learn More

- Browser Snapshots
- Configure End User Experience

Customize Your EUM Deployment

- Configuring Custom Deployments
 - To access optional custom configuration
- Alternate Geo Server Location
 - Download the Geo Server File
 - Configure the Geo Server Location
 - Create the IP Mapping File
 - Set Properties in web.xml
- Alternate JavaScript Agent for EUM Extension Location
- Alternate EUM Data Collector Location
- Learn More

Deployment customizations include:

- alternate geo server location
- alternate JavaScript agent extension location
- alternate EUM data collector location

Configuring Custom Deployments

To access optional custom configuration

1. Access the EUM configuration screen if you are not already there. See [To access EUM Configuration](#).
2. Click the JavaScript Instrumentation tab.
3. Expand **Advanced**.
4. Expand **Customize your Deployment**.

Alternate Geo Server Location

By default, end-users' locations are resolved using public geographic databases. You can host an alternate geo server for your countries, regions, and cities instead of using the default geo server hosted by AppDynamics.

The most common reasons for doing this are:

- You have an intranet application for which you do not want transactions to reference a URL outside your organization's network.
- The accessing browsers are coming from many locations through a single gateway, in which case a single IP address appears to the AppDynamics default geo server, hiding the actual locations of your end users.

To host a custom geo server:

1. Download the Geo Server File
2. Configure the Geo Server location
3. Create the IP Mapping File
4. Set Properties in web.xml

Download the Geo Server File

Download the GeoServer-2.0.zip file from AppDynamics at

<http://download.appdynamics.com/onpremise/public/latest/GeoServer.zip>

This file contains:

- a geo.war
- local-map.xml

Deploy geo.war in a web container.

Configure the Geo Server Location

Enter URL, including the context root, of your hosted geo server in the Geo Server URL field in the configuration screen. In the following configuration the context root is "/geo".

The screenshot shows a configuration interface with a text input field labeled "Geo Server URL (Optional)". The field contains the URL "http(s):// mygeo.acme.com/geo". To the right of the input field is a blue circular icon with a white question mark inside.

Create the IP Mapping File

The local-map.xml IP mapping file specifies the locations for which EUM provides geographic data. It maps IP addresses to geographic locations.

Edit the local-map.xml, which was downloaded with the geo.war file, for your environment. This file contains a <location> element for every location to be monitored. The file has the following format.

```
<config>
    <location network="239.0.64.0" subnet-mask="255.255.192.0">
        <country>United States of America</country>
        <region>California</region>
        <city>Mountain View</city>
    </location>
    ... more location entries
</config>
```

The <country>, <region> and <city> elements are required. If the values of <country>, <region> do not correspond to an actual geographic location in the geographic database, map support is not available for the location in the EUM map panel, but EUM metrics are displayed for the location in grid view of the geographic distribution, end user response time panel, trend graphs, browser distribution panel, and in the Metric Browser. The <city> element can be a string that represents the static location of the end-user.

This data is visible in browser snapshots and can be used to filter browser snapshots and to filter browser snapshots for specific locations:

The valid names for country and region are those used in the map in the geo dashboard. You can hover over a region in the dashboard to see the exact name (including spelling and case) of the region. See [Geo Dashboard](#).

Set Properties in web.xml

In the web.xml file, set the ip.mapping.config property to the path of the IP mapping file. The web.xml file is in the geo.war. You can also set the log directory for the geo server and the number of seconds that geo data should be cached,

Add the mapping information as follows:

```

<init-param>
    <param-name>logs.dir</param-name>
    <param-value>/opt/geo/logs</param-value>
</init-param>
<init-param>
    <param-name>ip.mapping.config</param-name>
    <param-value>/opt/geo/local-map.xml</param-value>
</init-param>
<init-param>
    <param-name>response.cache.seconds</param-name>
    <!-- Default is 1 day. Caching geo info longer than that is bad for mobile
devices. -->
    <param-value>86400</param-value>
</init-param>
```

This example assumes that you are using a modified local-map.xml file. If you created a new mapping file instead, use the name of that file in the <param-value> element instead of "local-map.xml" for the ip.mapping.config property.

Alternate JavaScript Agent for EUM Extension Location

AppDynamics hosts the JavaScript agent extension on the highly available Amazon CloudFront CDN infrastructure.

To host the JavaScript agent extension, download it yourself, download it from the download link in the Customize Your Deployment section of the configuration screen. You will get a version that is compatible with your version of the Controller.

Geo Server URL (Optional)	http(s):// (no default)
JavaScript Agent Extension URL (Optional)	http(s):// s3-us-west-1.amazonaws.com/jsagent-trunk Download adrum-ext.[version].js
EUM Data Collector URL (Optional)	http(s):// test1-828673714.us-east-1.elb.amazonaws.com
Additional information	

Save it in a Web container and enter the URL of the host in the JavaScript Agent Extension URL field. If you saved the agent file in a directory, for example "js", include the directory name but do not include the name of the actual agent extension file as this may change with subsequent versions. AppDynamics will supply the name of the file when it processes the URL.

JavaScript Agent Extension URL (Optional)	http(s):// ec2-184-72-146-37.compute-1.amazonaws.com/js	?
---	---	-------------------

Alternate EUM Data Collector Location

The AppDynamics JavaScript agent for EUM reports browser performance data to an EUM data collector. The default data collector stores the data in the Amazon US-WEST Region. Depending on the privacy laws of your country, you may choose a different instance of the EUM data collector.

Contact your AppDynamics sales representative or AppDynamics Support for an appropriate URL for an alternate data collector.

Learn More

- [Inject the JavaScript Agent for EUM](#)
- [AppDynamics Support](#)

Add Information to a Browser Snapshot

- [Add User Data](#)
- [Modify User Data Size Limit](#)

You can add user information to a browser snapshot. The information appears in a User Data section of the snapshot.

For example you can get the AWS zone or type of node from which the end-user request was served.

Add User Data

To add user data, add the following method to the pages for which you want the additional data to appear in the browser snapshots.

```
<static> ADRUM.commands ("addUserData", key, value)
```

The results appear in the browser snapshot in a special User Data section.

A screenshot of a browser snapshot interface. At the top, there's a header bar with the title 'Client Request GUID' and the value '069e5f91-9e23-4200-aa50-b471885420a1'. Below this, there's a section titled 'User Data' with two columns: 'keys' and 'values'. The 'keys' column contains 'algo', 'algo', and 'card'. The 'values' column contains 'LOG(N)', 'LOG(N)', and 'silver'. The entire interface has a torn paper effect at the bottom.

User Data	keys	values
	algo	LOG(N)
	algo	LOG(N)
	card	silver

Modify User Data Size Limit

The maximum size of all user data in a page is 100 bytes, unless you increase the limit using the `setMaxBeaconLength()` method.

```
<static> setMaxBeaconLength(nbytes)
```

You can set the user data size as high as 2000 bytes. Some browsers will not send packets larger than this, so increasing this value may cause data to be dropped.

If you modify the user data size, the amount of space allocated to all user data fields scales uniformly relative to their default sizes.

Configure Page Names of Any Format

In the AppDynamics console, you can configure the names of pages, iFrames and Ajax requests based on various parts of the page URL. See [Configure Page Identification and Naming](#).

To use any arbitrary string to name a page, not necessarily some part of the URL, add the `setPageName` method to the page that you want to name.

```
<static> ADRUM.commands ("setPageName", name)
```

The default page name is the DOM document title.

Handle the `window.onerror` Event

If any script on your monitored Web pages, including library code, sets the JavaScript `window.onerror` event, add the following method to the page immediately after setting `window.onerror`:

```
<static> ADRUM.listenForErrors()
```

The JavaScript agent for EUM (ADRUM) sets `window.onerror` to listen for uncaught JavaScript errors. If this listener is overwritten, errors will not be reported.

ADRUM will invoke your original `onerror` handler.

Inject the JavaScript Agent for EUM

The JavaScript Agent for EUM allows you to capture EUM metrics and browser snapshots for any web application.

It also can correlate browser snapshots with associated server-side business transaction data for applications that are instrumented by AppDynamics Java and .NET app agents.

Injection Overview

- [Summary of Injection Techniques](#)
- [Inject to Capture EUM Metrics and Browser Snapshots](#)
- [Inject to Get Server-Side Correlation](#)
 - [Getting Full Timing Data for Associated Business Transactions](#)
 - [Learn More](#)

Summary of Injection Techniques

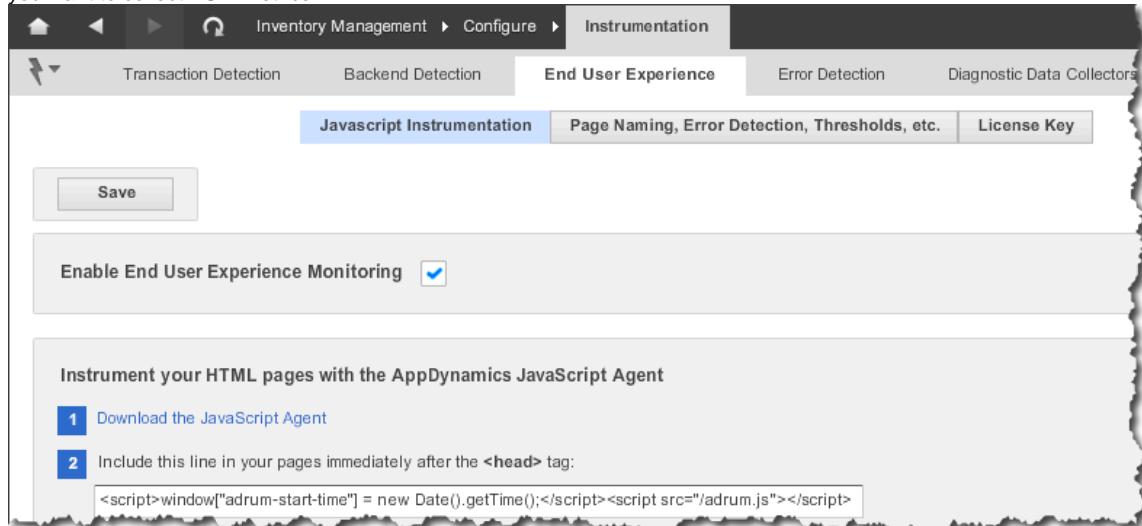
The JavaScript Agent for EUM must be injected into all of the web pages for which you want to capture EUM information.

There are several ways to inject the JavaScript Agent for EUM into your web pages. Not all types of injection are supported on all frameworks.

See the Script Injection columns in the End User Monitoring (EUM) Compatibility matrices for information about platforms that support the various types of injection.

- **Manual Injection**

For manual injection, you download the JavaScript Agent for EUM and include it in the head section of the web pages for which you want to collect EUM metrics.



However, this procedure only injects into the head, so it is not guaranteed to return the total execution time for business transactions associated with browser snapshots.

Manual injection is available for all platforms.

- **Automatic Injection**

Automatic injection configuration directs AppDynamics to inject the JavaScript Agent for EUM into the headers and footers on the web pages that you specify for the business transactions that you specify.

Automatic injection is available only for applications that are deployed on Application Servers based on the Apache Jasper JSP compiler for Java platforms or ASP.NET and ASPX for .NET platforms. See the Script Injection column of the matrices in [End User Monitoring \(EUM\) Compatibility](#) to find out if automatic injection is supported for your environment. If it is, it is still possible that automatic injection may not work due to some distinctive characteristics of your web pages. Always try automatic injection thoroughly in a test environment before you use it in production.

For .NET, see also [Configure the .NET App Agent for EUM](#) as configuring the app agent is required to support automatic injection on .NET.

- **Assisted Injection-Using Injection Rules (Java Only)**

Assisted injection using injection rules configuration directs AppDynamics to inject the JavaScript Agent for EUM into the headers and/or footers of pages that you designate through JavaScript injection rules that you create. The JavaScript injection rules specify when and where to inject the agent and the business transactions for which the injection is enabled.

Assisted injection using injection rules is available for Java platforms only.

- **Assisted Injection-Using Attribute Injection**

When enabled through the configuration, the JavaScript Agent for EUM populates attributes in the java.servlet.HttpServletRequest with the HTML to be output in the head section of the page and at the end of the body section.

▼ Advanced Instrumentation of your HTML Pages

Automatic JavaScript Injection Configure JavaScript Injection

▼ Create Injection Rules

Name

Request Attribute Injection [?](#)

See [Injecting code into the request object](#) for examples of the HTML.

Attribute injection using attribute injection is available only for applications built on Java servlet or ASP.NET.

Inject to Capture EUM Metrics and Browser Snapshots

The JavaScript Agent for EUM must be injected into the head sections of the web page for which you want to capture EUM metrics. You can inject the JavaScript Agent for EUM into the page using the following techniques:

- Manual Injection
- Automatic Injection
- Assisted Injection - Using Injection Rules
- Assisted Injection - Using Attribute Injection

Inject to Get Server-Side Correlation

For servers that are instrumented by an AppDynamics Java or .NET app agent, you can get correlation of browser snapshots with business transaction information on the server side for business transactions that are associated with browser snapshots. When such an association exists, this correlated data appears in the Server Side tab of a browser snapshot:

- business transaction information
In the Business Transactions section of the Server Side tab is the name and the time of business transactions associated with the browser snapshot.
- snapshot linking
In the Transaction Snapshots section of the Server Side tab is the list of transaction snapshots that are associated with the browser snapshot, if any transaction snapshots were taken at that time.
This gives you a continuous view of the user's experience at a certain point in time from the browser through to the backend via the transaction snapshot.

See [Browser Snapshots](#) and [Transaction Snapshots](#) for information about these types of snapshots.

Getting Full Timing Data for Associated Business Transactions

To ensure that you get the total execution time of a business transaction associated with a browser snapshot, it is necessary to inject the JavaScript Agent for EUM into the footers of the web pages as well as the headers.

You can inject the JavaScript Agent for EUM into the footer of a web page using the following techniques:

- If you use automatic injection for the injecting into the head section, you automatically get injection into the footer also.
- If you use manual injection for the head section, for applications built on Java platforms you can use assisted injection-using injection rules to inject into the footer. Or for applications built on Java servlet or ASP.NET platforms, you can use assisted

injection-using attribute injection.

Learn More

- Manual Injection
- Automatic Injection
- Configure the .NET App Agent for EUM
- Assisted Injection-Using Injection Rules (Java Only)
- Assisted Injection-Using Attribute Injection
- Selecting a JavaScript for EUM Instrumentation Strategy

Manual Injection

- Download and Include the Agent
 - To access the manual injection panel
 - To inject the JavaScript Agent for EUM
- Learn More

For manual injection, you download the JavaScript Agent for EUM and include it in the head section of the web pages for which you want to collect EUM data.

However, this procedure only injects into the header, so it does not guarantee complete business transaction timing information when browser snapshots are correlated with server-side business transactions.

Download and Include the Agent

You configure manual injection from the JavaScript Instrumentation tab of the EUM configuration screen.

To access the manual injection panel

1. In the left navigation menu, click **Configure -> Instrumentation**.
2. Click the End User Experience tab.
3. Click the JavaScript Instrumentation subtab if it is not already selected.
4. Scroll down to the Instrument your HTML pages with the AppDynamics JavaScript Agent panel.

To inject the JavaScript Agent for EUM

1. Click **Download the JavaScript Agent**.

2. Click **Save to File** to save it.

The name of the saved file should be "adrum.js". Save it in the root directory of the page into which you are injecting.

3. To include the JavaScript Agent for EUM in your page, copy the line in the text field in the second step and paste it into the <head> element of the pages that you want to monitor.

You will get EUM metrics for all pages in which you include this line. If you later decide that you do not want metrics for the page, remove the line.

4. Click **Save** in the configuration screen.

Instrument your HTML pages with the AppDynamics JavaScript Agent

- 1 Download the JavaScript Agent
- 2 Include this line in your pages immediately after the <head> tag:

```
<script>window["adrum-start-time"] = new Date().getTime();</script><script src="/adrum.js"></script>
```

This inclusion is highly preferable, for convenience, accuracy and maintenance, to copying the entire JavaScript agent into your web

pages inline.



The JavaScript for EUM agent is named `adrum.js`. This script invokes another script called `adrum-ext`, which performs most of the EUM logic. The `adrum-ext` script is hosted on Amazon CDN, but you have the option of hosting it at another location. See [Alternate Location for the JavaScript for EUM Agent](#) for information about configuring this option.

If you are using the same JavaScript agent for EUM (`adrum.js`) for multiple applications, add the following line before the line that includes the agent (`adrum.js`):

```
window[ "adrum-app-key" ];
```

This is necessary to override the application key set in the AppDynamics Controller.

Learn More

- [Injection Overview](#)
- [Configure End User Experience](#)
- [Inject the JavaScript Agent for EUM](#)
- [Automatic Injection](#)
- [Assisted Injection-Using Injection Rules \(Java Only\)](#)
- [Assisted Injection-Using Attribute Injection](#)

Automatic Injection

- To access the automatic injection configuration panel
- [Enable Automatic Injection](#)
 - To enable or disable automatic injection
- [Configuring Automatic Injection](#)
 - To Specify Business Transactions for Automatic Injection
 - To Create Match Rules for Automatic Injection
- [Learn More](#)

One reason to use automatic injection is to inject into the footer of your web pages to capture comprehensive business transaction timing for every end-user request for which there is an associated business transaction on an instrumented server. Automatic injection is one way to inject into the footer.

Another reason to use automatic injection is if you have a large number of web pages to instrument. Automatic injection saves you the trouble of injecting each web page.

Automatic injection is available only for applications built on a Jasper-supported JSP (Java) or ASP.NET or ASPX (.NET) framework. If you plan to use automatic injection and desire correlation with instrumented .NET nodes on the server side, the .NET app agent must be enabled for EUM. See [Configure the .NET App Agent for EUM](#).



Warning: If you configure automatic injection, AppDynamics *strongly* advises you run some traffic through your application in a *test environment* to verify that EUM works and that it is not interfering with your HTML *before you use it in production*.

To access the automatic injection configuration panel

1. In the left navigation menu, click **Configure -> Instrumentation**.
2. Click the End User Experience tab.
3. Click the JavaScript Instrumentation subtab if it is not already selected.
4. Scroll down to the Advanced panel and expand it if it is closed.
5. Expand **Advanced Instrumentation of your HTML Pages** if it is closed.
6. Click the Automatic JavaScript Injection subtab if it is not already selected.

▼ Advanced

► Customize your Deployment

▼ Advanced Instrumentation of your HTML Pages

Automatic JavaScript Injection

Configure JavaScript Injection

Enable Automatic Injection of JavaScript ?

Automatic injection must be enabled for specific Business Transactions. To enable automatic injection, click "Refresh List" and you should see the eligible Business Transactions.

Enable Automatic Injection

You configure automatic injection from the Advanced Instrumentation of your HTML Pages EUM configuration screen. See [To access the automatic injection configuration panel](#).

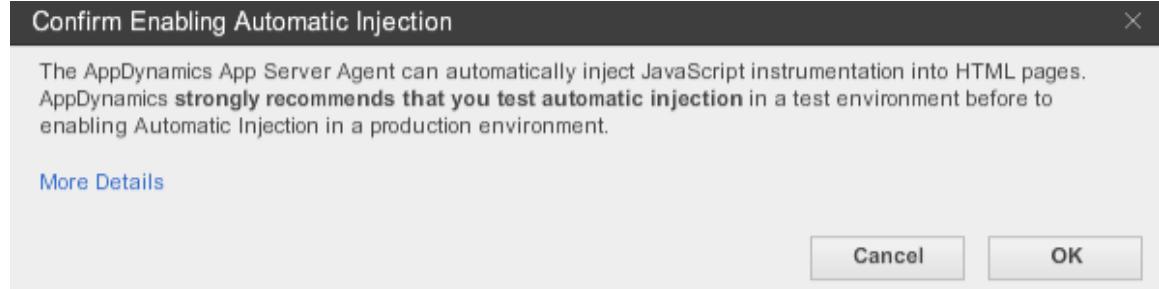
To enable or disable automatic injection

In the Automatic JavaScript Injection subtab:

1. Do one of the following:

- To enable automatic injection, check the Enable Automatic Injection of JavaScript check box.
- To disable automatic injection, clear the Enable Automatic Injection of JavaScript check box.

2. If you enabled automatic injection click **OK** to confirm your action.



Configuring Automatic Injection

After you have enabled automatic injection:

- You can specify the business transactions for which automatic JavaScript injection is enabled.
- You can qualify which pages to inject, by creating custom match and exclude rules for automatic injection. If you do not configure these rules, by default AppDynamics injects all pages visited by the enabled business transactions.

Use these rules to fine-tune which business transaction to include or exclude from injection based on match criteria. For example, you can exclude all business transactions that have a certain string in their URLs or servlet names or with a certain cookie. The configurations for include rules and exclude rules are similar. It depends on your situation whether it is more convenient to restrict transactions based on inclusion or exclusion.

To Specify Business Transactions for Automatic Injection

1. Access the Automatic JavaScript Injection subtab in the configuration screen. See [To access the automatic injection configuration panel](#) if you are not there.

If automatic injection is enabled, the lists of business transactions for which automatic injection is enabled and disabled are displayed.

You can filter the lists by entering a string to filter on in the field above the lists.
It is possible that not all your business transactions will appear in these lists. The lists contain only the business transactions that AppDynamics can parse for automatic injection, those which are Jasper-compiled JSPs or ASPX pages.

2. Specify the business transactions for which injection is automatically enabled by moving them from the Automatic injection possible but not enabled list to the Automatic injection enabled list.
3. Specify the business transactions for which EUM is disabled by moving them from the Automatic injection list to the Automatic injection possible but not enabled list.
4. Click **Save** in the configuration screen..

To Create Match Rules for Automatic Injection

1. Access the Automatic JavaScript Injection subtab in the configuration screen. See [To access the automatic injection configuration panel if you are not there](#).
2. Expand **Only enable Automatic Injection for certain Pages** if it is closed.
3. To create rules that define which pages to include for automatic injection, click the Add icon in the Request Match Rules section. To create rules that define which pages to exclude from automatic injection, click the Add icon in the Request Exclude Rules section.

You can later remove these rules by selecting the rule and clicking the delete icon. You can modify them by selecting the edit icon.

4. In the Request Match Rule or Request Exclude Rule screen:

- a. Check the check box next to the criterion to use to include or exclude a request from JavaScript injection.
- b. Configure the match condition for that criterion.

See [Match Rule Conditions](#) for general information about match rules.

Repeat this step for all the criteria that you wish to configure. You can configure multiple criteria, all of which would have to match to qualify the page for inclusion in or exclusion from for injection.

The screenshot shows the 'Create HTTP Request Match Rule' dialog box. The 'Match Criteria' section contains several fields for defining match rules. The 'URI' field is currently selected, showing 'Contains' as the condition and 'checkout' as the value. Other fields like 'Method', 'Header', and 'Cookie' also have dropdown menus and input fields for defining match conditions. At the bottom right of the dialog are 'Cancel' and 'Save' buttons.

c. Click **Save**.

5. Click **Save** in the outer configuration screen.

You can later edit or remove a match rule by selecting it in the list and clicking the edit or delete icon.

Learn More

- Injection Overview
- Configure the .NET App Agent for EUM
- Configure End User Experience
- Inject the JavaScript Agent for EUM
- Manual Injection
- Assisted Injection-Using Injection Rules (Java Only)
- Assisted Injection-Using Attribute Injection

Assisted Injection-Using Injection Rules (Java Only)

- To access the JavaScript injection rules configuration panel
- To create JavaScript injection rules
- Learn More

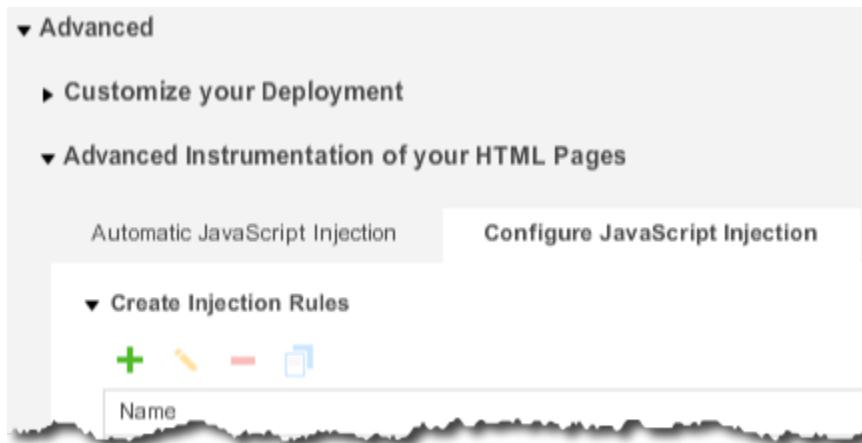
Assisted injection using injection rules configuration directs AppDynamics to inject the JavaScript Agent for EUM into the headers and/or footers of pages that you designate.

You configure JavaScript injection rules that tell AppDynamics when and where to inject the JavaScript and for which the business transactions the injection is enabled.

Assisted injection is available for Java frameworks only.

To access the JavaScript injection rules configuration panel

1. In the left navigation menu, click **Configure -> Instrumentation**.
2. Click the End User Experience tab.
3. Click the JavaScript Instrumentation subtab if it is not already selected.
4. Scroll down to the Advanced panel and expand it if it is closed.
5. Expand **Advanced Instrumentation of your HTML Pages** if it is closed.
6. Click the Configure JavaScript Injection Rules subtab if it is not already selected.



To create JavaScript injection rules

1. In the Configure JavaScript Injection Rules tab, expand **Create Injection Rules** if it is closed.
2. Click the add icon.

The Create Injection Rule screen is displayed.

3. Click the Where to Inject JavaScript tab
4. In the Name field, enter a name for the rule.
5. To enable the rule, check the Enabled check box.
To disable the rule, clear the Enabled check box.
6. In the Class and method to intercept section, define the match conditions for the class and method that write to the output stream. This is the class that AppDynamics intercepts for injection.
7. If the write method is overloaded:
 - a. Check the Is this Method Overloaded? check box.
 - b. Click **Add Parameter**.
 - c. Add the parameters that define the method.
8. In the Pointer to the writer section, select how to obtain the output writer stream in the intercepted method.
9. In the Injection options section, specify:
 - the output stream write method AppDynamics should use to inject
 - when to inject: when the method begins or when the method ends
 - where to inject: before the header or after the footer; to capture full business transaction timing information inject after the footer
 - optional prefix to output before writing the header or footer, such as <DOCTYPE. . .>
10. Optionally configure the business transactions for which the rule is enabled. By default the rule is enabled for all business transactions. To enable it for specific business transactions only:
 - a. Click the Inject for these Business Transactions tab.
 - b. Select These Business Transactions.
 - c. Specify the business transactions for which the injection rule is enabled by moving them from the Other Business Transactions list to the Selected Business Transactions list.
 - d. Specify the business transactions for which the injection rule is disabled by moving them to (or leaving them in) the Other Business Transactions list.
11. Click **Create Injection rule**.
12. Click **Save**.

You can later edit or remove an injection rule by selecting it in the list and clicking the edit or delete icon.

The following sample injection rule configures injection in the footer at the end of the intercepted method

Name: zk footer
Enabled:

Class and method to intercept: Class with a Class Name that equals org.zkoss.zk.ui.sys.HtmlPageRenderers
Method Name: outPageContent Is this Method Overloaded?

Method Parameters (optional): Add Parameter

Pointer to the writer: Object containing the writer: Method Parameter @ Index: 2

 Return Value
 Invoked Object
 Use Getter Chain: this
 for example: getResponse().getWriter()
[more help](#)

Injection options: Output Stream Write Method: write
 Inject on: method begin method end
 Inject: eum header eum footer
 Prefix: ?
 Cancel [Create Injection Rule](#)

Learn More

- [Injection Overview](#)
- [Configure End User Experience](#)
- [Inject the JavaScript Agent for EUM](#)
- [Manual Injection](#)
- [Automatic Injection](#)
- [Assisted Injection-Using Attribute Injection](#)

Assisted Injection-Using Attribute Injection

- [Enabling Attribute Injection](#)
 - To enable attribute injection
- [Injecting code into the request object](#)
- [Learn More](#)

There are two steps required to set up attribute injection:

- enable attribute injection in AppDynamics
- inject the code

Enabling Attribute Injection

To enable attribute injection

1. In the left navigation menu, click **Configure -> Instrumentation**.
2. Click the End User Experience tab.
3. Click the JavaScript Instrumentation subtab if it is not already selected.
4. Scroll down to the **Advanced** panel and expand it if it is closed.

5. Expand **Advanced Instrumentation of your HTML Pages** if it is closed.
6. Click the Configure JavaScript Injection Rules subtab if it is not already selected.
7. Check the Request Attribute Injection check box.

The screenshot shows a user interface for configuring JavaScript injection rules. At the top, there's a navigation bar with tabs: 'Automatic JavaScript Injection' and 'Configure JavaScript Injection'. Under 'Configure JavaScript Injection', there's a section titled 'Create Injection Rules'. Within this section, there is a checkbox labeled 'Request Attribute Injection' which is checked, indicated by a blue checkmark icon. A question mark icon is also present next to the checkbox.

8. Click **Save**.

Injecting code into the request object

If attribute injection is enabled, you can inject the AppDynamics_JS_HEADER and AppDynamics_JS_FOOTER objects into the request object. These objects can be accessed in your pages to place the header and footer where they are appropriate.

⚠ Warning: Be careful about where you inject your JavaScript, as it is possible to break your HTML pages. AppDynamics recommends that you always inject in the <head> element of the HTML and in the footer just before the closing </body> tag.

The following examples show code snippets that can be directly injected into the page:

Manual JSF

```
<h:outputText rendered="#{AppDynamics_JS_HEADER != null}"  
value="#{request.getAttribute("AppDynamics_JS_HEADER")}" escape="false"/>  
<h:outputText rendered="#{AppDynamics_JS_FOOTER != null}"  
value="#{request.getAttribute("AppDynamics_JS_FOOTER")}" escape="false"/>
```

Manual JSP

```
<% if (request.getAttribute("AppDynamics_JS_HEADER") != null) { %>  
<%=request.getAttribute("AppDynamics_JS_HEADER")%> <% } %>  
<% if (request.getAttribute("AppDynamics_JS_FOOTER") != null) { %>  
<%=request.getAttribute("AppDynamics_JS_FOOTER")%> <% } %>
```

Manual Servlet

```

if (request.getAttribute("AppDynamics_JS_HEADER") != null)
{
    out.write(request.getAttribute("AppDynamics_JS_HEADER"));
}
if (request.getAttribute("AppDynamics_JS_FOOTER") != null)
{
    out.write(request.getAttribute("AppDynamics_JS_FOOTER").toString());
}

```

Manual Groovy

```

<g:if test="${AppDynamics_JS_HEADER}">
    ${AppDynamics_JS_HEADER}
</g:if>

<g:if test="${AppDynamics_JS_FOOTER}">
    ${AppDynamics_JS_FOOTER}
</g:if>

```

Manual Velocity Template

```

#if ($AppDynamics_JS_HEADER)
    $AppDynamics_JS_HEADER
#end
#if ($AppDynamics_JS_FOOTER)
    $AppDynamics_JS_FOOTER
#end

```

Manual ASP.NET

```

<% if (Context.Items.Contains("AppDynamics_JS_HEADER"))
    Response.Write(Context.Items["AppDynamics_JS_HEADER"]); %>
<% if (Context.Items.Contains("AppDynamics_JS_FOOTER"))
    Response.Write(Context.Items["AppDynamics_JS_FOOTER"]); %>

```

Manual MVC Razor

```

@if(HttpContext.Current.Items.Contains("AppDynamics_JS_HEADER"))
{ @Html.Raw((string)HttpContext.Current.Items["AppDynamics_JS_HEADER"]) }

@if(HttpContext.Current.Items.Contains("AppDynamics_JS_FOOTER"))
{ @Html.Raw((string)HttpContext.Current.Items["AppDynamics_JS_FOOTER"]) }

```

[Learn More](#)

- Injection Overview
- Configure End User Experience
- Inject the JavaScript Agent for EUM
- Manual Injection
- Automatic Injection
- Assisted Injection-Using Injection Rules (Java Only)

Selecting a JavaScript for EUM Instrumentation Strategy

- Choosing an Injection Method
- Verifying Injection
- Reversing Injection
- Learn More

Injection into the head section of a web page is required to get EUM data for that page. Injection into the footer is optional. See [Getting Full Timing Data for Associated Business Transactions](#) for reasons for injecting into the footers.

Choosing an Injection Method

If you do not have an obvious preference for injection based on your platform and your requirements, and you do not care about getting the entire business transaction timing information in your browser snapshots, use manual injection to inject into the head section.

Automatic injection requires the least amount of effort because you do not have to instrument every page; however, you must test automatic injection thoroughly before using it in production. Check the matrices at [End User Monitoring \(EUM\) Compatibility](#) to see if automatic injection has been tested in your environment. Note that even if a particular script injection type has been tested on a particular platform, it still may not work for you because of some characteristics of your web pages.

If automatic injection is not an option and you want total business transaction timing data, the you can use manual injection to inject into the head section and assisted injection to inject into the footer. Another alternative is to use assisted injection to inject into both the headers and the footers.

Verifying Injection

If you configured manual injection and are not seeing EUM metrics after running some load for a while, check the web page to confirm that the JavaScript Agent for EUM is in the page. If it is not, try adding it again one time.

If after two attempts you still do not see EUM metrics, try one of the other injection schemes if they are available for your platform, or call AppDynamics Support.

Reversing Injection

If you try one way to inject and it does not work, AppDynamics recommends that you undo the current injection configuration before implementing another one.

- To undo automatic injection, just clear the Enable Automatic Injection of JavaScript check box.
- To undo manual and assisted injection using attribute injection, manually delete the JavaScript Agent for EUM from your web pages.
- To undo assisted injection using injection rules, clear the Enable check box for each injection rule in the injection rules list.

If multiple copies of the agent exist on a page, the second copy does not execute.

Learn More

- [Inject the JavaScript Agent for EUM](#)

EUM Supported Countries and Regions

EUM Supported Countries

AppDynamics reports EUM data from all of the following countries. See [EUM Supported Regions](#) for the regions supported within each countries. Some countries do not have regions.

A - D

Afghanistan
Aland Islands
Albania
Algeria
American Samoa
Andorra
Angola
Anguilla
Antarctica
Antigua and Barbuda
Argentina
Armenia
Aruba
Asia/Pacific Region
Australia
Austria
Azerbaijan
Bahamas
Bahrain
Bangladesh
Barbados
Belarus
Belgium
Belize
Benin
Bermuda
Bhutan
Bolivia
Bonaire, Saint Eustatius and Saba
Bosnia and Herzegovina
Botswana
Bouvet Island
Brazil
British Indian Ocean Territory
Brunei Darussalam
Bulgaria
Burkina Faso
Burundi
Cambodia
Cameroon
Canada
Cape Verde
Cayman Islands
Central African Republic
Chad
Chile
China
Christmas Island
Cocos (Keeling) Islands
Colombia
Comoros
Congo
Congo, The Democratic Republic of the
Cook Islands
Costa Rica
Cote d'Ivoire
Croatia
Cuba
Curacao
Cyprus
Czech Republic
Denmark
Djibouti
Dominica
Dominican Republic

E - K

Ecuador

Egypt
El Salvador
Equatorial Guinea
Eritrea
Estonia
Ethiopia
Europe
Falkland Islands (Malvinas)
Faroe Islands
Fiji
Finland
France
French Guiana
French Polynesia
French Southern Territories
Gabon
Gambia
Georgia
Germany
Ghana
Gibraltar
Greece
Greenland
Grenada
Guadeloupe
Guam
Guatemala
Guernsey
Guinea
Guinea-Bissau
Guyana
Haiti
Heard Island and McDonald Islands
Holy See (Vatican City State)
Honduras
Hong Kong
Hungary
Iceland
India
Indonesia
Iran, Islamic Republic of
Iraq
Ireland
Isle of Man
Israel
Italy
Jamaica
Japan
Jersey
Jordan
Kazakhstan
Kenya
Kiribati
Korea, Democratic People's Republic of
Korea, Republic of
Kuwait
Kyrgyzstan

L - Q

Lao People's Democratic Republic
Latvia
Lebanon
Lesotho
Liberia
Libyan Arab Jamahiriya
Liechtenstein
Lithuania
Luxembourg
Macao
Macedonia

Madagascar
Malawi
Malaysia
Maldives
Mali
Malta
Marshall Islands
Martinique
Mauritania
Mauritius
Mayotte
Mexico
Micronesia, Federated States of
Moldova, Republic of
Monaco
Mongolia
Montenegro
Montserrat
Morocco
Mozambique
Myanmar
Namibia
Nauru
Nepal
Netherlands
New Caledonia
New Zealand
Nicaragua
Niger
Nigeria
Niue
Norfolk Island
Northern Mariana Islands
Norway
Oman
Pakistan
Palau
Palestinian Territory
Panama
Papua New Guinea
Paraguay
Peru
Philippines
Pitcairn
Poland
Portugal
Puerto Rico
Qatar

R - Z

Reunion
Romania
Russian Federation
Rwanda
Saint Barthelemy
Saint Helena
Saint Kitts and Nevis
Saint Lucia
Saint Martin
Saint Pierre and Miquelon
Saint Vincent and the Grenadines
Samoa
San Marino
Sao Tome and Principe
Saudi Arabia
Senegal
Serbia
Seychelles
Sierra Leone
Singapore

Sint Maarten
Slovakia
Slovenia
Solomon Islands
Somalia
South Africa
South Georgia and the South Sandwich Islands
Spain
Sri Lanka
Sudan
Suriname
Svalbard and Jan Mayen
Swaziland
Sweden
Switzerland
Syrian Arab Republic
Taiwan
Tajikistan
Tanzania, United Republic of
Thailand
Timor-Leste
Togo
Tokelau
Tonga
Trinidad and Tobago
Tunisia
Turkey
Turkmenistan
Turks and Caicos Islands
Tuvalu
Uganda
Ukraine
United Arab Emirates
United Kingdom
United States
United States Minor Outlying Islands
Uruguay
Uzbekistan
Vanuatu
Venezuela
Vietnam
Virgin Islands, British
Virgin Islands, U.S.
Wallis and Futuna
Western Sahara
Yemen
Zambia
Zimbabwe

EUM Supported Regions

In the interactive EUM map, you can hover over the largest countries to see the supported regions in a particular country. See [EUM Supported Countries](#) for the list of countries.

However, AppDynamics reports EUM data from all of the following countries and regions, even when the regions are not displayed on the map. There may be additional countries and regions that are not on this list.

A - G

Afghanistan, Badakhshan
Afghanistan, Badghis
Afghanistan, Baghlan
Afghanistan, Balkh
Afghanistan, Bamian
Afghanistan, Daykundi
Afghanistan, Farah
Afghanistan, Faryab
Afghanistan, Ghazni
Afghanistan, Ghowr
Afghanistan, Helmand

Afghanistan, Herat
Afghanistan, Jowzjan
Afghanistan, Kabul
Afghanistan, Kandahar
Afghanistan, Kapisa
Afghanistan, Khowst
Afghanistan, Konar
Afghanistan, Kondoz
Afghanistan, Laghman
Afghanistan, Lowgar
Afghanistan, Nangarhar
Afghanistan, Nimruz
Afghanistan, Nurestan
Afghanistan, Oruzgan
Afghanistan, Paktia
Afghanistan, Paktika
Afghanistan, Panjshir
Afghanistan, Parvan
Afghanistan, Samangan
Afghanistan, Sar-e Pol
Afghanistan, Takhar
Afghanistan, Vardak
Afghanistan, Zabol
Albania, Berat
Albania, Diber
Albania, Durres
Albania, Elbasan
Albania, Fier
Albania, Gjirokaster
Albania, Korce
Albania, Kukes
Albania, Lezhe
Albania, Shkoder
Albania, Tirane
Albania, Vlore
Algeria, Adrar
Algeria, Ain Defla
Algeria, Ain Temouchent
Algeria, Alger
Algeria, Annaba
Algeria, Batna
Algeria, Bechar
Algeria, Bejaia
Algeria, Biskra
Algeria, Blida
Algeria, Bordj Bou Arreridj
Algeria, Bouira
Algeria, Boumerdes
Algeria, Chlef
Algeria, Constantine
Algeria, Djelfa
Algeria, El Bayadh
Algeria, El Oued
Algeria, El Tarf
Algeria, Ghardaia
Algeria, Guelma
Algeria, Illizi
Algeria, Jijel
Algeria, Khencelia
Algeria, Laghouat
Algeria, M'sila
Algeria, Mascara
Algeria, Medea
Algeria, Mila
Algeria, Mostaganem
Algeria, Naama
Algeria, Oran
Algeria, Ouargla
Algeria, Oum el Bouaghi
Algeria, Relizane
Algeria, Saida

Algeria, Setif
Algeria, Sidi Bel Abbes
Algeria, Skikda
Algeria, Souk Ahras
Algeria, Tamanghasset
Algeria, Tebessa
Algeria, Tiaret
Algeria, Tindouf
Algeria, Tipaza
Algeria, Tissemsilt
Algeria, Tizi Ouzou
Algeria, Tlemcen
Andorra, Andorra la Vella
Andorra, Canillo
Andorra, Encamp
Andorra, Escaldes-Engordany
Andorra, La Massana
Andorra, Ordino
Andorra, Sant Julia de Loria
Angola, Bengo
Angola, Benguela
Angola, Bie
Angola, Cabinda
Angola, Cuando Cubango
Angola, Cuanza Norte
Angola, Cuanza Sul
Angola, Cunene
Angola, Huambo
Angola, Huila
Angola, Luanda
Angola, Lunda Norte
Angola, Lunda Sul
Angola, Malanje
Angola, Moxico
Angola, Namibe
Angola, Uige
Angola, Zaire
Antigua and Barbuda, Barbuda
Antigua and Barbuda, Redonda
Antigua and Barbuda, Saint George
Antigua and Barbuda, Saint John
Antigua and Barbuda, Saint Mary
Antigua and Barbuda, Saint Paul
Antigua and Barbuda, Saint Peter
Antigua and Barbuda, Saint Philip
Argentina, Buenos Aires
Argentina, Catamarca
Argentina, Chaco
Argentina, Chubut
Argentina, Cordoba
Argentina, Corrientes
Argentina, Distrito Federal
Argentina, Entre Rios
Argentina, Formosa
Argentina, Jujuy
Argentina, La Pampa
Argentina, La Rioja
Argentina, Mendoza
Argentina, Misiones
Argentina, Neuquen
Argentina, Rio Negro
Argentina, Salta
Argentina, San Juan
Argentina, San Luis
Argentina, Santa Cruz
Argentina, Santa Fe
Argentina, Santiago del Estero
Argentina, Tierra del Fuego
Argentina, Tucuman
Armenia, Aragatsotn
Armenia, Ararat

Armenia, Armavir
Armenia, Geghark'unik'
Armenia, Kotayk'
Armenia, Lorri
Armenia, Shirak
Armenia, Syunik'
Armenia, Tavush
Armenia, Vayots' Dzor
Armenia, Yerevan
Australia, Australian Capital Territory
Australia, New South Wales
Australia, Northern Territory
Australia, Queensland
Australia, South Australia
Australia, Tasmania
Australia, Victoria
Australia, Western Australia
Austria, Burgenland
Austria, Karnten
Austria, Niederosterreich
Austria, Oberosterreich
Austria, Salzburg
Austria, Steiermark
Austria, Tirol
Austria, Vorarlberg
Austria, Wien
Azerbaijan, Abseron
Azerbaijan, Agcabadi
Azerbaijan, Agdam
Azerbaijan, Agdas
Azerbaijan, Agstafa
Azerbaijan, Agsu
Azerbaijan, Ali Bayramli
Azerbaijan, Astara
Azerbaijan, Baki
Azerbaijan, Balakan
Azerbaijan, Barda
Azerbaijan, Beylaqan
Azerbaijan, Bilasuvar
Azerbaijan, Cabrayil
Azerbaijan, Calilabad
Azerbaijan, Daskasan
Azerbaijan, Davaci
Azerbaijan, Fuzuli
Azerbaijan, Gadabay
Azerbaijan, Ganca
Azerbaijan, Goranboy
Azerbaijan, Goycay
Azerbaijan, Haciqabul
Azerbaijan, Imisli
Azerbaijan, Ismayilli
Azerbaijan, Kalbacar
Azerbaijan, Kurdamir
Azerbaijan, Lacin
Azerbaijan, Lankaran
Azerbaijan, Lankaran
Azerbaijan, Lerik
Azerbaijan, Masalli
Azerbaijan, Mingacevir
Azerbaijan, Naftalan
Azerbaijan, Naxcivan
Azerbaijan, Neftcalı
Azerbaijan, Oguz
Azerbaijan, Qabala
Azerbaijan, Qax
Azerbaijan, Qazax
Azerbaijan, Qobustan
Azerbaijan, Quba
Azerbaijan, Qubadli
Azerbaijan, Qusar
Azerbaijan, Saatli

Azerbaijan, Sabirabad
Azerbaijan, Saki
Azerbaijan, Saki
Azerbaijan, Salyan
Azerbaijan, Samaxi
Azerbaijan, Samkir
Azerbaijan, Samux
Azerbaijan, Siyazan
Azerbaijan, Sumqayit
Azerbaijan, Susa
Azerbaijan, Susa
Azerbaijan, Tartar
Azerbaijan, Tovuz
Azerbaijan, Ucar
Azerbaijan, Xacmaz
Azerbaijan, Xankandi
Azerbaijan, Xanlar
Azerbaijan, Xizi
Azerbaijan, Xocali
Azerbaijan, Xocavand
Azerbaijan, Yardimli
Azerbaijan, Yevlax
Azerbaijan, Yevlax
Azerbaijan, Zangilan
Azerbaijan, Zaqatala
Azerbaijan, Zardab
Bahrain, Al Asimah
Bahrain, Al Hadd
Bahrain, Al Janubiyah
Bahrain, Al Manamah
Bahrain, Al Mintaqah al Gharbiyah
Bahrain, Al Mintaqah al Wusta
Bahrain, Al Mintaqah ash Shamaliyah
Bahrain, Al Muharraq
Bahrain, Al Wusta
Bahrain, Ar Rifa
Bahrain, Ash Shamaliyah
Bahrain, Jidd Hafs
Bahrain, Madinat
Bahrain, Madinat Hamad
Bahrain, Mintaqat Juzur Hawar
Bahrain, Sitrah
Bangladesh, Barisal
Bangladesh, Chittagong
Bangladesh, Dhaka
Bangladesh, Khulna
Bangladesh, Rajshahi
Bangladesh, Sylhet
Barbados, Christ Church
Barbados, Saint Andrew
Barbados, Saint George
Barbados, Saint James
Barbados, Saint John
Barbados, Saint Joseph
Barbados, Saint Lucy
Barbados, Saint Michael
Barbados, Saint Peter
Barbados, Saint Philip
Barbados, Saint Thomas
Belarus, Brestskaya Voblasts'
Belarus, Homyl'skaya Voblasts'
Belarus, Hrodzyenskaya Voblasts'
Belarus, Mahilyowskaya Voblasts'
Belarus, Minsk
Belarus, Minskaya Voblasts'
Belarus, Vitsyebskaya Voblasts'
Belgium, Antwerpen
Belgium, Brabant Wallon
Belgium, Brussels Hoofdstedelijk Gewest
Belgium, Hainaut
Belgium, Liege

Belgium, Limburg
Belgium, Luxembourg
Belgium, Namur
Belgium, Oost-Vlaanderen
Belgium, Vlaams-Brabant
Belgium, West-Vlaanderen
Belize, Belize
Belize, Cayo
Belize, Corozal
Belize, Orange Walk
Belize, Stann Creek
Belize, Toledo
Benin, Alibori
Benin, Atakora
Benin, Atlantique
Benin, Borgou
Benin, Collines
Benin, Donga
Benin, Kouffo
Benin, Littoral
Benin, Mono
Benin, Oueme
Benin, Plateau
Benin, Zou
Bermuda, Devonshire
Bermuda, Hamilton
Bermuda, Hamilton
Bermuda, Paget
Bermuda, Pembroke
Bermuda, Saint George
Bermuda, Saint George's
Bermuda, Sandys
Bermuda, Smiths
Bermuda, Southampton
Bermuda, Warwick
Bhutan, Bumthang
Bhutan, Chhukha
Bhutan, Chirang
Bhutan, Daga
Bhutan, Geylegphug
Bhutan, Ha
Bhutan, Lhuntshi
Bhutan, Mongar
Bhutan, Paro
Bhutan, Pemagatsel
Bhutan, Punakha
Bhutan, Samchi
Bhutan, Samdrup
Bhutan, Shemgang
Bhutan, Tashigang
Bhutan, Thimphu
Bhutan, Tongsa
Bhutan, Wangdi Phodrang
Bolivia, Chuquisaca
Bolivia, Cochabamba
Bolivia, El Beni
Bolivia, La Paz
Bolivia, Oruro
Bolivia, Pando
Bolivia, Potosi
Bolivia, Santa Cruz
Bolivia, Tarija
Bosnia and Herzegovina, Federation of Bosnia and Herzegovina
Bosnia and Herzegovina, Republika Srpska
Botswana, Central
Botswana, Ghanzi
Botswana, Kgalagadi
Botswana, Kgatleng
Botswana, Kweneng
Botswana, North-East
Botswana, North-West

Botswana, South-East
Botswana, Southern
Brazil, Acre
Brazil, Alagoas
Brazil, Amapa
Brazil, Amazonas
Brazil, Bahia
Brazil, Ceara
Brazil, Distrito Federal
Brazil, Espirito Santo
Brazil, Goias
Brazil, Maranhao
Brazil, Mato Grosso
Brazil, Mato Grosso do Sul
Brazil, Minas Gerais
Brazil, Para
Brazil, Paraiba
Brazil, Parana
Brazil, Pernambuco
Brazil, Piaui
Brazil, Rio de Janeiro
Brazil, Rio Grande do Norte
Brazil, Rio Grande do Sul
Brazil, Rondonia
Brazil, Roraima
Brazil, Santa Catarina
Brazil, Sao Paulo
Brazil, Sergipe
Brazil, Tocantins
Brunei Darussalam, Alibori
Brunei Darussalam, Belait
Brunei Darussalam, Brunei and Muara
Brunei Darussalam, Collines
Brunei Darussalam, Donga
Brunei Darussalam, Kouffo
Brunei Darussalam, Littoral
Brunei Darussalam, Oueme
Brunei Darussalam, Plateau
Brunei Darussalam, Temburong
Brunei Darussalam, Tutong
Brunei Darussalam, Zou
Bulgaria, Blagoevgrad
Bulgaria, Burgas
Bulgaria, Dobrich
Bulgaria, Gabrovo
Bulgaria, Grad Sofiya
Bulgaria, Khaskovo
Bulgaria, Kurdzhali
Bulgaria, Kyustendil
Bulgaria, Lovech
Bulgaria, Mikhaylovgrad
Bulgaria, Montana
Bulgaria, Pazardzhik
Bulgaria, Pernik
Bulgaria, Pleven
Bulgaria, Plovdiv
Bulgaria, Razgrad
Bulgaria, Ruse
Bulgaria, Shumen
Bulgaria, Silistra
Bulgaria, Sliven
Bulgaria, Smolyan
Bulgaria, Sofiya
Bulgaria, Stara Zagora
Bulgaria, Turgovishte
Bulgaria, Varna
Bulgaria, Veliko Turnovo
Bulgaria, Vidin
Bulgaria, Vratsa
Bulgaria, Yambol
Burkina Faso, Bale

Burkina Faso, Bam
Burkina Faso, Banwa
Burkina Faso, Bazega
Burkina Faso, Bougouriba
Burkina Faso, Boulgou
Burkina Faso, Boulkiemde
Burkina Faso, Ganzourgou
Burkina Faso, Gnagna
Burkina Faso, Gourma
Burkina Faso, Houet
Burkina Faso, Ioba
Burkina Faso, Kadiogo
Burkina Faso, Kenedougou
Burkina Faso, Komoe
Burkina Faso, Komondjari
Burkina Faso, Kompienga
Burkina Faso, Kossi
Burkina Faso, Koulpelogo
Burkina Faso, Kouritenga
Burkina Faso, Kourweogo
Burkina Faso, Leraba
Burkina Faso, Loroum
Burkina Faso, Mouhoum
Burkina Faso, Namentenga
Burkina Faso, Naouri
Burkina Faso, Nayala
Burkina Faso, Noumbiel
Burkina Faso, Oubritenga
Burkina Faso, Oudalan
Burkina Faso, Passore
Burkina Faso, Poni
Burkina Faso, Sanguie
Burkina Faso, Sanmatenga
Burkina Faso, Seno
Burkina Faso, Sissili
Burkina Faso, Soum
Burkina Faso, Sourou
Burkina Faso, Tapoa
Burkina Faso, Tuy
Burkina Faso, Yagha
Burkina Faso, Yatenga
Burkina Faso, Ziro
Burkina Faso, Zondoma
Burkina Faso, Zoundweogo
Burundi, Bubanza
Burundi, Bujumbura
Burundi, Bururi
Burundi, Cankuzo
Burundi, Cibitoke
Burundi, Gitega
Burundi, Karuzi
Burundi, Kayanza
Burundi, Kirundo
Burundi, Makamba
Burundi, Muramvya
Burundi, Muyinga
Burundi, Mwaro
Burundi, Ngozi
Burundi, Rutana
Burundi, Ruyigi
Cambodia, Banteay Meanchey
Cambodia, Batdambang
Cambodia, Batdambang
Cambodia, Kampong Cham
Cambodia, Kampong Chhnang
Cambodia, Kampong Speu
Cambodia, Kampong Thum
Cambodia, Kampot
Cambodia, Kandal
Cambodia, Koh Kong
Cambodia, Kracheh

Cambodia, Mondulkiri
Cambodia, Pailin
Cambodia, Phnum Penh
Cambodia, Preah Vihear
Cambodia, Prey Veng
Cambodia, Pursat
Cambodia, Ratanakiri Kiri
Cambodia, Siem Reap
Cambodia, Stung Treng
Cambodia, Svay Rieng
Cambodia, Takeo
Cameroon, Adamaoua
Cameroon, Centre
Cameroon, Est
Cameroon, Extreme-Nord
Cameroon, Littoral
Cameroon, Nord
Cameroon, Nord-Ouest
Cameroon, Ouest
Cameroon, Sud
Cameroon, Sud-Ouest
Canada, Alberta
Canada, British Columbia
Canada, Manitoba
Canada, New Brunswick
Canada, Newfoundland
Canada, Northwest Territories
Canada, Nova Scotia
Canada, Nunavut
Canada, Ontario
Canada, Prince Edward Island
Canada, Quebec
Canada, Saskatchewan
Canada, Yukon Territory
Cape Verde, Boa Vista
Cape Verde, Brava
Cape Verde, Maio
Cape Verde, Mosteiros
Cape Verde, Paul
Cape Verde, Praia
Cape Verde, Ribeira Grande
Cape Verde, Sal
Cape Verde, Santa Catarina
Cape Verde, Santa Cruz
Cape Verde, Sao Domingos
Cape Verde, Sao Filipe
Cape Verde, Sao Miguel
Cape Verde, Sao Nicolau
Cape Verde, Sao Vicente
Cape Verde, Tarrafal
Cayman Islands, Creek
Cayman Islands, Eastern
Cayman Islands, Midland
Cayman Islands, South Town
Cayman Islands, Spot Bay
Cayman Islands, Stake Bay
Cayman Islands, West End
Cayman Islands, Western
Central African Republic, Bamingui-Bangoran
Central African Republic, Bangui
Central African Republic, Basse-Kotto
Central African Republic, Cuvette-Ouest
Central African Republic, Haut-Mbomou
Central African Republic, Haute-Kotto
Central African Republic, Kemo
Central African Republic, Lobaye
Central African Republic, Mambere-Kadei
Central African Republic, Mbomou
Central African Republic, Nana-Grebizi
Central African Republic, Nana-Mambere
Central African Republic, Ombella-Mpoko

Central African Republic, Ouaka
Central African Republic, Ouham
Central African Republic, Ouham-Pende
Central African Republic, Sangha-Mbaere
Chad, Batha
Chad, Biltine
Chad, Borkou-Ennedi-Tibesti
Chad, Chari-Baguirmi
Chad, Guera
Chad, Kanem
Chad, Lac
Chad, Logone Occidental
Chad, Logone Oriental
Chad, Mayo-Kebbi
Chad, Moyen-Chari
Chad, Ouaddai
Chad, Salamat
Chad, Tandjile
Chile, Aisen del General Carlos Ibanez del Campo
Chile, Antofagasta
Chile, Araucania
Chile, Arica y Parinacota
Chile, Atacama
Chile, Bio-Bio
Chile, Coquimbo
Chile, Libertador General Bernardo O'Higgins
Chile, Los Lagos
Chile, Los Lagos
Chile, Los Rios
Chile, Magallanes y de la Antartica Chilena
Chile, Maule
Chile, Region Metropolitana
Chile, Tarapaca
Chile, Tarapaca
Chile, Valparaiso
China, Anhui
China, Beijing
China, Chongqing
China, Fujian
China, Gansu
China, Guangdong
China, Guangxi
China, Guizhou
China, Hainan
China, Hebei
China, Heilongjiang
China, Henan
China, Hubei
China, Hunan
China, Jiangsu
China, Jiangxi
China, Jilin
China, Liaoning
China, Nei Mongol
China, Ningxia
China, Qinghai
China, Shaanxi
China, Shandong
China, Shanghai
China, Shanxi
China, Sichuan
China, Tianjin
China, Xinjiang
China, Xizang
China, Yunnan
China, Zhejiang
Colombia, Amazonas
Colombia, Antioquia
Colombia, Arauca
Colombia, Atlantico
Colombia, Bolivar

Colombia, Bolivar Department
Colombia, Boyaca
Colombia, Boyaca Department
Colombia, Caldas
Colombia, Caldas Department
Colombia, Caqueta
Colombia, Casanare
Colombia, Cauca
Colombia, Cesar
Colombia, Choco
Colombia, Cordoba
Colombia, Cundinamarca
Colombia, Distrito Especial
Colombia, Guainia
Colombia, Guaviare
Colombia, Huila
Colombia, La Guajira
Colombia, Magdalena
Colombia, Magdalena Department
Colombia, Meta
Colombia, Narino
Colombia, Norte de Santander
Colombia, Putumayo
Colombia, Quindio
Colombia, Risaralda
Colombia, San Andres y Providencia
Colombia, Santander
Colombia, Sucre
Colombia, Tolima
Colombia, Valle del Cauca
Colombia, Vaupes
Colombia, Vichada
Comoros, Anjouan
Comoros, Grande Comore
Comoros, Moheli
Congo, Bouenza
Congo, Brazzaville
Congo, Cuvette
Congo, Cuvette-Ouest
Congo, Kouilou
Congo, Lekoumou
Congo, Likouala
Congo, Niari
Congo, Plateaux
Congo, Pool
Congo, Sangha
Congo, The Democratic Republic of the, Bandundu
Congo, The Democratic Republic of the, Bas-Congo
Congo, The Democratic Republic of the, Equateur
Congo, The Democratic Republic of the, Kasai-Oriental
Congo, The Democratic Republic of the, Katanga
Congo, The Democratic Republic of the, Kinshasa
Congo, The Democratic Republic of the, Maniema
Congo, The Democratic Republic of the, Nord-Kivu
Congo, The Democratic Republic of the, Orientale
Congo, The Democratic Republic of the, Sud-Kivu
Costa Rica, Alajuela
Costa Rica, Cartago
Costa Rica, Guanacaste
Costa Rica, Heredia
Costa Rica, Limon
Costa Rica, Puntarenas
Costa Rica, San Jose
Cote D'Ivoire, Agneby
Cote D'Ivoire, Bafing
Cote D'Ivoire, Bas-Sassandra
Cote D'Ivoire, Denguele
Cote D'Ivoire, Dix-Huit Montagnes
Cote D'Ivoire, Fromager
Cote D'Ivoire, Haut-Sassandra
Cote D'Ivoire, Lacs

Cote D'Ivoire, Lagunes
Cote D'Ivoire, Marahoue
Cote D'Ivoire, Moyen-Cavally
Cote D'Ivoire, Moyen-Comoé
Cote D'Ivoire, N'zi-Comoé
Cote D'Ivoire, Savanes
Cote D'Ivoire, Sud-Bandama
Cote D'Ivoire, Sud-Comoé
Cote D'Ivoire, Vallée du Bandama
Cote D'Ivoire, Worodougou
Cote D'Ivoire, Zanzan
Croatia, Bjelovarsko-Bilogorska
Croatia, Brodsko-Posavska
Croatia, Dubrovacko-Neretvanska
Croatia, Grad Zagreb
Croatia, Istarska
Croatia, Karlovacka
Croatia, Koprivnicko-Krizevacka
Croatia, Krapinsko-Zagorska
Croatia, Licko-Senjska
Croatia, Medimurska
Croatia, Osjecko-Baranjska
Croatia, Pozesko-Slavonska
Croatia, Primorsko-Goranska
Croatia, Sibensko-Kninska
Croatia, Sisacko-Moslavacka
Croatia, Splitsko-Dalmatinska
Croatia, Varazdinska
Croatia, Viroviticko-Podravska
Croatia, Vukovarsko-Srijemska
Croatia, Zadarska
Croatia, Zagrebacka
Cuba, Camaguey
Cuba, Ciego de Avila
Cuba, Cienfuegos
Cuba, Ciudad de la Habana
Cuba, Granma
Cuba, Guantanamo
Cuba, Holguin
Cuba, Isla de la Juventud
Cuba, La Habana
Cuba, Las Tunas
Cuba, Matanzas
Cuba, Pinar del Rio
Cuba, Sancti Spiritus
Cuba, Santiago de Cuba
Cuba, Villa Clara
Cyprus, Famagusta
Cyprus, Kyrenia
Cyprus, Larnaca
Cyprus, Limassol
Cyprus, Nicosia
Cyprus, Paphos
Czech Republic, Hlavni mesto Praha
Czech Republic, Jihocesky kraj
Czech Republic, Jihomoravsky kraj
Czech Republic, Karlovarsky kraj
Czech Republic, Kralovehradecky kraj
Czech Republic, Liberecky kraj
Czech Republic, Moravskoslezsky kraj
Czech Republic, Olomoucky kraj
Czech Republic, Pardubicky kraj
Czech Republic, Plzensky kraj
Czech Republic, Stredocesky kraj
Czech Republic, Ustecky kraj
Czech Republic, Vysocina
Czech Republic, Zlinsky kraj
Denmark, Hovedstaden
Denmark, Midtjylland
Denmark, Nordjylland
Denmark, Sjælland

Denmark, Syddanmark
Djibouti, Ali Sabieh
Djibouti, Arta
Djibouti, Dikhil
Djibouti, Djibouti
Djibouti, Obock
Djibouti, Tadjoura
Dominica, Saint Andrew
Dominica, Saint David
Dominica, Saint George
Dominica, Saint John
Dominica, Saint Joseph
Dominica, Saint Luke
Dominica, Saint Mark
Dominica, Saint Patrick
Dominica, Saint Paul
Dominica, Saint Peter
Dominican Republic, Azua
Dominican Republic, Baoruco
Dominican Republic, Barahona
Dominican Republic, Dajabon
Dominican Republic, Distrito Nacional
Dominican Republic, Distrito Nacional
Dominican Republic, Duarte
Dominican Republic, El Seibo
Dominican Republic, Elias Pina
Dominican Republic, Espaillat
Dominican Republic, Hato Mayor
Dominican Republic, Independencia
Dominican Republic, La Altagracia
Dominican Republic, La Romana
Dominican Republic, La Vega
Dominican Republic, Maria Trinidad Sanchez
Dominican Republic, Monseñor Nouel
Dominican Republic, Monte Cristi
Dominican Republic, Monte Plata
Dominican Republic, Pedernales
Dominican Republic, Peravia
Dominican Republic, Peravia
Dominican Republic, Puerto Plata
Dominican Republic, Salcedo
Dominican Republic, Samana
Dominican Republic, San Cristobal
Dominican Republic, San Jose de Ocoa
Dominican Republic, San Juan
Dominican Republic, San Pedro De Macoris
Dominican Republic, Sanchez Ramirez
Dominican Republic, Santiago
Dominican Republic, Santiago Rodriguez
Dominican Republic, Santo Domingo
Dominican Republic, Valverde
Ecuador, Azuay
Ecuador, Bolívar
Ecuador, Canar
Ecuador, Carchi
Ecuador, Chimborazo
Ecuador, Cotopaxi
Ecuador, El Oro
Ecuador, Esmeraldas
Ecuador, Galapagos
Ecuador, Guayas
Ecuador, Imbabura
Ecuador, Loja
Ecuador, Los Ríos
Ecuador, Manabí
Ecuador, Morona-Santiago
Ecuador, Napo
Ecuador, Orellana
Ecuador, Pastaza
Ecuador, Pichincha
Ecuador, Sucumbíos

Ecuador, Tungurahua
Ecuador, Zamora-Chinchipe
Egypt, Ad Daqahliyah
Egypt, Al Bahr al Ahmar
Egypt, Al Buhayrah
Egypt, Al Fayyum
Egypt, Al Gharbiyah
Egypt, Al Iskandariyah
Egypt, Al Isma'iliyah
Egypt, Al Jizah
Egypt, Al Minufiyah
Egypt, Al Minya
Egypt, Al Qahirah
Egypt, Al Qalyubiyah
Egypt, Al Wadi al Jadid
Egypt, As Suways
Egypt, Ash Sharqiyah
Egypt, Aswan
Egypt, Asyut
Egypt, Bani Suwayf
Egypt, Bur Sa'id
Egypt, Dumyat
Egypt, Janub Sina'
Egypt, Kafr ash Shaykh
Egypt, Matruh
Egypt, Qina
Egypt, Shamal Sina'
Egypt, Suhaj
El Salvador, Ahuachapan
El Salvador, Cabanas
El Salvador, Chalatenango
El Salvador, Cuscatlan
El Salvador, La Libertad
El Salvador, La Paz
El Salvador, La Union
El Salvador, Morazan
El Salvador, San Miguel
El Salvador, San Salvador
El Salvador, San Vicente
El Salvador, Santa Ana
El Salvador, Sonsonate
El Salvador, Usulutan
Equatorial Guinea, Annobon
Equatorial Guinea, Bioko Norte
Equatorial Guinea, Bioko Sur
Equatorial Guinea, Centro Sur
Equatorial Guinea, Kie-Ntem
Equatorial Guinea, Litoral
Equatorial Guinea, Wele-Nzas
Eritrea, Anseba
Eritrea, Debub
Eritrea, Debubawi K'eyih Bahri
Eritrea, Gash Barka
Eritrea, Ma'akel
Eritrea, Semenawi K'eyih Bahri
Estonia, Harjumaa
Estonia, Hiiumaa
Estonia, Ida-Virumaa
Estonia, Jarvamaa
Estonia, Jõgevamaa
Estonia, Kohtla-Jarve
Estonia, Laane-Virumaa
Estonia, Laanemaa
Estonia, Narva
Estonia, Parnu
Estonia, Pärnumaa
Estonia, Polvamaa
Estonia, Raplamaa
Estonia, Saaremaa
Estonia, Sillamae
Estonia, Tallinn

Estonia, Tartu
Estonia, Tartumaa
Estonia, Valgamaa
Estonia, Viljandimaa
Estonia, Vorumaa
Ethiopia, Adis Abeba
Ethiopia, Afar
Ethiopia, Amara
Ethiopia, Binshangul Gumuz
Ethiopia, Dire Dawa
Ethiopia, Gambela Hizboch
Ethiopia, Hareri Hizb
Ethiopia, Oromiya
Ethiopia, Sumale
Ethiopia, Tigray
Ethiopia, YeDeubub Biheroch Bihereseboch na Hizboch
Fiji, Central
Fiji, Eastern
Fiji, Northern
Fiji, Rotuma
Fiji, Western
Finland, Aland
Finland, Eastern Finland
Finland, Lapland
Finland, Oulu
Finland, Southern Finland
Finland, Western Finland
France, Alsace
France, Aquitaine
France, Auvergne
France, Basse-Normandie
France, Bourgogne
France, Bretagne
France, Centre
France, Champagne-Ardenne
France, Corse
France, Franche-Comte
France, Haute-Normandie
France, Ile-de-France
France, Languedoc-Roussillon
France, Limousin
France, Lorraine
France, Midi-Pyrenees
France, Nord-Pas-de-Calais
France, Pays de la Loire
France, Picardie
France, Poitou-Charentes
France, Provence-Alpes-Cote d'Azur
France, Rhone-Alpes
Gabon, Estuaire
Gabon, Haut-Ogooue
Gabon, Moyen-Ogooue
Gabon, Ngounie
Gabon, Nyanga
Gabon, Ogooue-Ivindo
Gabon, Ogooue-Lolo
Gabon, Ogooue-Maritime
Gabon, Woleu-Ntem
Gambia, Banjul
Gambia, Central River
Gambia, Lower River
Gambia, North Bank
Gambia, Upper River
Gambia, Western
Georgia, Abashis Raioni
Georgia, Abkhazia
Georgia, Adigenis Raioni
Georgia, Ajaria
Georgia, Akhalgoris Raioni
Georgia, Akhalk'alak's Raioni
Georgia, Akhalts'ikhis Raioni

Georgia, Akhmetis Raioni
Georgia, Ambrolauris Raioni
Georgia, Aspindzis Raioni
Georgia, Baghdat'is Raioni
Georgia, Bolnisis Raioni
Georgia, Borjomis Raioni
Georgia, Ch'khorotsqus Raioni
Georgia, Ch'okhatauris Raioni
Georgia, Chiat'ura
Georgia, Dedop'listsqaros Raioni
Georgia, Dmanisis Raioni
Georgia, Dushet'is Raioni
Georgia, Gardabanis Raioni
Georgia, Gori
Georgia, Goris Raioni
Georgia, Gurjaanis Raioni
Georgia, Javis Raioni
Georgia, K'arelis Raioni
Georgia, K'ut'aisi
Georgia, Kaspis Raioni
Georgia, Kharagaulis Raioni
Georgia, Khashuris Raioni
Georgia, Khobis Raioni
Georgia, Khonis Raioni
Georgia, Lagodekhis Raioni
Georgia, Lanch'khut'is Raioni
Georgia, Lentekhis Raioni
Georgia, Marneulis Raioni
Georgia, Martvilis Raioni
Georgia, Mestiai Raioni
Georgia, Mts'khet'is Raioni
Georgia, Ninotsmindis Raioni
Georgia, Onis Raioni
Georgia, Ozurget'is Raioni
Georgia, P'ot'i
Georgia, Qazbegis Raioni
Georgia, Qvarlis Raioni
Georgia, Rust'avi
Georgia, Sach'keris Raioni
Georgia, Sagarejos Raioni
Georgia, Samtrediai Raioni
Georgia, Senakis Raioni
Georgia, Sighnaghis Raioni
Georgia, Tbilisi
Georgia, T'elavis Raioni
Georgia, Terjolis Raioni
Georgia, T'et'ritsqaros Raioni
Georgia, T'ianet'is Raioni
Georgia, Tqibuli
Georgia, Ts'ageris Raioni
Georgia, Tsalenjikhis Raioni
Georgia, Tsalkis Raioni
Georgia, Tsqaltubo
Georgia, Vanis Raioni
Georgia, Zestap'onis Raioni
Georgia, Zugdidi
Georgia, Zugdidis Raioni
Germany, Baden-Wurttemberg
Germany, Bayern
Germany, Berlin
Germany, Brandenburg
Germany, Bremen
Germany, Hamburg
Germany, Hessen
Germany, Mecklenburg-Vorpommern
Germany, Niedersachsen
Germany, Nordrhein-Westfalen
Germany, Rheinland-Pfalz
Germany, Saarland
Germany, Sachsen
Germany, Sachsen-Anhalt

Germany, Schleswig-Holstein
Germany, Thuringen
Ghana, Ashanti
Ghana, Brong-Ahafo
Ghana, Central
Ghana, Eastern
Ghana, Greater Accra
Ghana, Northern
Ghana, Upper East
Ghana, Upper West
Ghana, Volta
Ghana, Western
Greece, Aitolia kai Akarnania
Greece, Akhaia
Greece, Argolis
Greece, Arkadia
Greece, Arta
Greece, Attiki
Greece, Dhadhekanisos
Greece, Drama
Greece, Evritania
Greece, Evros
Greece, Evvoia
Greece, Florina
Greece, Fokis
Greece, Fthiotis
Greece, Grevena
Greece, Ilia
Greece, Imathia
Greece, Ioannina
Greece, Iraklion
Greece, Karditsa
Greece, Kastoria
Greece, Kavala
Greece, Kefallinia
Greece, Kerkira
Greece, Khalkidiki
Greece, Khania
Greece, Khios
Greece, Kikladhes
Greece, Kilkis
Greece, Korinthia
Greece, Kozani
Greece, Lakonia
Greece, Larisa
Greece, Lasithi
Greece, Lesvos
Greece, Levkas
Greece, Magnisia
Greece, Messinia
Greece, Pella
Greece, Pieria
Greece, Preveza
Greece, Rethimni
Greece, Rodhopi
Greece, Samos
Greece, Serrai
Greece, Thesprotia
Greece, Thessaloniki
Greece, Trikala
Greece, Voiotia
Greece, Xanthi
Greece, Zakynthos
Greenland, Nordgronland
Greenland, Ostgronland
Greenland, Vestgronland
Grenada, Saint Andrew
Grenada, Saint David
Grenada, Saint George
Grenada, Saint John
Grenada, Saint Mark

Grenada, Saint Patrick
Guatemala, Alta Verapaz
Guatemala, Baja Verapaz
Guatemala, Chimaltenango
Guatemala, Chiquimula
Guatemala, El Progreso
Guatemala, Escuintla
Guatemala, Guatemala
Guatemala, Huehuetenango
Guatemala, Izabal
Guatemala, Jalapa
Guatemala, Jutiapa
Guatemala, Peten
Guatemala, Quetzaltenango
Guatemala, Quiche
Guatemala, Retalhuleu
Guatemala, Sacatepequez
Guatemala, San Marcos
Guatemala, Santa Rosa
Guatemala, Solola
Guatemala, Suchitepequez
Guatemala, Totonicapan
Guatemala, Zacapa
Guinea-Bissau, Bafata
Guinea-Bissau, Biombo
Guinea-Bissau, Bissau
Guinea-Bissau, Bolama
Guinea-Bissau, Cacheu
Guinea-Bissau, Gabu
Guinea-Bissau, Oio
Guinea-Bissau, Quinara
Guinea-Bissau, Tombali
Guinea, Beyla
Guinea, Boffa
Guinea, Boke
Guinea, Conakry
Guinea, Coyah
Guinea, Dabola
Guinea, Dalaba
Guinea, Dingiraye
Guinea, Dubreka
Guinea, Faranah
Guinea, Forecariah
Guinea, Fria
Guinea, Gaoual
Guinea, Gueckedou
Guinea, Kankan
Guinea, Kerouane
Guinea, Kindia
Guinea, Kissidougou
Guinea, Kouribia
Guinea, Koundara
Guinea, Kouroussa
Guinea, Labe
Guinea, Lelouma
Guinea, Lola
Guinea, Macenta
Guinea, Mali
Guinea, Mamou
Guinea, Mandiana
Guinea, Nzerekore
Guinea, Pita
Guinea, Sigiri
Guinea, Telimele
Guinea, Tougue
Guinea, Yomou
Guyana, Barima-Waini
Guyana, Cuyuni-Mazaruni
Guyana, Demerara-Mahaica
Guyana, East Berbice-Corentyne
Guyana, Essequibo Islands-West Demerara

Guyana, Mahaica-Berbice
Guyana, Pomeroon-Supenaam
Guyana, Potaro-Siparuni
Guyana, Upper Demerara-Berbice
Guyana, Upper Takutu-Upper Essequibo

H - M

Haiti, Artibonite
Haiti, Centre
Haiti, Grand' Anse
Haiti, Nippes
Haiti, Nord
Haiti, Nord-Est
Haiti, Nord-Ouest
Haiti, Ouest
Haiti, Sud
Haiti, Sud-Est
Honduras, Atlantida
Honduras, Choluteca
Honduras, Colon
Honduras, Comayagua
Honduras, Copan
Honduras, Cortes
Honduras, El Paraiso
Honduras, Francisco Morazan
Honduras, Gracias a Dios
Honduras, Intibuca
Honduras, Islas de la Bahia
Honduras, La Paz
Honduras, Lempira
Honduras, Ocotepeque
Honduras, Olancho
Honduras, Santa Barbara
Honduras, Valle
Honduras, Yoro
Hungary, Bacs-Kiskun
Hungary, Baranya
Hungary, Bekes
Hungary, Bekescsaba
Hungary, Borsod-Abauj-Zemplen
Hungary, Budapest
Hungary, Csongrad
Hungary, Debrecen
Hungary, Dunaujvaros
Hungary, Eger
Hungary, Erd
Hungary, Fejer
Hungary, Gyor
Hungary, Gyor-Moson-Sopron
Hungary, Hajdu-Bihar
Hungary, Heves
Hungary, Hodmezovasarhely
Hungary, Jasz-Nagykun-Szolnok
Hungary, Kaposvar
Hungary, Kecskemet
Hungary, Komarom-Esztergom
Hungary, Miskolc
Hungary, Nagykanizsa
Hungary, Nograd
Hungary, Nyiregyhaza
Hungary, Pecs
Hungary, Pest
Hungary, Salgotarjan
Hungary, Somogy
Hungary, Sopron
Hungary, Szabolcs-Szatmar-Bereg
Hungary, Szeged
Hungary, Szekesfehervar
Hungary, Szekszard
Hungary, Szolnok

Hungary, Szombathely
Hungary, Tatabanya
Hungary, Tolna
Hungary, Vas
Hungary, Veszprem
Hungary, Veszprem
Hungary, Zala
Hungary, Zalaegerszeg
Iceland, Arnessysla
Iceland, Austur-Hunavatnssysla
Iceland, Austur-Skaftafellssysla
Iceland, Borgarfjardarsysla
Iceland, Eyjafjardarsysla
Iceland, Gullbringusysla
Iceland, Kjosarsysla
Iceland, Myrasysla
Iceland, Nordur-Mulasysla
Iceland, Nordur-Tingeyjarsysla
Iceland, Norourland Eystra
Iceland, Norourland Vestra
Iceland, Rangarvallasysla
Iceland, Skagafjardarsysla
Iceland, Snafellsnes- og Hnappadalssysla
Iceland, Strandasysla
Iceland, Sudur-Mulasysla
Iceland, Sudur-Tingeyjarsysla
Iceland, Suourland
Iceland, Suournes
Iceland, Vestfiroir
Iceland, Vestur-Bardastrandarsysla
Iceland, Vestur-Hunavatnssysla
Iceland, Vestur-Isafjardarsysla
Iceland, Vestur-Skaftafellssysla
Iceland, Vesturland
India, Andaman and Nicobar Islands
India, Andhra Pradesh
India, Arunachal Pradesh
India, Assam
India, Bihar
India, Chandigarh
India, Chhattisgarh
India, Dadra and Nagar Haveli
India, Daman and Diu
India, Delhi
India, Goa
India, Gujarat
India, Haryana
India, Himachal Pradesh
India, Jammu and Kashmir
India, Jharkhand
India, Karnataka
India, Kerala
India, Lakshadweep
India, Madhya Pradesh
India, Maharashtra
India, Manipur
India, Meghalaya
India, Mizoram
India, Nagaland
India, Orissa
India, Puducherry
India, Punjab
India, Rajasthan
India, Sikkim
India, Tamil Nadu
India, Tripura
India, Uttar Pradesh
India, Uttarakhand
India, West Bengal
Indonesia, Aceh
Indonesia, Bali

Indonesia, Banten
Indonesia, Bengkulu
Indonesia, Gorontalo
Indonesia, Irian Jaya Barat
Indonesia, Jakarta Raya
Indonesia, Jambi
Indonesia, Jawa Barat
Indonesia, Jawa Barat
Indonesia, Jawa Tengah
Indonesia, Jawa Timur
Indonesia, Kalimantan Barat
Indonesia, Kalimantan Selatan
Indonesia, Kalimantan Tengah
Indonesia, Kalimantan Timur
Indonesia, Kepulauan Bangka Belitung
Indonesia, Kepulauan Riau
Indonesia, Lampung
Indonesia, Maluku
Indonesia, Maluku
Indonesia, Maluku Utara
Indonesia, Nusa Tenggara Barat
Indonesia, Nusa Tenggara Timur
Indonesia, Papua
Indonesia, Papua
Indonesia, Riau
Indonesia, Riau
Indonesia, Sulawesi Barat
Indonesia, Sulawesi Selatan
Indonesia, Sulawesi Selatan
Indonesia, Sulawesi Tengah
Indonesia, Sulawesi Tenggara
Indonesia, Sulawesi Utara
Indonesia, Sulawesi Utara
Indonesia, Sumatera Barat
Indonesia, Sumatera Selatan
Indonesia, Sumatera Selatan
Indonesia, Sumatera Utara
Indonesia, Yogyakarta
Iran, Islamic Republic of, Ardabil
Iran, Islamic Republic of, Azarbayjan-e Bakhtari
Iran, Islamic Republic of, Bakhtaran
Iran, Islamic Republic of, Bushehr
Iran, Islamic Republic of, Chahar Mahall va Bakhtiari
Iran, Islamic Republic of, East Azarbaijan
Iran, Islamic Republic of, Esfahan
Iran, Islamic Republic of, Fars
Iran, Islamic Republic of, Gilan
Iran, Islamic Republic of, Golestan
Iran, Islamic Republic of, Hamadan
Iran, Islamic Republic of, Hormozgan
Iran, Islamic Republic of, Ilam
Iran, Islamic Republic of, Kerman
Iran, Islamic Republic of, Kerman
Iran, Islamic Republic of, Khorasan
Iran, Islamic Republic of, Khorasan-e Janubi
Iran, Islamic Republic of, Khorasan-e Razavi
Iran, Islamic Republic of, Khorasan-e Shemali
Iran, Islamic Republic of, Khuzestan
Iran, Islamic Republic of, Kohkiluyeh va Buyer Ahmadi
Iran, Islamic Republic of, Kordestan
Iran, Islamic Republic of, Lorestan
Iran, Islamic Republic of, Markazi
Iran, Islamic Republic of, Markazi
Iran, Islamic Republic of, Mazandaran
Iran, Islamic Republic of, Mazandaran
Iran, Islamic Republic of, Qazvin
Iran, Islamic Republic of, Qom
Iran, Islamic Republic of, Semnan
Iran, Islamic Republic of, Semnan Province
Iran, Islamic Republic of, Sistan va Baluchestan

Iran, Islamic Republic of, Tehran
Iran, Islamic Republic of, Yazd
Iran, Islamic Republic of, Yazd
Iran, Islamic Republic of, Zanjan
Iran, Islamic Republic of, Zanjan
Iran, Islamic Republic of, Zanjan
Iraq, Al Anbar
Iraq, Al Basrah
Iraq, Al Muthanna
Iraq, Al Qadisiyah
Iraq, An Najaf
Iraq, Arbil
Iraq, As Sulaymaniyah
Iraq, At Ta'mim
Iraq, Babil
Iraq, Baghdad
Iraq, Dahuk
Iraq, Dhi Qar
Iraq, Diyala
Iraq, Karbala'
Iraq, Maysan
Iraq, Ninawa
Iraq, Salah ad Din
Iraq, Wasit
Ireland, Carlow
Ireland, Cavan
Ireland, Clare
Ireland, Cork
Ireland, Donegal
Ireland, Dublin
Ireland, Galway
Ireland, Kerry
Ireland, Kildare
Ireland, Kilkenny
Ireland, Laois
Ireland, Leitrim
Ireland, Limerick
Ireland, Longford
Ireland, Louth
Ireland, Mayo
Ireland, Meath
Ireland, Monaghan
Ireland, Offaly
Ireland, Roscommon
Ireland, Sligo
Ireland, Tipperary
Ireland, Waterford
Ireland, Westmeath
Ireland, Wexford
Ireland, Wicklow
Israel, HaDarom
Israel, HaMerkaz
Israel, HaZafon
Israel, Hefa
Israel, Tel Aviv
Israel, Yerushalayim
Italy, Abruzzi
Italy, Basilicata
Italy, Calabria
Italy, Campania
Italy, EmiliaRomagna
Italy, Friuli-Venezia Giulia
Italy, Lazio
Italy, Liguria
Italy, Lombardia
Italy, Marche
Italy, Molise
Italy, Piemonte
Italy, Puglia
Italy, Sardegna
Italy, Sicilia

Italy, Toscana
Italy, Trentino-Alto Adige
Italy, Umbria
Italy, Valle d'Aosta
Italy, Veneto
Jamaica, Clarendon
Jamaica, Hanover
Jamaica, Kingston
Jamaica, Manchester
Jamaica, Portland
Jamaica, Saint Andrew
Jamaica, Saint Ann
Jamaica, Saint Catherine
Jamaica, Saint Elizabeth
Jamaica, Saint James
Jamaica, Saint Mary
Jamaica, Saint Thomas
Jamaica, Trelawny
Jamaica, Westmoreland
Japan, Aichi
Japan, Akita
Japan, Aomori
Japan, Chiba
Japan, Ehime
Japan, Fukui
Japan, Fukuoka
Japan, Fukushima
Japan, Gifu
Japan, Gumma
Japan, Hiroshima
Japan, Hokkaido
Japan, Hyogo
Japan, Ibaraki
Japan, Ishikawa
Japan, Iwate
Japan, Kagawa
Japan, Kagoshima
Japan, Kanagawa
Japan, Kochi
Japan, Kumamoto
Japan, Kyoto
Japan, Mie
Japan, Miyagi
Japan, Miyazaki
Japan, Nagano
Japan, Nagasaki
Japan, Nara
Japan, Niigata
Japan, Oita
Japan, Okayama
Japan, Okinawa
Japan, Osaka
Japan, Saga
Japan, Saitama
Japan, Shiga
Japan, Shimane
Japan, Shizuoka
Japan, Tochigi
Japan, Tokushima
Japan, Tokyo
Japan, Tottori
Japan, Toyama
Japan, Wakayama
Japan, Yamagata
Japan, Yamaguchi
Japan, Yamanashi
Jordan, Al Balqa'
Jordan, Al Karak
Jordan, Al Mafraq
Jordan, Amman
Jordan, Amman Governorate

Jordan, At Tafilah
Jordan, Az Zarqa
Jordan, Irbid
Jordan, Ma
Kazakhstan, Almaty
Kazakhstan, Almaty City
Kazakhstan, Aqmola
Kazakhstan, Aqtobe
Kazakhstan, Astana
Kazakhstan, Atyrau
Kazakhstan, Bayqonyr
Kazakhstan, East Kazakhstan
Kazakhstan, Mangghystau
Kazakhstan, North Kazakhstan
Kazakhstan, Pavlodar
Kazakhstan, Qaraghandy
Kazakhstan, Qostanay
Kazakhstan, Qyzylorda
Kazakhstan, South Kazakhstan
Kazakhstan, West Kazakhstan
Kazakhstan, Zhambyl
Kenya, Central
Kenya, Coast
Kenya, Eastern
Kenya, Nairobi Area
Kenya, North-Eastern
Kenya, Nyanza
Kenya, Rift Valley
Kenya, Western
Kiribati, Gilbert Islands
Kiribati, Line Islands
Kiribati, Phoenix Islands
Korea, Democratic People's Republic of, Chagang-do
Korea, Democratic People's Republic of, Hamgyong-bukto
Korea, Democratic People's Republic of, Hamgyong-namdo
Korea, Democratic People's Republic of, Hwanghae-bukto
Korea, Democratic People's Republic of, Hwanghae-namdo
Korea, Democratic People's Republic of, Kaesong-si
Korea, Democratic People's Republic of, Kangwon-do
Korea, Democratic People's Republic of, Naja'n Sonbong-si
Korea, Democratic People's Republic of, Namp'o-si
Korea, Democratic People's Republic of, P'yongan-bukto
Korea, Democratic People's Republic of, P'yongan-namdo
Korea, Democratic People's Republic of, P'yongyang-si
Korea, Democratic People's Republic of, Yanggang-do
Korea, Republic of, Ch'ungch'ong-bukto
Korea, Republic of, Ch'ungch'ong-namdo
Korea, Republic of, Cheju-do
Korea, Republic of, Cholla-bukto
Korea, Republic of, Cholla-namdo
Korea, Republic of, Inch'on-jikhalsi
Korea, Republic of, Kangwon-do
Korea, Republic of, Kwangju-jikhalsi
Korea, Republic of, Kyonggi-do
Korea, Republic of, Kyongsang-bukto
Korea, Republic of, Kyongsang-namdo
Korea, Republic of, Pusan-jikhalsi
Korea, Republic of, Seoul-t'ukpyolsi
Korea, Republic of, Taegu-jikhalsi
Korea, Republic of, Taejon-jikhalsi
Korea, Republic of, Ulsan-gwangyoksi
Kuwait, Al Ahmadi
Kuwait, Al Farwaniyah
Kuwait, Al Jahra
Kuwait, Al Kuwayt
Kuwait, Hawalli
Kuwait, Mubarak al Kabir
Kyrgyzstan, Batken
Kyrgyzstan, Bishkek
Kyrgyzstan, Chuy
Kyrgyzstan, Jalal-Abad

Kyrgyzstan, Naryn
Kyrgyzstan, Osh
Kyrgyzstan, Osh
Kyrgyzstan, Talas
Kyrgyzstan, Ysyk-Kol
Lao People's Democratic Republic, Attapu
Lao People's Democratic Republic, Champasak
Lao People's Democratic Republic, Houaphan
Lao People's Democratic Republic, Khammouan
Lao People's Democratic Republic, Louang Namtha
Lao People's Democratic Republic, Louangphrabang
Lao People's Democratic Republic, Oudomxai
Lao People's Democratic Republic, Phongsali
Lao People's Democratic Republic, Saravan
Lao People's Democratic Republic, Savannakhet
Lao People's Democratic Republic, Vientiane
Lao People's Democratic Republic, Xaignabouri
Lao People's Democratic Republic, Xiangkhoang
Latvia, Aizkraukles
Latvia, Aluksnes
Latvia, Balvu
Latvia, Bauskas
Latvia, Cesu
Latvia, Daugavpils
Latvia, Daugavpils
Latvia, Dobeles
Latvia, Gulbenes
Latvia, Jekabpils
Latvia, Jelgava
Latvia, Jelgavas
Latvia, Jurmala
Latvia, Kraslavas
Latvia, Kuldigas
Latvia, Liepaja
Latvia, Liepajas
Latvia, Limbazu
Latvia, Ludzas
Latvia, Madonas
Latvia, Ogres
Latvia, Preiliu
Latvia, Rezekne
Latvia, Rezeknes
Latvia, Riga
Latvia, Rigas
Latvia, Saldus
Latvia, Talsu
Latvia, Tukuma
Latvia, Valkas
Latvia, Valmieras
Latvia, Ventspils
Latvia, Ventspils
Lebanon, Aakk
Lebanon, Al Janub
Lebanon, Baalbek-Hermel
Lebanon, Beqaa
Lebanon, Beqaa
Lebanon, Beyrouth
Lebanon, Liban-Nord
Lebanon, Liban-Nord
Lebanon, Liban-Sud
Lebanon, Mont-Liban
Lebanon, Nabatiye
Lesotho, Berea
Lesotho, Butha-Buthe
Lesotho, Leribe
Lesotho, Mafeteng
Lesotho, Maseru
Lesotho, Mohales Hoek
Lesotho, Mokhotlong
Lesotho, Qachas Nek
Lesotho, Quthing

Lesotho, Thaba-Tseka
Liberia, Bong
Liberia, Gbarpolu
Liberia, Grand Bassa
Liberia, Grand Cape Mount
Liberia, Grand Cape Mount
Liberia, Grand Gedeh
Liberia, Lofa
Liberia, Lofa
Liberia, Margibi
Liberia, Maryland
Liberia, Maryland
Liberia, Monrovia
Liberia, Montserrado
Liberia, Nimba
Liberia, River Cess
Liberia, River Gee
Liberia, Sino
Libyan Arab Jamahiriya, Ajdabiya
Libyan Arab Jamahiriya, Al Aziziyah
Libyan Arab Jamahiriya, Al Fatih
Libyan Arab Jamahiriya, Al Jabal al Akhdar
Libyan Arab Jamahiriya, Al Jufrah
Libyan Arab Jamahiriya, Al Khums
Libyan Arab Jamahiriya, Al Kufrah
Libyan Arab Jamahiriya, An Nuqat al Khams
Libyan Arab Jamahiriya, Ash Shati'
Libyan Arab Jamahiriya, Awbari
Libyan Arab Jamahiriya, Az Zawiyah
Libyan Arab Jamahiriya, Banghazi
Libyan Arab Jamahiriya, Darnah
Libyan Arab Jamahiriya, Ghadamis
Libyan Arab Jamahiriya, Gharyan
Libyan Arab Jamahiriya, Misratah
Libyan Arab Jamahiriya, Murzuq
Libyan Arab Jamahiriya, Sabha
Libyan Arab Jamahiriya, Sawfajjin
Libyan Arab Jamahiriya, Surt
Libyan Arab Jamahiriya, Tarabulus
Libyan Arab Jamahiriya, Tarhunah
Libyan Arab Jamahiriya, Tubruq
Libyan Arab Jamahiriya, Yafran
Libyan Arab Jamahiriya, Zlitan
Liechtenstein, Balzers
Liechtenstein, Eschen
Liechtenstein, Gamprin
Liechtenstein, Gbarpolu
Liechtenstein, Mauren
Liechtenstein, Planken
Liechtenstein, River Gee
Liechtenstein, Ruggell
Liechtenstein, Schaan
Liechtenstein, Schellenberg
Liechtenstein, Triesen
Liechtenstein, Triesenberg
Liechtenstein, Vaduz
Lithuania, Alytaus Apskritis
Lithuania, Kauno Apskritis
Lithuania, Klaipedos Apskritis
Lithuania, Marijampoles Apskritis
Lithuania, Panevezio Apskritis
Lithuania, Siauli Apskritis
Lithuania, Taurages Apskritis
Lithuania, Telsiu Apskritis
Lithuania, Utenos Apskritis
Lithuania, Vilniaus Apskritis
Luxembourg, Diekirch
Luxembourg, Grevenmacher
Luxembourg, Luxembourg
Macau, Ilhas
Macau, Macau

Macedonia, Aracinovo
Macedonia, Bac
Macedonia, Belcista
Macedonia, Berovo
Macedonia, Bistrica
Macedonia, Bitola
Macedonia, Blatec
Macedonia, Bogdanci
Macedonia, Bogomila
Macedonia, Bogovinje
Macedonia, Bosilovo
Macedonia, Brvenica
Macedonia, Cair
Macedonia, Capari
Macedonia, Caska
Macedonia, Cegrane
Macedonia, Centar
Macedonia, Centar Zupa
Macedonia, Cesinovo
Macedonia, Cucer-Sandevo
Macedonia, Debar
Macedonia, Delcevo
Macedonia, Delogozdi
Macedonia, Demir Hisar
Macedonia, Demir Kapija
Macedonia, Dobrusevo
Macedonia, Dolna Banjica
Macedonia, Dolneni
Macedonia, Dorce Petrov
Macedonia, Drugovo
Macedonia, Dzepciste
Macedonia, Gazi Baba
Macedonia, Gevgelija
Macedonia, Gostivar
Macedonia, Gradsko
Macedonia, Ilinden
Macedonia, Izvor
Macedonia, Jegunovce
Macedonia, Kamenjane
Macedonia, Karbinci
Macedonia, Karpos
Macedonia, Kavadarci
Macedonia, Kicevo
Macedonia, Kisela Voda
Macedonia, Klecevce
Macedonia, Kocani
Macedonia, Konce
Macedonia, Kondovo
Macedonia, Konopiste
Macedonia, Kosel
Macedonia, Kratovo
Macedonia, Kriva Palanka
Macedonia, Krivogastani
Macedonia, Krusevo
Macedonia, Kuklis
Macedonia, Kukurecani
Macedonia, Kumanovo
Macedonia, Labunista
Macedonia, Lipkovo
Macedonia, Lozovo
Macedonia, Lukovo
Macedonia, Makedonska Kamenica
Macedonia, Makedonski Brod
Macedonia, Mavrovi Anovi
Macedonia, Meseista
Macedonia, Miravci
Macedonia, Mogila
Macedonia, Murtino
Macedonia, Negotino
Macedonia, Negotino-Polosko
Macedonia, Novaci

Macedonia, Novo Selo
Macedonia, Oblisevo
Macedonia, Ohrid
Macedonia, Orasac
Macedonia, Orizari
Macedonia, Oslomej
Macedonia, Pehcevo
Macedonia, Petровец
Macedonia, Plasnica
Macedonia, Podares
Macedonia, Prilep
Macedonia, Probistip
Macedonia, Radovis
Macedonia, Rankovce
Macedonia, Resen
Macedonia, Rosoman
Macedonia, Rostusa
Macedonia, Samokov
Macedonia, Saraj
Macedonia, Sipkovica
Macedonia, Sopiste
Macedonia, Sopotnica
Macedonia, Srbinovo
Macedonia, Star Dojran
Macedonia, Staravina
Macedonia, Staro Nagoricane
Macedonia, Stip
Macedonia, Struga
Macedonia, Strumica
Macedonia, Studenicani
Macedonia, Suto Orizari
Macedonia, Sveti Nikole
Macedonia, Tearce
Macedonia, Tetovo
Macedonia, Topolcani
Macedonia, Valandovo
Macedonia, Vasilevo
Macedonia, Veles
Macedonia, Velesta
Macedonia, Vevcani
Macedonia, Vinica
Macedonia, Vitoliste
Macedonia, Vranestica
Macedonia, Vrapciste
Macedonia, Vratnica
Macedonia, Vrutok
Macedonia, Zajas
Macedonia, Zelenikovo
Macedonia, Zelino
Macedonia, Zitose
Macedonia, Zletovo
Macedonia, Znovci
Madagascar, Antananarivo
Madagascar, Antsiranana
Madagascar, Fianarantsoa
Madagascar, Mahajanga
Madagascar, Toamasina
Madagascar, Toliara
Malawi, Balaka
Malawi, Blantyre
Malawi, Chikwawa
Malawi, Chiradzulu
Malawi, Chitipa
Malawi, Dedza
Malawi, Dowa
Malawi, Karonga
Malawi, Kasungu
Malawi, Likoma
Malawi, Lilongwe
Malawi, Machinga
Malawi, Mangochi

Malawi, Mchinji
Malawi, Mulanje
Malawi, Mwanza
Malawi, Mzimba
Malawi, Nkhata Bay
Malawi, Nkhotakota
Malawi, Nsanje
Malawi, Ntcheu
Malawi, Ntchisi
Malawi, Phalombe
Malawi, Rumphi
Malawi, Salima
Malawi, Thyolo
Malawi, Zomba
Malaysia, Johor
Malaysia, Kedah
Malaysia, Kelantan
Malaysia, Kuala Lumpur
Malaysia, Labuan
Malaysia, Melaka
Malaysia, Negeri Sembilan
Malaysia, Pahang
Malaysia, Perak
Malaysia, Perlis
Malaysia, Pulau Pinang
Malaysia, Putrajaya
Malaysia, Sabah
Malaysia, Sarawak
Malaysia, Selangor
Malaysia, Terengganu
Maldives, Alifu
Maldives, Baa
Maldives, Dhaalu
Maldives, Faafu
Maldives, Gaafu Alifu
Maldives, Gaafu Dhaalu
Maldives, Gnaviyani
Maldives, Haa Alifu
Maldives, Haa Dhaalu
Maldives, Kaafu
Maldives, Laamu
Maldives, Lhaviyani
Maldives, Maale
Maldives, Meemu
Maldives, Noonu
Maldives, Raa
Maldives, Seenu
Maldives, Shaviyani
Maldives, Thaa
Maldives, Vaavu
Mali, Bamako
Mali, Gao
Mali, Kayes
Mali, Kidal
Mali, Koulikoro
Mali, Mopti
Mali, Segou
Mali, Sikasso
Mali, Tombouctou
Mauritania, Adrar
Mauritania, Assaba
Mauritania, Brakna
Mauritania, Dakhlet Nouadhibou
Mauritania, Gorgol
Mauritania, Guidimaka
Mauritania, Hodh Ech Chargui
Mauritania, Hodh El Gharbi
Mauritania, Inchiri
Mauritania, Tagant
Mauritania, Tiris Zemmour
Mauritania, Trarza

Mauritius, Agalega Islands
Mauritius, Black River
Mauritius, Cargados Carajos
Mauritius, Flacq
Mauritius, Grand Port
Mauritius, Moka
Mauritius, Pamplemousses
Mauritius, Plaines Wilhems
Mauritius, Port Louis
Mauritius, Riviere du Rempart
Mauritius, Rodrigues
Mauritius, Savanne
Mexico, Aguascalientes
Mexico, Baja California
Mexico, Baja California Sur
Mexico, Campeche
Mexico, Chiapas
Mexico, Chihuahua
Mexico, Coahuila de Zaragoza
Mexico, Colima
Mexico, Distrito Federal
Mexico, Durango
Mexico, Guanajuato
Mexico, Guerrero
Mexico, Hidalgo
Mexico, Jalisco
Mexico, Mexico
Mexico, Michoacan de Ocampo
Mexico, Morelos
Mexico, Nayarit
Mexico, Nuevo Leon
Mexico, Oaxaca
Mexico, Puebla
Mexico, Queretaro de Arteaga
Mexico, Quintana Roo
Mexico, San Luis Potosi
Mexico, Sinaloa
Mexico, Sonora
Mexico, Tabasco
Mexico, Tamaulipas
Mexico, Tlaxcala
Mexico, Veracruz-Llave
Mexico, Yucatan
Mexico, Zacatecas
Micronesia, Chuuk
Micronesia, Kosrae
Micronesia, Pohnpei
Micronesia, Yap
Moldova, Republic of, Anenii Noi
Moldova, Republic of, Balti
Moldova, Republic of, Basarabeasca
Moldova, Republic of, Bender
Moldova, Republic of, Briceni
Moldova, Republic of, Cahul
Moldova, Republic of, Calarasi
Moldova, Republic of, Cantemir
Moldova, Republic of, Causeni
Moldova, Republic of, Chisinau
Moldova, Republic of, Cimislia
Moldova, Republic of, Criuleni
Moldova, Republic of, Donduseni
Moldova, Republic of, Drochia
Moldova, Republic of, Dubasari
Moldova, Republic of, Edinet
Moldova, Republic of, Falesti
Moldova, Republic of, Floresti
Moldova, Republic of, Gagauzia
Moldova, Republic of, Glodeni
Moldova, Republic of, Hincesti
Moldova, Republic of, Ialoveni
Moldova, Republic of, Leova

Moldova, Republic of, Nisporeni
Moldova, Republic of, Ocnița
Moldova, Republic of, Rezina
Moldova, Republic of, Rîșcani
Moldova, Republic of, Singerei
Moldova, Republic of, Soldanesti
Moldova, Republic of, Soroca
Moldova, Republic of, Stefan-Voda
Moldova, Republic of, Stinga Nistrului
Moldova, Republic of, Strășeni
Moldova, Republic of, Taraclia
Moldova, Republic of, Telenesti
Moldova, Republic of, Ungheni
Monaco, La Condamine
Monaco, Monaco
Monaco, Monte-Carlo
Mongolia, Arhangay
Mongolia, Bayan-Olgii
Mongolia, Bayanhongor
Mongolia, Bulgan
Mongolia, Darhan
Mongolia, Darhan-Uul
Mongolia, Dornod
Mongolia, Dornogovi
Mongolia, Dundgoví
Mongolia, Dzavhan
Mongolia, Erdenet
Mongolia, Govi-Altay
Mongolia, Govisumber
Mongolia, Hentiy
Mongolia, Hovd
Mongolia, Hovsgol
Mongolia, Omnogovi
Mongolia, Orhon
Mongolia, Ovorhangay
Mongolia, Selenge
Mongolia, Suhbaatar
Mongolia, Tov
Mongolia, Ulaanbaatar
Mongolia, Uvs
Montserrat, Saint Anthony
Montserrat, Saint Georges
Montserrat, Saint Peter
Morocco, Chaouia-Ouardigha
Morocco, Doukkala-Abda
Morocco, Fes-Boulemane
Morocco, Gharb-Chrarda-Beni Hssen
Morocco, Grand Casablanca
Morocco, Guelmim-Es Smara
Morocco, La
Morocco, Marrakech-Tensift-Al Haouz
Morocco, Meknes-Tafilalet
Morocco, Oriental
Morocco, Rabat-Sale-Zemmour-Zaer
Morocco, Souss-Massa-Dr
Morocco, Tadla-Azilal
Morocco, Tanger-Tetouan
Morocco, Taza-Al Hoceima-Taounate
Mozambique, Cabo Delgado
Mozambique, Gaza
Mozambique, Inhambane
Mozambique, Manica
Mozambique, Maputo
Mozambique, Maputo
Mozambique, Nampula
Mozambique, Niassa
Mozambique, Sofala
Mozambique, Tete
Mozambique, Zambezí
Myanmar, Chin State
Myanmar, Irrawaddy

Myanmar, Kachin State
Myanmar, Karan State
Myanmar, Kayah State
Myanmar, Magwe
Myanmar, Mandalay
Myanmar, Mon State
Myanmar, Pegu
Myanmar, Rakhine State
Myanmar, Rangoon
Myanmar, Sagaing
Myanmar, Shan State
Myanmar, Tenasserim
Myanmar, Yangon

N - S

Namibia, Bethanien
Namibia, Boesmanland
Namibia, Caprivi
Namibia, Caprivi Oos
Namibia, Damaraland
Namibia, Erongo
Namibia, Gobabis
Namibia, Grootfontein
Namibia, Hardap
Namibia, Hereroland Oos
Namibia, Hereroland Wes
Namibia, Kaokoland
Namibia, Karas
Namibia, Karasburg
Namibia, Karibib
Namibia, Kavango
Namibia, Keetmanshoop
Namibia, Kunene
Namibia, Luderitz
Namibia, Maltahohe
Namibia, Mariental
Namibia, Namaland
Namibia, Ohangwena
Namibia, Okahandja
Namibia, Okavango
Namibia, Omaheke
Namibia, Omaruru
Namibia, Omusati
Namibia, Oshana
Namibia, Oshikoto
Namibia, Otjiwarongo
Namibia, Otjozondjupa
Namibia, Outjo
Namibia, Owambo
Namibia, Rehoboth
Namibia, Swakopmund
Namibia, Tsumeb
Namibia, Windhoek
Nauru, Aiwo
Nauru, Anabar
Nauru, Anetan
Nauru, Anibare
Nauru, Baiti
Nauru, Boe
Nauru, Buada
Nauru, Denigomodu
Nauru, Ewa
Nauru, Ijuw
Nauru, Meneng
Nauru, Nibok
Nauru, Uaboe
Nauru, Yaren
Nepal, Bagmati
Nepal, Bheri
Nepal, Dhawalagiri

Nepal, Gandaki
Nepal, Janakpur
Nepal, Karnali
Nepal, Kosi
Nepal, Lumbini
Nepal, Mahakali
Nepal, Mechi
Nepal, Narayani
Nepal, Rapti
Nepal, Sagarmatha
Nepal, Seti
Netherlands, Drenthe
Netherlands, Flevoland
Netherlands, Friesland
Netherlands, Gelderland
Netherlands, Groningen
Netherlands, Limburg
Netherlands, Noord-Brabant
Netherlands, Noord-Holland
Netherlands, Overijssel
Netherlands, Overijssel
Netherlands, Utrecht
Netherlands, Zeeland
Netherlands, Zuid-Holland
New Zealand, Auckland
New Zealand, Bay of Plenty
New Zealand, Canterbury
New Zealand, Chatham Islands
New Zealand, Gisborne
New Zealand, Hawke's Bay
New Zealand, Manawatu-Wanganui
New Zealand, Marlborough
New Zealand, Nelson
New Zealand, Northland
New Zealand, Otago
New Zealand, Southland
New Zealand, Taranaki
New Zealand, Waikato
New Zealand, Wellington
New Zealand, West Coast
Nicaragua, Autonoma Atlantico Norte
Nicaragua, Boaco
Nicaragua, Carazo
Nicaragua, Chinandega
Nicaragua, Chontales
Nicaragua, Esteli
Nicaragua, Granada
Nicaragua, Jinotega
Nicaragua, Leon
Nicaragua, Madriz
Nicaragua, Managua
Nicaragua, Masaya
Nicaragua, Matagalpa
Nicaragua, Nueva Segovia
Nicaragua, Region Autonoma Atlantico Sur
Nicaragua, Rio San Juan
Nicaragua, Rivas
Nicaragua, Zelaya
Niger, Agadez
Niger, Diffa
Niger, Dosso
Niger, Maradi
Niger, Niamey
Niger, Niamey
Niger, Tahoua
Niger, Zinder
Nigeria, Abia
Nigeria, Adamawa
Nigeria, Akwa Ibom
Nigeria, Anambra
Nigeria, Bauchi

Nigeria, Bayelsa
Nigeria, Benue
Nigeria, Borno
Nigeria, Cross River
Nigeria, Delta
Nigeria, Ebonyi
Nigeria, Edo
Nigeria, Ekiti
Nigeria, Enugu
Nigeria, Federal Capital Territory
Nigeria, Gombe
Nigeria, Imo
Nigeria, Jigawa
Nigeria, Kaduna
Nigeria, Kano
Nigeria, Katsina
Nigeria, Kebbi
Nigeria, Kogi
Nigeria, Kwara
Nigeria, Lagos
Nigeria, Nassarawa
Nigeria, Niger
Nigeria, Ogun
Nigeria, Ondo
Nigeria, Osun
Nigeria, Oyo
Nigeria, Plateau
Nigeria, Rivers
Nigeria, Sokoto
Nigeria, Taraba
Nigeria, Yobe
Nigeria, Zamfara
Norway, Akershus
Norway, Aust-Agder
Norway, Buskerud
Norway, Finnmark
Norway, Hedmark
Norway, Hordaland
Norway, More og Romsdal
Norway, Nord-Trøndelag
Norway, Nordland
Norway, Oppland
Norway, Oslo
Norway, Østfold
Norway, Rogaland
Norway, Sogn og Fjordane
Norway, Sor-Trøndelag
Norway, Telemark
Norway, Troms
Norway, Vest-Agder
Norway, Vestfold
Oman, Ad Dakhiliyah
Oman, Al Batinah
Oman, Al Wusta
Oman, Ash Sharqiyah
Oman, Az Zahirah
Oman, Masqat
Oman, Musandam
Oman, Zufar
Pakistan, Azad Kashmir
Pakistan, Balochistan
Pakistan, Federally Administered Tribal Areas
Pakistan, Islamabad
Pakistan, North-West Frontier
Pakistan, Northern Areas
Pakistan, Punjab
Pakistan, Sindh
Palestinian Territory, Occupied, Gaza
Palestinian Territory, Occupied, West Bank
Panama, Bocas del Toro
Panama, Chiriquí

Panama, Coclé
Panama, Colón
Panama, Darién
Panama, Herrera
Panama, Los Santos
Panama, Panamá
Panama, San Blas
Panama, Veraguas
Papua New Guinea, Central
Papua New Guinea, Chimbu
Papua New Guinea, East New Britain
Papua New Guinea, East Sepik
Papua New Guinea, Eastern Highlands
Papua New Guinea, Enga
Papua New Guinea, Gulf
Papua New Guinea, Madang
Papua New Guinea, Manus
Papua New Guinea, Milne Bay
Papua New Guinea, Morobe
Papua New Guinea, National Capital
Papua New Guinea, New Ireland
Papua New Guinea, North Solomons
Papua New Guinea, Northern
Papua New Guinea, Sandaun
Papua New Guinea, Southern Highlands
Papua New Guinea, West New Britain
Papua New Guinea, Western
Papua New Guinea, Western Highlands
Paraguay, Alto Paraguay
Paraguay, Alto Paraná
Paraguay, Amambay
Paraguay, Boquerón
Paraguay, Caaguazú
Paraguay, Caazapá
Paraguay, Canindeyú
Paraguay, Central
Paraguay, Chaco
Paraguay, Concepción
Paraguay, Cordillera
Paraguay, Guairá
Paraguay, Itapúa
Paraguay, Misiones
Paraguay, Neembucú
Paraguay, Nueva Asunción
Paraguay, Paraguari
Paraguay, Presidente Hayes
Paraguay, San Pedro
Peru, Amazonas
Peru, Ancash
Peru, Apurímac
Peru, Arequipa
Peru, Ayacucho
Peru, Cajamarca
Peru, Callao
Peru, Cusco
Peru, Huancavelica
Peru, Huanuco
Peru, Ica
Peru, Junín
Peru, La Libertad
Peru, Lambayeque
Peru, Lima
Peru, Loreto
Peru, Madre de Dios
Peru, Moquegua
Peru, Pasco
Peru, Piura
Peru, Puno
Peru, San Martín
Peru, Tacna
Peru, Tumbes

Peru, Ucayali
Philippines, Abra
Philippines, Agusan del Norte
Philippines, Agusan del Sur
Philippines, Aklan
Philippines, Albay
Philippines, Angeles
Philippines, Antique
Philippines, Aurora
Philippines, Bacolod
Philippines, Bago
Philippines, Baguio
Philippines, Bais
Philippines, Basilan
Philippines, Basilan City
Philippines, Bataan
Philippines, Batanes
Philippines, Batangas
Philippines, Batangas City
Philippines, Benguet
Philippines, Bohol
Philippines, Bukidnon
Philippines, Bulacan
Philippines, Butuan
Philippines, Cabanatuan
Philippines, Cadiz
Philippines, Cagayan
Philippines, Cagayan de Oro
Philippines, Calbayog
Philippines, Caloocan
Philippines, Camarines Norte
Philippines, Camarines Sur
Philippines, Camiguin
Philippines, Canlaon
Philippines, Capiz
Philippines, Catanduanes
Philippines, Cavite
Philippines, Cavite City
Philippines, Cebu
Philippines, Cebu City
Philippines, Cotabato
Philippines, Dagupan
Philippines, Danao
Philippines, Dapitan
Philippines, Davao
Philippines, Davao City
Philippines, Davao del Sur
Philippines, Davao Oriental
Philippines, Dipolog
Philippines, Dumaguete
Philippines, Eastern Samar
Philippines, General Santos
Philippines, Ginggoog
Philippines, Ifugao
Philippines, Iligan
Philippines, Ilocos Norte
Philippines, Ilocos Sur
Philippines, Iloilo
Philippines, Iloilo City
Philippines, Iriga
Philippines, Isabela
Philippines, Kalinga-Apayao
Philippines, La Carlota
Philippines, La Union
Philippines, Laguna
Philippines, Lanao del Norte
Philippines, Lanao del Sur
Philippines, Laoag
Philippines, Lapu-Lapu
Philippines, Legaspi
Philippines, Leyte

Philippines, Lipa
Philippines, Lucena
Philippines, Maguindanao
Philippines, Mandaue
Philippines, Manila
Philippines, Marawi
Philippines, Marinduque
Philippines, Masbate
Philippines, Mindoro Occidental
Philippines, Mindoro Oriental
Philippines, Misamis Occidental
Philippines, Misamis Oriental
Philippines, Mountain
Philippines, Naga
Philippines, Negros Occidental
Philippines, Negros Occidental
Philippines, Negros Oriental
Philippines, North Cotabato
Philippines, Northern Samar
Philippines, Nueva Ecija
Philippines, Nueva Vizcaya
Philippines, Olongapo
Philippines, Ormoc
Philippines, Oroquieta
Philippines, Ozamis
Philippines, Pagadian
Philippines, Palawan
Philippines, Palayan
Philippines, Pampanga
Philippines, Pangasinan
Philippines, Pasay
Philippines, Puerto Princesa
Philippines, Quezon
Philippines, Quezon City
Philippines, Quirino
Philippines, Rizal
Philippines, Romblon
Philippines, Roxas
Philippines, Samar
Philippines, San Carlos
Philippines, San Carlos
Philippines, San Jose
Philippines, San Pablo
Philippines, Silay
Philippines, Siquijor
Philippines, Sorsogon
Philippines, South Cotabato
Philippines, Southern Leyte
Philippines, Sultan Kudarat
Philippines, Sulu
Philippines, Surigao
Philippines, Surigao del Norte
Philippines, Surigao del Sur
Philippines, Tacloban
Philippines, Tagaytay
Philippines, Tagbilaran
Philippines, Tangub
Philippines, Tarlac
Philippines, Tawitawi
Philippines, Toledo
Philippines, Trece Martires
Philippines, Zambales
Philippines, Zamboanga
Philippines, Zamboanga del Norte
Philippines, Zamboanga del Sur
Poland, Dolnoslaskie
Poland, Kujawsko-Pomorskie
Poland, Lodzkie
Poland, Lubelskie
Poland, Lubuskie
Poland, Malopolskie

Poland, Mazowieckie
Poland, Opolskie
Poland, Podkarpackie
Poland, Podlaskie
Poland, Pomorskie
Poland, Slaskie
Poland, Swietokrzyskie
Poland, Warminsko-Mazurskie
Poland, Wielkopolskie
Poland, Zachodniopomorskie
Portugal, Aveiro
Portugal, Azores
Portugal, Beja
Portugal, Braga
Portugal, Braganca
Portugal, Castelo Branco
Portugal, Coimbra
Portugal, Evora
Portugal, Faro
Portugal, Guarda
Portugal, Leiria
Portugal, Lisboa
Portugal, Madeira
Portugal, Portalegre
Portugal, Porto
Portugal, Santarem
Portugal, Setubal
Portugal, Viana do Castelo
Portugal, Vila Real
Portugal, Viseu
Qatar, Ad Dawhah
Qatar, Al Ghuwariyah
Qatar, Al Jumaliyah
Qatar, Al Khawr
Qatar, Al Wakrah
Qatar, Al Wakrah Municipality
Qatar, Ar Rayyan
Qatar, Jariyan al Batnah
Qatar, Madinat ach Shamal
Qatar, Umm Sa'id
Qatar, Umm Salal
Romania, Alba
Romania, Arad
Romania, Arges
Romania, Bacau
Romania, Bihor
Romania, Bistrita-Nasaud
Romania, Botosani
Romania, Braila
Romania, Brasov
Romania, Bucuresti
Romania, Buzau
Romania, Calarasi
Romania, Caras-Severin
Romania, Cluj
Romania, Constanta
Romania, Covasna
Romania, Dambovita
Romania, Dolj
Romania, Galati
Romania, Giurgiu
Romania, Gorj
Romania, Harghita
Romania, Hunedoara
Romania, Ialomita
Romania, Iasi
Romania, Ilfov
Romania, Maramures
Romania, Mehedinți
Romania, Mureș
Romania, Neamț

Romania, Olt
Romania, Prahova
Romania, Salaj
Romania, Satu Mare
Romania, Sibiu
Romania, Suceava
Romania, Teleorman
Romania, Timis
Romania, Tulcea
Romania, Valcea
Romania, Vaslui
Romania, Vrancea
Russian Federation, Adygeya
Russian Federation, Aginsky Buryatsky AO
Russian Federation, Altaisky krai
Russian Federation, Amur
Russian Federation, Arkhangel'sk
Russian Federation, Astrakhan'
Russian Federation, Bashkortostan
Russian Federation, Belgorod
Russian Federation, Bryansk
Russian Federation, Buryat
Russian Federation, Chechnya
Russian Federation, Chechnya Republic
Russian Federation, Chelyabinsk
Russian Federation, Chita
Russian Federation, Chukot
Russian Federation, Chuvashia
Russian Federation, Dagestan
Russian Federation, Evenk
Russian Federation, Gorno-Altay
Russian Federation, Ingush
Russian Federation, Irkutsk
Russian Federation, Ivanovo
Russian Federation, Kabardin-Balkar
Russian Federation, Kaliningrad
Russian Federation, Kalmyk
Russian Federation, Kaluga
Russian Federation, Kamchatka
Russian Federation, Karachay-Cherkess
Russian Federation, Karelia
Russian Federation, Kemerovo
Russian Federation, Khabarovsk
Russian Federation, Khakass
Russian Federation, Khanty-Mansi
Russian Federation, Kirov
Russian Federation, Komi
Russian Federation, Komi-Permyak
Russian Federation, Koryak
Russian Federation, Kostroma
Russian Federation, Krasnodar
Russian Federation, Krasnoyarsk
Russian Federation, Krasnoyarskiy Kray
Russian Federation, Kurgan
Russian Federation, Kursk
Russian Federation, Leningrad
Russian Federation, Lipetsk
Russian Federation, Magadan
Russian Federation, Mariy-El
Russian Federation, Mordovia
Russian Federation, Moscow City
Russian Federation, Moskva
Russian Federation, Murmansk
Russian Federation, Nenets
Russian Federation, Nizhegorod
Russian Federation, North Ossetia
Russian Federation, Novgorod
Russian Federation, Novosibirsk
Russian Federation, Omsk
Russian Federation, Orel
Russian Federation, Orenburg

Russian Federation, Penza
Russian Federation, Perm'
Russian Federation, Permskiy Kray
Russian Federation, Primor'ye
Russian Federation, Pskov
Russian Federation, Rostov
Russian Federation, Ryazan'
Russian Federation, Saint Petersburg City
Russian Federation, Sakha
Russian Federation, Sakhalin
Russian Federation, Samara
Russian Federation, Saratov
Russian Federation, Smolensk
Russian Federation, Stavropol'
Russian Federation, Sverdlovsk
Russian Federation, Tambovskaya oblast
Russian Federation, Tatarstan
Russian Federation, Taymyr
Russian Federation, Tomsk
Russian Federation, Tula
Russian Federation, Tuva
Russian Federation, Tver'
Russian Federation, Tyumen'
Russian Federation, Udmurt
Russian Federation, Ul'yanovsk
Russian Federation, Ust-Orda Buryat
Russian Federation, Vladimir
Russian Federation, Volgograd
Russian Federation, Vologda
Russian Federation, Voronezh
Russian Federation, Yamal-Nenets
Russian Federation, Yaroslavl'
Russian Federation, Yevrey
Rwanda, Butare
Rwanda, Est
Rwanda, Gitarama
Rwanda, Kibungo
Rwanda, Kigali
Rwanda, Kigali
Rwanda, Nord
Rwanda, Ouest
Rwanda, Sud
Saint Helena, Ascension
Saint Helena, Saint Helena
Saint Helena, Tristan da Cunha
Saint Kitts and Nevis, Christ Church Nichola Town
Saint Kitts and Nevis, Saint Anne Sandy Point
Saint Kitts and Nevis, Saint George Basseterre
Saint Kitts and Nevis, Saint George Gingerland
Saint Kitts and Nevis, Saint James Windward
Saint Kitts and Nevis, Saint John Capisterre
Saint Kitts and Nevis, Saint John Figtree
Saint Kitts and Nevis, Saint Mary Cayon
Saint Kitts and Nevis, Saint Paul Capisterre
Saint Kitts and Nevis, Saint Paul Charlestown
Saint Kitts and Nevis, Saint Peter Basseterre
Saint Kitts and Nevis, Saint Thomas Lowland
Saint Kitts and Nevis, Saint Thomas Middle Island
Saint Kitts and Nevis, Trinity Palmetto Point
Saint Lucia, Anse-la-Raye
Saint Lucia, Castries
Saint Lucia, Choiseul
Saint Lucia, Dauphin
Saint Lucia, Denney
Saint Lucia, Gros-Islet
Saint Lucia, Laborie
Saint Lucia, Micoud
Saint Lucia, Praslin
Saint Lucia, Soufriere
Saint Lucia, Vieux-Fort
Saint Vincent and the Grenadines, Charlotte

Saint Vincent and the Grenadines, Grenadines
Saint Vincent and the Grenadines, Saint Andrew
Saint Vincent and the Grenadines, Saint David
Saint Vincent and the Grenadines, Saint George
Saint Vincent and the Grenadines, Saint Patrick
Samoa, Aiga-i-le-Tai
Samoa, Atua
Samoa, Fa
Samoa, Gaga
Samoa, Gagaifomauga
Samoa, Palauli
Samoa, Satupa
Samoa, Tuamasaga
Samoa, Va
Samoa, Vaisigano
San Marino, Acquaviva
San Marino, Borgo Maggiore
San Marino, Chiesanuova
San Marino, Domagnano
San Marino, Faetano
San Marino, Fiorentino
San Marino, Monte Giardino
San Marino, San Marino
San Marino, Serravalle
Sao Tome and Principe, Principe
Sao Tome and Principe, Sao Tome
Saudi Arabia, Al Bahah
Saudi Arabia, Al Hudud ash Shamaliyah
Saudi Arabia, Al Jawf
Saudi Arabia, Al Jawf
Saudi Arabia, Al Madinah
Saudi Arabia, Al Qasim
Saudi Arabia, Al Qurayyat
Saudi Arabia, Ar Riyad
Saudi Arabia, Ash Sharqiyah
Saudi Arabia, Ha'il
Saudi Arabia, Jizan
Saudi Arabia, Makkah
Saudi Arabia, Najran
Saudi Arabia, Tabuk
Senegal, Dakar
Senegal, Diourbel
Senegal, Fatick
Senegal, Kaolack
Senegal, Kolda
Senegal, Louga
Senegal, Matam
Senegal, Saint-Louis
Senegal, Tambacounda
Senegal, Thies
Senegal, Ziguinchor
Serbia, Kosovo
Serbia, Vojvodina
Seychelles, Anse aux Pins
Seychelles, Anse Boileau
Seychelles, Anse Etoile
Seychelles, Anse Louis
Seychelles, Anse Royale
Seychelles, Baie Lazare
Seychelles, Baie Sainte Anne
Seychelles, Beau Vallon
Seychelles, Bel Air
Seychelles, Bel Ombre
Seychelles, Cascade
Seychelles, Glacis
Seychelles, Grand' Anse
Seychelles, Grand' Anse
Seychelles, La Digue
Seychelles, La Riviere Anglaise
Seychelles, Mont Buxton
Seychelles, Mont Fleuri

Seychelles, Plaisance
Seychelles, Pointe La Rue
Seychelles, Port Glaud
Seychelles, Saint Louis
Seychelles, Takamaka
Sierra Leone, Eastern
Sierra Leone, Northern
Sierra Leone, Southern
Sierra Leone, Western Area
Slovakia, Banska Bystrica
Slovakia, Bratislava
Slovakia, Kosice
Slovakia, Nitra
Slovakia, Presov
Slovakia, Trencin
Slovakia, Trnava
Slovakia, Zilina
Slovenia, Ajdovscina
Slovenia, Beltinci
Slovenia, Bled
Slovenia, Bohinj
Slovenia, Borovnica
Slovenia, Bovec
Slovenia, Brda
Slovenia, Brezice
Slovenia, Brezovica
Slovenia, Celje
Slovenia, Cerkle na Gorenjskem
Slovenia, Cerknica
Slovenia, Cerkno
Slovenia, Crenovci
Slovenia, Crna na Koroškem
Slovenia, Crnomelj
Slovenia, Divaca
Slovenia, Dobrepolje
Slovenia, Dobrova-Horjul-Polhov Gradec
Slovenia, Dol pri Ljubljani
Slovenia, Domzale
Slovenia, Dornava
Slovenia, Dravograd
Slovenia, Duplek
Slovenia, Gorenja Vas-Poljane
Slovenia, Gorisnica
Slovenia, Gornja Radgona
Slovenia, Gornji Grad
Slovenia, Gornji Petrovci
Slovenia, Grosuplje
Slovenia, Hrastnik
Slovenia, Hrpelje-Kozina
Slovenia, Idrija
Slovenia, Ig
Slovenia, Ilirska Bistrica
Slovenia, Ivancna Gorica
Slovenia, Izola-Isola
Slovenia, Jesenice
Slovenia, Jursinci
Slovenia, Kamnik
Slovenia, Kanal
Slovenia, Kidricevo
Slovenia, Kobarid
Slovenia, Kobilje
Slovenia, Kocevje
Slovenia, Komen
Slovenia, Koper-Capodistria
Slovenia, Kozje
Slovenia, Kranj
Slovenia, Kranjska Gora
Slovenia, Krsko
Slovenia, Kungota
Slovenia, Kuzma
Slovenia, Lasko

Slovenia, Lenart
Slovenia, Litija
Slovenia, Ljubljana
Slovenia, Ljubno
Slovenia, Ljutomer
Slovenia, Logatec
Slovenia, Loska Dolina
Slovenia, Loski Potok
Slovenia, Luce
Slovenia, Lukovica
Slovenia, Majsperek
Slovenia, Maribor
Slovenia, Medvode
Slovenia, Menges
Slovenia, Metlika
Slovenia, Mezica
Slovenia, Miren-Kostanjevica
Slovenia, Mislinja
Slovenia, Moravce
Slovenia, Moravske Toplice
Slovenia, Mozirje
Slovenia, Murska Sobota
Slovenia, Muta
Slovenia, Naklo
Slovenia, Nazarje
Slovenia, Nova Gorica
Slovenia, Novo Mesto
Slovenia, Odranci
Slovenia, Ormoz
Slovenia, Osilnica
Slovenia, Pesnica
Slovenia, Piran
Slovenia, Pivka
Slovenia, Podcetrtek
Slovenia, Postojna
Slovenia, Preddvor
Slovenia, Ptuj
Slovenia, Puconci
Slovenia, Racam
Slovenia, Radece
Slovenia, Radenci
Slovenia, Radlje ob Dravi
Slovenia, Radovljica
Slovenia, Ribnica
Slovenia, Rogaska Slatina
Slovenia, Rogasovci
Slovenia, Rogatec
Slovenia, Ruse
Slovenia, Semic
Slovenia, Sencur
Slovenia, Sentilj
Slovenia, Sentjernej
Slovenia, Sentjur pri Celju
Slovenia, Sevnica
Slovenia, Sezana
Slovenia, Skocjan
Slovenia, Skofja Loka
Slovenia, Skofljica
Slovenia, Slovenj Gradec
Slovenia, Slovenska Bistrica
Slovenia, Slovenske Konjice
Slovenia, Smarje pri Jelsah
Slovenia, Smartno ob Paki
Slovenia, Sostanj
Slovenia, Starse
Slovenia, Store
Slovenia, Sveti Jurij
Slovenia, Tolmin
Slovenia, Trbovlje
Slovenia, Trebnje
Slovenia, Trzic

Slovenia, Turnisce
Slovenia, Velenje
Slovenia, Velike Lasce
Slovenia, Videm
Slovenia, Vipava
Slovenia, Vitanje
Slovenia, Vodice
Slovenia, Vojnik
Slovenia, Vrhnika
Slovenia, Vuzenica
Slovenia, Zagorje ob Savi
Slovenia, Zalec
Slovenia, Zavrc
Slovenia, Zelezniki
Slovenia, Ziri
Slovenia, Zrece
Solomon Islands, Central
Solomon Islands, Choiseul
Solomon Islands, Guadalcanal
Solomon Islands, Isabel
Solomon Islands, Makira
Solomon Islands, Malaita
Solomon Islands, Rennell and Bellona
Solomon Islands, Temotu
Solomon Islands, Western
Somalia, Awdal
Somalia, Bakool
Somalia, Banaadir
Somalia, Bari
Somalia, Bay
Somalia, Galguduud
Somalia, Gedo
Somalia, Hiiraan
Somalia, Jubbada Dhexe
Somalia, Jubbada Hoose
Somalia, Mudug
Somalia, Nugaal
Somalia, Nugaal
Somalia, Sanaag
Somalia, Shabeellaha Dhexe
Somalia, Shabeellaha Hoose
Somalia, Sool
Somalia, Togdheer
Somalia, Woqooyi Galbeed
Somalia, Woqooyi Galbeed
South Africa, Eastern Cape
South Africa, Free State
South Africa, Gauteng
South Africa, KwaZulu-Natal
South Africa, Limpopo
South Africa, Mpumalanga
South Africa, North-West
South Africa, North-Western Province
South Africa, Northern Cape
South Africa, Western Cape
Spain, Andalucia
Spain, Aragon
Spain, Asturias
Spain, Canarias
Spain, Cantabria
Spain, Castilla y Leon
Spain, Castilla-La Mancha
Spain, Catalonia
Spain, Comunidad Valenciana
Spain, Extremadura
Spain, Galicia
Spain, Islas Baleares
Spain, La Rioja
Spain, Madrid
Spain, Murcia
Spain, Navarra

Spain, Pais Vasco
Sri Lanka, Amparai
Sri Lanka, Anuradhapura
Sri Lanka, Badulla
Sri Lanka, Batticaloa
Sri Lanka, Central
Sri Lanka, Colombo
Sri Lanka, Galle
Sri Lanka, Gampaha
Sri Lanka, Hambantota
Sri Lanka, Jaffna
Sri Lanka, Kalutara
Sri Lanka, Kandy
Sri Lanka, Kegalla
Sri Lanka, Kurunegala
Sri Lanka, Mannar
Sri Lanka, Matale
Sri Lanka, Matara
Sri Lanka, Moneragala
Sri Lanka, Mullaitivu
Sri Lanka, North Central
Sri Lanka, North Western
Sri Lanka, Northern
Sri Lanka, Nuwara Eliya
Sri Lanka, Polonnaruwa
Sri Lanka, Puttalam
Sri Lanka, Ratnapura
Sri Lanka, Sabaragamuwa
Sri Lanka, Southern
Sri Lanka, Trincomalee
Sri Lanka, Uva
Sri Lanka, Vavuniya
Sri Lanka, Western
Sudan, Al Istiwa'iyah
Sudan, Al Khartum
Sudan, Al Wahadah State
Sudan, Al Wusta
Sudan, Ash Shamaliyah
Sudan, Ash Sharqiyah
Sudan, Bahr al Ghazal
Sudan, Central Equatoria State
Sudan, Darfur
Sudan, Kurdufan
Sudan, Upper Nile
Suriname, Brokopondo
Suriname, Commewijne
Suriname, Coronie
Suriname, Marowijne
Suriname, Nickerie
Suriname, Para
Suriname, Paramaribo
Suriname, Saramacca
Suriname, Sipaliwini
Suriname, Wanica
Swaziland, Hhohho
Swaziland, Lubombo
Swaziland, Manzini
Swaziland, Praslin
Swaziland, Shiselweni
Sweden, Blekinge Lan
Sweden, Dalarnas Lan
Sweden, Gavleborgs Lan
Sweden, Gotlands Lan
Sweden, Hallands Lan
Sweden, Jamtlands Lan
Sweden, Jonkopings Lan
Sweden, Kalmar Lan
Sweden, Kronobergs Lan
Sweden, Norrbottens Lan
Sweden, Orebro Lan
Sweden, Ostergotlands Lan

Sweden, Skane Lan
Sweden, Sodermanlands Lan
Sweden, Stockholms Lan
Sweden, Uppsala Lan
Sweden, Varmlands Lan
Sweden, Vasterbottens Lan
Sweden, Vasternorrlands Lan
Sweden, Vastmanlands Lan
Sweden, Vastra Gotaland
Switzerland, Aargau
Switzerland, Ausser-Rhoden
Switzerland, Basel-Landschaft
Switzerland, Basel-Stadt
Switzerland, Bern
Switzerland, Fribourg
Switzerland, Geneve
Switzerland, Glarus
Switzerland, Graubunden
Switzerland, Inner-Rhoden
Switzerland, Jura
Switzerland, Luzern
Switzerland, Neuchatel
Switzerland, Nidwalden
Switzerland, Obwalden
Switzerland, Sankt Gallen
Switzerland, Schaffhausen
Switzerland, Schwyz
Switzerland, Solothurn
Switzerland, Thurgau
Switzerland, Ticino
Switzerland, Uri
Switzerland, Valais
Switzerland, Vaud
Switzerland, Zug
Switzerland, Zurich
Syrian Arab Republic, Al Hasakah
Syrian Arab Republic, Al Ladhiqiyah
Syrian Arab Republic, Al Qunaytirah
Syrian Arab Republic, Ar Raqqah
Syrian Arab Republic, As Suwayda'
Syrian Arab Republic, Dar
Syrian Arab Republic, Dayr az Zawr
Syrian Arab Republic, Dimashq
Syrian Arab Republic, Halab
Syrian Arab Republic, Hamah
Syrian Arab Republic, Hims
Syrian Arab Republic, Idlib
Syrian Arab Republic, Rif Dimashq
Syrian Arab Republic, Tartus

T - Z

Taiwan, Fu-chien
Taiwan, Kao-hsiung
Taiwan, T'ai-pei
Taiwan, T'ai-wan
Tajikistan, Khatlon
Tajikistan, Kuhistoni Badakhshon
Tajikistan, Sughd
Tanzania, Arusha
Tanzania, Dar es Salaam
Tanzania, Dodoma
Tanzania, Iringa
Tanzania, Kagera
Tanzania, Kigoma
Tanzania, Kilimanjaro
Tanzania, Lindi
Tanzania, Manyara
Tanzania, Mara
Tanzania, Mbeya
Tanzania, Morogoro

Tanzania, Mtwara
Tanzania, Mwanza
Tanzania, Pemba North
Tanzania, Pemba South
Tanzania, Pwani
Tanzania, Rukwa
Tanzania, Ruvuma
Tanzania, Shinyanga
Tanzania, Singida
Tanzania, Tabora
Tanzania, Tanga
Tanzania, Zanzibar Central
Tanzania, Zanzibar North
Tanzania, Zanzibar Urban
Thailand, Amnat Charoen
Thailand, Ang Thong
Thailand, Buriram
Thailand, Chachoengsao
Thailand, Chai Nat
Thailand, Chaiyaphum
Thailand, Chanthaburi
Thailand, Chiang Mai
Thailand, Chiang Rai
Thailand, Chon Buri
Thailand, Chumphon
Thailand, Kalasin
Thailand, Kamphaeng Phet
Thailand, Kanchanaburi
Thailand, Khon Kaen
Thailand, Krabi
Thailand, Krung Thep
Thailand, Lampang
Thailand, Lamphun
Thailand, Loei
Thailand, Lop Buri
Thailand, Mae Hong Son
Thailand, Maha Sarakham
Thailand, Mukdahan
Thailand, Nakhon Nayok
Thailand, Nakhon Pathom
Thailand, Nakhon Phanom
Thailand, Nakhon Phanom
Thailand, Nakhon Ratchasima
Thailand, Nakhon Sawan
Thailand, Nakhon Si Thammarat
Thailand, Nan
Thailand, Narathiwat
Thailand, Nong Bua Lamphu
Thailand, Nong Khai
Thailand, Nonthaburi
Thailand, Pathum Thani
Thailand, Pattani
Thailand, Phangnga
Thailand, Phatthalung
Thailand, Phayao
Thailand, Phetchabun
Thailand, Phetchaburi
Thailand, Phichit
Thailand, Phitsanulok
Thailand, Phra Nakhon Si Ayutthaya
Thailand, Phrae
Thailand, Phuket
Thailand, Prachin Buri
Thailand, Prachuap Khiri Khan
Thailand, Ranong
Thailand, Ratchaburi
Thailand, Rayong
Thailand, Roi Et
Thailand, Sa Kaeo
Thailand, Sakon Nakhon
Thailand, Samut Prakan

Thailand, Samut Sakhon
Thailand, Samut Songkhram
Thailand, Saraburi
Thailand, Satun
Thailand, Sing Buri
Thailand, Sisaket
Thailand, Songkhla
Thailand, Sukhothai
Thailand, Suphan Buri
Thailand, Surat Thani
Thailand, Surin
Thailand, Tak
Thailand, Trang
Thailand, Trat
Thailand, Ubon Ratchathani
Thailand, Ubon Ratchathani
Thailand, Udon Thani
Thailand, Uthai Thani
Thailand, Uttaradit
Thailand, Yala
Thailand, Yasothon
The Bahamas, Acklins and Crooked Islands
The Bahamas, Bimini
The Bahamas, Cat Island
The Bahamas, Exuma
The Bahamas, Freeport
The Bahamas, Fresh Creek
The Bahamas, Governor's Harbour
The Bahamas, Green Turtle Cay
The Bahamas, Harbour Island
The Bahamas, High Rock
The Bahamas, Inagua
The Bahamas, Kemps Bay
The Bahamas, Long Island
The Bahamas, Marsh Harbour
The Bahamas, Mayaguana
The Bahamas, New Providence
The Bahamas, Nichollstown and Berry Islands
The Bahamas, Ragged Island
The Bahamas, Rock Sound
The Bahamas, San Salvador and Rum Cay
The Bahamas, Sandy Point
Togo, Centrale
Togo, Kara
Togo, Maritime
Togo, Plateaux
Togo, Savanes
Tonga, Ha
Tonga, Tongatapu
Tonga, Vava
Trinidad and Tobago, Arima
Trinidad and Tobago, Caroni
Trinidad and Tobago, Mayaro
Trinidad and Tobago, Nariva
Trinidad and Tobago, Port-of-Spain
Trinidad and Tobago, Saint Andrew
Trinidad and Tobago, Saint David
Trinidad and Tobago, Saint George
Trinidad and Tobago, Saint Patrick
Trinidad and Tobago, San Fernando
Trinidad and Tobago, Tobago
Trinidad and Tobago, Victoria
Tunisia, Aiana
Tunisia, Al Mahdia
Tunisia, Al Munastir
Tunisia, Bahaj
Tunisia, Ben Arous
Tunisia, Bizerte
Tunisia, El Kef
Tunisia, Gabes
Tunisia, Jendouba

Tunisia, Kairouan
Tunisia, Kasserine
Tunisia, Kebili
Tunisia, Madanin
Tunisia, Manouba
Tunisia, Nabeul
Tunisia, Qafsa
Tunisia, Sfax
Tunisia, Sidi Bou Zid
Tunisia, Siliana
Tunisia, Sousse
Tunisia, Tataouine
Tunisia, Tozeur
Tunisia, Tunis
Tunisia, Zaghouan
Turkey, Adana
Turkey, Adiyaman
Turkey, Afyonkarahisar
Turkey, Agri
Turkey, Aksaray
Turkey, Amasya
Turkey, Ankara
Turkey, Antalya
Turkey, Ardahan
Turkey, Artvin
Turkey, Aydin
Turkey, Balikesir
Turkey, Bartin
Turkey, Batman
Turkey, Bayburt
Turkey, Bilecik
Turkey, Bingol
Turkey, Bitlis
Turkey, Bolu
Turkey, Burdur
Turkey, Bursa
Turkey, Canakkale
Turkey, Cankiri
Turkey, Corum
Turkey, Denizli
Turkey, Diyarbakir
Turkey, Duzce
Turkey, Edirne
Turkey, Elazig
Turkey, Erzincan
Turkey, Erzurum
Turkey, Eskisehir
Turkey, Gaziantep
Turkey, Giresun
Turkey, Gumushane
Turkey, Hakkari
Turkey, Hatay
Turkey, Igdir
Turkey, Isparta
Turkey, Istanbul
Turkey, Izmir
Turkey, Kahramanmaraş
Turkey, Karabuk
Turkey, Karaman
Turkey, Kars
Turkey, Kastamonu
Turkey, Kayseri
Turkey, Kilis
Turkey, Kirikkale
Turkey, Kırklareli
Turkey, Kırşehir
Turkey, Kocaeli
Turkey, Konya
Turkey, Kutahya
Turkey, Malatya
Turkey, Manisa

Turkey, Mardin
Turkey, Mersin
Turkey, Mugla
Turkey, Mus
Turkey, Nevsehir
Turkey, Nigde
Turkey, Ordu
Turkey, Osmaniye
Turkey, Rize
Turkey, Sakarya
Turkey, Samsun
Turkey, Sanliurfa
Turkey, Siirt
Turkey, Sinop
Turkey, Sirnak
Turkey, Sivas
Turkey, Tekirdag
Turkey, Tokat
Turkey, Trabzon
Turkey, Tunceli
Turkey, Usak
Turkey, Van
Turkey, Yalova
Turkey, Yozgat
Turkey, Zonguldak
Turkmenistan, Ahal
Turkmenistan, Balkan
Turkmenistan, Dashoguz
Turkmenistan, Lebap
Turkmenistan, Mary
Uganda, Adjumani
Uganda, Apac
Uganda, Arua
Uganda, Bugiri
Uganda, Bundibugyo
Uganda, Bushenyi
Uganda, Busia
Uganda, Gulu
Uganda, Hoima
Uganda, Iganga
Uganda, Jinja
Uganda, Kabarole
Uganda, Kaberamaido
Uganda, Kalangala
Uganda, Kampala
Uganda, Kamuli
Uganda, Kamwenge
Uganda, Kanungu
Uganda, Kapchorwa
Uganda, Kasese
Uganda, Katakwi
Uganda, Kayunga
Uganda, Kibale
Uganda, Kiboga
Uganda, Kisoro
Uganda, Kitgum
Uganda, Kotido
Uganda, Kumi
Uganda, Kyenjojo
Uganda, Lira
Uganda, Luwero
Uganda, Masaka
Uganda, Masindi
Uganda, Mayuge
Uganda, Mbale
Uganda, Mbarara
Uganda, Moroto
Uganda, Moyo
Uganda, Mpigi
Uganda, Mubende
Uganda, Mukono

Uganda, Nakapiripirit
Uganda, Nakasongola
Uganda, Nebbi
Uganda, Ntungamo
Uganda, Pader
Uganda, Pallisa
Uganda, Rakai
Uganda, Rukungiri
Uganda, Sembabule
Uganda, Sironko
Uganda, Soroti
Uganda, Tororo
Uganda, Wakiso
Uganda, Yumbe
Ukraine, Cherkas'ka Oblast'
Ukraine, Chernihivs'ka Oblast'
Ukraine, Chernivets'ka Oblast'
Ukraine, Dnipropetrovs'ka Oblast'
Ukraine, Donets'ka Oblast'
Ukraine, Ivano-Frankivs'ka Oblast'
Ukraine, Kharkivs'ka Oblast'
Ukraine, Khersons'ka Oblast'
Ukraine, Khmel'nyts'ka Oblast'
Ukraine, Kirovohrads'ka Oblast'
Ukraine, Krym
Ukraine, Kyiv
Ukraine, Kyyivs'ka Oblast'
Ukraine, L'vivs'ka Oblast'
Ukraine, Luhans'ka Oblast'
Ukraine, Mykolayivs'ka Oblast'
Ukraine, Odes'ka Oblast'
Ukraine, Poltavs'ka Oblast'
Ukraine, Rivnens'ka Oblast'
Ukraine, Sevastopol'
Ukraine, Sums'ka Oblast'
Ukraine, Ternopil's'ka Oblast'
Ukraine, Vinnyts'ka Oblast'
Ukraine, Volyns'ka Oblast'
Ukraine, Zakarpats'ka Oblast'
Ukraine, Zaporiz'ka Oblast'
Ukraine, Zhytomyr's'ka Oblast'
United Arab Emirates, Abu Dhabi
United Arab Emirates, Ajman
United Arab Emirates, Dubai
United Arab Emirates, Fujairah
United Arab Emirates, Ras Al Khaimah
United Arab Emirates, Sharjah
United Arab Emirates, Umm Al Quwain
United Kingdom, Aberdeen City
United Kingdom, Aberdeenshire
United Kingdom, Angus
United Kingdom, Antrim
United Kingdom, Ards
United Kingdom, Argyll and Bute
United Kingdom, Armagh
United Kingdom, Ballymena
United Kingdom, Ballymoney
United Kingdom, Banbridge
United Kingdom, Barking and Dagenham
United Kingdom, Barnet
United Kingdom, Barnsley
United Kingdom, Bath and North East Somerset
United Kingdom, Bedfordshire
United Kingdom, Belfast
United Kingdom, Bexley
United Kingdom, Birmingham
United Kingdom, Blackburn with Darwen
United Kingdom, Blackpool
United Kingdom, Blaenau Gwent
United Kingdom, Bolton
United Kingdom, Bournemouth

United Kingdom, Bracknell Forest
United Kingdom, Bradford
United Kingdom, Brent
United Kingdom, Bridgend
United Kingdom, Brighton and Hove
United Kingdom, Bristol
United Kingdom, Bromley
United Kingdom, Buckinghamshire
United Kingdom, Bury
United Kingdom, Caerphilly
United Kingdom, Calderdale
United Kingdom, Cambridgeshire
United Kingdom, Camden
United Kingdom, Cardiff
United Kingdom, Carmarthenshire
United Kingdom, Carrickfergus
United Kingdom, Castlereagh
United Kingdom, Ceredigion
United Kingdom, Cheshire
United Kingdom, Clackmannanshire
United Kingdom, Coleraine
United Kingdom, Conwy
United Kingdom, Cookstown
United Kingdom, Cornwall
United Kingdom, Coventry
United Kingdom, Craigavon
United Kingdom, Croydon
United Kingdom, Cumbria
United Kingdom, Darlington
United Kingdom, Denbighshire
United Kingdom, Derby
United Kingdom, Derbyshire
United Kingdom, Derry
United Kingdom, Devon
United Kingdom, Doncaster
United Kingdom, Dorset
United Kingdom, Down
United Kingdom, Dudley
United Kingdom, Dumfries and Galloway
United Kingdom, Dundee City
United Kingdom, Dungannon
United Kingdom, Durham
United Kingdom, Ealing
United Kingdom, East Ayrshire
United Kingdom, East Dunbartonshire
United Kingdom, East Lothian
United Kingdom, East Renfrewshire
United Kingdom, East Riding of Yorkshire
United Kingdom, East Sussex
United Kingdom, Edinburgh
United Kingdom, Eilean Siar
United Kingdom, Enfield
United Kingdom, Essex
United Kingdom, Falkirk
United Kingdom, Fermanagh
United Kingdom, Fife
United Kingdom, Flintshire
United Kingdom, Gateshead
United Kingdom, Glasgow City
United Kingdom, Gloucestershire
United Kingdom, Greenwich
United Kingdom, Gwynedd
United Kingdom, Hackney
United Kingdom, Halton
United Kingdom, Hammersmith and Fulham
United Kingdom, Hampshire
United Kingdom, Haringey
United Kingdom, Harrow
United Kingdom, Hartlepool
United Kingdom, Havering
United Kingdom, Herefordshire

United Kingdom, Hertford
United Kingdom, Highland
United Kingdom, Hillingdon
United Kingdom, Hounslow
United Kingdom, Inverclyde
United Kingdom, Isle of Anglesey
United Kingdom, Isle of Wight
United Kingdom, Islington
United Kingdom, Kensington and Chelsea
United Kingdom, Kent
United Kingdom, Kingston upon Hull
United Kingdom, Kingston upon Thames
United Kingdom, Kirklees
United Kingdom, Knowsley
United Kingdom, Lambeth
United Kingdom, Lancashire
United Kingdom, Larne
United Kingdom, Leeds
United Kingdom, Leicester
United Kingdom, Leicestershire
United Kingdom, Lewisham
United Kingdom, Limavady
United Kingdom, Lincolnshire
United Kingdom, Lisburn
United Kingdom, Liverpool
United Kingdom, London
United Kingdom, Luton
United Kingdom, Magherafelt
United Kingdom, Manchester
United Kingdom, Medway
United Kingdom, Merthyr Tydfil
United Kingdom, Merton
United Kingdom, Middlesbrough
United Kingdom, Midlothian
United Kingdom, Milton Keynes
United Kingdom, Monmouthshire
United Kingdom, Moray
United Kingdom, Moyle
United Kingdom, Neath Port Talbot
United Kingdom, Newcastle upon Tyne
United Kingdom, Newham
United Kingdom, Newport
United Kingdom, Newry and Mourne
United Kingdom, Newtownabbey
United Kingdom, Norfolk
United Kingdom, North Ayrshire
United Kingdom, North Down
United Kingdom, North East Lincolnshire
United Kingdom, North Lanarkshire
United Kingdom, North Lincolnshire
United Kingdom, North Somerset
United Kingdom, North Tyneside
United Kingdom, North Yorkshire
United Kingdom, Northamptonshire
United Kingdom, Northumberland
United Kingdom, Nottingham
United Kingdom, Nottinghamshire
United Kingdom, Oldham
United Kingdom, Omagh
United Kingdom, Orkney
United Kingdom, Oxfordshire
United Kingdom, Pembrokeshire
United Kingdom, Perth and Kinross
United Kingdom, Peterborough
United Kingdom, Plymouth
United Kingdom, Poole
United Kingdom, Portsmouth
United Kingdom, Powys
United Kingdom, Reading
United Kingdom, Redbridge
United Kingdom, Redcar and Cleveland

United Kingdom, Renfrewshire
United Kingdom, Rhondda Cynon Taff
United Kingdom, Richmond upon Thames
United Kingdom, Rochdale
United Kingdom, Rotherham
United Kingdom, Rutland
United Kingdom, Salford
United Kingdom, Sandwell
United Kingdom, Scottish Borders
United Kingdom, Sefton
United Kingdom, Sheffield
United Kingdom, Shetland Islands
United Kingdom, Shropshire
United Kingdom, Slough
United Kingdom, Solihull
United Kingdom, Somerset
United Kingdom, South Ayrshire
United Kingdom, South Gloucestershire
United Kingdom, South Lanarkshire
United Kingdom, South Tyneside
United Kingdom, Southampton
United Kingdom, Southend-on-Sea
United Kingdom, Southwark
United Kingdom, St. Helens
United Kingdom, Staffordshire
United Kingdom, Stirling
United Kingdom, Stockport
United Kingdom, Stockton-on-Tees
United Kingdom, Stoke-on-Trent
United Kingdom, Strabane
United Kingdom, Suffolk
United Kingdom, Sunderland
United Kingdom, Surrey
United Kingdom, Sutton
United Kingdom, Swansea
United Kingdom, Swindon
United Kingdom, Tameside
United Kingdom, Telford and Wrekin
United Kingdom, Thurrock
United Kingdom, Torbay
United Kingdom, Torfaen
United Kingdom, Tower Hamlets
United Kingdom, Trafford
United Kingdom, Vale of Glamorgan
United Kingdom, Wakefield
United Kingdom, Walsall
United Kingdom, Waltham Forest
United Kingdom, Wandsworth
United Kingdom, Warrington
United Kingdom, Warwickshire
United Kingdom, West Berkshire
United Kingdom, West Dunbartonshire
United Kingdom, West Lothian
United Kingdom, West Sussex
United Kingdom, Westminster
United Kingdom, Wigan
United Kingdom, Wiltshire
United Kingdom, Windsor and Maidenhead
United Kingdom, Wirral
United Kingdom, Wokingham
United Kingdom, Wolverhampton
United Kingdom, Worcestershire
United Kingdom, Wrexham
United Kingdom, York
United States, Alabama
United States, Alaska
United States, American Samoa
United States, Arizona
United States, Arkansas
United States, Armed Forces Americas
United States, Armed Forces Europe

United States, Armed Forces Pacific
United States, California
United States, Colorado
United States, Connecticut
United States, Delaware
United States, District of Columbia
United States, Federated States of Micronesia
United States, Florida
United States, Georgia
United States, Guam
United States, Hawaii
United States, Idaho
United States, Illinois
United States, Indiana
United States, Iowa
United States, Kansas
United States, Kentucky
United States, Louisiana
United States, Maine
United States, Marshall Islands
United States, Maryland
United States, Massachusetts
United States, Michigan
United States, Minnesota
United States, Mississippi
United States, Missouri
United States, Montana
United States, Nebraska
United States, Nevada
United States, New Hampshire
United States, New Jersey
United States, New Mexico
United States, New York
United States, North Carolina
United States, North Dakota
United States, Northern Mariana Islands
United States, Ohio
United States, Oklahoma
United States, Oregon
United States, Palau
United States, Pennsylvania
United States, Puerto Rico
United States, Rhode Island
United States, South Carolina
United States, South Dakota
United States, Tennessee
United States, Texas
United States, Utah
United States, Vermont
United States, Virgin Islands
United States, Virginia
United States, Washington
United States, West Virginia
United States, Wisconsin
United States, Wyoming
Uruguay, Artigas
Uruguay, Canelones
Uruguay, Cerro Largo
Uruguay, Colonia
Uruguay, Durazno
Uruguay, Flores
Uruguay, Florida
Uruguay, Lavalleja
Uruguay, Maldonado
Uruguay, Montevideo
Uruguay, Paysandu
Uruguay, Rio Negro
Uruguay, Rivera
Uruguay, Rocha
Uruguay, Salto
Uruguay, San Jose

Uruguay, Soriano
Uruguay, Tacuarembo
Uruguay, Treinta y Tres
Uzbekistan, Andijon
Uzbekistan, Bukhoro
Uzbekistan, Farghona
Uzbekistan, Jizzakh
Uzbekistan, Khorazm
Uzbekistan, Namangan
Uzbekistan, Navoiy
Uzbekistan, Qashqadaryo
Uzbekistan, Qoraqalpoghiston
Uzbekistan, Samarcand
Uzbekistan, Sirdaryo
Uzbekistan, Surkhondaryo
Uzbekistan, Toshkent
Uzbekistan, Toshkent
Vanuatu, Ambrym
Vanuatu, Aoba
Vanuatu, Efate
Vanuatu, Epi
Vanuatu, Malakula
Vanuatu, Malampa
Vanuatu, Paama
Vanuatu, Penama
Vanuatu, Pentecote
Vanuatu, Sanma
Vanuatu, Shefa
Vanuatu, Shepherd
Vanuatu, Tafea
Vanuatu, Torba
Venezuela, Amazonas
Venezuela, Anzoategui
Venezuela, Apure
Venezuela, Aragua
Venezuela, Barinas
Venezuela, Bolivar
Venezuela, Carabobo
Venezuela, Cojedes
Venezuela, Delta Amacuro
Venezuela, Dependencias Federales
Venezuela, Distrito Federal
Venezuela, Falcon
Venezuela, Guarico
Venezuela, Lara
Venezuela, Merida
Venezuela, Miranda
Venezuela, Monagas
Venezuela, Nueva Esparta
Venezuela, Portuguesa
Venezuela, Sucre
Venezuela, Tachira
Venezuela, Trujillo
Venezuela, Vargas
Venezuela, Yaracuy
Venezuela, Zulia
Vietnam, An Giang
Vietnam, An Giang
Vietnam, Ba Ria-Vung Tau
Vietnam, Ben Tre
Vietnam, Binh Dinh
Vietnam, Binh Thuan
Vietnam, Can Tho
Vietnam, Cao Bang
Vietnam, Da Nang
Vietnam, Dac Lac
Vietnam, Dak Lak
Vietnam, Dak Nong
Vietnam, Dien Bien
Vietnam, Dong Nai
Vietnam, Dong Thap

Vietnam, Dong Thap
Vietnam, Ha Giang
Vietnam, Ha Nam
Vietnam, Ha Noi
Vietnam, Ha Tay
Vietnam, Ha Tinh
Vietnam, Hai Duong
Vietnam, Hai Phong
Vietnam, Hau Giang
Vietnam, Ho Chi Minh
Vietnam, Ho Chi Minh
Vietnam, Hoa Binh
Vietnam, Hung Yen
Vietnam, Khanh Hoa
Vietnam, Kien Giang
Vietnam, Kien Giang
Vietnam, Kon Tum
Vietnam, Lai Chau
Vietnam, Lam Dong
Vietnam, Lang Son
Vietnam, Lao Cai
Vietnam, Long An
Vietnam, Nam Dinh
Vietnam, Nam Ha
Vietnam, Nghe An
Vietnam, Ninh Binh
Vietnam, Ninh Thuan
Vietnam, Phu Tho
Vietnam, Phu Yen
Vietnam, Quang Binh
Vietnam, Quang Nam
Vietnam, Quang Ngai
Vietnam, Quang Ninh
Vietnam, Quang Tri
Vietnam, Quang Tri
Vietnam, Soc Trang
Vietnam, Son La
Vietnam, Song Be
Vietnam, Tay Ninh
Vietnam, Thai Binh
Vietnam, Thai Nguyen
Vietnam, Thanh Hoa
Vietnam, Thua Thien
Vietnam, Tien Giang
Vietnam, Tra Vinh
Vietnam, Tuyen Quang
Vietnam, Vinh Long
Vietnam, Vinh Phu
Vietnam, Vinh Puc Province
Yemen, Abyan
Yemen, Adan
Yemen, Al Bayda'
Yemen, Al Ghaydah
Yemen, Al Hudaydah
Yemen, Al Jawf
Yemen, Al Mahrah
Yemen, Al Mahwit
Yemen, Dhamar
Yemen, Hadramawt
Yemen, Hajjah
Yemen, Ibb
Yemen, Lahij
Yemen, Ma'rib
Yemen, Sa
Yemen, San
Yemen, Shabwah
Yemen, Ta
Zambia, Central
Zambia, Copperbelt
Zambia, Eastern
Zambia, Luapula

Zambia, Lusaka
Zambia, North-Western
Zambia, Northern
Zambia, Southern
Zambia, Western
Zimbabwe, Bulawayo
Zimbabwe, Harare
Zimbabwe, Manicaland
Zimbabwe, Mashonaland Central
Zimbabwe, Mashonaland East
Zimbabwe, Mashonaland West
Zimbabwe, Masvingo
Zimbabwe, Matabeleland North
Zimbabwe, Matabeleland South
Zimbabwe, Midlands

EUM License

- [Learn More](#)

EUM requires a special EUM license which is separate from the regular AppDynamics license. You will be required to provide the EUM license key when you enable EUM.

If you are an existing AppDynamics customer upgrading to 3.7, contact your AppDynamics sales representative to provision your account for EUM.

If you are a new AppDynamics customer with 3.7 and you purchased AppDynamics through a sales representative, your EUM license key is included in your welcome email from AppDynamics.

If you purchased AppDynamics directly from the website, your trial version of AppDynamics has EUM disabled. If you want to enable EUM, contact AppDynamics Support to obtain a license key.

Learn More

- [Configure End User Experience](#)

Background Task Monitoring

- [Background Tasks](#)
 - [Separating Background Tasks from Business Transactions](#)
 - [Background Tasks in the UI](#)
- [Enabling and Configuring Background Task Monitoring](#)
- [Learn More](#)

Background Tasks

A background task or a batch job is a scheduled program that runs without user intervention. These programs automate tasks that are performed on a regular basis. Batch jobs that are required to be processed on a regular basis are incorporated into batch schedules.

By default AppDynamics does not automatically discover most background tasks. In some cases AppDynamics may discover a task and interpret it as a business transaction.

Separating Background Tasks from Business Transactions

In some environments AppDynamics will automatically discover a background task and initially identify it as a business transaction. In this case you want to reclassify the task so that it is not included in the business transaction metrics.

AppDynamics lets you treat background tasks differently than transaction-based requests because background tasks usually take more time to execute. Monitoring background tasks together with the business transactions would skew the averages for response time for the tier and for the node where these background tasks are executed. Therefore, AppDynamics recommends that you monitor background tasks separately from business transactions.

Background tasks are not counted toward the average response time of the tier or the node.

Background Tasks in the UI

Although background tasks are monitored separately, they are displayed along with the other business transactions in the [Business Transactions List](#). They show the background task icon: .

AppDynamics provides visibility into the key performance metrics and code level visibility for background tasks. You can monitor background tasks in the following areas of the UI:

- The [Business Transactions List](#) shows currently executing background tasks and their statistics.
- The [Metric Browser](#) shows the response time for each background task.
- AppDynamics creates transaction snapshots of each execution of a background task. However, if a particular job runs too frequently then AppDynamics may not capture all details for each execution.

The tier and application level metrics are not collected for background tasks. On an application dashboard, background tasks will not show any call details.

Enabling and Configuring Background Task Monitoring

When you know that there are background tasks in your application environment and you want to monitor them, or to reclassify a business transaction as a background task, see [Configure Background Tasks](#).

Learn More

- [Configure Background Tasks](#)
- [Transaction Snapshots](#)

Configure Background Tasks

- [Enabling Automatic Discovery for Background Tasks](#)
 - To enable discovery for a background task using a common framework
- [Configuring Thresholds for Background Tasks](#)
 - To configure thresholds for all background tasks
 - To configure thresholds for an individual background task
- [Re-configuring a Business Transaction as a Background Task](#)
 - To re-configure a business transaction as a background task
- [Learn More](#)

Configuring background tasks consists of two basic steps:

1. [Enabling automatic discovery](#) so that AppDynamics will detect the background process and monitor it.
2. [Configuring thresholds](#) for what you consider to be slow, very slow, or stalled performance behavior.

Sometimes AppDynamics will discover a background task and interpret it to be a business transaction. You can instruct AppDynamics to [Configuring a Business Transaction as a Background Task](#).

Enabling Automatic Discovery for Background Tasks

Automatic discovery of background tasks is disabled by default. When you know that there are background tasks in your application environment and you want to monitor them, first enable automatic discovery so that AppDynamics will detect the task.

AppDynamics provides preconfigured support for some common frameworks. If your application is not using one of the default frameworks you can create a custom match rule.

To enable discovery for a background task using a common framework

1. In the left navigation pane, click [Configure -> Instrumentation](#).
2. On the Transaction Detection tab, select the tier for which you want to enable monitoring.
3. Click [Use Custom Configuration for this Tier](#).
4. Scroll down to the Custom Match Rules pane.
5. Do one of the following

- If you are using a pre-configured framework, select the row of the framework and click the pencil icon, or double-click on the row to open the Business Transaction Match Rule window. By default the values are populated with rule name and the class and method names for the particular framework. Verify that those are the correct names for your environment.
- OR
- If you are using a custom framework, select the match criteria and enter the Class Name and Method Name.

The Background Task check box should be already checked.

6. Check **Enabled**.

7. Click **Save**.

The custom match rule for the background task will take effect and the background task will display in the Business Transaction List.

Once you enable discovery, every background task is identified based on following attributes:

- Implementation class name
- Parameter to the execution method name

For additional details see:

[Configure Background Tasks \(Java\)](#)

Not Supported for .NET

Not Supported for PHP

Configuring Thresholds for Background Tasks

The out-of-the-box stall thresholds for background tasks are disabled. This is because the default configuration for stall detection is 45 seconds, which is usually not enough time to detect stalled background tasks.

AppDynamics recommends that you configure a threshold that is suitable for the background task in your environment.

AppDynamics also recommends that you use static thresholds for slow and very slow background tasks, if they have infrequent load patterns such as once every night. This is because the dynamic moving average-based thresholds are more suitable for production load scenarios and will automatically classify a background process as slow or very slow.

To configure thresholds for all background tasks

1. In the left navigation pane click **Configure -> Slow Transaction Thresholds**.
2. Click the Background Tasks Thresholds tab.
3. Set the thresholds for all background tasks.
4. If you want these thresholds applied to existing background tasks, check **Apply to all Existing Background Task Business Transactions**.
5. Click **Save Default Background Task Thresholds**.

All new background tasks will use the thresholds that are configured here.

You can override default thresholds for an individual background tasks by configuring individual task thresholds.

To configure thresholds for an individual background task

1. In the Business Transactions List, right-click on the background task and select **Configure Thresholds**. AppDynamics displays the threshold settings for that background task.
2. Update the thresholds.
3. Save the changes.

Re-configuring a Business Transaction as a Background Task

You can re-configure an auto-discovered business transaction as a background task.

To re-configure a business transaction as a background task

1. In the left navigation pane click **Business Transactions**.
2. In the Business Transactions List select the business transaction that you want to mark as a background task.
3. Right-click on the selected business transaction and select **Set as Background Task**.
4. Verify that **Set as Background Task** is selected in the dialog.
5. Click **OK**.

If you discover that a background task is better represented as a regular business transaction, repeat steps 1 and 2 then right-click and select **Set as User Transaction**.

Learn More

- [Configure Background Tasks \(Java\)](#)

Backend Monitoring

A backend is a service in your application environment that AppDynamics does not instrument directly; it monitors calls from the code of instrumented nodes out to the services and responses from the services.

Typical examples of backends are a database instance or messaging service. AppDynamics automatically discovers commonly used backends and you can configure it to discover others. See [Supported Backends for the Java Agent](#) and [Supported Backends for the .NET Agent](#).

Monitor Databases

- [Automatic Database Detection](#)
 - [Java Environment](#)
 - SQL Databases
 - NoSQL
 - [.NET Environment](#)
- [Database Visibility](#)
- [Slow Database Calls](#)
- [Deep Dive into Database Metrics](#)
- [Configuring Database Detection](#)
- [Learn More](#)

Calls from an instrumented node to a database are recognized as traffic related to specific business transactions. AppDynamics monitors the performance of calls to databases in two ways:

- Overall performance of calls to individual databases
- Call performance for specific business transactions

Metrics for the database calls and response times are collected at three levels:

- Business transaction metrics - the metrics for a specific business transaction for a specific database are visible on the [Transaction Flow Map](#).
- Tier metrics - the metrics for all calls from a tier to the specified database are visible on the [Tier Flow Map](#).
- Database metrics - the overall database access metrics across the application (all business transactions) are visible on the [Application Flow Map](#) and the [Database Dashboard](#).

Automatic Database Detection

Many databases are automatically detected when database calls are made from JVMs and CLRs instrumented with AppDynamics agents.

Java Environment

SQL Databases

AppDynamics automatically discovers JDBC databases. The databases are named according to a set of default rules, but can be renamed with a more easily understood display name. For a list of auto-discovered JDBC databases, see [Supported Databases for the Java Agent](#).

NoSQL

AppDynamics automatically discovers Cassandra databases and supports custom configuration for MongoDB.

Non-JDBC databases show up on flow maps as remote services.

.NET Environment

- ADO.NET
- Windows Azure Blob Storage

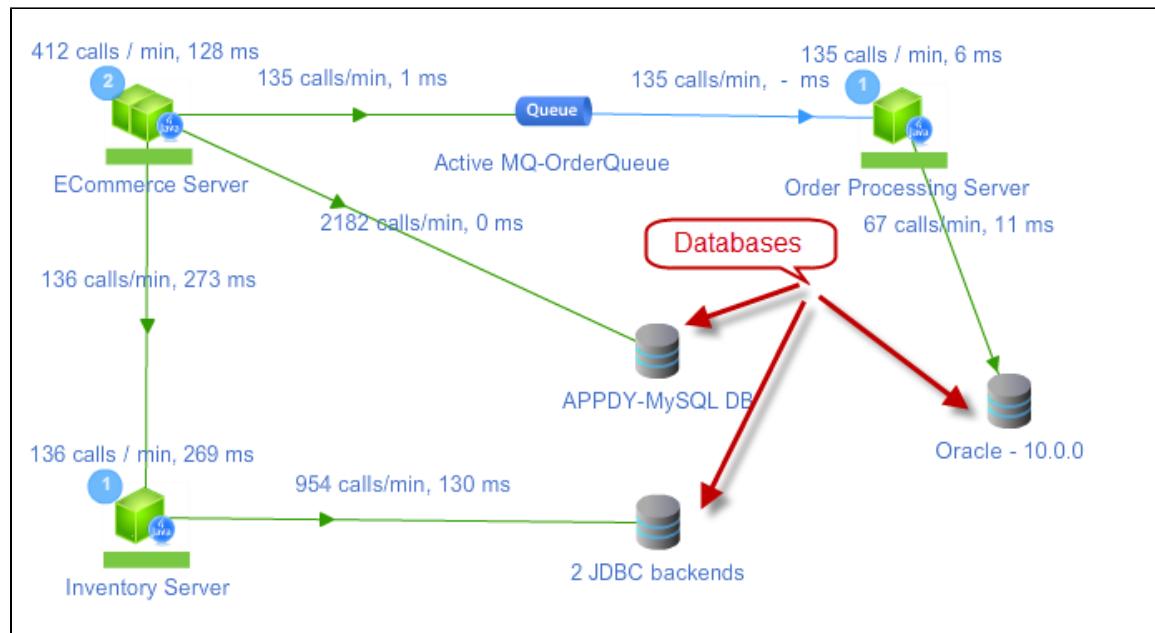
ADO.NET data providers implementing standard Microsoft interfaces are automatically discovered as backends. For a complete list, see [Supported ADO.NET Clients for the .NET Agent](#).

Because the ADO.NET API is interface-based, by default AppDynamics instruments all ADO.NET database providers that implement these interfaces. For database identification, AppDynamics uses the ADO.NET connection string. The connection string specifies the server address and schema or the local file name. Most connection strings are formatted according to well-known rules that can be parsed and distilled to a database name. However, because there is no standard on the connection string, it is up to the ADO.NET provider implementer to choose the format. For some providers, AppDynamics may fail to parse the connection string and generate an **Unknown** backend. If you see a backend named **Unknown**, you can use the connection string found in the backend properties to come up with actual backend name. The backend properties can be viewed on the **Database Dashboard**.

Database Visibility

- Application Flow Map

The detected databases show up on the Application Flow Map. You can view all the detected databases in the context of the entire application's transaction flow.



- Database List

You can view a list of the detected database servers that displays key performance indicators, such as response time, total calls, calls per minute, errors, and errors per minute.

Name	Response Time (ms)	Calls	Calls / min	Errors	Errors / min
APPDY-MySQL DB	0	268,124	2,234	0	0
INVENTORY-MySQL DB	0	98,595	822	0	0
Oracle - 10.0.0	313	16,419	137	0	0
Oracle - 10.0.0	11	8,014	67	0	0

- **Database Dashboard**

From the database list, you can select a database and click **View Dashboard** to see the Database Dashboard. The dashboard displays a Database Flow Map, database properties, and graphs of the key performance indicators (KPIs). The database properties are its identity and define what shows up in the display map.

Host:	LOCALHOST
Major Version:	5.5.16-log
Port:	3388
Schema:	APPDY
Url:	jdbc:mysql://localhost:3388/appdy
Vendor:	MySQL DB

Load 33,329 calls 2,222 / min

9:07 AM 9:11 AM 9:15 AM 9:19 AM

Response Time (ms) 0 ms average

9:07 AM 9:11 AM 9:15 AM 9:19 AM

Errors 0% 0 total 0 / min

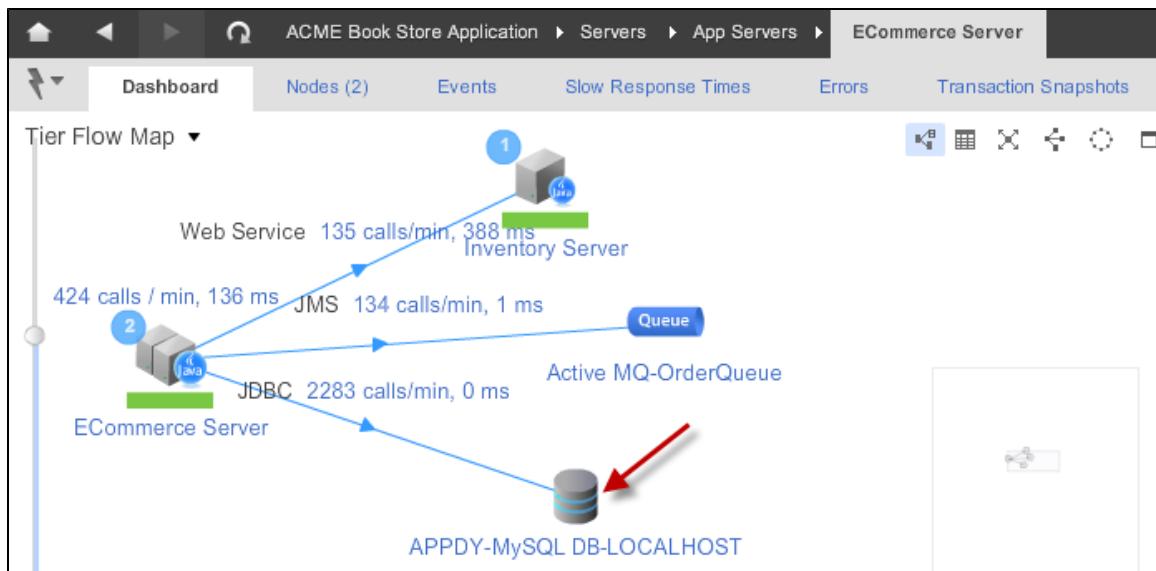
9:07 AM 9:11 AM 9:15 AM 9:19 AM

From this dashboard you can select the Slow Database Calls tab to view specific call details and find related business transaction snapshots to troubleshoot issues and find the root cause of database bottlenecks.

These are the calls with largest observed individual execution time (Max Time) during the specified time range.				
Calling Tier	Call	Avg. Time per Call (ms)	Number of Calls	Max Time (ms)
From	DELETE FROM CART	51.3	3	73
Calls from All Tiers	SELECT THIS_ID AS ID1_0_, THIS_TITLE AS TITLE1_0_, THIS_IMAGEPATH # 70	1	70	View snapshots
ECommerce Server				

- **Tier Flow Map**

The detected databases show up on the Tier Flow Map. You can view the detected databases in the context of the traffic on this specific tier. For example, the ECommerce Server tier, as shown in the following screen capture.



Slow Database Calls

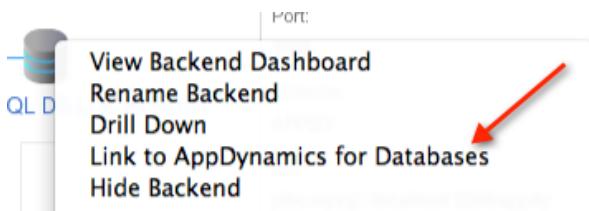
AppDynamics displays a list of the slowest database calls with call details so you can drill down to find problems. The Slowest Database Calls tab lists up to ten calls to the database with the longest execution time over the selected time frame, by tier and for all tiers. Each call shows the SQL Query, Average Time, Number of Calls during the time range, and maximum execution time (Max Time). The Max Time is used to determine which calls are displayed in the Slowest Database Calls list. For JDBC calls, the Max Time must exceed 10 ms before the call is tracked as a potential candidate for this list. The Agent aggregates and reports database call data every 15 minutes. See [Troubleshoot Slow Response Times](#).

To monitor call performance to a database, first make sure it shows up in the Database Server List and has its own [dashboard](#).

If a database is not appearing, check the configuration. See [Configure Backend Detection](#).

Deep Dive into Database Metrics

Users of AppDynamics for Databases can link to that product by right-clicking on a database from the database list or from the database icon on any flow map.



Configuring Database Detection

You can customize the detection and naming rules and configure the detection of additional databases:

- Configure Backend Detection
- Configure Custom Exit Points (Java)
- Configure Custom Exit Points (.NET)
- Configurations for Custom Exit Points
- Match Rule Conditions
- Integrate with AppDynamics for Databases

Learn More

- Configure Stale Backend Removal
- Troubleshoot Slow Response Times
- Backend Monitoring

Monitor Remote Services

- Automatic Remote Service Detection
 - Java
 - .NET
- Remote Service Visibility
- Slow Remote Services Calls
- Configuring Remote Service Detection
- Learn More

Calls from an instrumented node to a remote service, such as a message queue or web service, are recognized as traffic related to specific business transactions. AppDynamics monitors the performance of calls to remote services in two ways:

- Overall performance of calls to individual remote services
- Call performance for specific business transactions

A remote service provides a service to a distributed application. It resides outside of the application server. Examples are a Java Message Service or Web Service. The remote services servers are not instrumented directly, but you can monitor calls to them from instrumented app servers.

Metrics about the remote service calls and response times are collected at three levels:

- Business transaction metrics - the metrics for a specific business transaction for a specific service are visible on the [Transaction Flow Map](#).
- Tier metrics - the metrics for all calls from a tier to the specified service are visible on the [Tier Flow Map](#).
- Remote Service metrics - the overall remote service metrics across the application (all business transactions) are visible on the [Application Flow Map](#) and the [Remote Services Dashboard](#).

Automatic Remote Service Detection

Java

AppDynamics automatically discovers the following remote service types:

- IBM WebSphere MQ
- HTTP
- JMS
- RMI
- Binary Remoting
- Web Services

For a list of supported backends see [Supported Backends for the Java Agent](#).

.NET

In a .NET environment, AppDynamics automatically discovers the following remote service types.

- WCF
- HTTP
- Web Services, including SOAP
- Directory Services, including LDAP*
- Queues
 - Apache ActiveMQ
 - IBM WebSphere MQ (also known as IBM XMS)
 - Microsoft Message Queuing (MSMQ)*
 - .NET Remoting*
 - Tibco Enterprise Message Service (EMS)
 - Tibco Rendezvous (RV)
 - MicrosoftServiceBus (Windows Azure Service Bus)
 - MicrosoftServiceBusQueue (Windows Azure Service Bus Queues)
 - AzureQueue (Windows Azure Queues)

* Notes: Correlation is supported unless otherwise indicated below.

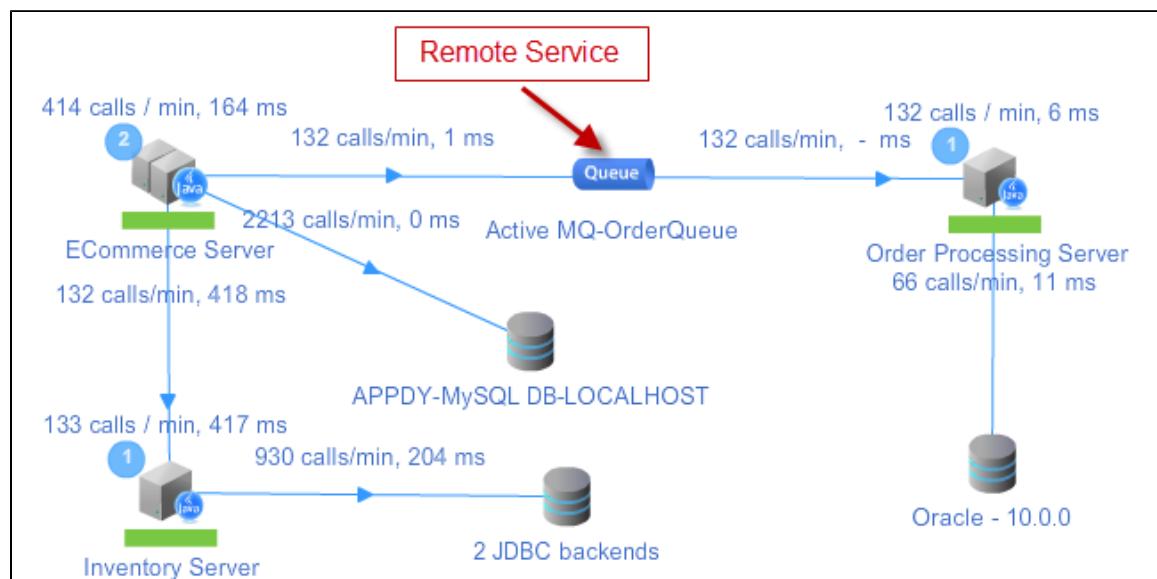
- For LDAP, correlation is non-applicable
- For MSMQ, correlation for downstream calls is not supported.
- For .NET remoting, additional configuration is needed to get downstream correlation. See [Enable Correlation for .NET Remoting](#).

For a list of supported backends see [Supported Backends for the .NET Agent](#)

Remote Service Visibility

- Application Flow Map

The detected remote services show up on the Application Flow Map. You can view all the detected services in the context of the entire application's transaction flow.



- Remote Services List

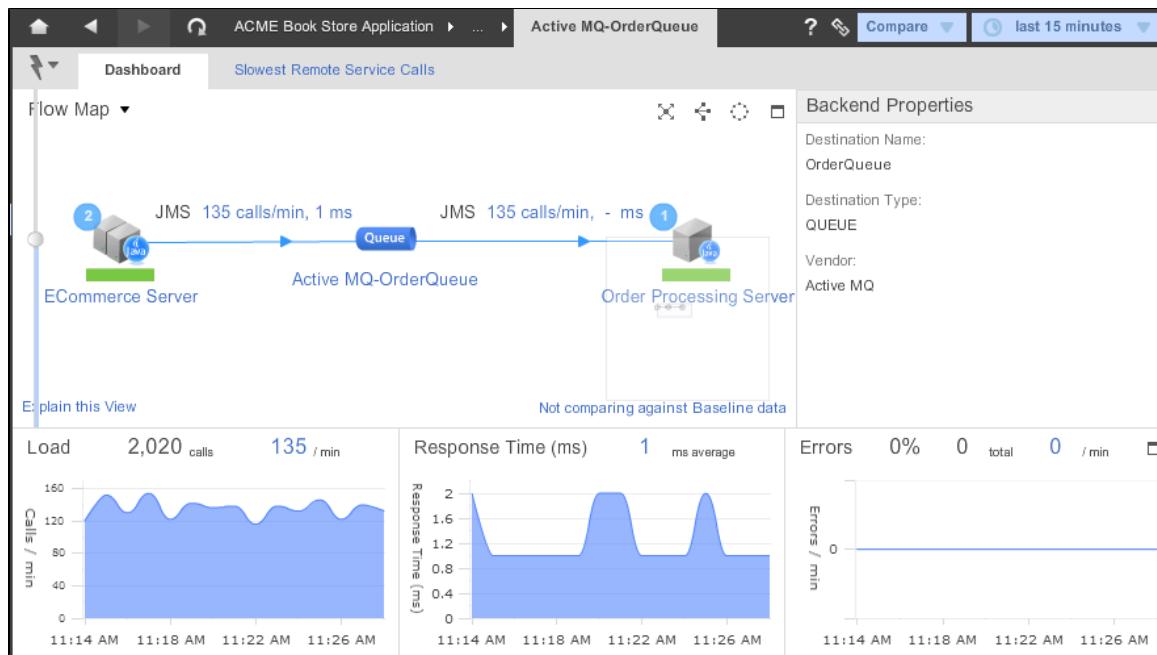
You can view a list of the detected remote services that displays key performance indicators, such as response time, total calls, calls per minute, errors, and errors per minute.

The screenshot shows the AppDynamics interface for the ACME Book Store Application. The left sidebar shows the navigation path: ACME Book Store Application > Servers > Remote Services. The main content area displays a table titled "Showing 1 of 1 Remote Services".

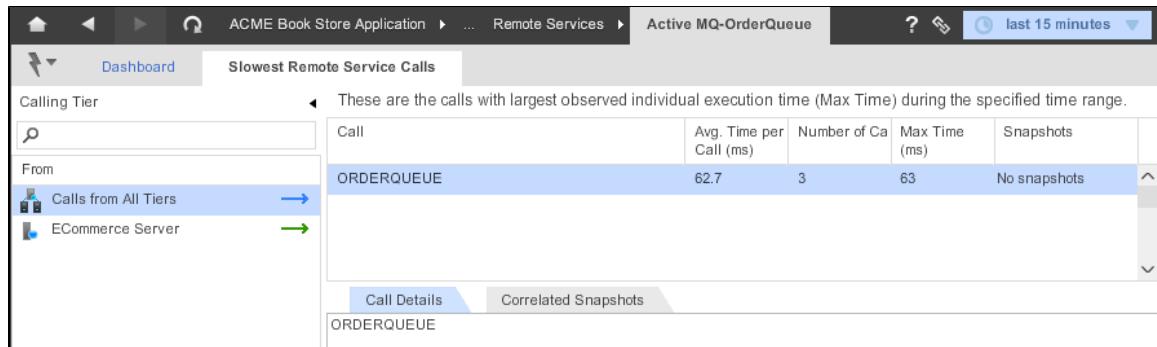
Name	Response Time (ms)	Calls	Calls / min	Errors	Errors / min
Active MQ-OrderQueue	1	2,010	134	0	0

- Remote Services Dashboard

From the list, you can select a remote service and click **View Dashboard** to see the Remote Service Dashboard. The dashboard displays a Flow Map, properties, and graphs of the key performance indicators (KPIs).

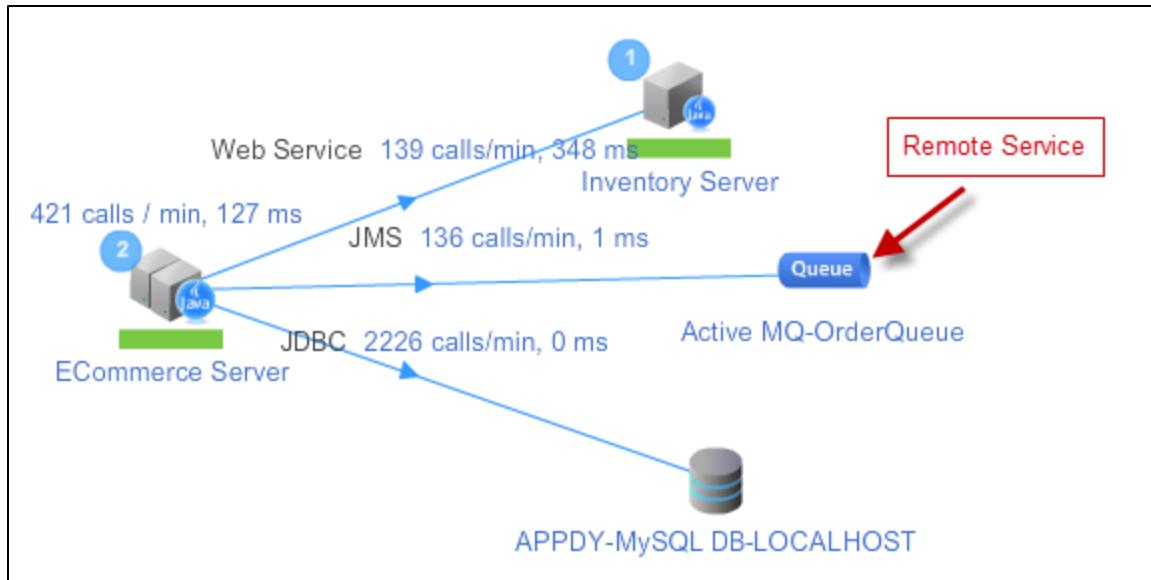


From this dashboard you can select the Slowest Remote Service Calls tab to view specific details and find related business transaction snapshots to troubleshoot issues and find the root cause of database bottlenecks.



• Tier Flow Map

The detected remote services show up on the Tier Flow Map. You can view the detected services in the context of the traffic on this specific tier. For example, the ECommerce Server tier, as shown in the following screen capture.



Slow Remote Services Calls

AppDynamics displays a list of the slowest remote service calls with call details so you can drill down to find problems. The Slowest Remote Service Calls tab lists the ten calls to the service with the longest execution time, by tier and for all tiers. Each call shows the Average Time, Number of Calls during the time range, and Max Time. The Max Time is used to determine which calls are in the list. For remote service calls, the Max Time must exceed 50 ms before the call is tracked as a potential candidate for this list. The Agent aggregates and reports call data every 15 minutes. See [Troubleshoot Slow Response Times](#).

To monitor a remote service, first make sure it shows up in the Remote Service List and has its own dashboard.

If a remote service is not appearing, check the configuration. See [Configure Backend Detection](#).

Configuring Remote Service Detection

You can customize the discovery and naming rules and configure the detection of additional remote services. For details see:

- [Configure Backend Detection](#)
- [Configure Custom Exit Points \(Java\)](#)
- [Configure Custom Exit Points \(.NET\)](#)
- [Configurations for Custom Exit Points](#)
- [Match Rule Conditions](#)

Learn More

- [Remote Services Dashboard](#)
- [Configure Stale Backend Removal](#)
- [Troubleshoot Slow Response Times](#)

Configure Backend Detection

- [Auto-Discovery](#)
- [Backend Discovery and Naming](#)
 - To view the discovery rules for an auto-discovered backend
 - [Default Properties for Backend Detection in Java Environments](#)
 - [Default Properties for Backend Detection in .NET Environments](#)
- [Customizing Backend Discovery Rules](#)
 - To copy an entire backend detection configuration to all tiers
 - To copy an entire backend detection configuration to another tier or application
 - To edit a backend discovery rule
- [Creating New Discovery Rules for Auto-Discovered Backends](#)
 - To create a custom discovery rule for an auto-discovered backend type
- [Examples](#)
- [Learn More](#)

In AppDynamics, databases and remote services are collectively known as backends. Backends are born from exit points. An exit point is an exit call from an instrumented node to an uninstrumented node or other service, such as a message queue. Such an exit call results in backend discovery.

Each type of supported remote service and database has a list of properties associated with it. Each supported backend is identified by its type and the related properties. The set of configured properties is referred to as a backend auto-discovery rule. AppDynamics uses auto-discovery rules to identify and name the databases and remote services for monitoring.

The preconfigured auto-discovery rules might not always match exactly how you want to measure performance in your specific application. You can change the properties and how they are used to aggregate activity to measure what you choose.

Auto-Discovery

AppDynamics uses auto-discovery rules to identify databases and remote services (collectively known as backends) for monitoring. By default, the configuration for backend detection is inherited from the tier or application level configuration. See [Hierarchical Configuration Model](#).

The backends that can be configured are listed, with their default configurations, in the Automatic Backend Discovery list in the **Backend Detection** tab. There is a different configuration for each auto-discovered backend type.

The default discovery rules for each backend type include the following:

- Whether automatic discovery is enabled
- Whether correlation with the application is enabled
- Properties used to identify and name the backend

Sometimes you might need a different set of discovery rules to meet your specific criteria. In these cases you can create a custom rule that meets those criteria. Or you might want to disable some or all of the default rules and create custom rules for detecting all your backends. AppDynamics provides flexibility for configuring backend detection.

For example, detection of HTTP backends is enabled by default. The backends are identified by the host and port and correlation with the application is enabled. To change the detection for HTTP backends in some way - disable correlation, or omit the port number from the detected name, or use only certain segments of the host in the name, you can edit the automatic discovery rule for the HTTP type. In most cases, you can achieve the level of customization that you need by editing the default rules.

Backend Discovery and Naming

For each auto-discovered backend type, there is a set of configurable properties that are used to identify the backend. You can change the default configuration for the backend types shown in the UI.

For example, you can ignore the Vendor property for JMS backend detection. If you do, then all JMS backends with the same destination and destination type would be identified as the same backend and Vendor is ignored for the purposes of identification. The metrics for all JMS backends with the same destination and destination type would be aggregated as one backend.

For properties that you do want to use, you can also configure how AppDynamics should use the property. For example, if the URL property is used to identify a backend and your URLs include a file name, using the default configuration, AppDynamics identifies a separate backend for every file. To reduce the number of backends identified using the URL property, you can configure which parts of the URL to use. For example, you could run a regular expression on the URL to discard the file name, or any other parts of the URL that you do not want used for identification.

You can modify the database and remote service discovery configuration in the following ways:

- Use the default configuration
- Use one or more specified segments of a property
- Run a regular expression on a property
- Execute a method on a property

To view the discovery rules for an auto-discovered backend

1. Access the backend detection screen using these steps:
 - a. Select the application.
 - b. In the left navigation pane, click **Configure -> Instrumentation**.
 - c. Click the **Backend Detection** tab.
 - d. Select the application and the tab corresponding to the backend platform.
The Automatic Backend Discovery default configurations are listed.

2. In the Automatic Backend Discovery list, click the configuration to view and a summary of the configuration appears on the right.

For example, the following figure shows that JMS backends are auto-discovered using the Destination, Destination Type, and Vendor.

The screenshot shows the 'Java - Backend Detection' tab selected in the 'Automatic Backend Discovery' configuration for the 'AcmeOnlineBookStore' application. The 'JMS' row is selected in the list, showing its configuration details:

- Enabled:** checked
- Correlation Enabled:** checked
- Name JMS Backends Using:** Destination, DestinationType, Vendor

Default Properties for Backend Detection in Java Environments

The following table lists the auto-discovered backend types that are configurable in a Java environment, the available configurable properties and whether the property is used in the default auto-discovery rules.

Type	Configurable Properties	Property Used by Default in Detection and Naming
IBM MQ	Destination	Yes
	Destination Type	Yes
	Host	Yes
	Port	Yes
	Major Version	Yes
	Vendor	Yes
HTTP	Host	Yes
	Port	Yes
	URL	No
	Query String	No
JDBC	URL	Yes
	Host	Yes
	Port	Yes
	Database	Yes
	Version	Yes
	Vendor	Yes
JMS	Destination	Yes
	Destination Type	Yes
	Vendor	Yes
Cassandra	Host	Yes
	Port	Yes

	transport	Yes
	keyspace	Yes
RMI	URL	Yes
Binary Remoting	Host	Yes
	Port	Yes
	transport	Yes
Web Service	Service	Yes
	URL	No
	Operation	No
	Soap Action	No
	Vendor	No

Default Properties for Backend Detection in .NET Environments

The following table lists the auto-discovered backend types configurable in a .NET environment, the available configurable properties and whether the property is used in the default auto-discovery rules.

Type	Configurable Properties	Property Used by Default
		in Detection and Naming
WCF	Remote Address	Yes
	Operation Contract	No
	URL	No
	Host	No
	Port	No
	Soap Action	No
Queues	Host	No
	Destination	Yes
	Destination Type	No
	Vendor	No
HTTP	Host	No
	Port	No
	URL	Yes
	Query String	No
Web Service	Service	No
	URL	Yes
	Operation	No
	Soap Action	No
ADO.NET	Host	Yes
	Port	No
	Database	Yes

	Vendor	No
	Connection String	No

Customizing Backend Discovery Rules

You can customize backend visibility to suit your monitoring needs. You can configure the detection of databases and remote services (collectively known as backends) in the following ways:

- [Edit the default discovery rules](#)
- [Create new discovery rules](#)
- [Configure custom exit points to monitor calls and enable detection for backend types that are not auto-discovered.](#)

The precise configurations vary according to the type of the backend, but the following general features are configurable:

- Discovery Enabled - You can enable and disable auto-discovery for the backend type. Undiscovered backends are not monitored.
- Correlation Enabled - You can enable and disable correlation. Enabling correlation lets AppDynamics pass information about the application to and through the backend. If you do not care about activity downstream from the backend, you may want to disable correlation.
- Backend Naming

Custom rules include the following additional settings:

- Name for the custom rule
- Priority - Used to set precedence for custom rules.
- Match Conditions - Used to identify backends that are identified by custom rules. See [Match Rule Conditions](#)

To copy an entire backend detection configuration to all tiers

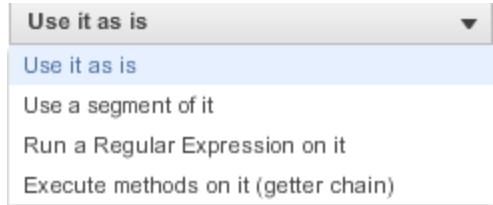
1. Access the backend detection screen. See [To access the backend detection tab](#).
2. In the left panel select the application or tier whose configuration you want to copy.
3. Click **Configure all Tiers to use this Configuration**.

To copy an entire backend detection configuration to another tier or application

1. Access the backend detection screen. See [To access the backend detection tab](#).
2. In the left panel select the application or tier whose configuration you want to copy.
3. Click **Copy**.
4. In the Application/Tier browser, choose the application or tier to copy the configuration to.
5. Click **OK**.

To edit a backend discovery rule

1. In the Automatic Backend Discovery list, select the backend type to modify. The rule summary appears in the Automatic Discovery panel on the right.
2. Click **Edit Automatic Discovery**. The Edit Automatic Backend Discovery Rule window appears
3. For each property that you want to configure:
 - Select the property in the property list.
 - Check the property check box to use the property for detection; clear the check box to omit it.
 - If you are using the property, choose how the property is used from the drop-down list.



4. Check **Enabled** to enable the rule; clear the check box to disable it.
5. Check **Correlation Enabled** to enable correlation.
6. Click **OK**.

Creating New Discovery Rules for Auto-Discovered Backends

To create a custom discovery rule for an auto-discovered backend type

1. In the Automatic Backend Discovery list, select the backend type.
The Custom Discovery Rules editor appears in the right panel below the Automatic Discovery panel.
2. Click **Add** (the + icon) to create a new rule or select an existing rule from the list and click the edit icon to modify one.
3. Enter a name for the custom rule.
4. Enter the priority for the custom rule among all other custom rules for this backend type. The higher the number, the higher the priority. A value of 0 (zero) indicates that the default rule should be used.
5. Configure the match conditions that define which backends.
Match conditions defined on the naming properties are used to identify which backends should use the custom rule. Backends that do not meet all the defined match conditions are discovered according to the default rule.

For example, to use the custom rule for JDBC backends with port numbers that start with 80, you would define the match condition:

Port Starts With 80

The comparison operators for defining the match conditions are: Equals, Starts With, Ends With, Contains, and Matches Regular Expression.

6. Configure the Enabled, Correlation Enabled, and Backend Naming Configuration as described in [To edit a default backend discovery rule](#).
7. Click **OK** to save the configuration.

Examples

A common use case for changing the backend discovery rules is to combine auto-discovered backends of the same type that share certain properties.

For example, AppDynamics auto-discovers JDBC backends based on host, port, URL, database, version and vendor values. To combine all JDBC databases that contain "EC2" in their host names to be monitored as a single database, follow steps 1 through 4 described above in [To create a custom discovery rule for an automatically discovered backend type](#) and use the following match condition:

Host Contains EC2

Similarly, you can group all Web Service remote services that share the same service value, or all RMI backends that have URLs that match a regular expression, and so on.

Learn More

- [Configure Custom Exit Points](#)
- [Match Rule Conditions](#)

- Hierarchical Configuration Model

Configure Custom Exit Points

- To create a custom exit point
 - To split an exit point
 - To group an exit point
- To define custom metrics for a custom exit point
- To define transaction snapshot data collected
- Learn More

For additional details see:

[Configure Custom Exit Points \(Java\)](#)

[Configure Custom Exit Points \(.NET\)](#)

Not Supported for PHP

Custom exit points provide identification for backend types that are not automatically detected, such as file systems, mainframes etc. For example, you can define a custom exit call to monitor the file system read method. Custom exit points appear as unresolved backends in the flow maps. Unresolved backends are shown on flow maps with this icon .

You define a custom exit point by specifying the class and method used to identify the backend. If the method is overloaded, you need to add the parameters to identify the method uniquely.

You can restrict the method invocations for which you want AppDynamics to collect metrics by specifying match conditions for the method. The match conditions can be based on a parameter or the invoked object.

You can also optionally split the exit point based on a method parameter, the return value, or the invoked object.

You can also configure custom metrics and transaction snapshot data to collect for the backend.

See [Configurations for Custom Exit Points](#) for suggested custom configurations for some common backends.

To create a custom exit point

1. In the left navigation panel, click **Configure -> Instrumentation**.
2. Click the **Backend Detection** tab.
3. Click the tab corresponding to the backend platform.
4. Select the application or tier for which you are configuring the custom exit point. Backend detection configuration is applied on a hierarchical inheritance model. See [Hierarchical Configuration Model](#).
5. Scroll down to Custom Exit Points.
6. Click **Add** (the + icon).
7. In the Create Custom Exit Point window, click the **Identification** tab if it is not selected.
8. Enter a name for the exit point. This is the name that identifies the backend.
9. Select the type of backend from the Type drop-down menu or check Use Custom if the type is not listed.
10. Configure the class and method name that identify the custom exit point. If the method is overloaded, check the Overloaded check box and add the parameters.
11. If you want to restrict metric collection based on a set of criteria that are evaluated at runtime, click **Add Match Condition** and define the match condition(s). For example, you may want to collect data only if the value of a specific method parameter contains a certain value.

12. Click **Save**.

The following screenshot shows a custom exit point of Type Cache. This exit point is defined on the `getAll()` method of the specified class. The exit point appears in flow maps as an unresolved backend named CoherenceGetAll.

Edit Custom Exit Point

Name: <input type="text" value="CoherenceGetAll"/>	Type: <input type="text" value="Cache"/>				
<input checked="" type="radio"/> Identification <input type="radio"/> Custom Metrics <input type="radio"/> Snapshot Data					
Define the class and method name which, when called, will be identified as a Custom Exit Point					
Class <input type="text" value="that implements an Interface which"/>	equals <input type="text" value="com.tangosol.net.NamedCache"/>				
Method Name <input type="text" value="getAll"/>	<input type="checkbox"/> Is this Method Overloaded?				
Method Parameters (optional)					
<input type="button" value="Add Parameter"/>					
Match Conditions (optional)					
<input type="button" value="Add Match Condition"/>					
Calls to the specified class and method name can be further split based by a combination of match conditions.					
<table border="1"><thead><tr><th>Name</th><th>Description</th></tr></thead><tbody><tr><td>Cachename</td><td>Collect data from the invoked object and capture the result of the call.</td></tr></tbody></table>		Name	Description	Cachename	Collect data from the invoked object and capture the result of the call.
Name	Description				
Cachename	Collect data from the invoked object and capture the result of the call.				
<input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/>					
<input type="button" value="Cancel"/> <input type="button" value="Save"/>					

To split an exit point

1. Click **Add**.
2. Enter a display name for the split exit point.
3. Specify the source of the data (parameter, return value, or invoked object).
4. Specify the operation to invoke on the source of the data (`toString()` or getter chain for complex objects).
5. Click **Save**.

The following example shows a pplit configuration of the previously created CoherenceGetAll exit point based on the `getCacheName()` method of the invoked object.

Edit Custom Exit Point Identifier

Specify the parameter index or indicate if it the return value of the diagnostic data to be collected.
Simple getters without parameters can be used on the parameter or the return value to be displayed against the display name specified here.

Display Name

Create your own name for the data collected. This will be the display name for the data in Request Snapshots

Collect Data From Method Parameter @ Index:
 Return Value
 Invoked Object

Operation on Invoked Object Use `toString()`

Use Getter Chain

for example: `getAccount().getBalance()`

To group an exit point

You can group methods as a single exit point. The only requirement is that these methods point to the same key.

For example, ACME Online has an exit point for `NamedCache.getAll`. This exit point has a split configuration of `getCacheName()` on the invoked object as illustrated in the previous screenshot.

Suppose we also define another exit point for `NamedCache.entrySet`. This is another exit point, but it has the split configuration that has `getCacheName()` method of the invoked object.

Edit Custom Exit Point

Name:	CoherenceCacheAcess	Type:	Cache				
<input checked="" type="radio"/> Identification <input type="radio"/> Custom Metrics <input type="radio"/> Snapshot Data							
Define the class and method name which, when called, will be identified as a Custom Exit Point:							
Class	that implements an Interface which	equals	com.tangosol.net.NamedCache				
Method Name	entrySet	<input type="checkbox"/> Is this Method Overloaded?					
Method Parameters (optional)							
<input type="button" value="Add Parameter"/>							
Match Conditions (optional)							
<input type="button" value="Add Match Condition"/>							
Calls to the specified class and method name can be further split based by a combination of method parameters and match conditions.							
<table border="1"> <thead> <tr> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>CacheName</td> <td>Collect data from the invoked object and capture the result of getCacheName().</td> </tr> </tbody> </table>				Name	Description	CacheName	Collect data from the invoked object and capture the result of getCacheName().
Name	Description						
CacheName	Collect data from the invoked object and capture the result of getCacheName().						
<input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/>							

If the getAll() and the entrySet() methods point to the same cache name, they will point to the same backend.

Matching name-value pairs identify the back-end. In this case, there is only one key, i.e. cache name, has to be matched. So, here both exit points have the same name for the cache and they resolve to the same backend.

To define custom metrics for a custom exit point

Custom metrics are collected in addition to the standard metrics.

The result of the data collected from the method invocation must be an integer value, which will either be averaged or added per minute, depending on your selection of data roll-up.

To configure custom business metrics that can be generated from the Java method invocation:

1. Click the **Custom Metrics** tab.
2. Click **Add**.
3. In the Add Custom Metric window, enter a name for the metric.
4. Select the Collect Data From radio button to specify the source of the metric data.
5. Select the Operation on Method Parameter to specify how the metric data is processed.
6. Select how the data should be rolled up (average or sum) from the Data Rollup drop-down menu.
7. Click **Create Custom Metric**.

To define transaction snapshot data collected

1. Click the **Snapshot Data** tab.
2. Click **Add**.
3. In the Add Snapshot Data window, enter a display name for the snapshot data.
4. Select the Collect Data From radio button to specify the source of the snapshot data.
5. Select the Operation on Method Parameter to specify how the snapshot data is processed.
5. Click **Save**.

Learn More

- Configurations for Custom Exit Points

Configure Stale Backend Removal

- Stale or Orphaned Backends
 - To configure automatic stale backend deletion
- Learn More

Stale or Orphaned Backends

A stale backend (also called an orphaned backend) is an unresolved backend for which AppDynamics has previously captured metrics, but which has experienced no metric activity for the configured time period.

You can configure AppDynamics to remove stale backends (database and remote service servers) automatically at a regular interval by setting the Controller-level global `backend.permanent.deletion.period` property.

If you are sharing a Controller on a SaaS account, contact [AppDynamics Support](#).

When this global property is enabled, AppDynamics removes all the stale backends in the managed environment, including any metrics previously collected for them and any health rules referencing their metrics. After a backend is removed, its metrics no longer appear in the Metric Browser and health rules that reference those metrics do not fire. You should remove any health rules conditions that reference metrics in stale backends.

By default, automatic removal of stale backends is enabled with a default interval of one month. The beginning of the interval is the last time that activity was reported on the backend (that Calls per Minute > 0) and the end of the interval is the time at which the backend is deleted.

You can modify the interval. This may be advisable, especially for large installations, since the maximum number of backends removed in one pass is 50. The minimum interval is one week.

You can also disable automatic removal by setting the interval to 0.

The automatic backend removal is logged in the `server.log` with the message:

```
BACKEND PURGER deleting unresolved stale backend ids:
```

followed by a list of the IDs being deleted.

To configure automatic stale backend deletion

1. Log into the Controller Admin console using the admin account.

```
http://<controller-installer-host>:<port>/controller/admin.html
```

2. Select Controller Settings.
3. Scroll down to the `backend.permanent.deletion.period` property.
4. Enter the new interval in hours.

5. Click **Save**.

Learn More

- Backend Monitoring

Infrastructure Monitoring

- Application Infrastructure Monitoring and Metrics
- Infrastructure Metrics Can Help Identify Problems
- Learn More

Application Infrastructure Monitoring and Metrics

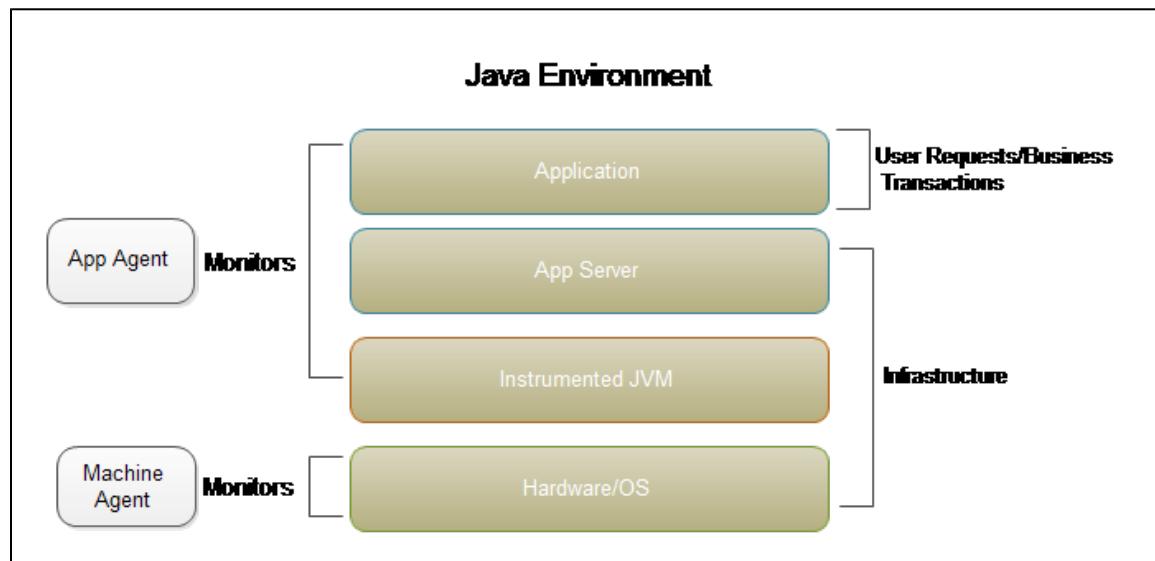
Monitoring business transaction health gives you an inside view of how applications are performing. AppDynamics encourages you to monitor your business transactions and make them the focus of your performance monitoring strategy. However, infrastructure performance naturally affects business transaction performance, and so infrastructure metrics can provide important clues to the root causes of your application performance issues. AppDynamics can alert you to the problem at the business transaction level and at the infrastructure level. In addition, you can correlate metrics with each other to see the relationships among different events that occur in the system.

AppDynamics provides preconfigured application infrastructure metrics and default health rules to enable you to discover and correct infrastructure problems. You can also configure additional persistent metrics to implement a monitoring strategy specific to your business needs and application architecture.

Application infrastructure includes:

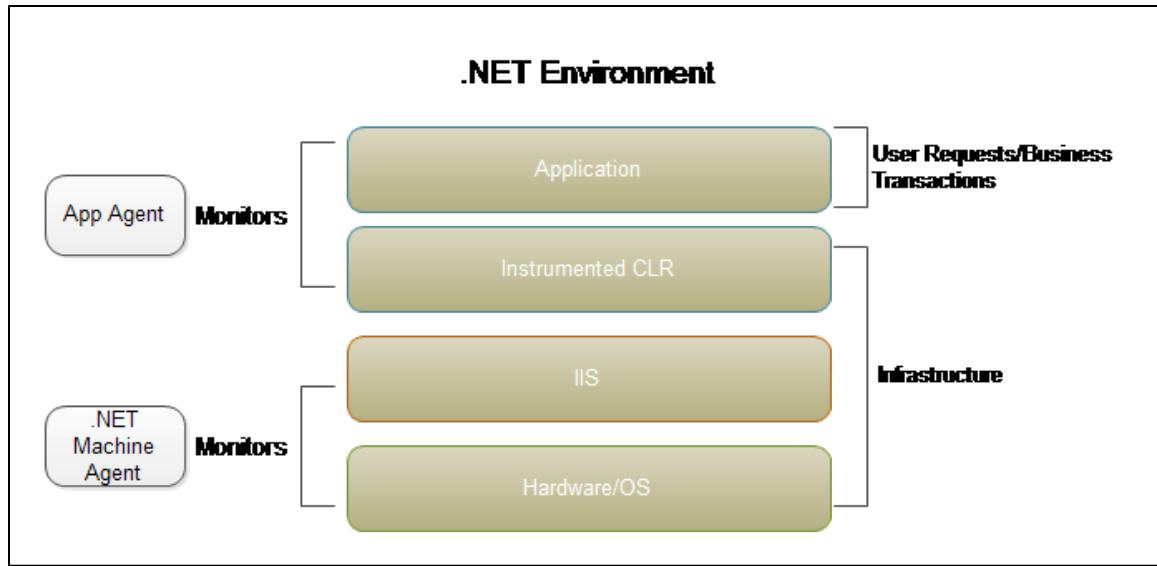
- Application servers and IIS
- JVMs and .NET CLRs
- Hardware
- OS

The following diagram illustrates the infrastructure associated with a Java application environment.



The App Agent for Java provides both infrastructure metrics for the JVM and the App Server, and application metrics for your business application. When a machine agent is installed, metrics are collected for CPU usage, disk I/O, network I/O, and memory usage.

The following diagram illustrates the infrastructure associated with a .NET application environment.



Infrastructure Metrics Can Help Identify Problems

Having the right infrastructure monitoring strategy is important because many times the root cause of application issues is most obvious by looking at infrastructure metrics. For example, the following infrastructure issues can slow down your application:

- Too much time spent in garbage collection of temporary objects
- Contention in connection pools
- Some other process spinning on the CPU

In all these cases, transaction snapshots are able to correlate the infrastructure metrics for the specific node so that you can identify the root cause of slow or stalled transactions. For example, from the Node Dashboard, you can view the **JVM** tab, the **JMX** tab showing the connection pool metrics, and the **Hardware** tab (when the machine agent is installed) respectively.

You can configure health rules on metrics such as garbage collection time, connection pool contention, or CPU usage to catch issues early in the cycle before there is an impact on your business transactions. In all cases, with the right monitoring strategy in place, you can be alerted to problems and fix them before user transactions are affected.

Learn More

- Monitor Java App Servers
- Monitor Hardware
- Monitor JVMs
- Alert and Respond
- Infrastructure Metrics

Infrastructure Metrics

- View of Infrastructure Metrics
 - Infrastructure Metric Collection
 - Long-term Metrics for Baseline, Monitoring, and Alerting
 - Short-term Metrics for Correlating and Troubleshooting
- Using Metrics in External Systems
- Learn More

AppMan Advice



You can't simulate every use case or condition in dev and test, so you must understand your application performance in the real world.

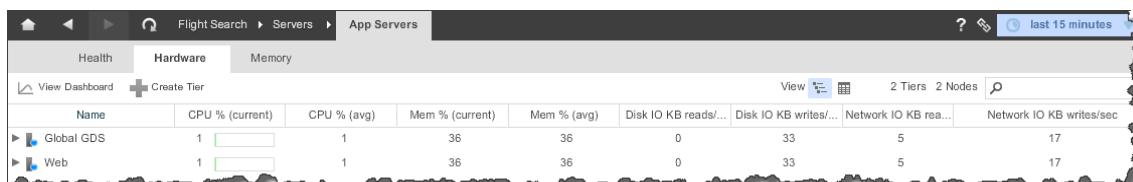
View of Infrastructure Metrics

The AppDynamics user interface organizes the infrastructure metrics for your application level by tiers. For each application, you can see metrics gathered by the machine agent (when installed) and app server agents. In general, agents upload metrics to the Controller once a minute at staggered intervals. The Controller rolls up the data and makes it available in the UI. You can get a view of the health of the nodes in a tier using the App Servers Dashboard.

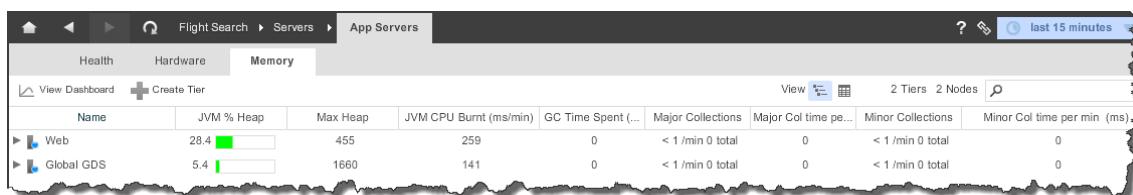
To access this dashboard, in the left navigation pane, click **Servers -> App Servers**.



Click the **Hardware** tab to see hardware metrics.



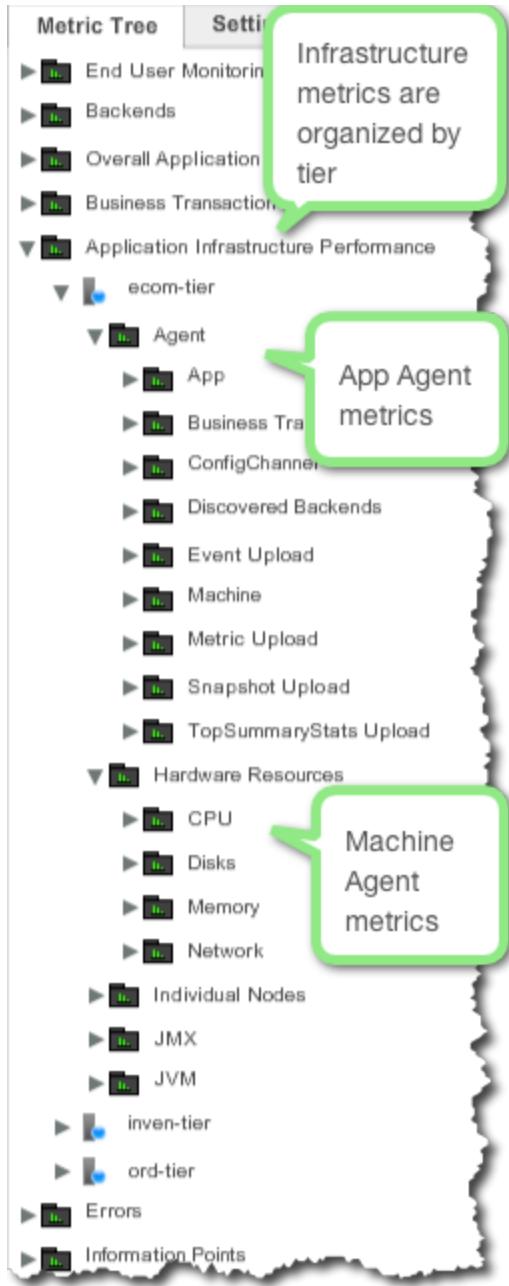
Click the **Memory** tab to see some of the key memory metrics.



You can see more detailed infrastructure metrics from the Node Dashboard. To access this dashboard, in the left navigation pane, click **Servers -> App Servers -> <tier> -> <node>**.



You can also view metrics in the **Metric Browser**.



Infrastructure Metric Collection

JVM and JMX infrastructure metrics are averaged over a period of one minute at 15-second intervals. As the minute boundary starts, we take the value, and take three more values every 15 seconds until the end of the minute. The four values are averaged and reported to the controller. Therefore, every minute the agent reports a tuple consisting of min, max, avg, sum, current for that minute to the controller. For example, if we took 10,20,40,30 as values for that minute we have 10,40,25,100,30 for that particular metric. In other words:

- min=10
- max=40
- avg=25
- sum=100
- current=30

Machine agent CPU and memory metrics are gathered every 2 seconds and averaged over a period of one minute. Machine agent network and disk metrics are gathered at one minute intervals.

Long-term Metrics for Baselining, Monitoring, and Alerting

Usually it is best to focus on detecting anomalies in business transactions, because infrastructure problems always have an impact on the business transaction metrics. However there may be cases where you want to monitor infrastructure directly.

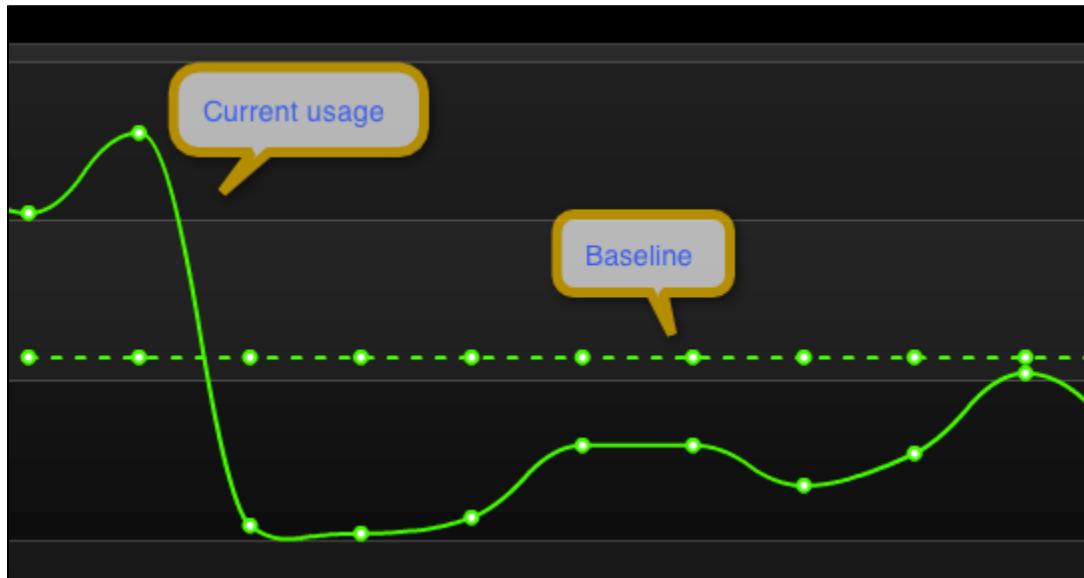
By default AppDynamics generates long-term metrics for the key machine, app server, and app server attributes that represent the health of an application. By default AppDynamics includes health rules at the node level for infrastructure metrics. You can modify these or create new health rules to be notified of actual or impending problems. The default infrastructure health rules are shown in the following screen capture.

The screenshot shows the 'Health Rules' section of the AppDynamics interface. On the left, a navigation tree under 'Inventory Management' includes 'Business Transactions', 'Servers', 'Events', 'Troubleshoot', 'Alert & Respond' (which is expanded), 'Policies', 'Health Rules' (which is selected and highlighted in blue), 'Actions', 'Email Digests', 'Cloud Auto-Scaling', 'Workflows', 'Tasks', and 'Shared Variables'. The main panel displays a table of health rules with columns for 'Name', 'Type', and 'Enabled'. There are seven rules listed, all of which are enabled (indicated by green checkmarks). The rules are:

Name	Type	Enabled
Business Transaction response time is much higher than normal	Business Transaction Performance	✓
Business Transaction error rate is much higher than normal	Business Transaction Performance	✓
CPU utilization is too high	Node Health - Hardware, JVM, CLR	✓
Memory utilization is too high	Node Health - Hardware, JVM, CLR	✓
JVM Heap utilization is too high	Node Health - Hardware, JVM, CLR	✓
JVM Garbage Collection Time is too high	Node Health - Hardware, JVM, CLR	✓
CLR Garbage Collection Time is too high	Node Health - Hardware, JVM, CLR	✓

For instructions to modify or add health rules see [Health Rules](#).

Analyzing metrics over the long term enables AppDynamics to define baselines against which anomalies are identified. A health rule can use a baseline as a condition. You can also compare baselines against current metrics in the Metric Browser.

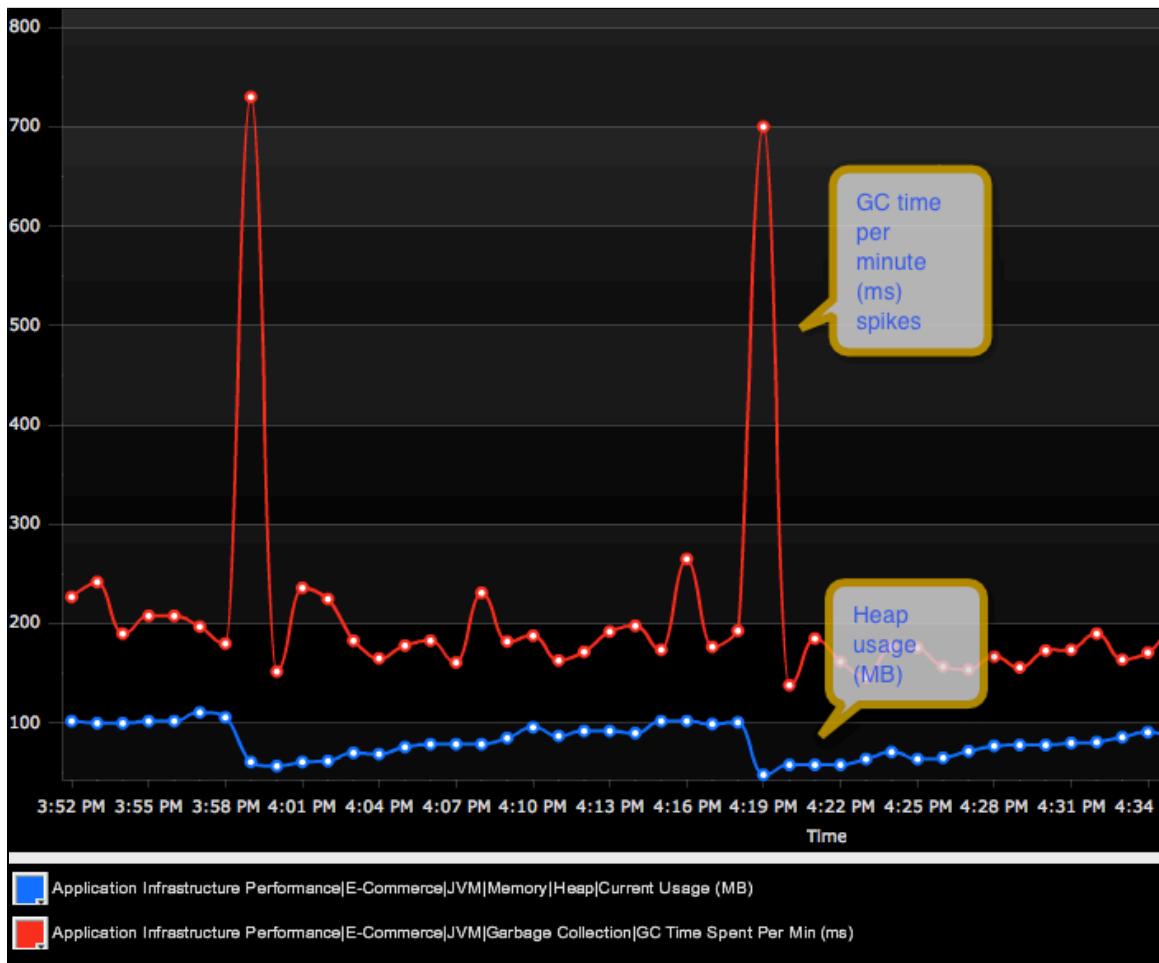


For more information about using baselines see [Behavior Learning and Anomaly Detection](#).

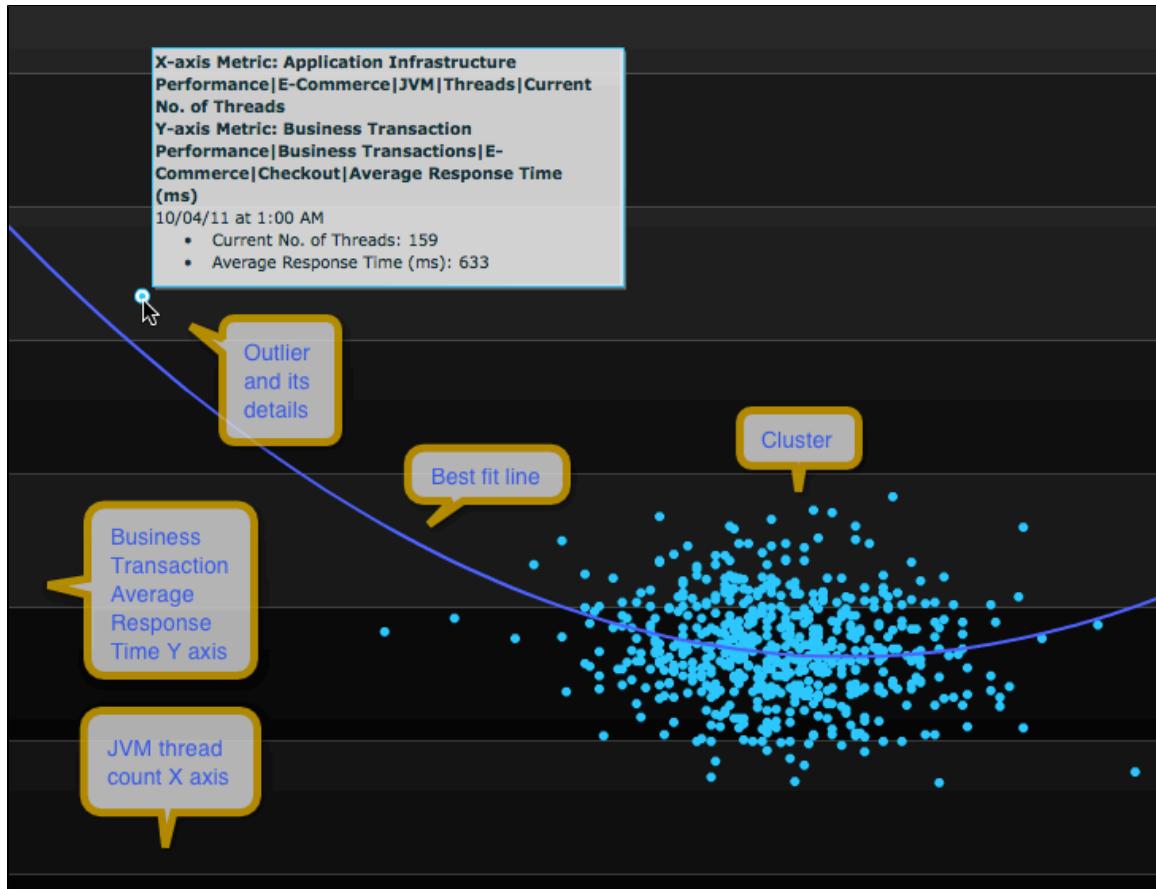
Short-term Metrics for Correlating and Troubleshooting

When a problem is discovered in the application infrastructure, you can drill down to find the root cause. The investigation may involve looking at the short-term trends of specific metrics, defining new metrics, or comparing different metrics to find correlations between behavior in different parts of the application environment.

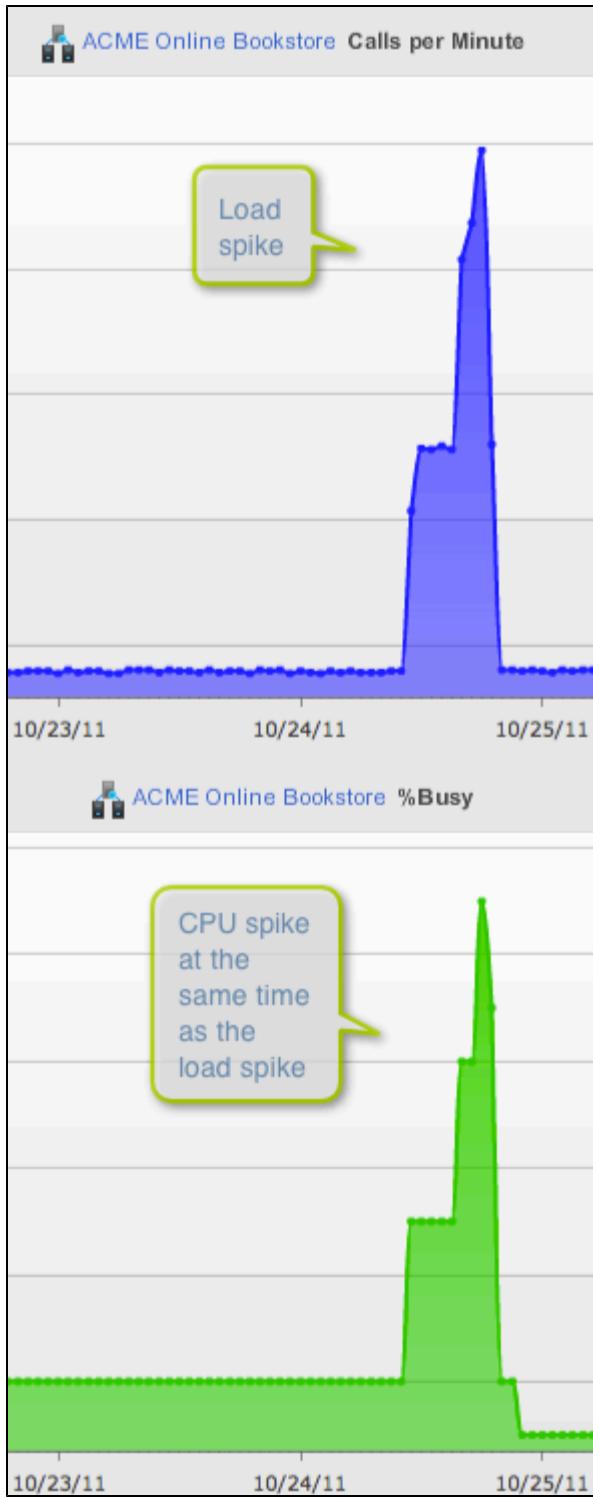
The Metric Browser displays the metrics and enables you to graph different metrics onto the same panel so that you can compare different data that was gathered in the same timeframe.



The built-in Correlation Analysis graph is similar to the basic metric graph, but it enables you to assign two metrics to X and Y coordinates. By default it uses a scatter plot view. The scatter plot view also calculates the Best Fit line, which uses the quadratic least squares algorithm to find the best fit curve line.



A built-in Scalability Analysis graph compares the CPU usage to the load (calls per minute) on the application, business transaction, tier, or node level. This comparison may be more easily understood in the graphs view shown here.



For details about how to compare and correlate metrics see Scalability Analysis and Correlation Analysis.

Using Metrics in External Systems

You may want to add metrics to external reports, dashboards or systems. AppDynamics helps you do this in a variety of ways:

- Exporting and importing application configuration baselines: Metric baselines (not the metric data) are included when you export AppDynamics application configuration data.
For details see [Export and Import Business Application Configurations](#).

- Exporting metrics as comma-separated values (CSV): The Metrics Browser has an option to export metrics on the graph to a CSV file. For each metric you can specify the columns to use, and you see a preview that you can also copy to the clipboard. For details about how to export data see [To Export Metrics Data](#).
- Accessing metrics using the REST API: The REST API provides access to all performance data gathered by the Controller. You can also use a POST request to create an event of type "APPLICATION_DEPLOYMENT" in your managed environment. For details about the REST API see [Use the AppDynamics REST API](#).

Learn More

- Behavior Learning and Anomaly Detection

Monitor App Servers

See:

- Monitor Java App Servers
- Monitor CLRs

Monitor Hardware

- Default Hardware Metrics
 - CPU Utilization
 - Memory Utilization
 - Disk I/O
 - Network I/O
- Accessing Hardware Metrics
- Adding Custom Metrics
 - Java
 - .NET
- Learn More

Default Hardware Metrics

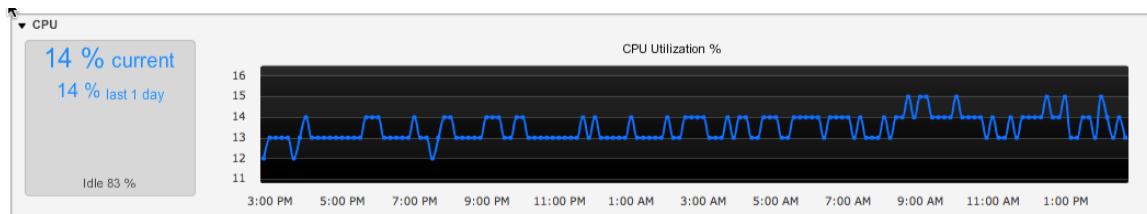
AppDynamics gathers information about the operating systems and machines in the monitored environment using the Machine Agent. By default, the following metrics are collected:

- CPU activity
- Memory usage
- Disk reads and writes
- Network traffic

The scope of the monitoring is a node associated with the Machine Agent. There is one Machine Agent per machine and there can be multiple nodes on the same machine.

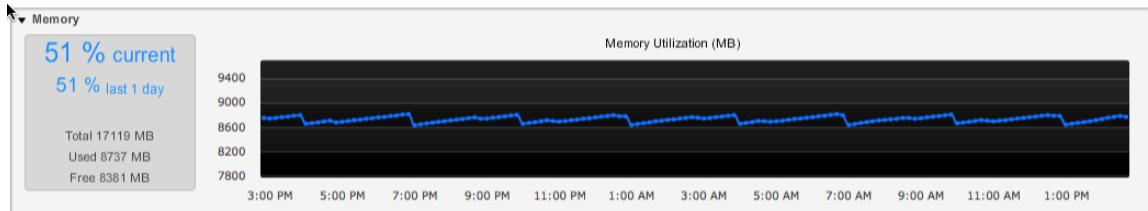
CPU Utilization

This trend displays average, minimum, and maximum values in percentage for CPU Busy Time for the selected point in the graph.



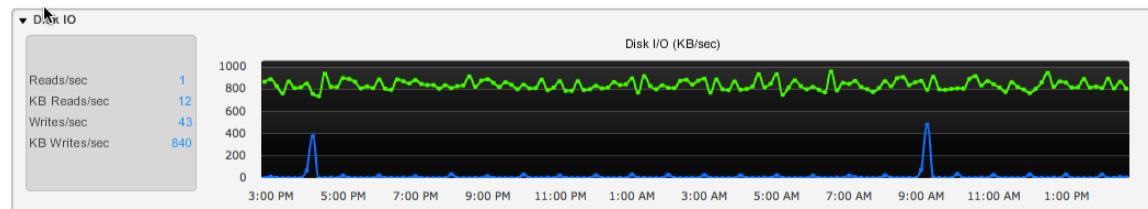
Memory Utilization

This trend displays average, minimum, and maximum values in KB for Memory Used for the selected point in the graph.



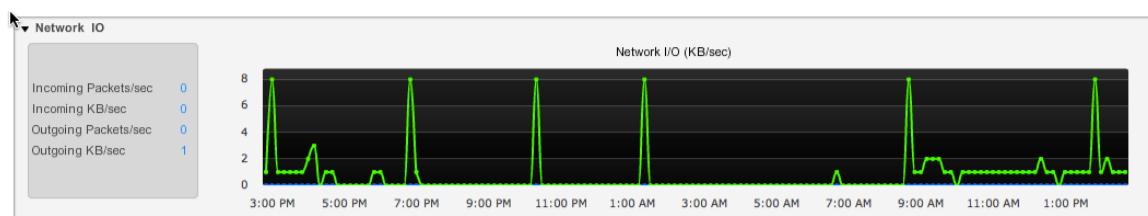
Disk I/O

This trend displays average, minimum, and maximum values for data (KB) written/sec for the selected point in the graph.



Network I/O

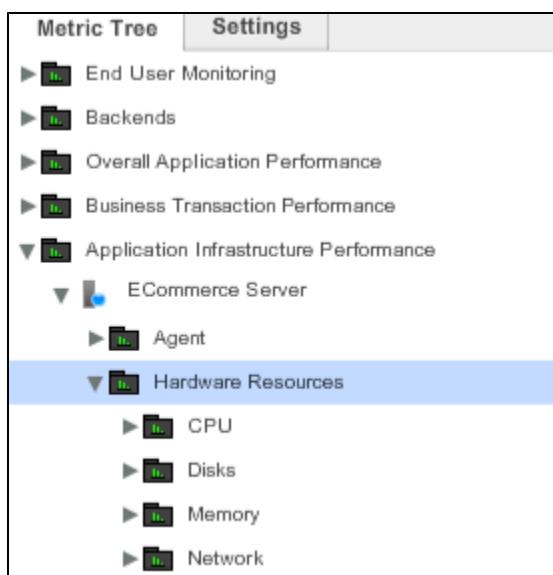
This trend displays average, minimum, and maximum values for outgoing data in KB/sec for the selected point in the graph.



Accessing Hardware Metrics

You can view hardware metrics from the **Hardware** tab of the **Node Dashboard** if a machine agent is installed on the machine that hosts the node.

You can also view hardware metrics in the **Metric Browser** in the **Hardware Resources** branch of a tier, as shown in this example.



Adding Custom Metrics

Java

You can modify the machine agent data collection metrics. See [Modify Machine Agent Data Collection Metrics](#).

You can add script-based custom monitors. See [Add Metrics Using Custom Monitors](#).

You can send metrics to the Machine Agent using its HTTP listener. See [Machine Agent HTTP Listener](#).

For a more complete discussion of adding custom metrics, see [Integration Basics](#).

.NET

The embedded .NET machine agent does not support adding script-based custom monitors. To use extensions such as custom monitors and HTTP listener in the .NET environment, it is necessary to install the AppDynamics Standalone Machine Agent, which is a Java application. See [Configure the .NET Machine Agent](#) for details on how to download, install, and configure the Standalone Machine Agent for .NET.

Learn More

- [Install the Machine Agent](#)
- [Configure the .NET Machine Agent](#)
- [Machine Agent Install and Admin FAQ](#)
- [Metric Browser](#)
- [Add Metrics Using Custom Monitors](#)

Alert and Respond

Alerts let you know when problems exist and help you anticipate problems that might be developing. Responses let you automate preventative actions to address those problems before they cause a severe slowdown or outage. Think of alert and respond as the automation of your runbooks.

AppDynamics recognizes some broad-based health issues commonly experienced by applications, such as "Business Transaction response time is much higher than normal" or "Memory utilization is too high". These are configured as default health rules, which define how high is "much higher than normal" or "too high", to which you can attach your alerts (whom to notify) and responses (what to do) when these problems exist. You can use these rules "as is" or modify them for your environment. See [Default Health Rules](#).

In addition to the broad-based rules, you can customize precise automatic alerts and responses for very narrowly circumscribed situations. This lets you finely tune your system, ensuring that the right alert goes to the right person, the right action is taken for the right problem on the right cluster or server.

For example:

- You do not want to alert your team if performance in a few clusters is lagging, but if more than 20% of the clusters are unhealthy, or if servers in particular clusters or servers that meet certain criteria are performing poorly, you do want to trigger an alert. The ability to configure health rules that are evaluated across specific tiers and nodes produces health rule violation events that inform you exactly which entity is experiencing problems and therefore whom to alert.

Select what Nodes this Health Rule affects

Type of nodes **Java and .NET nodes** ▾

Nodes matching the following Criteria: ▾

Nodes with a Name that Contains 800 ⚙

Nodes with the following Meta-Info Properties ?

+ Add Meta Info Criteria

Nodes with the following Environment Variables ?

+ Add Environment Variable Match Criteria

Nodes with the following JVM System Environment Properties (java nodes only) ?

+ Add JVM System Property Match Criteria

Select what Tiers this Health Rule affects

These specific Tiers ▾

Selected Tiers (2)



	Name	Type
	GDS	Application Server
	Web	Application Server

- Performance is deteriorating in one business transaction so you want to view snapshots for that one transaction.

Create Diagnostic Session Action

Name	GetSnapshotOnAddToCart		
Duration	10 minutes		
Snapshot rate	5 snapshots per minute		
Business Transactions to run on	<input type="radio"/> Affected Business Transactions ? <input checked="" type="radio"/> These Business Transactions: Selected Business Transactions (1)		
	 <table border="1"> <tr> <td>Name</td> <td>ViewCart.addToCart</td> </tr> </table>	Name	ViewCart.addToCart
Name	ViewCart.addToCart		

- You have a large operation with several development teams, each responsible for a different service. You create a health rule for one service and then it triggers when you create different policies in which you can pair each copy of the health rule to an alert addressed to the appropriate team.
- You have an application that performs well for normal load. However, peak loads can cause the application to slow. During peak load, AppDynamics not only detects the connection pool contention, but also allows you to create a remediation script that can automate increasing or decreasing the size of connection pool. You can require human approval to run this script or simply configure it to execute automatically when it is triggered. Create a Runbook and associate it with a policy so that it will fire when the connection pool is exhausted.

To learn more about using alerts, health rules, and actions:

- See these topics:
- Watch this walk through the process.
-  Notifications
-  Actions

Policies

- Policy Structure
 - Policy Triggers
 - Policy Actions
- Policy List
- Learn More

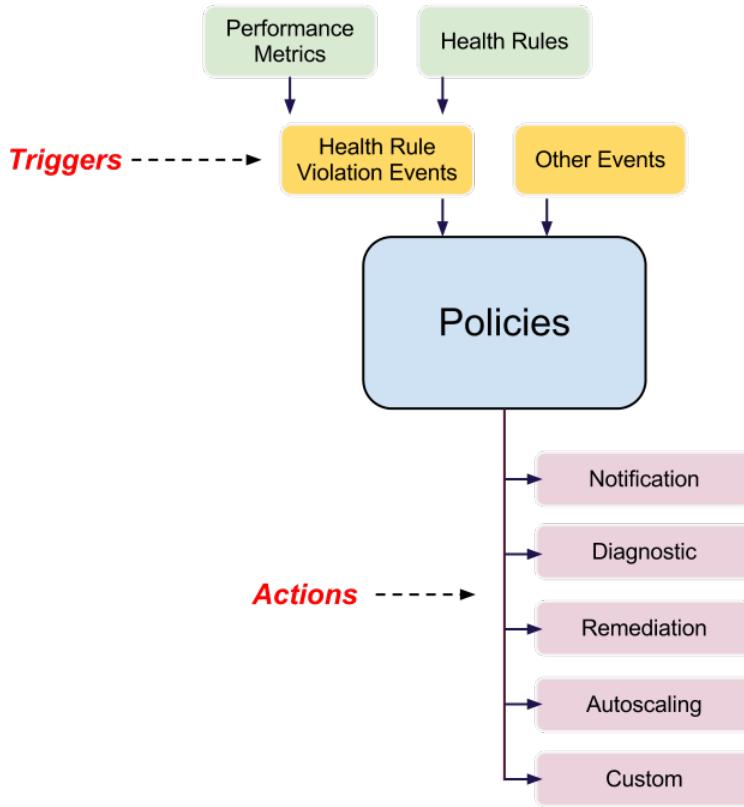
Policies let you anticipate problems and take actions to address those problems before they cause a severe slowdown or outage.

Policies provide a mechanism for automating monitoring and problem remediation. Instead of continually scanning metrics and events for the many conditions that could suggest problems, you can proactively define the events that are of greatest concern in keeping your applications running smoothly and then create policies that specify actions to start automatically when those events occur.

Policy Structure

A policy has two parts:

- triggers
- actions



Policy Triggers

Policy triggers are the events that cause the policy to fire. The events can be health rule violation events or other types of events. See [Health Rules](#), [Troubleshoot Health Rule Violations](#) and [Events](#).

You can define the triggering events broadly as events affecting any object in the application or very narrowly as those affecting only specific objects. You can create a policy that fires when an event involving all tiers in the application occurs, or only involving specific tiers. You can create a policy that fires on events affecting only certain nodes, or only certain business transactions or certain errors. You can very finely tune policies for different entities and situations.

For example, the broadly-defined ConnectionPoolPolicy fires whenever a resource pool limit is reached for any object in the application.

Policy Configuration		
	Name: ConnectionPoolPolicy	Enabled: <input checked="" type="checkbox"/>
TRIGGER	This Policy will fire when any of these Events occur on any object	
ACTIONS	Health Rule Violation Events <input type="checkbox"/> Health Rule Violation Started - Warning <input type="checkbox"/> Health Rule Violation Started - Critical <input type="checkbox"/> Health Rule Violation Upgraded - Warning to Critical <input type="checkbox"/> Health Rule Violation Downgraded - Critical to Warning <input type="checkbox"/> Health Rule Violation Ended	
	Other Events <ul style="list-style-type: none"> ▶ <input type="checkbox"/> Slow Transactions ▼ <input checked="" type="checkbox"/> Code Problems <ul style="list-style-type: none"> <input type="checkbox"/> Code Deadlock <input checked="" type="checkbox"/> Resource Pool Limit Reached ▶ <input type="checkbox"/> Application Changes ▶ <input type="checkbox"/> AppDynamics Config Warnings 	

The narrowly-defined JVMViolationInWebTier policy fires only when the JVM heap utilization or JVM garbage collection time is too high in the WebTier tier.

Here is the configuration of the triggering events for this policy:

TRIGGER Name: JVMViolationInWebTier Enabled:

This Policy will fire when ▶ any of these Events occur on ▶ any object

ACTIONS

Health Rule Violation Events

- Health Rule Violation Started - Warning
- Health Rule Violation Started - Critical
- Health Rule Violation Upgraded - Warning to Critical
- Health Rule Violation Downgraded - Critical to Warning
- Health Rule Violation Ended

Other Events

- ▶ Slow Transactions
- ▶ Code Problems
- ▶ Application Changes
- ▶ AppDynamics Config Warnings

What Health Rules?

- Any Health Rule
- These Health Rules:

JVM Heap utilization is too high	-
JVM Garbage Collection Time is too high	-

[Select Health Rule](#) 2 of 7 Selected [Create New Health Rule](#)

and here is the configuration of the triggering entities:

TRIGGER Name: JVMViolationInWebTier Enabled:

This Policy will fire when ▶ any of these Events occur on ▶ these specified objects

ACTIONS

Any objects

Any of these specified objects:

- Business Transactions
- Tiers and Nodes
- Exceptions

Events that are associated to ANY of the specified objects will make this Policy fire

Select Tiers or Nodes

- Tiers
- Nodes

Select Tiers

These specific Tiers

Selected Tiers (1)

	Name	Type
	WebTier	Application Server

A policy violation exists when at least one of the specified triggering events occurs on at least one of the specified objects.

Policy Actions

You can assign one or more actions to be automatically taken in response to a policy violation.

For example, the actions for the ConnectPoolPolicy violation are to take a thread dump and run a script to increase the pool size.

TRIGGER Name: ConnectionPoolPolicy Enabled:

Actions to Execute

ACTIONS

Name	Type
ThreadDump	Thread dump
IncreasePool	Run local script

How to see when Actions are executed?
If this Policy fires in response to an Event, the Actions that are executed will be visible in the 'Actions' column on the Events screen.
[View Events](#)

Other common actions are to restart an application server if it crashes, purge a message queue that is blocked, or trigger collection of transaction snapshots. You can also trigger a custom action to invoke third party systems. See [Integrate using Custom Action Scripts](#) for information about custom actions.

See [Actions](#) for more information about the different types of actions.

The maximum number of actions that an agent will process is xxx actions per minute.

Because the definition of health rules is separate from the definition of actions, and both health rules and actions can be very precisely defined, you can take different actions for breaching the same thresholds based on context, for example, which tier or node the violation occurred in.

Policy List

To access the list of policies in an application, select **Alert & Respond ->Policies**.

The policy list lists all the policies created for your application, with its triggers and actions taken. You can view and edit an action assigned to a specific policy by clicking the action in the policy list.

A screenshot of a web-based application interface titled 'Inventory Management > Alert & Respond > Policies'. The top navigation bar includes icons for Create Policy, Edit, Delete, Copy, Enable, and Disable. Below the navigation is a table with columns: Name, Trigger, Enabled, and Actions to Execute. A single row is visible for 'ConnectionPoolPolicy' with 'Other Events: Resource Pool Limit Reached.' Under the 'Actions to Execute' column, there are two entries: 'ThreadDump' and 'IncreasePool'. A green callout box with the text 'Click here to view or edit the action.' points to the 'Actions to Execute' column for the 'ConnectionPoolPolicy' row.

Learn More

- [Actions](#)
- [Configure Policies](#)
- [Custom Actions](#)
- [Diagnostic Sessions](#)
- [Events](#)
- [Health Rules](#)
- [Workflow Automation](#)
- [Integrate using Custom Action Scripts](#)

Configure Policies

- [Structure of the Policy Wizard](#)
 - To access the Policy Wizard
- [Configuring the Policy Trigger](#)
 - To configure policy triggers
- [Configuring the Policy Actions](#)
 - To configure policy actions
- [Learn More](#)

Structure of the Policy Wizard

The Policy Wizard contains two panels:

- **Trigger:** Sets the policy name, enabled status, events that trigger the policy, entities that are affected by the policy
- **Actions:** Sets the actions to take when the policy is triggered.

To access the Policy Wizard

1. Click **Alert & Respond -> Policies** in the left navigation pane.
2. Do one of the following:
 - To edit an existing policy, select the policy and click the edit icon.
 - To remove an existing policy, select the policy and click the delete icon.
 - To create a new policy, click the "+" icon.

Configuring the Policy Trigger

The policy trigger panel defines the events and objects generating the events that cause the policy to fire and invoke its actions.

For policy triggers that depend on health violation events, the health rules must be created before you can create a policy on them. See [Health Rules](#) and [Troubleshoot Health Rule Violations](#).

To configure policy triggers

1. Open the Policy Wizard for creating a new policy or for editing an existing one. See [To access the Policy Wizard](#).
2. Enter a name for the policy in the Name field.
3. To enable the policy, check the Enabled check box. To disable the policy, clear the Enabled check box.
4. On the left, click **Trigger** if it is not already selected.
5. Check the check boxes for the events that will trigger the policy. You may need to click the down arrow to expose specific events within an event category.
If you select any health rule violation events, you can choose whether any (all) health rule violations or only specific health rule violations will trigger the policy.
To designate specific health rule violations, select These Health Rules, click the "+" icon, and then choose the health rules from the embedded health rule browser.

This Policy will fire when ▾ any of these Events occur on ▶ any object

Health Rule Violation Events

- Health Rule Violation Started - Warning
- Health Rule Violation Started - Critical
- Health Rule Violation Upgraded - Warning to Critical
- Health Rule Violation Downgraded - Critical to Warning
- Health Rule Violation Ended

Other Events

- ▼ Slow Transactions
 - Slow Transactions
 - Very Slow Transactions
 - Stalled Transactions
- ▼ Code Problems
 - Code Deadlock
 - Resource Pool Limit Reached
- ▼ Application Changes
 - Application Deployment
 - App Server Restart
 - Application Configuration Change
- ▶ AppDynamics Config Warnings

What Health Rules?

Any Health Rule

These Health Rules:

 Business Transaction response time is much higher than normal -

+ Select Health Rule 1 of 12 Selected Create New Health Rule

6. When you have finished selecting the events that trigger the policy, click "any object" to configure the objects of events that will trigger the policy.

This Policy will fire when ▾ any of these Events occur on ▶ any object



If you select **Any Objects** the policy will be triggered by the configured events when they occur on any object in your application. To restrict the policy to specific objects, select **Any of these specified objects** and then choose the objects from the embedded object browser.
For example, the following policy fires only on the specified events generated on Node_8002 and Node_8003. You can similarly restrict the objects to specific tiers, business transactions or exceptions.

7. Click **Save** to save the policy configuration.

Configuring the Policy Actions

The policy actions panel defines the actions that the policy automatically initiates when the trigger fires.

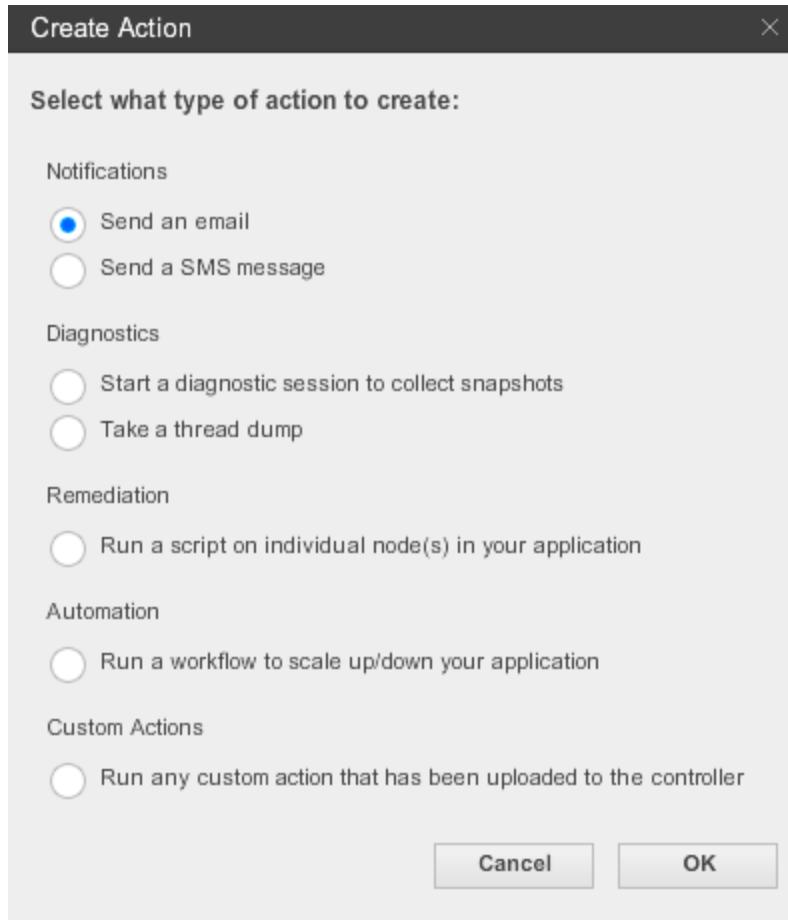
The actions must be created before you can create a policy that fires them. See [Actions](#) and the documentation for individual types of actions (notification actions, remediation actions, etc.) for information on creating an action.

To configure policy actions

1. If you have not already done so, open the policy wizard and edit the policy to which you want to add actions. See [To access the Policy Wizard](#).
2. On the left, click **Actions** if it is not already selected.
3. Click "+" icon and then **Select Action**.
The list of actions appears. You can filter the list by checking the check boxes for the types of actions you want to see.
4. In the list of actions, select the action that you want this policy to fire and click **Select**.

Name	Approval?	Type
ThreadDump	Yes	Thread dump
IncreasePool	Yes	Run local script

If you do not see an appropriate action for your needs, click **Create Action** to create an action. See [Actions](#).



Then select the newly-created action to assign it to the policy that you are configuring.

5. Click **Save** in the Policy Wizard.

Learn More

- Policies
- Events
- Health Rules
- Troubleshoot Health Rule Violations
- Actions
- Notification Actions
- Diagnostic Actions
- Remediation Actions (Java only)
- Auto-Scaling Actions
- Custom Actions
- Create a Workflow
- Workflow Automation

Health Rules

- Health Rule Types
- Health Rule Conditions
 - Critical and Warning Conditions
- Health Rule Applicability
 - Entities Affected by a Health Rule
- Health Rule Evaluation Scope
- Health Rule Schedules
 - Health Rule Enabled Schedule
 - Health Rule Evaluation Window
 - Health Rule Wait Time After Violation

- Default Health Rules
- Process for Setting Up Health Rules
- Health Rule Management
- Learn More

A health rule defines a condition or set of conditions in terms of metrics. The condition compares the performance metrics that AppDynamics collects with some static or dynamic threshold that you define. If performance exceeds the threshold, a health rule violation event is triggered.

You can create policies based on health rule violation events to initiate an automated action that responds to the violation.

You create health rules using a health rule wizard. See [Configure Health Rules](#).

Health Rule Types

AppDynamics categorizes health rules according to type.

The health rule types are:

- **Overall Application Performance:** monitor metrics related to load, response time, slow calls, stalls, etc. at the application level
- **Business Transaction Performance:** monitor metrics related to load, response time, slow calls, stalls, etc. at the business transaction level
- **Node Health-Transaction Performance:** monitor metrics related to load, response time, slow calls, stalls, etc. at the node level
- **Node Health-Hardware, JVM, CLR :** monitor metrics related to CPU, heap, disk i/o, JVM, CLR, etc. at the node level
- **Node Health-JMX:** monitor metrics related to connection pools, tread pools, etc. at the node level (Java only)
- **Error Rate:** monitor exception and error rates at the application or tier level

The health rule type determines which metrics AppDynamics presents in the health rule wizard, thereby simplifying the health rule creation process by presenting you with a manageable number of relevant options.

You can also specify **Custom** as the health rule type, in which case all the metrics that AppDynamics collects are available.

Health Rule Conditions

A condition consists of a Boolean statement based on metrics. The values to compare in these statements can be static or dynamic thresholds.

Static thresholds are straightforward. For example, is a Business Transaction's average response time greater than 200 ms?

A dynamic threshold is a baseline for a metric that is a rolled-up value for every hour of a rolling time window. This roll-up operation depends on the type of baseline. For example, a daily trend baseline rolls up values for the hours for each day whereas a weekly trend baseline rolls up values for same the hour of the day of the week. The threshold evaluation is performed by comparing against that baseline. For more information about baselines, see [Behavior Learning and Anomaly Detection](#).

You can define a threshold for a health rule based on a single metric value or on a mathematical expression built from multiple metric values.

Examples of conditions are:

- IF the value of the Average Response Time is greater than the default baseline by 3 X the Baseline Standard Deviation . . .
- IF the count of the Errors Per Minute is greater than 1000 . . .
- IF the number of MB of Free Memory is less than 2 X the Default Baseline . . .
- IF the value of Errors per Minute/Calls per Minute over the last 15 days > 0.2 . . .

The last example combines two metrics in a single condition. You can use the expression builder inside the health rule engine to the create conditions based on a complex expression comprising multiple interdependent metrics

Often a condition consists of multiple statements that evaluate multiple metrics. A health rule is violated either when one of its condition evaluates to true or when all of its conditions evaluate to true, depending on how it is configured.

You can correlate multiple metrics to simulate the health rules used in your managed environment. For example, a health rule that measures response time (average response time greater than some baseline value) makes more business sense if it is correlated with the application load (for example, 50 concurrent users) on the system. You may not want to use the response time condition alone in a

policy that triggers a remedial action if the load is low, even if the response time threshold is reached. To configure this correlation, the first part of the condition would evaluate the actual performance measurement and the second part would ensure that the health rule is violated only when there is sufficient load.

Critical and Warning Conditions

Conditions are classified as critical or warning.

Critical conditions are evaluated before warning conditions. If you have defined a critical condition and a warning condition in the same health rule, the warning condition is evaluated only if the critical condition is not true.

The configuration procedures for critical and warning conditions are identical, but you configure these two types of conditions in separate panels. You can copy a critical condition configuration to a warning configuration and vice-versa and then adjust the metrics in the copy to differentiate them. For example, in the Critical Condition panel you can create a critical condition based on the rule:

- IF the Average Response Time is greater than 1000

Then from the Warning Condition panel, copy that condition and edit it to be:

- IF the Average Response Time is greater than 500

As performance changes, a health rule violation can be upgraded from warning to critical if performance deteriorates to the higher threshold or downgraded from critical to warning if performance improves to the warning threshold.

Health Rule Applicability

A health rule can be broadly applied to an entire application. For example, you can create business transaction performance health rules that evaluate certain metrics for all business transactions in the application or node health rules that cover all the nodes in the application or all the nodes in specified tiers. The default health rules are in this category of broadly-applied policies.

You can also create health rules that are very narrowly applied to a limited set of entities in the application, or even a single entity such as a node or a JMX object or an error. For example, you can create a JMX health rule that evaluates the initial pool size and number of active connections for specific connection pools in nodes that share certain system properties.

The health rule wizard lets you specify precisely which entities in the application the health rule affects, enabling the creation of very specific health rules. For example, for a business transaction you can limit the tiers that the health rule applies to or specific business transactions by name or by names that match certain criteria.

Create Health Rule

OVERVIEW

AFFECTS

CRITICAL CONDITION

WARNING CONDITION

Select what Business Transactions this Health Rule applies to

All Business Transactions in the Application

All Business Transactions in the Application

Business Transactions within the specified Tiers:

These specified Business Transactions:

Business Transactions matching the following criteria:

For the node health rules, you can specify that a health rule applies only to nodes that meet certain criteria:

Create Health Rule

OVERVIEW

AFFECTS Nodes Tiers

CRITICAL CONDITION

WARNING CONDITION

What Does this Health Rule affects?

Select what Nodes this Health Rule affects

Type of nodes **Java and .NET nodes**

Nodes matching the following Criteria:

Nodes with a Name that Equals

Nodes with the following Meta-Info Properties

+ Add Meta Info Criteria

Nodes with the following Environment Variables

+ Add Environment Variable Match Criteria

Nodes with the following JVM System Environment Properties (java nodes only)

+ Add JVM System Property Match Criteria

You can see the entities affected by a rule in the health rule list and in the health rule detail. Click **View Dashboard During Health Rule Violation** to see what was happening in the application at the time that the rule was violated.

Health Rule Violation Event

Summary [Actions Executed](#)

Type	
Health Rule	 CallCount
Health Rule Type	Business Transaction Performance (load, response time, slow calls, etc)
Affects	 /RunBookDemo/inventory

[View Dashboard During Health Rule Violation](#)



Entities Affected by a Health Rule

To reduce noise from health rule violations on entities that you do not want to monitor, you can specify precisely which entities are affected by a health rule.

For an Overall Application Performance health rule type, the health rule applies the entire application, regardless of business transaction, tier, or node.

For a Business Transaction Performance health rule type, you can apply the health rule to:

- All Business Transactions in the application
- All Business Transactions within tiers that you select
- Individual Business Transactions that you select
- Business Transactions with names that have patterns matching criteria that you specify (such as all Business Transactions with names that start with "INV")

For a Node Health--Transaction Performance or Node Health-Hardware, JVM, CLR health rule type, you can apply the health rule to:

- All tiers in the application
- Individual tiers that you specify
- All nodes in the application
- Nodes within tiers that you specify
- Individual nodes that you specify
- Nodes with names, meta-data, environment variables or JVM system environment properties with matching criteria that you specify

For a Node Health--JMX health rule type, you select

- the JMX objects on which the health rule is evaluated

and apply the health rule to:

- All nodes in the application
- Nodes within tiers that you specify
- Individual nodes that you specify
- Nodes with names matching criteria that you specify

For an Error Rates health rule type, you can apply the health rule to:

- All errors in the application
- Errors within tiers that you specify
- Individual errors that you specify
- Errors with names matching criteria that you specify; can include exceptions that are not in the system

For custom policies, affected entities are restricted indirectly by the hard-coded metrics used to configure the health rule's conditions.

Health Rule Evaluation Scope

The evaluation scope applies only to business transaction performance type policies and node health policies in which the affected entities are defined at the tier level.

The health rule evaluation scope defines how many nodes in the affected entities must violate the condition before the health rule is considered violated.

For example, you may have a critical condition in which the condition is unacceptable for any node, or you may want to trigger the violation only if the condition is true for 50% or more of the nodes in a tier.

Options for the evaluation scope are:

- any node - If any node exceeds the threshold(s), the violation fires.
- percentage of the nodes - If x% of the nodes exceed the threshold(s), the violation fires.
- number of nodes - If x nodes exceed the threshold(s), the violation fires.
- the tier average - Evaluation is performed on the tier average instead of the individual nodes.

Health Rule Schedules

You can configure the times that a health rule is enabled, the evaluation time period, and how much recent data to use to evaluate the health rule.

Health Rule Enabled Schedule

A health rule can be enabled at all times or you can configure the schedule during which the rule is in effect.
Built-in schedules are:

- End of business hour
- Weekday lunch
- Weekday mornings
- Weekdays
- Weekends

You can also create a new schedule based on UNIX Chron expressions using your custom values.

The default is for a health rule to be enabled at all times (always).

Health Rule Evaluation Window

Set this value to specify the amount of recent data to use to evaluate the health rule. This is the period of time over which AppDynamics evaluates the health rule to determine whether a violation exists.

The default is 30 minutes.

For metrics based on an average calculation, such as average response time, the determination of a health rule violation is established by the average over the evaluation window time. For example, if the window is set to five minutes, AppDynamics averages the average response time over that five-minute period to compare with the threshold. For metrics based on a sum calculation, such as number of calls, AppDynamics totals the number of calls over the five-minute period and compares that total with the threshold.

Health Rule Wait Time After Violation

Normally health rules are evaluated every minute.

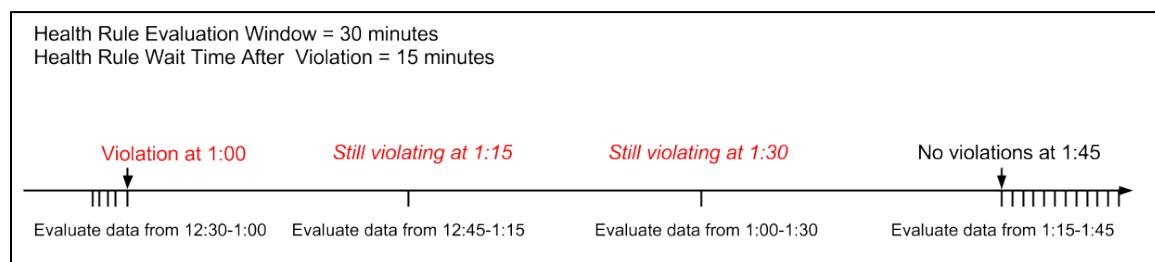
Set this value to how long to wait, in minutes, before re-evaluating a health rule that has been violated for a particular entity. This prevents generation of a large number of alerts and actions triggered by the violated rule before enough time has elapsed for the problem to be fixed.

The default for the wait time after violation is 30 minutes.

After the wait time has elapsed, the violated health rule is re-evaluated on the affected entity where the violation occurred. By now, the evaluation window has moved, so the data is evaluated for a later time period than before. At this point, one of three situations can arise:

1. The health rule is no longer being violated. In this case the evaluation returns to its normal schedule which is to evaluate to health rule every minute.
2. The health rule is still being violated, but there has been no state change. In other words, if there was a critical violation, there is still a critical violation. In this case AppDynamics waits another wait time period before re-evaluating the health rule.
3. The health rule is still being violated, but there has been a state change; for example the violation has escalated from warning to critical or de-escalated from critical to warning. In this case AppDynamics waits another wait time period before re-evaluating the health rule.

The timeline below illustrates the evaluation frequency with a configuration of a 30-minute evaluation window and a 15-minute wait period after violation. In this scenario, AppDynamics is evaluating the rule every minute until 1:00, when it detects a violation. It then switches to a 15-minute evaluation frequency, which it maintains until the violation is no longer detected, at which time it switches back to a 1-minute evaluation frequency. Note that the evaluation window is still moving during this time, so that only the last 30 minutes of data is evaluated. The slice of data that was evaluated when the violation was first detected is not the same set of data that was evaluated when the violation ceased.



Default Health Rules

The default health rules provided by AppDynamics are:

Health Rule Name	Health Rule Type
Business Transaction response time is much higher than normal	Business Transaction Performance
Business Transaction error rate is much higher than normal	Business Transaction Performance
CPU utilization is too high	Node Health - Hardware, JVM, CLR Performance

Memory utilization is too high	Node Health - Hardware, JVM, CLR Performance
JVM Memory Heap is too high	Node Health - Hardware, JVM, CLR Performance
JVM Garbage Collection Time is too high	Node Health - Hardware, JVM, CLR Performance
CLR Garbage Collection Time is too high	Node Health - Hardware, JVM, CLR Performance

The predefined default health rules appear in the dashboards when they are violated.

In many cases the default health rules may be the only health rules that you need. However, the thresholds may not be configured appropriately for your application, but you can edit them. You can also disable the default health rules.

Process for Setting Up Health Rules

AppDynamics recommends the following process for setting up health rules for your application:

1. Identify the key metrics that you need to monitor for business transaction health, node health, and error rates. These metrics should be representative of the overall health of your application.

2. Click **Alert & Respond->Health Rules** to examine the default health rules that are provided by AppDynamics. Compare your list of metrics with the metrics configured in these rules.

If the default health rules cover all the key metrics you need, determine whether the pre-configured thresholds are applicable to your environment. If necessary, modify the thresholds for your needs.

You can also view the list of affected entities for each default health rule and modify the entities affected by a default health rule.

3. If the health rules do not cover all your needs or if you need very finely-applied health rules to cover specific use cases, create new health rules.

First, identify the type of the health rule that you want to create. See [Health Rule Types](#). Then decide which entities should be affected by the new rule. See [Entities Affected by a Health Rule](#). Then define the conditions to monitor.

4. Create schedules for health rules, if needed.

In some situations a health rule is more useful if it runs at a particular time. See [Health Rule Schedules](#).

5. Configure policies that trigger specific actions when the health rules are violated. See [Policies](#).

Health Rule Management

To view current health rules in an application, including the default health rule, and to access the health rule wizard, click **Alert & Respond->Health Rules**.

Current health rules are listed in the left panel. If you click one of these rules, an information table appears in the right panel with one row for each entity affected by the health rule. To see detailed information by tier, click the tier cell for the affected entity.

In the left panel you can directly delete or duplicate a health rule. From here you can also access the health rule wizard to add a new rule or edit an existing one.

To delete an existing health rule:

1. Select the health rule in the left panel.

2. Click the delete icon.

The health rule is removed.

To duplicate an existing health rule:

1. Select the health rule in the left panel.

2. Click the duplicate icon.

The health rule is duplicated under the name you assign.

To edit an existing health rule:

1. Select the health rule in the left panel.

2. Click the edit icon.

The health rule wizard appears, with its values configured. You can modify these values in the wizard.

To create a new health rule:

1. Click the add icon.
2. The health rule wizard appears with some default values configured. You define the health rule in the wizard.

See [Configure Health Rules](#) for details on using the health rule wizard.

Learn More

- [Notification Actions](#)
- [Configure Health Rules](#)
- [Configure Baselines](#)
- [Events](#)
- [Policies](#)

Configure Health Rules

- [Structure of the Health Rule Wizard](#)
- [Configuring Generic Health Rule Settings](#)
 - [To Configure Generic Health Rule Settings](#)
 - [To Create a New Health Rule Schedule](#)
- [Configuring Affected Entities](#)
 - [To Configure Affected Entities](#)
 - [To Configure Affected Entities for Custom Health Rule Types](#)
- [Configuring Health Rule Conditions](#)
 - [To Configure a Condition](#)
 - [To Configure a Condition Component](#)
 - [To build an expression](#)
- [Configuring the Evaluation Scope](#)

This topic describes the detailed steps for configuring health rules using the Health Rule Wizard.

Structure of the Health Rule Wizard

To access the Health Rule Wizard:

1. Click **Alert & Respond -> Health Rules**.
2. To edit an existing health rule, in the left panel of the health rule list, select the health rule and click the Edit icon.
3. To remove an existing health rule click the delete icon.
4. To create a new health rule, click "+" .

The Health Rule Wizard opens. It contains four panels:

1. **Overview:** Sets the health rule name, enabled status, health rule type, health rule enabled period, health rule evaluation time
2. **Affects:** Sets the entities evaluated by the health rule. The options presented vary according to the health rule type set in the Overview panel.
3. **Critical Condition:** Sets the conditions, whether all or any of the conditions constitutes a health rule violation, the evaluation scope (BT and node health policies defined at the tier level only); includes an expression builder to create complex expressions containing multiple metrics.
4. **Warning Condition:** Settings are identical to Critical Condition, but configured separately.

You can navigate among these panels using the **Back** and **Next** buttons at the bottom of each panel or by clicking their entries in the left panel of the wizard. You should configure the panels in order, because the configuration of the health rule type in the Overview panel determines the available affected entities in the Affects panel as well as the available metrics in the Condition panels.

Configuring Generic Health Rule Settings

You configure generic settings for all health rules in the Overview panel.

To Configure Generic Health Rule Settings

1. In the Overview panel of the Health Rule Wizard, if you are creating a new health rule, enter the health rule name in the Name field. If you are editing an existing health rule, the name will already be there. You can change the name of the health rule in the Name field.
2. To enable the health rule check the Enabled check box. To disable the health rule, clear the Enabled check box.

3. Click one of the health rule types in the health rule type list to select the health rule type.

This setting affects metrics offered for configuration in subsequent windows in the wizard, so you must select a health rule type before continuing to other windows. If none of the predefined health rules are applicable for your needs, select Custom. Custom health rule types are offered all metrics for subsequent configuration in the Condition panels.

4. If the health rule is always (24/7) enabled, check the Always check box. If the health rule is enabled only at certain times, clear the Always check box and either:

- Select a predefined time interval from the drop-down menu.

or

- Click **Create New Schedule**.

The Create Schedule window opens. See [To Create a New Health Rule Schedule](#).

The time is the time at the site of the controller, not the app agent. For example, if the enabled time is set to 5pm-6pm, Mon-Fri and the controller is in San Francisco but the app agent is in Dubai, the health rule engine uses San Francisco time.

5. Click the dropdown menu and select a value between 1 and 360 minutes for the evaluation window. This is the amount of recent data to use to determine whether a health rule violation exists. This value applies to both critical and warning conditions. See [Health Rule Evaluation Window](#).

6. In the Wait Time after Violation section, enter the number of minutes to wait before evaluating the rule again for the same affected entity in which the violation occurred. See [Health Rule Wait Time After Violation](#).

7. Click **Save**.

8. Click **Next**.

To Create a New Health Rule Schedule

1. In the Overview window of the Health Wizard, clear the Always check box if it is checked.

2. Click **Create New Schedule**.

2. Enter a name for the schedule.

3. Enter an optional description of the schedule.

3. Enter the start and end times for the schedule as cron expressions. See <http://www.quartz-scheduler.org/documentation/quartz-2.1.x/tutorials/crontrigger> for details about and examples of cron syntax.

5. Click **Save**.

Configuring Affected Entities

This feature lets you define health rules broadly or fine-tune them very precisely to affect specific entities in your application.

The choices offered for configuring affected entities vary according to the health rule type previously configured in the Overview panel.

To Configure Affected Entities

1. In the Affects panel, select the entities affected by this health rule from the drop-down menu.

The entity affected and the choices presented in the menu depend on the health rule type configured in the Overview window.

See [Entities Affected by a Health Rule](#) for information about the types of entities that can be affected by the various health rule types.

If you have selected nodes based on matching criteria, specify the matching criteria. Nodes can be matched based on the node name, meta-information properties, environment variables, and JVM system environment properties (Java only).

If you are configuring a JMX health rule, select the JMX objects that the health rule is evaluated on.

If the configured health rule type is custom, see [To Configure Affected Entities for Custom Health Rule Types](#).

2. Click **Save**.

3. Click **Next**.

To Configure Affected Entities for Custom Health Rule Types

1. In the Affects window of the Health Rule Wizard, select the entity level that the health rule affects: Business Transaction Performance, Node Performance, or Application Performance.
2. If the health rule affects Business Transaction Performance, select the Business Transaction performance radio button and the Business Transaction affected from the drop-down menu. You can use the search field in the Select a Business Transaction browser that opens to locate the specific Business Transaction.
3. If the health rule affects Node Performance:
 - a. Select the Node Performance radio button.
 - b. Click **Node (for a new configuration) or *Change** (to modify an existing configuration).
 - c. In the Node browser that opens, navigate to the node which you want the health rule to affect.
 - d. Click **Select**.
3. If the health rule is not specific to a Business Transaction or Node, select Application Performance.

Configuring Health Rule Conditions

The configuration processes for critical and warning conditions are identical.

The Health Rule Wizard provides the ability to copy settings between Critical and Warning condition panels to facilitate the creation of similar health rules on the same metrics, possibly with different thresholds, or to assign different alerts or actions. For example, if you have already defined a critical condition and you want to create a warning condition that is similar, in the Warning Condition window click **Copy from Critical Condition** to populate the fields with settings from the Critical condition.

The high-level process for configuring conditions is:

1. Determine how many metrics and which metrics the health rule will evaluate. For each metric for which you want to use to evaluate performance, create a condition. You can use a single metric or build a mathematical expression based on multiple metrics and operations.
2. Decide whether the health rule is violated if all of the tests are true or if any single test is true.
3. For business transaction performance health rules and node health rule types that specify affected entities at the tier level, decide how many of the nodes must be violating the health rule to constitute a violation. See [Health Rule Evaluation Scope](#).
4. To configure a critical condition use the Critical Condition window. To configure a warning condition use the Warning Condition window.

To Configure a Condition

1. In the Conditional Condition or Warning Condition window, click **+ Add Condition** to add a new condition component. The row defining the component opens. See [To Configure a Condition Component](#). Continue to add components to the condition as needed.
2. From the drop-down menu above the components, select All if all of the components must evaluate to true to constitute violation of the rule. Select Any if a health rule violation exists if any single component is true.
3. For Business Transaction and node health rules that define the affected entities by tier rather than node, configure the evaluation scope. See [Health Rule Evaluation Scope](#).

Health Rule will violate if the conditions above evaluate to true for:

- the Tier Average (the aggregate of all Nodes)
- Any Node
- % of the Nodes
- of the Nodes

To Configure a Condition Component

1. In the first field of the condition row, name the condition.

This name is used in the generated notification text and in the AppDynamics console to identify the violation.

2. To select the metric on which the condition is based, do one of the following:

- To specify a simple metric, click the metric icon to open a small metric browser.
The browser displays metrics appropriate to the health rule type. Select the metric to monitor and click **Select Metric**.
The selected metric appears in the test configuration.

or

- To build an expression using multiple metrics, click the gear icon at the end of the row and select **Use a mathematical expression of 2 or more metric values**.
This opens the mathematical expression builder where you can construct the expression to use as the metric. See [To Build an Expression](#).

4. From the Value drop-down menu before the metric, select the qualifier to apply to the metric from the following options:

- Minimum
- Maximum
- Value
- Sum
- Count
- Current

Value refers to a metric over the entire evaluation time length configured in the Overview window. Current is the latest value.

5. From the drop-down menu after the metric, select the value against which the metric is evaluated.

- To compare the metric with a literal value, select < **Specific value** or > **Specific Value** from the menu, then enter the specific value in the text field. For example:

Value of Errors per Minute > 100

- To compare the metric with a baseline, select < **Baseline** or > **Baseline** from the drop-down menu, and then select the baseline to use, the numeric qualifier of the unit of evaluation and the unit of evaluation. For example:

Maximum of Average Response Time is > Baseline of the Daily Trend by 3 X the Baseline Standard Deviation

See [Baselines and Periodic Trends](#) for information about the baseline options.

6. Click **Save**.

You can remove a component from the condition by clicking the delete icon.

To build an expression

To access the expression builder to create a complex expression as the basis of a condition, click the gear icon at the end of the row and select **Use a mathematical expression of 2 or more metric values**.

In the expression builder, use the Expression pane to construct the expression.

Use the Variable Declaration pane to define variables based on metrics to use in the expression.

1. In Variable Declaration pane of the Mathematical Expression builder, click **+ Add variable** to add a variable.

2. In the Variable Name field enter a name for the variable.

3. Click **Select a metric** to open a small metric browser

Follow the instructions in [To configure a test](#) to select a metric.

4. From the drop-down menu select the qualifier for the metric.

5. Repeat steps 1 through 4 for each metric that you will use in the expression.

You can remove a variable by clicking the delete icon.

6. Build the expression by typing the expression in the Expression pane. Click the **Insert Variable** button to insert variables created in

the Variable Declaration pane.

The screenshot shows the 'Mathematical Expression' dialog box. In the 'Expression' section, the text '{numSlows} + {numVerySlows} + is entered. In the 'Variable Declaration' section, there is a table:

Variable Name	Variable Definition
numStalls	Value of Stall Count
numVerySlows	Value of Number of Slow Calls
numSlows	Value of Number of Slow Calls

Below the table is a button '+ Add Variable'. At the bottom are 'Cancel', 'Use Expression', 'Back', and 'Next' buttons. A tooltip 'Insert Variable...' is visible above the 'Use Expression' button.

7. When the expression is built, click **Use Expression**.

The expression appears as the metric in the condition configuration window.

The screenshot shows the condition configuration window. It displays the expression '{numSlows} + {numVerySlows} + {numStalls}' in the 'Edit Expression' field. Below it, the condition is set to 'is > Specific Value' with the value '100' entered.

Configuring the Evaluation Scope

This setting is applicable to business transaction performance type policies and node health policies that specify the affected entities at the tier level.

If the **Health Rule will violate if the conditions above evaluate to true** section is visible, click the appropriate radio button to set the evaluation scope.

If you select percentage of nodes, enter the percentage. If you select number of nodes, enter the absolute number of nodes.

Troubleshoot Health Rule Violations

- Troubleshoot Health Rule Violations
 - To troubleshoot health violations
 - View Health Violations in the AppDynamics Console

- Learn More

"Health" throughout the AppDynamics UI refers to the extent to which the component being monitored adheres to health rules.

You can create health rules to automate pro-active monitoring and problem mediation in your managed environment. In addition, AppDynamics provides a set of default health rules, so even if you have not explicitly created any health rules, you may see health rule violations reported for your application unless you have disabled the default health rules.

A health rule violation exists when the conditions that define the rule are true. For example, if a health rule condition is defined as the CPU%Busy rate should not exceed 90%, if that rate exceeds 90%, a health rule violation exists.

For general information about health rules see [Health Rules](#).

You can troubleshoot health rule violations by getting a list of the health rule violations from the **Troubleshoot->Health Rule Violations** menu item or by clicking on a health rule violation that is displayed in the AppDynamics console.

Troubleshoot Heath Rule Violations

You can access the list of health rule violations in your application for the selected time range.

To troubleshoot health violations

1. In the left navigation pane, click **Troubleshoot -> Health Rule Violations**.

The list of health violations displays.

The screenshot shows the AppDynamics Troubleshoot > Health Rule Violations page. The left sidebar has 'Hide Filters' and 'Health Rule Violation Details' buttons. The main area has a search bar and a 'last 15 minutes' filter. Two radio buttons are shown: 'View All Health Rule Violations in the Time Range' (selected) and 'View Only Health Rule Violations Open Now'. A table lists four violations:

Status	Health Rule	Description	Start Time	End Time	Duration	Affects
Open	CallCount Business Transaction Performance (load, response time, slow calls, etc)	Health rule violation status chan	01/29/13 11:44:55	-	Ongoing (2 hours, 45 minutes)	/RunBookDemo/inventory
Open	JVM Heap utilization is too high Node Health - Hardware, JVM, CLR (cpu, heap, disk I/O, etc)	Health rule violation status chan	01/29/13 12:35:55	-	Ongoing (1 hour, 54 minutes)	WebTier AdminServer
Open	CPU utilization is too high Node Health - Hardware, JVM, CLR (cpu, heap, disk I/O, etc)	Health rule violation status chan	01/29/13 12:35:55	-	Ongoing (1 hour, 54 minutes)	WebTier AdminServer
Open	Memory utilization is too high Node Health - Hardware, JVM, CLR (cpu, heap, disk I/O, etc)	Health rule violation status chan	01/29/13 12:35:55	-	Ongoing (1 hour, 54 minutes)	WebTier AdminServer

Buttons at the bottom include 'Refresh Tree' and 'View' next to each row.

2. Select **View All Health Rule Violations in the Time Range** or **View Only Health Rule Violations Open Now**.

It is possible that health rule violations that were reported are no longer open because remedial action has been taken or performance has improved on its own.

3. To see the filters click **Show Filters**. To hide them click **Hide Filters**.

With the filters showing in the left filters panel you can select the health rule violations that you want to troubleshoot. You can view all health rule violations or expand the nodes in the tree to select by health rule type (such as business transaction health rules or node health rules) or affected entity (such as business transaction, tier or node).

You can filter health rule violations by entering the health rule in the search field.

The list of health rule violations is displayed in the right panel, with their status, description, start time, end time and duration (if ended), and affected entity.

4. To see the health rule that was violated for a specific violation, select the health rule violation from the list and click the link to the health rule configuration in the Health Rule column.

- To see the dashboard for the entity affected by the violation, click the link to the entity in the In the Affects column.
 - To troubleshoot a specific health rule violation, select the health rule violation from the list and click **Health Rule Violation Details**.
- The Health Rule Violations Event window displays a summary of the violation and any actions that were executed to respond to it.

The screenshot shows the 'Health Rule Violation Event' window with two tabs: 'Summary' and 'Actions Executed'. The 'Summary' tab is selected, displaying details about the violation:

- Type:** Health Rule (BT-1)
- Health Rule Type:** Business Transaction Performance (load, response time, slow calls, etc)
- Affects:** /BasicSample/1.pojoServlet
- View Dashboard During Health Rule Violation:** A button to view the dashboard at the time of the violation.

On the right side, there is a section for 'Actions Executed' which is currently empty, indicated by the message 'No Actions Executed' and a question 'What does this mean?'. Below this, there is a list of icons representing different actions.

Description:

Health rule violation status changed from None to Open (Critical level) for health rule 'BT-1' of type Business Transaction Performance. All of the following conditions were evaluated for each node on tier8086 for business transaction performance /BasicSample/1.pojoServlet and 1 nodes were found to be violating thresholds. For Node node8086:1) condition 1The condition 1 observed value 450 was greater than the threshold 10 for the last 3 minutes.

You can click the **View Dashboard During Health Rule Violation** in the details window to view the dashboard at the time the violation occurred. The time range in this and all other dashboards is set to the time range of the health rule violation. From the dashboard you can get an overall picture of the application at the time of the violation as well as access to snapshots using the dashboard's Transaction Snapshots tab, which allows you to drill down to the root cause of the problem.

View Health Violations in the AppDynamics Console

If you see a health rule violation reported in the console, you can click it to get more information about the violation.

Healthy activities are green; warning-level activities are yellow/orange; critical-level activities are red. For example:

Here are the health summary bars on the dashboards:



There is a health column in the business transaction list:

	Name	Health	Server Time (ms)
ViewItems.getAllItems	✓	33	
ViewCart.sendItems	⚠	756	
UserLogOut.memberLogOut	✓	10	
UserLogin.memberLogin	✓	16	

Health violations are recorded as events.



Click a health rule violation from the Events list to see a summary of the violation, a link to the appropriate dashboard at the time of the violation, and actions executed in response to the violation if any were executed.

Health Rule Violation Started - Critical

Summary Actions Executed (0)

Event Type: ✗ Health Rule Violation Started - Critical

Health Rule: ☒ Slow Requests

Health Rule Type: Business Transaction Performance (load, response time, slow calls, etc.)

Affects: ☒ Supplier Search

View Dashboard During Health Rule Violation

Violation State: Open
Duration: Ongoing (6 minutes)
From: 05/06/13 4:48:55 PM To: -

No Actions Executed [What does this mean?](#)

Description:
Health rule violation status changed from None to Open (Critical level) for health rule 'Slow Requests' of type Business Transaction Performance.
All of the following conditions were evaluated for each node onE-Commerce for business transaction performance Supplier Search and 1 nodes were found to be violating thresholds.
For Node E-Commerce-Node-8004:

Learn More

- Health Rules
- Configure Health Rules
- Policies
- Alert and Respond
- Transaction Snapshots

Actions

- Types of Actions
- Actions Limits

- Actions Requiring Approval
- Viewing and Creating Actions
 - To view and edit existing actions
 - To create an action
- Learn More

An action is a predefined, reusable, automated response to an event. You can use actions to automate your runbooks.

A policy can trigger an action in response to any event. You configure which actions are triggered by which events when you configure policies. See [Policies](#).

Types of Actions

You can create the following types of actions:

- Notifications
- Diagnostics
- Remediation
- Custom Actions
- Cloud Auto-Scaling

Not all actions are applicable to all application environments or to all situations. Below are some general guidelines concerning different types of actions. For more details, see the pages on the specific actions before you assign an action to a policy.

- The diagnostic thread dump actions can be performed only on nodes running a Java agent.
- The diagnostic session actions can be triggered only by violations of business transaction health rules or slow or stalled transaction events, since these are the events that produce a view into transaction snapshots.
- Remediation actions run a local script in a node and are available on nodes running the machine agent. See [Install the Machine Agent](#).
- Custom Actions require a dedicated controller, deployed using either the on-premise or SaaS option. This feature is not supported for accounts on multi-tenant SaaS controllers.
- Cloud Auto-Scaling actions require a previously created workflow. See [Workflow Automation](#).

Actions Limits

The Controller limits the actions invoked based on the number of triggering events per event type. There is a maximum of ten events for any single event type that can trigger actions in a given minute. If the number of triggering events per type exceeds the limit, the actions that would have been triggered by the excess events are not started. You will not see a visual indication that these actions are not being started.

For example, your application can have up to ten Health Violation Started events triggering actions and up to ten Resource Pool Limit Reached events triggering actions within the same minute. But if you have eleven Health Violation Started events firing, the action that would be triggered by the eleventh event is not started.

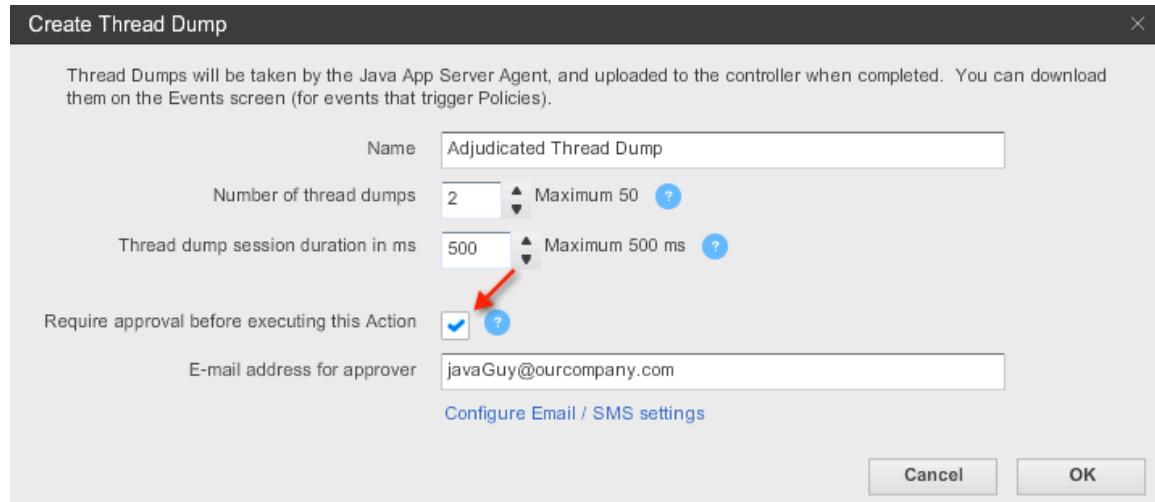
To reduce unnecessary actions, there is a configurable limit on the number of diagnostic and remediation actions that AppDynamics will invoke. The default limit is five actions per minute per machine for each type of action.

If, for example, a policy is configured on all the nodes, where there are 100 nodes triggering actions AppDynamics randomly selects five of the actions to execute.

To avoid exceeding the limits, design your policies so that they do not trigger an excessive number of actions for any particular event. You can generate fewer events by configuring the affected entities of your health rules at the tier level. See [Entities Affected by a Health Rule](#).

Actions Requiring Approval

For actions that take thread dumps or run a local script, you can optionally require email approval to run the action whenever it is triggered. If you configure this option, human intervention is required before the "automated" action actually starts.



If you specify the approval required option when you configure the action, when the action is triggered an email containing a link is sent to the configured email address. The link presents a login screen (if the user is not already logged in to AppDynamics) and after the user logs in, a dialog requesting approval to take the thread dump or run the script. The user can click in this dialog to approve and start the action or cancel the action.

If you do not check the Require approval option before executing the Action check box, the action will start automatically.

Viewing and Creating Actions

To view and edit existing actions

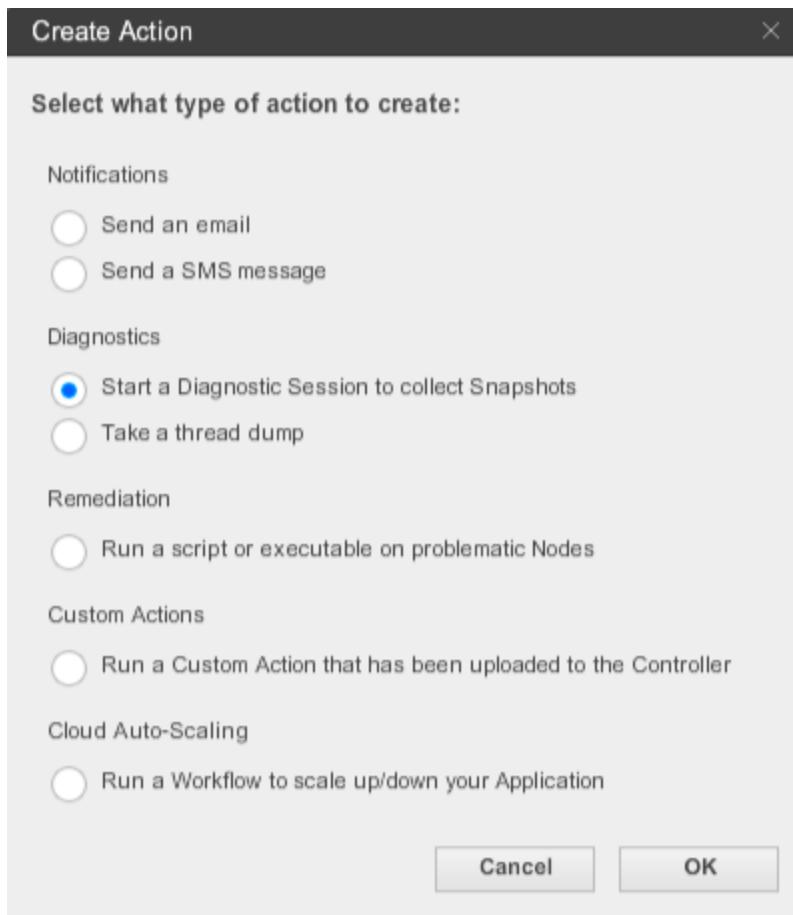
1. Click **Alert & Respond -> Actions** in the left navigation pane.
The list of actions in the application appears.

To filter the types of actions displayed in the list, select the type at the top of the list. For example, to see only diagnostic and remediation actions, check Diagnostics and Remediation and clear the other check boxes.

2. To examine or modify an action, select the action in the list and double-click or click **Edit**.
3. To delete an action, select the action in the list and click **Delete**.

To create an action

1. Click **Alert & Respond -> Actions** in the left navigation pane.
2. Click the **Create Action** button.



3. Select the type of action that you want to create.

4. Click **OK**.

The instructions beyond this point vary depending on the type of action you are creating. See the topics for the action type you have selected.

Learn More

- Policies
- Notification Actions
- Diagnostic Actions
- Remediation Actions
- Cloud Auto-Scaling Actions
- Custom Actions
- Install the Machine Agent

Notification Actions

- Email Notifications
 - To create an email notification
- SMS Notifications
 - To create an SMS notification
- Learn More

A notification action sends an email or SMS to a recipient list. The text of the notification is automatically generated by the event that triggered the action.

Email Notifications

An email notification contains a deep link to the details of the event that triggered it. Clicking this deep link takes you directly to the place to start troubleshooting the problem.

To create an email notification

1. Follow the instructions in [To Create an Action](#), selecting **Notifications->Send an email** in the Create Action window.
2. Enter the email address to which to send the notification.
3. Click **Save**.

If email and SMS settings have not been configured for AppDynamics, configure them now. See [Configure the SMTP Server](#).

SMS Notifications

An SMS notification includes the phone number of the recipient.

To create an SMS notification

1. Follow the instructions in [To Create an Action](#), selecting **Notifications->Send an SMS message** in the Create Action window.
2. Enter the phone number to which to send the notification.
3. Click **Save**.

If email and SMS settings have not been configured for AppDynamics, configure them now. See [Configure the SMTP Server](#).

Learn More

- [Actions](#)
- [Email Digests](#)
- [Policies](#)

Diagnostic Actions

- [Diagnostic Action Results](#)
 - To get the details of a diagnostic session or a thread dump that has been initiated by an action
- [Diagnostic Session Actions](#)
 - To create a diagnostic session action
- [Thread Dump Actions \(Java only\)](#)
 - [Agent Limit on Thread Dumps](#)
 - To create a thread dump action
- [Learn More](#)

A diagnostic action can:

- start a diagnostic session to collect snapshots
- take a thread dump (Java only)

When performance is slow or your application is experiencing a lot of errors you can start a diagnostic action to get to the root cause.

A diagnostic session gives you a view into captured transaction snapshots with full call graphs. These snapshots help you diagnose violations of business transaction performance health rules or slow or stalled transaction events. The affected entity of the event triggering a diagnostic session must be a business transaction.

A thread dump is a general-purpose snapshot of the state of all threads that are part of a JVM process. The state of each thread is presented with a stack trace that shows the contents of each thread's stack. Thread dumps are used for diagnosing JVM performance problems, such as code deadlocks.

Thread dumps are not supported for .NET agents.

Diagnostic Action Results

The results of a diagnostic action that has executed are available in the events list for the event that triggered the action.

To get the details of a diagnostic session or a thread dump that has been initiated by an action

1. Click **Events** in the left navigation pane.
2. Locate the row for the event that triggered the action for which you want to see the results.
3. In the Actions Executed column, click the diagnostic sessions or thread dump icon for the event that you want to troubleshoot.

Diagnostic Session Actions

A diagnostic session is always associated with a business transaction. It shows transaction snapshots with full call graphs to help you drill down to the root cause of a problem.

To create a diagnostic session action

1. Follow the instructions in [To create an action](#), selecting **Diagnostics->Start a diagnostic sessions to collect snapshots** in the Create Action window.
2. Enter a name for the action, the duration of the diagnostic session in minutes, and the number of snapshots to take per minute.
3. Select whether a diagnostic session will be started for any business transaction affected by an event or specific business transactions.
If you choose specific business transactions, specify the business transactions that will trigger the diagnostic session by moving them from the "available" list to the "selected" list. The business transactions that you can specify are not limited to those that triggered the action.
4. Click **OK**.

Thread Dump Actions (Java only)

You can direct the agent to take a thread dump for a specified number of samples (maximum of 50) with each sample lasting for a specified number of milliseconds (maximum of 500 ms). The thread dump is executed on the node.

Thread dump actions are not supported on .NET.

Agent Limit on Thread Dumps

One thread dump operation is executed at a time. They are not executed in parallel. If additional thread dump requests are received while one is being executed, they are queued with a limit of five per agent.

If the five thread dumps per agent limit is exceeded, the console shows an event with a thread dump operation that was skipped because of the limit and the associated action dialog for the executed policy links to this event.

To create a thread dump action

1. Follow the instructions in [To create an action](#), selecting **Diagnostics->Take a thread dump** in the Create Action window.
2. Enter a name for the action, the number of samples to take, and the duration of the samples in milliseconds.
3. If you want to require approval before the thread dump action can be started, check the **Require approval before this Action** check box and enter the email address of the individual or group that is authorized to approve the action. See [Actions Requiring Approval](#) for more information.
4. Click **OK**.

Learn More

- Actions
- Call Graphs
- Diagnostic Sessions
- Policies
- Transaction Snapshots
- Install the Machine Agent

Remediation Actions

- Prerequisites for Local Script Actions
- Remediation Scripts
 - Guidelines for Remediation Scripts
 - Troubleshooting Remediation Scripts

- Remediation Example
- Creating a Local Script (Remediation) Action
 - To create a local Script Action
 - To see the output of the local script
- Learn More

A remediation action runs a local script in a node. The script executes on the machine from which it was invoked or on the machine specified by the remediation action configuration. You can use this type of action to automate your runbook procedures.

By default the script is a shell script in /bin/sh invoked with the **-ex** option, unless the script has a header, in which case the interpreter in the header is used. For example, if the script header is "#!/bin/perl", the PERL interpreter is invoked.

You can optionally configure the remediation action to require human approval before the script is started. See [Actions Requiring Approval](#).

Remediation actions can be performed on nodes running the Machine Agent. See [Install the Machine Agent](#).

Prerequisites for Local Script Actions

- The standalone (Java-based) Machine Agent must be installed running on the host on which the script executes. To see a list of installed machine agents for your application, click **View machines with machine-agent installed** in the bottom right corner of the remediation script configuration window. See [Install the Machine Agent](#) if you need to install a machine agent.
- The machine agent OS user must have full permissions to the script file and the log files generated by the script and/or its associated child processes.
- The script must be placed in a sub-directory of the machine agent installation directory that is named "local-scripts".
- The script must be available on the host on which it executes.
- Processes spawned from the scripts must be daemon processes

Remediation Scripts

A remediation script is run on the nodes affected by the violation that triggered it. You can configure the policy that triggers the action to run the script on 100% of the nodes affected by the triggering violation or specify a certain number of the affected nodes.

Guidelines for Remediation Scripts

A process exit code of zero indicates that the script execution succeeded. A non-zero exit code indicates that it failed.

The script should be written as generically as possible to allow it run on any of the nodes for which it is invoked. AppDynamics exports the following environment variables to the script runtime to provide context regarding the environment and the event that triggered the action.

Environment Variable	Cardinality (1 or N)	Notes
APP_ID	1	Name of the Application
EVENT_TIME	1	Timestamp of the event
EVENT_ID	1	Event Id
EVENT_TYPE	1	type of event, such as: ERROR, APPLICATION_ERROR, APPLICATION_INFO, STALL, BT_SLA_VIOLATION, DEADLOCK, MEMORY_LEAK, MEMORY_LEAK_DIAGNOSTICS, LOW_HEAP_MEMORY, ALERT, CUSTOM, APP_SERVER_RESTART, BT_SLOW, SYSTEM_LOG, INFO_INSTRUMENTATION_VISIBILITY, AGENT_EVENT, INFO_BT_SNAPSHOT, AGENT_STATUS, SERIES_SLOW, SERIES_ERROR, ACTIVITY_TRACE, OBJECT_CONTENT_SUMMARY, DIAGNOSTIC_SESSION, HIGH_END_TO_END_LATENCY, APPLICATION_CONFIG_CHANGE, APPLICATION_DEPLOYMENT, AGENT_DIAGNOSTICS, MEMORY, LICENSE
ENV_STARTUP_ARGS	1	Process args
ENV_SYSTEM_PROPERTIES	1	JVM System Props (When Java)

AFFECTED_ENTITY	1	Affected Entity that triggered the event
-----------------	---	--

Troubleshooting Remediation Scripts

To troubleshoot your remediation script, look for the process in the machine agent log. The log is located at

<Machine_Agent_Installation_Directory>/logs/machine-agent.log

The snippet below from the machine agent log shows both error and success messages from running a local script named "script.sh".

```
[Agent-Scheduler-1] 07 May 2013 18:20:24,580 ERROR RunLocalScriptEventHandler while executing run local script operation
[Agent-Scheduler-1] 07 May 2013 18:20:24,580 INFO RunLocalScriptRequestHandler - Received run local script request: opId=27,
actionGuid=f117d181-a3a0-407e-b0ac-0e3878547f2f
[Agent-Scheduler-1] 07 May 2013 18:20:24,581 ERROR ScriptExecutor - Script '/Users/akilman/script.sh' must reside in
'/Users/akilman/Work/cart-tmp/machineagent/local-scripts'
[Agent-Scheduler-1] 07 May 2013 18:20:24,593 ERROR RunLocalScriptEventHandler - Error occurred while executing run local script operation
[Agent-Scheduler-1] 07 May 2013 18:20:24,594 INFO RunLocalScriptRequestHandler - Received run local script request: opId=28,
actionGuid=45202a5b-af43-4f2a-9308-1bce7f2408b2
[Agent-Scheduler-1] 07 May 2013 18:20:24,594 ERROR ScriptExecutor - Script '/Users/akilman/script.sh' must reside in
'/Users/akilman/Work/cart-tmp/machineagent/local-scripts'
[Agent-Scheduler-1] 07 May 2013 18:20:24,606 ERROR RunLocalScriptEventHandler - Error occurred while executing run local script operation
[Agent-Scheduler-1] 07 May 2013 18:26:24,689 INFO RunLocalScriptRequestHandler - Received run local script request: opId=29,
actionGuid=b5b80d3e-7859-400f-927c-d42c1b495d0c
[Agent-Scheduler-1] 07 May 2013 18:26:24,693 INFO ScriptExecutor - Executing: [/Users/akilman/Work/cart-tmp/machineagent/local-
scripts/script.sh]
[Agent-Scheduler-1] 07 May 2013 18:26:24,693 INFO ScriptExecutor - Using working directory: /Users/akilman/Work/cart-tmp/machineagent
[Agent-Scheduler-1] 07 May 2013 18:26:26,582 INFO RunLocalScriptEventHandler - Run local script request completed successfully
[Agent-Scheduler-1] 07 May 2013 18:26:26,582 INFO RunLocalScriptRequestHandler - Received run local script request: opId=30,
actionGuid=29169ecf-fdf4-4a59-9a95-987d92a7610
[Agent-Scheduler-1] 07 May 2013 18:26:26,584 INFO ScriptExecutor - Executing: [/Users/akilman/Work/cart-tmp/machineagent/local-
scripts/script.sh]
[Agent-Scheduler-1] 07 May 2013 18:26:26,584 INFO ScriptExecutor - Using working directory: /Users/akilman/Work/cart-tmp/machineagent
[Agent-Scheduler-1] 07 May 2013 18:26:28,248 INFO RunLocalScriptEventHandler - Run local script request completed successfully
[Agent-Scheduler-1] 07 May 2013 18:26:28,249 INFO RunLocalScriptRequestHandler - Received run local script request: opId=31,
actionGuid=8c2e6530-184a-40ed-b672-1cf28aa7120d
[Agent-Scheduler-1] 07 May 2013 18:26:28,250 INFO ScriptExecutor - Executing: [/Users/akilman/Work/cart-tmp/machineagent/local-
scripts/script.sh]
[Agent-Scheduler-1] 07 May 2013 18:26:28,250 INFO ScriptExecutor - Using working directory: /Users/akilman/Work/cart-tmp/machineagent
[Agent-Scheduler-1] 07 May 2013 18:26:29,913 INFO RunLocalScriptRequestHandler - Run local script request completed successfully
```

Script is not in
correct directory.

RunLocalScript
succeeded.

Remediation Example

The following remediation action, named increasePool, executes a local script named runbook.sh, which increases the size of the connection pool on the JVM.

Create Remediation Script Action

You can specify any script or executable and the Machine Agent will execute it, and upload the results to the controller. You can download the script output on the Events screen (for events that trigger Policies).

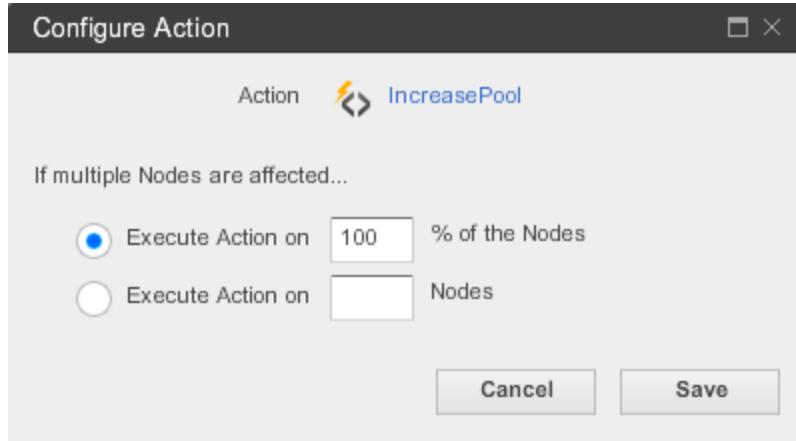
Name	IncreasePool
Relative path to script	<code>\$(machine.agent.directory)/local-scripts/runBook.sh</code>
Absolute paths to log files	<input style="margin-right: 10px;" type="button" value="+"/> <input style="margin-right: 10px;" type="button" value="X"/> <input style="margin-right: 10px;" type="button" value="-"/> <input style="margin-right: 10px;" type="button" value="?"/> Path <code>/tmp/script.out</code>
Script timeout in minutes	2
Require approval before executing this Action	<input type="checkbox"/>
E-mail address for approver	<input type="text"/>
Configure Email / SMS settings	
View machines with machine-agent installed Cancel OK	

A policy named ConnectionPoolPolicy triggers this action when the Resource Pool Limit Event fires:

Edit Policy - ConnectionPoolPolicy

Name	ConnectionPoolPolicy	Enabled	<input checked="" type="checkbox"/>
TRIGGER	This Policy will fire when ▾ any of these Events occur on ▶ any object		
ACTIONS	Health Rule Violation Events <input type="checkbox"/> Health Rule Violation Started - Warning <input type="checkbox"/> Health Rule Violation Started - Critical <input type="checkbox"/> Health Rule Violation Upgraded - Warning to Critical <input type="checkbox"/> Health Rule Violation Downgraded - Critical to Warning <input type="checkbox"/> Health Rule Violation Ended Other Events <input type="checkbox"/> Slow Transactions <input type="checkbox"/> Code Problems <input type="checkbox"/> Code Deadlock <input checked="" type="checkbox"/> Resource Pool Limit Reached		

The Action part of this policy specifies that the script should execute on all of the nodes affected by the Resource Pool Limit Reached event.



Creating a Local Script (Remediation) Action

To create a local Script Action

1. Follow the instructions in [To create an action](#), selecting **Remediation->Run a script or executable on problematic Nodes** in the Create Action window.
2. Enter a name for the action.
3. In the field that terminates the Relative path to script entry, enter the rest of the path to the executable script. Remediation scripts must be stored in a sub-directory of the machine agent installation. The sub-directory must be named "local-scripts". The following paths are all valid:

```

${machine.agent.directory}/local-scripts/runMe.sh
${machine.agent.directory}/local-scripts/johns_scripts/runMe.sh
${machine.agent.directory}/local-scripts/ops/johns_scripts/runMe.sh

```

4. Click the **+** to enter the absolute paths of any log files that the script writes to that you want included in the script output.
5. Enter the timeout period for the script process in minutes.
6. If you want to require approval before the script action can be started, check the **Require approval before this Action** check box and enter the email address of the individual or group that is authorized to approve the action. See [Actions Requiring Approval](#) for more information.
7. Click **OK**.

To see the output of the local script

1. Click **Events** in the left navigation pane to navigate to the Events list.
2. Locate the row for the event that triggered the action for which you want to see the results.
3. In the Actions column, click the remediation script icon.

Type	Summary	Time	Business Transaction	Tier	Node	Actions
×	Health Rule Violat Policy TooSlow has violated	01/31/13 2:00	/RunB...	-	Admin...	-
×	Health Rule Violat Policy TooSlow has violated	01/31/13 12:5	/RunB...	-	Admin...	-
×	Health Rule Violat Policy TooSlow has violated	01/31/13 11:3	/RunB...	-	Admin...	-
×	Health Rule Violat Policy TooSlow has violated	01/31/13 9:13	/RunB...	-	Admin...	-
×	Health Rule Violat Policy TooSlow has violated	01/31/13 9:07	/RunB...	-	Admin...	

2 Remediation Script(s) were executed.

4. In the script result list, select the script output that you want and click **Download Local Script Result**.

Select Local Script Result

Download Local Script Result

Time	Summary
01/31/13 9:09:55 AM	Run local script request completed successfully
01/31/13 9:09:48 AM	Run local script request completed successfully
01/31/13 9:09:55 AM	Run local script request completed successfully
01/31/13 9:09:48 AM	Run local script request completed successfully

Learn More

- [Actions](#)
- [Policies](#)
- [Install the Machine Agent](#)

Cloud Auto-Scaling Actions

- [To Create a Cloud Auto-Scaling Action](#)
- [Learn More](#)

A cloud auto-scaling action scales your application to your current capacity requirements, using a workflow for adding or removing one or more application servers. The workflow must already exist before you can create an action that uses it.

See [Workflow Automation](#) and [Create a Workflow](#).

To Create a Cloud Auto-Scaling Action

1. Follow the instructions in [To create an action](#), selecting **Cloud Auto-Scaling->Run a workflow to scale up/down your application** in the Create Action window.
2. Enter a name for the action.
3. Select the workflow from the dropdown list.
4. Click **OK**.

Learn More

- [Actions](#)
- [Workflow Automation](#)
- [Create a Workflow](#)
- [Policies](#)

Custom Actions

- [Creating the Custom Action](#)
 - [To create a custom action](#)
- [Learn More](#)

A custom action is typically used to integrate with third party alerting and ticketing systems. A custom action is different from other actions in that it executes just once on an on-premise controller.

Custom Actions require a dedicated controller deployed using either the on-premise or SaaS deployment options. Custom actions are not available to customers with accounts on multi-tenant SaaS controllers.

The custom action script and custom.xml file must already exist before you can create an action that uses it. The custom action scripts include parameters for specifying the affected entity, for example the tier, node, business transaction, etc. See [Integrate using Custom Action Scripts](#).

Custom actions are commonly used when you want to trigger a human work flow or leverage an existing alerting system that is external to AppDynamics. For example, you could use a custom action to file a JIRA ticket when AppDynamics reports that a connection pool is near saturation.

Creating the Custom Action

After the custom action script and custom.xml files have been tested manually and installed on the Controller, you can create the custom action.

To create a custom action

1. Follow the instructions in [To create an action](#), selecting **Automation->Run any custom action that has been uploaded to the controller** in the Create Action window.
2. Enter a name for the action.
3. Select the custom action from the dropdown list.
4. Click **OK**.

The custom action is now available for assignment to a policy.

Learn More

- [Integrate using Custom Action Scripts](#)

Email Digests

An email digest is a compilation of messages sent to a recipient list by email at a configured time interval.

The purpose of the digest is to notify the recipients of events that have occurred in the application.

The contents of the digest are automatically generated for the context of the events that it reports.

See [Configure Email Digests](#) for configuration details.

Configure Email Digests

- Structure of the Email Digest Wizard
- Configuring the Email Digest
 - To configure content settings
 - To configure digest recipients
 - To configure the digest interval
- [Learn More](#)

Structure of the Email Digest Wizard

To access the Email Digest Wizard:

1. Click **Alert & Respond -> Email Digests**.
2. To edit an existing email digest, select the digest and click the Edit icon.
3. To remove an existing digest click the delete icon.
4. To create a new digest, click "+" .

The Email Digest Wizard opens. It contains four panels:

1. Contents: Sets the digest name, enabled status, health rule type, events and objects that trigger the sending of the digest
2. Recipients Adds the email addresses of the digest recipients.
4. How Often: Sets how often the digest is sent, in hours.

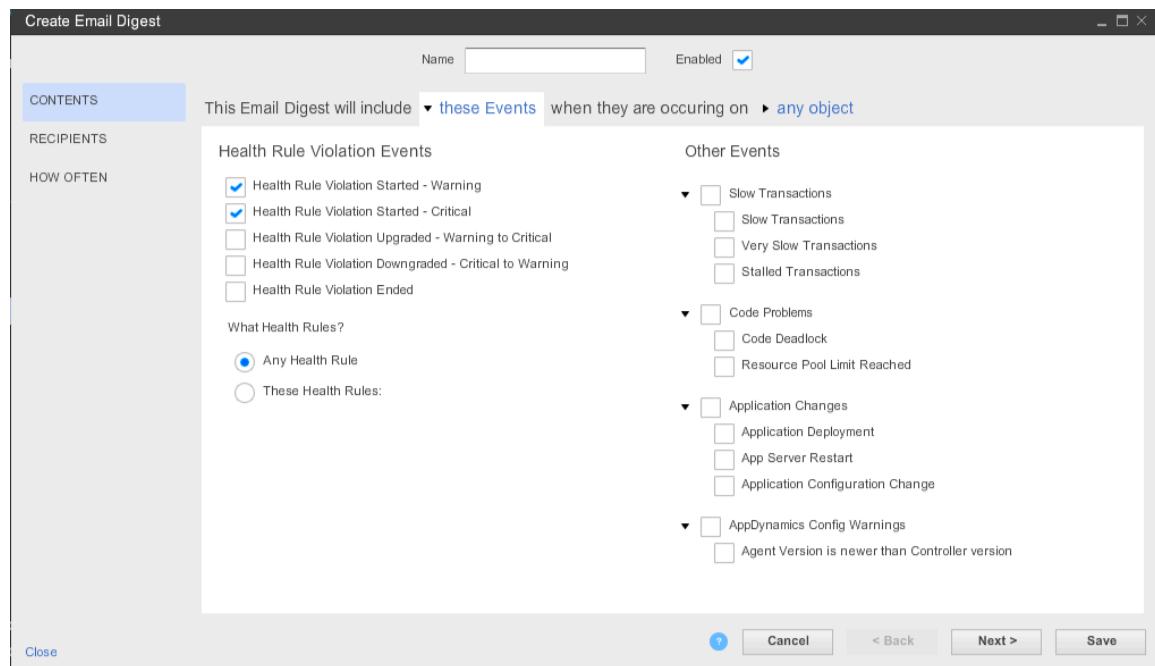
You can navigate among these panels using the Back and Next buttons at the bottom of each panel or by clicking their entries in the left panel of the wizard.

Configuring the Email Digest

To configure content settings

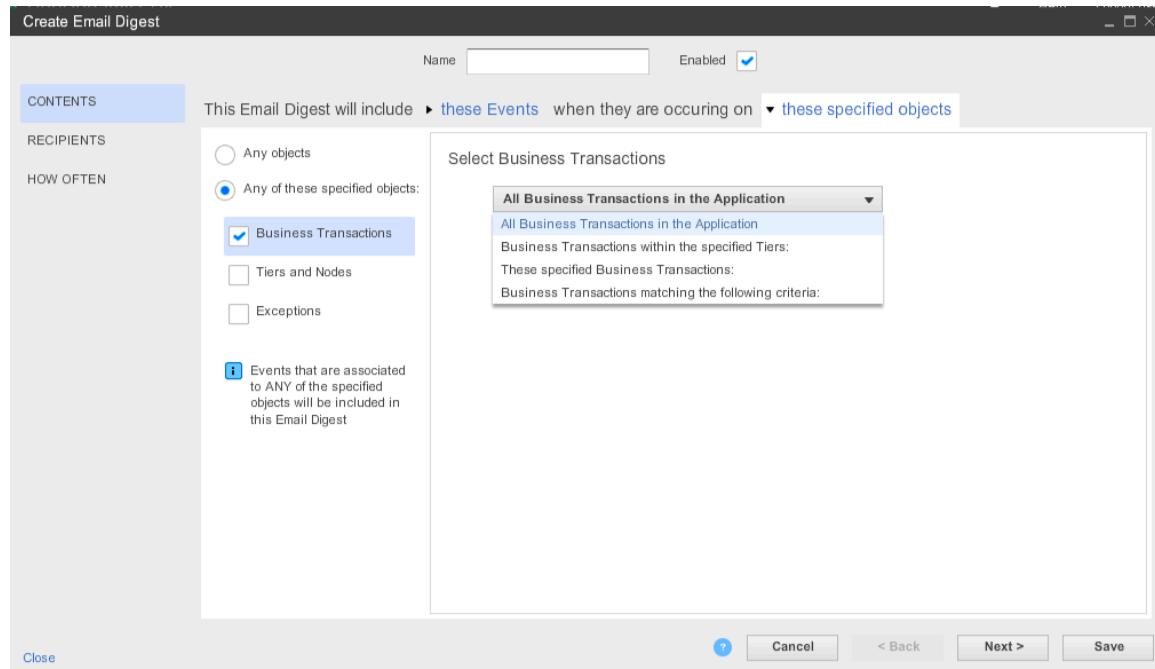
1. In the Content panel of the Email Digest Wizard, if you are creating a new digest, enter the digest name in the Name field. If you are editing an existing digest, the name will already be there. You can change the name of the digest in the Name field.
2. To enable sending the digest check the Enabled check box. To disable the digest, clear the Enabled check box.

3. Check the check boxes for the events that will be included in the digest. You may need to click the down arrow to expose specific events within an event category.



4. When you have finished selecting the events in the digest, you can click "any object" to refine the contents by specifying only events that affect certain objects in the application. If you select Any Objects the digest will include configured events when they occur on any object in your application.

To restrict the policy to specific objects, select Any of these specified objects and then choose the objects from the embedded object browser.



5. Click **Save** to save the digest configuration.

To configure digest recipients

Configuration of digest recipients involves selecting or creating an email notification action for every recipient of the digest. You can

create these notification actions from this Email Digest Wizard as well as from the **Actions** menu.

1. In the Recipient panel of the Email Digest Wizard, if you are creating a new digest, enter the digest name in the Name field. If you are modifying an existing digest, double-click the digest in the list.
2. To edit an existing recipient, select the recipient from the list and double-click or click the Edit icon.
3. To remove an existing recipient from the email digest, select the recipient from the list and click the delete icon.
4. To add a recipient, click the "+" sign.
5. Do one of the following:

- To add a recipient who has already been configured (i.e. an existing email notification action), click **Select Action**, select the email notification from the list, and click **Select**.

or

- Click **Create Email Action**, enter the email address of the recipient in the text field.

6. In the Add Action screen, you can also add an optional note to include in the email.

7. Click **Save** to add the recipient.

8. Click **Save** in the Email Digest Wizard to save the digest configuration.

To configure the digest interval

1. Click How Often to access the How Often panel.

2 In the text field enter an integer to indicate the number of hours between digests.

3. Click **Save** to save the digest configuration.

Learn More

- [Email Digests](#)
- [Policies](#)
- [Health Rules](#)
- [Notification Actions](#)

Alerting Wizard

- To access the alerting wizard
- To edit or delete the generated policy
- [Learn More](#)

You can configure an alert to be triggered when a health rule starts, escalates, and de-escalates, using the Alerting Wizard. The alert is sent to a single email address.

To access the alerting wizard

1. Click **Alert & Respond** in the left navigation pane.
2. Click the **Alert and Respond - Getting Started Wizard** at the bottom of the Alert & Respond screen.
3. If your SMTP server is not set up, click **Configure SMTP Server**.
See [Configure the SMTP Server](#) for instructions on configuring the SMTP server for email and SMS notifications.
4. Enter the email address to which alerts will be sent.
5. Click **Save**.
This wizard creates a policy named My Policy or My Policy *n* that sends email to the configured address whenever any health rule violation is started in the application.

Getting Started Wizard

Alert & Respond - Getting Started Wizard

This wizard will setup email alerts for when problems are detected.

- ✓ 1 Configure AppDynamics to send emails
 - [Configure SMTP server](#)
- ✓ 2 Enter email address to receive alerts
 - Email
- ✓ 3 Done!

You have created a [Policy](#) that will send emails to everyone@appdynamics.com when [Health Rule](#) violations occur.

AppDynamics has automatic Health Rules which compare the health of Business Transactions and Servers against their baselines. When a Health Rule violation happens, notifications can be sent, diagnostics can be performed, shell scripts can be executed, and more.

To edit or delete the generated policy

1. Click **Alert & Respond -> Policies** to access the policy list.

Name	Trigger	Enabled	Actions to Execute
My Policy	Health Rule Violation Events: Health Rule Violation Started - Critical , Health Rule Violation	✓	everyone@appD.com

2. Do one of the following:

- To delete the automatically generated policy, select the policy and click the delete icon.
- To edit the automatically generated policy, select the in the list and click the edit icon.
You can fine-tune the types of violations and events that trigger the alert by editing the policy manually. For example, you can change the name of the policy, add or remove events that trigger the notification, add additional notification email addresses or other actions to be triggered by the policy.

You can also create entirely new policies with different triggers and actions. See [Configure Policies](#).

3. Click **Save**.

Learn More

- Policies
- Actions
- Notification Actions
- Email Digests
- Configure Policies
- Configure Email Digests

Rapid Troubleshooting

The dashboards show you when problems occur and you need to take action. For example the [Transaction Scorecard](#) monitors business transactions and categorizes their performance according to [thresholds](#).



When AppDynamics indicates a problem you can easily go right into troubleshooting mode.

- From the All Applications dashboard click the Troubleshoot link.

The dashboard shows the ACME Book Store Application. The navigation bar includes Events, End User Experience, Alert & Respond, Troubleshoot (underlined), and Analyze. The main area displays 'Health Rule Violations' (7) and 'Load' (3.0m, 2.1k). A '2 Critical' alert is indicated by a red dot.

- Alternatively, from the left navigation menu click Troubleshoot.

The left navigation menu includes Servers, Events, End User Experience (selected), Troubleshoot (selected), Alert & Respond, Analyze, and Configure. The Troubleshoot section contains Slow Response Times, Errors, and Health Rule Violations (7). The main content area is titled 'Troubleshoot your Application' and lists three items: Slow Response Times, Errors, and Health Rule Violations, each with a corresponding icon.

See:

- Troubleshoot Slow Response Times
- Troubleshoot Errors
- Troubleshoot Health Rule Violations

Troubleshoot Slow Response Times

- Slow and Stalled Transactions
 - To troubleshoot slow and stalled transactions
- Slow Database and Remote Service Calls
 - To troubleshoot slow database and remote service calls
- Learn More

For tutorials see:

Tutorial for Java - Slow Transactions

Super-Simple Java Troubleshooting using Events

When you click **Troubleshoot -> Slow Response Times** the Slow Response Times window opens showing two tabs. You can drill down into transaction issues in the **Slow Transactions** tab, and into database or remote services issues in the **Slowest DB & Remote Services** tab.

Slow and Stalled Transactions

There are many reasons why a business transaction may be slow or stalled. The Slow Response Times tab helps you find the root cause whether that be resource or thread contention, deadlock, race condition, or something else.

By default AppDynamics considers a slow transaction one that lasts longer than 3 times the standard deviation for the last two hours and a very slow transaction 4 times the baseline for the last two hours.

By default AppDynamics considers a transaction that lasts longer than 45 seconds (4500 milliseconds) to be stalled.

You can configure these thresholds to better match your environment. See [Thresholds](#) and [Configure Thresholds](#).

To troubleshoot slow and stalled transactions

1. Click **Troubleshoot -> Slow Response Times**

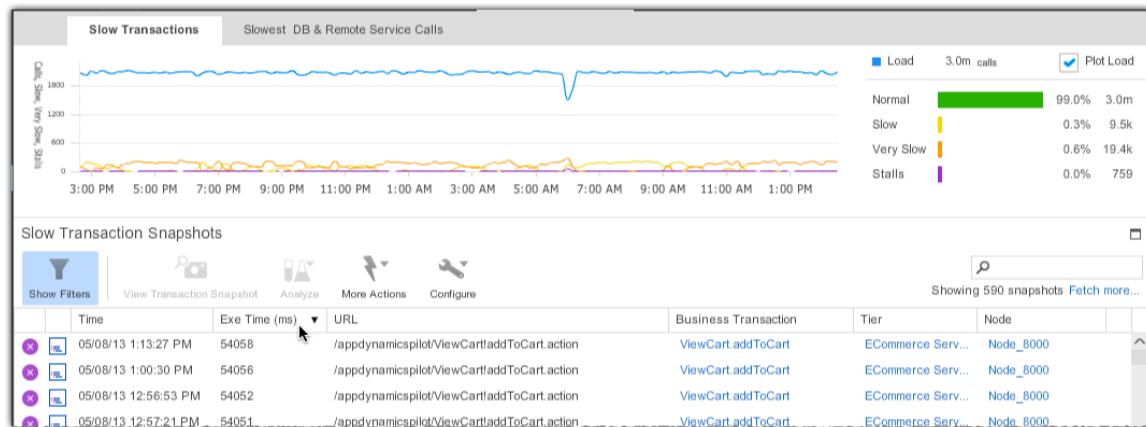
You can also access this information from tabs in the various dashboards.

2. Click the **Slow Transactions** tab if it is not selected.

In the upper pane AppDynamics displays a graph of the slow, very slow, and stalled transactions for the time period specified in the Time Range drop-down menu. Click the Plot Load checkbox to see the load.

In the lower pane AppDynamics displays the transaction snapshots for slow, very slow, and stalled transactions.

Click the Exe Time column to sort the transactions from slowest to fastest.



To drill down, select a snapshot from the list and click **View Transaction Snapshot**. See [Transaction Snapshots](#).

Slow Database and Remote Service Calls

Although AppDynamics does not instrument database and remote service servers directly, it collects metrics about calls to these backends from the instrumented app servers. This allows you to drill down to the root cause of slow database and remote service calls.

To troubleshoot slow database and remote service calls

1. Click **Troubleshoot -> Slow Response Times**.

You can also access this information from tabs in the various dashboards.

2. Click the **Slowest DB & Remote Service Calls** tab if it is not selected.

Call	Avg. Time per Call (ms)	Number of Calls	Max Time (ms)	Snapshots
ORDERSERVICE.CREATEORDER	9922.9	25933	91875	View snapshots
INSERT INTO ORDERREQUEST (ITEM_ID, NOTES) VALUES (?, ?)	302.5	10053	10003	View snapshots
GET POOLED CONNECTION FROM DATASOURCE	648.3	73	4002	View snapshots
ORDERQUEUE	337.9	44	1913	View snapshots

3. In the Call Type panel select the type of call for which you want to see information or select All Calls. The Call panel displays the call or query with the average time per call, number of calls, and maximum execution time (Max Time) for the calls with the longest execution time.

If transaction snapshots are available for a slow call, you can click **View Snapshots** link or you can select the call and click the **Correlated Snapshots** tab in the lower panel. From there you can select a snapshot and click **View Transaction Snapshot** to drill down to the root cause of the slow call.

	Time	Exe Time (r▼)	URL	Business Transaction	Tier	Node
[Icon]	05/08/13 3:00:17 PM	10091	/appdynar	ViewCart.sendItems	ECommerce Server	Node_8000
[Icon]	05/08/13 3:02:25 PM	10064	/appdynar	ViewCart.sendItems	ECommerce Server	Node_8000
[Icon]	05/08/13 3:02:25 PM	10064	/appdynar	ViewCart.sendItems	ECommerce Server	Node_8000
[Icon]	05/08/13 3:02:21 PM	10053	/appdynar	ViewCart.sendItems	ECommerce Server	Node_8000

See Transaction Snapshots.

Learn More

- Transaction Snapshots
- Configure Thresholds

Troubleshoot Errors

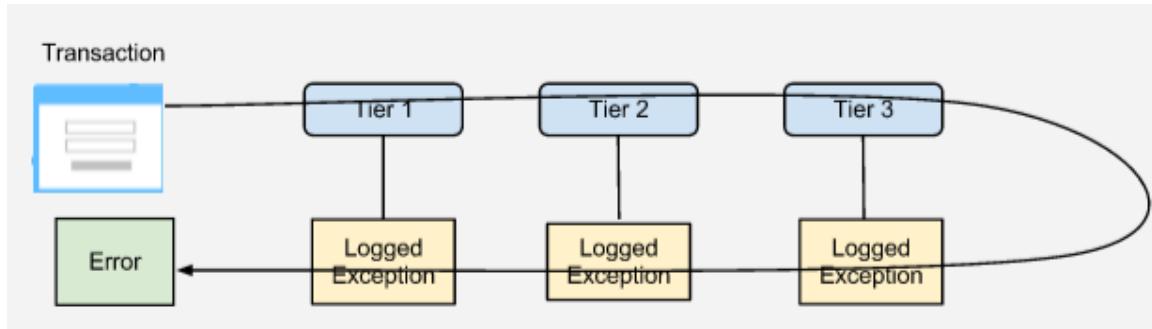
- Error Transactions and Exceptions
 - To Troubleshoot Error Transactions
 - To Troubleshoot Exceptions
- Learn More

Error Transactions and Exceptions

An error transaction is a transaction that experienced an error during transaction execution. The error can be a logged error or a thrown exception. If a transaction experiences an error, it is counted as an error transaction and not as a slow, very slow or stalled transaction even if the transaction was also slow or stalled.

An application server exception is a code-logged message outside the context of a business transaction.

There is not a one-to-one correspondence between the number of errors and the number of exceptions. For example, a business transaction may experience a single code 500 error in which several exceptions were logged as the transaction passed through multiple tiers.



You can configure the types of errors that AppDynamics detects as well as the types of exceptions to ignore. See [Configure Error Detection](#).

To Troubleshoot Error Transactions

1. Click **Troubleshoot -> Errors** in the left navigation panel.

The error viewer opens.

2. Click the **Error Transactions** tab if it is not already selected.

3. From the time range drop-down menu select the time range for which you want to view information about error transactions.

A graph of the error transactions displays at the top of the viewer. You can get an exact count of the errors per minute at a particular point in time by hovering with your pointing device on the line in the graph.

To the right of the graph is a summary of the load and the error transactions.

Check the **Plot** check box if you want the graph at the top of the viewer to show the load over the selected time period. Clear this check box is clear, if you want the graph to show only the error transactions .

4. The error transaction snapshots are listed in the lower part of the viewer. To filter this list click **Show Filters** and select the filter criteria.

5. To examine the root cause of an error, select the snapshot from the list and click **View Transaction Snapshot**. See [Transaction Snapshots](#) for information about examining snapshots.

6. To identify the most expensive calls or queries, select a snapshot from the list and click **Analyze** and then click **Identify the most expensive calls / SQL statements in a group of snapshots**.

The Most Expensive Methods / SQL Statements viewer opens.

7. In the lower panel click the **Expensive Methods** tab to view the methods with their total and average execution times and call counts. Click the **Expensive SQL** tab to view the queries with their counts and execution times.

To Troubleshoot Exceptions

1. Click **Troubleshoot -> Errors** in the left navigation panel.

2. Click the **Exceptions** tab if it is not already selected.

The total exception count, HTTP Error Codes and Error Page Redirects for the selected time range are reported in the upper panel. You can get an exact count of the errors per minute at a particular point in time by hovering with your pointing device on the line in the graphs.

The exceptions list is displayed in the lower panel.

- To filter the exception list, enter the filter term in the filter text field.
- To see only errors with performance data, clear the Show Exceptions with 0 count checkbox.

3. To view details of a particular exception, select the exception in the lower panel and click **View Details**. The exception detail window displays.

4. To view transaction snapshots for an exception:

- a. In the exception detail window, click the **Occurrences of this Exception** tab.

- b. Select a snapshot from the list.
- c. Click **View Details**.
- d. In the snapshot flow map that displays, click **Drill Down**. See [Transaction Snapshots](#).

5. To view a stack trace for an exception:

- a. In the exception detail window, click the **Stack Traces for this Exception** tab.
- b. Click an exception in the left panel.

The right panel displays the stack trace for the selected exception.

Learn More

- [Troubleshoot Errors](#)
- [Configure Error Detection](#)
- [Transaction Snapshots](#)

Troubleshoot Node Problems

- [To Access the Node Problem Viewer](#)
- [Node Problem Viewer](#)
- [Filter Options for Node Problem Analysis](#)
 - [Baseline to Use](#)
 - [Data to Analyze](#)
 - [Analysis Type](#)
- [Learn More](#)

AppDynamics categorizes the ten items that deviate the most from the baseline performance as node problems.

You can analyze node problems from the Node Problems viewer.

To Access the Node Problem Viewer

To access the Node Problems viewer do one of the following:

- In the Node dashboard, from the Actions drop-down menu click **Analyze Node Problems**.
- or
- In the left navigation panel of the Snapshot viewer, click **NODE PROBLEMS**.

Node Problem Viewer

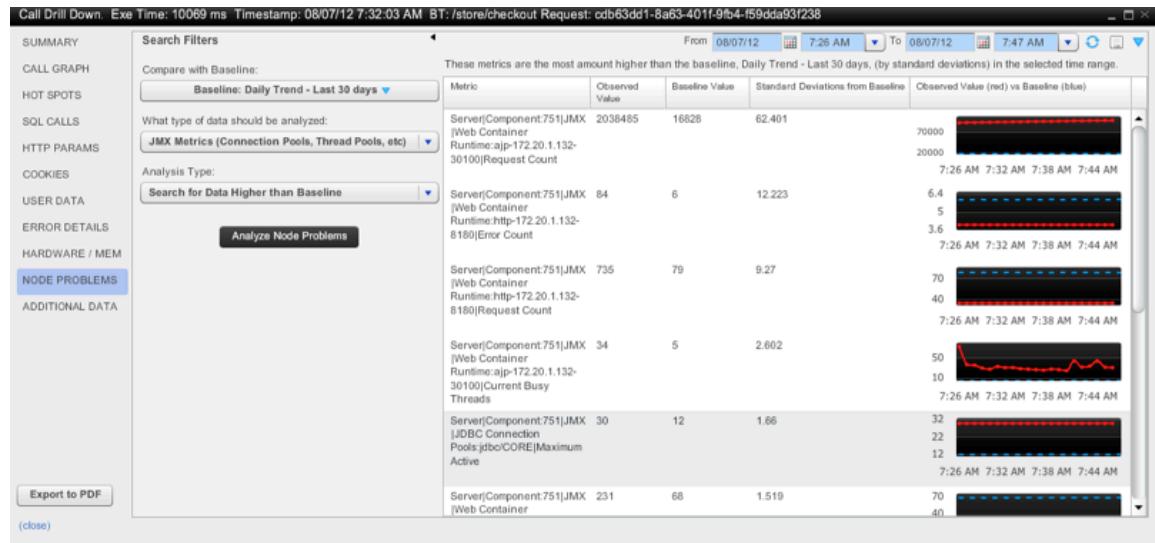
The right panel of the Node Problem viewer displays the metrics that deviate the most from their baselines for the specified time range.

The left panel of the Node Problem viewer lets you filter the types of data reported as node problems.

If accessed from the Snapshot viewer, the Node Problem viewer displays node problems for the time range of the snapshot. If accessed from the Node dashboard, it uses the time range set in the Node dashboard. You can edit the time range in the Node Problem viewer and then apply the new time range. You can also define and save a custom time range.

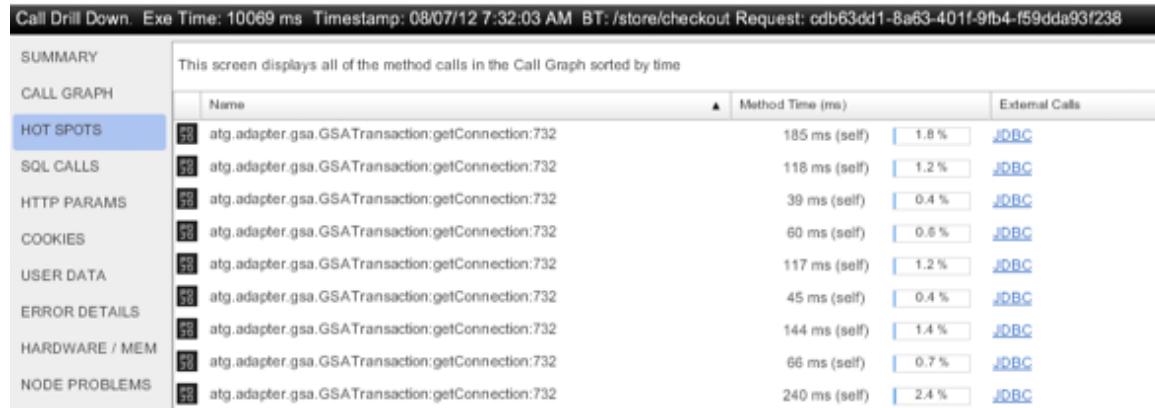
The selection of metrics displayed depends on the configured filter options.

The following Node Problem viewer shows a transaction that took over 10 seconds (10069 ms) to execute.



It shows that the request count was 62 standard deviations from its normal baseline and that the DB connection pool was 1.6 standard deviations above its normal baseline. It maxed out at 30 concurrent connections.

The **Hot Spots** tab for this snapshot shows the details of the connection pool latency.



In this case, the Node Problems viewer reveals how excessive web requests saturated the DB connection pool.

Filter Options for Node Problem Analysis

You can filter the following options in the Filter Options panel of the Node Problem viewer:

- Baseline to Use
- Data to Analyze
- Analysis Type

Baseline to Use

Specify which baseline to use to define node problems:

- No baseline
- All data - Last 15 days
- Daily Trend - Last 30 days (default)
- Weekly Trend - Last 6 months
- Monthly Trend - Last year

For information about baselines, see [Behavior Learning and Anomaly Detection](#).

Data to Analyze

Select the type of data to analyze from the drop-down menu. AppDynamics analyzes the ten items that deviate the most from the baseline performance for the specified type.

Analysis Type

Specify whether to display problems with values that are:

- higher than baseline
- lower than the baseline
- higher and lower than baseline

For example, if you are only interested in CPU that is too high, set the Analysis Type to Higher for Hardware Resources. On the other hand, if you want to monitor the load on your machine continuously because low CPU usage would also be a problem, set the Analysis Type to Higher and Lower.

Learn More

- [Behavior Learning and Anomaly Detection](#)

Call Graphs

- Call Graphs
 - To view call graphs
 - To Filter and Search for a Class Name in a Call Graph
- Diagnosing the Root Cause of Problems Using Call Graphs
 - Troubleshoot Problems using Call Graphs
 - Understanding Data Captured in Call Graphs
 - Class/Method Names and Time spent
 - External Calls Invoked by a Method
 - Viewing Call Graph Hot Spots
 - Advanced Options
 - Classes displayed in the call graphs
 - Packages and Namespaces excluded from the call graph
 - Configure instrumentation
- Learn More

This topic describes call graphs for code-level diagnostics.

Call Graphs

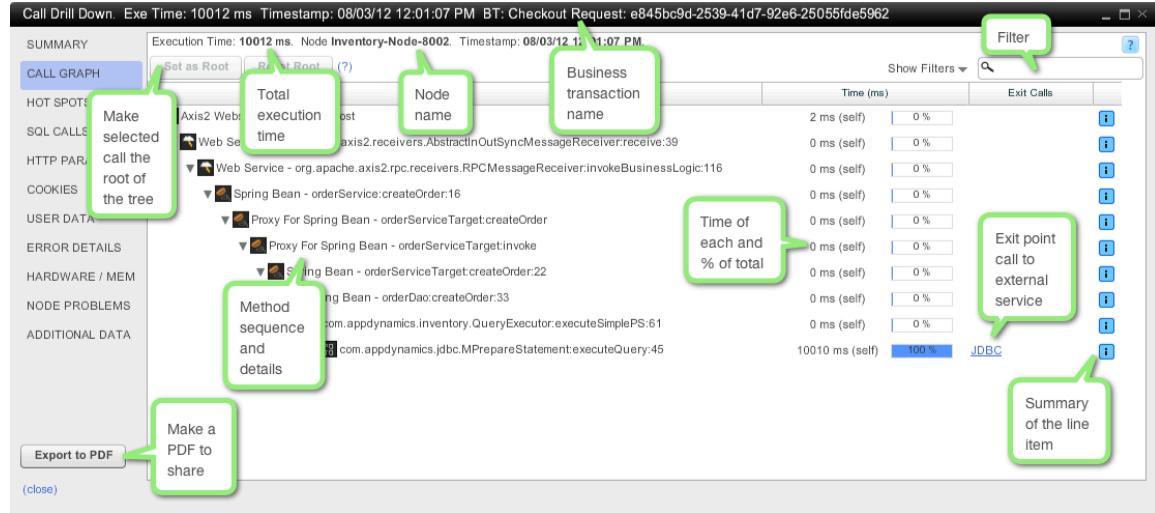
A call graph lists the methods in a call stack and provides information about each call. Call graphs help you diagnose performance issues and optimize the flow of a complex business transaction.

AppDynamics call graphs provide the following information:

- **Call information:** This includes the total execution time, the node name for the call graph, and the time stamp for the start of execution.
- **Method execution sequence:** An easily readable method execution sequence with the class names and the method names of each method.
 - **Application component information:** Application component information such as Servlet names, Struts Action classes, EJB names, Spring Bean IDs and proxies, message listeners, etc. POJOs are represented by class name, unless the POJO is a special type of component such as a struts action invoker. The class name is available for all call graph elements including those having logical class names.
 - **Code line number:** The line number of the element in the source code, if available.
- **Time distribution for each method:** The time distribution for each method along with the percentage of the total time in the call graph.
- **Exit call information:** Links to more information about exit calls such as JDBC, JMS, and Web Services.
- **Application component summary information:** Summary information for each element in the call graph.

In addition you can:

- **Filter the call graph list** using criteria such as application components, time, or all or part of the name.
- **Select an element of the list and set it as the root of the list** to drill down into large call graphs more efficiently.
- **Export the call graph to PDF** to share the information with other team members.



To view call graphs

1. From a dashboard, click the **Transaction Snapshots** tab.
2. Do one of the following:
 - Click **All Snapshots** to see all snapshots.
 - Click **Slow and Error Transactions** to see snapshots for slow and error transactions.
 - Click **Periodic Collection** to see snapshots collected periodically.
2. Select a particular transaction snapshot and click **View Transaction Snapshot**.

3. Click the **Drill Down** icon to see the call graph for the snapshot.

By default the originating call graph in a business transaction, generated by the entry point on the entry point tier, displays.

To Filter and Search for a Class Name in a Call Graph

The call graph is displayed when you click on the **drill down** option on the snapshot. You can filter and search for a particular class name on the call graph. To do this, use the search box available on the left side of the call graph.

Diagnosing the Root Cause of Problems Using Call Graphs

Troubleshoot Problems using Call Graphs

You can use call graphs to troubleshoot the problems in the flow map. Drill down into each tier using the **Drill Down** option. If there is only one invocation, AppDynamics displays the call graph.

When there are multiple invocations for that tier which participated in the transaction, the **Action -> Drill Down** menu displays a list of all call graphs for the node. Each call graph represents a call into the node. For example, if JVM A makes two remote calls to JVM B, then JVM B will have two call graphs.

Understanding Data Captured in Call Graphs

Class/Method Names and Time spent

Call graphs represent the sequence of execution of Java code on a JVM and the .NET code on the CLR. Each row represents a method call and the tree represents the execution sequence.

The total time spent in the call graph is displayed on the top left corner of the call graph.

Execution Time: 10066 ms. Node ECommerce-Node-8004. Timestamp: 08/31/12 7:11:30 AM.

[Set as Root](#) [Reset Root](#) ([?](#))

The Time (ms) column in the call graph shows the time spent in each method.

External Calls Invoked by a Method

If a method invokes external calls outside of the app server, such as a JDBC query or a Web Service call, there is a link to the call in the call graph

When you click on a link in Exit Calls column, you see the details associated with the call.

Type	Details	Count	Time (ms)	% Time	From Tier	To Tier	Downstream Call Time (ms)
Web Service	OrderSe	1	10025	99.7	ECommerce	Inventory	10021 ms

10025 ms

Web Service Name: OrderService
Operation Name: createOrder

Details: OrderService.createOrder

Drill Down into Downstream Call

If the detail screen has a Drill Down link, you can get a call graph for the calls downstream from the original call.

Viewing Call Graph Hot Spots

The most expensive methods are listed in the Hot Spots section of the transaction snapshot.

Advanced Options

Classes displayed in the call graphs

The sequence of Java method invocations in a call graph can include hundreds of classes. These classes are categorized as:

- Application classes
- AppServer/Container/Middleware classes (For example: Tomcat runtime classes, Websphere runtime classes, etc.)
- JDBC Driver/External infrastructure library classes (For example: Oracle Driver classes, Axis Web Service runtime classes, Hibernate classes, etc.)
- Third party utility libraries (For example: Apache commons libraries)

- Java/J2EE core libraries

Adding all of these classes together in a call graph is counterproductive for troubleshooting. The only mandatory classes required, are the classes from first category- the Application classes. You might require the other classes only in certain situations. To make the call graph more readable and to avoid the overhead of processing the timing information for all the non-application classes, other classes are not shown in the call graph.

Packages and Namespaces excluded from the call graph

A call graph can have a list of packages (Java) or namespaces (.NET) that are not displayed. To access this list, click on the message for "Some packages/namespaces have been excluded from the Call graph" message.

The screenshot shows a call graph visualization with a tooltip overlay titled 'Excluded Packages'. The tooltip contains the following text:

In order to reduce the call graph size, certain packages may be excluded.
[Configure Excluded Packages for all Call Graphs](#)

These packages were excluded from this call graph:

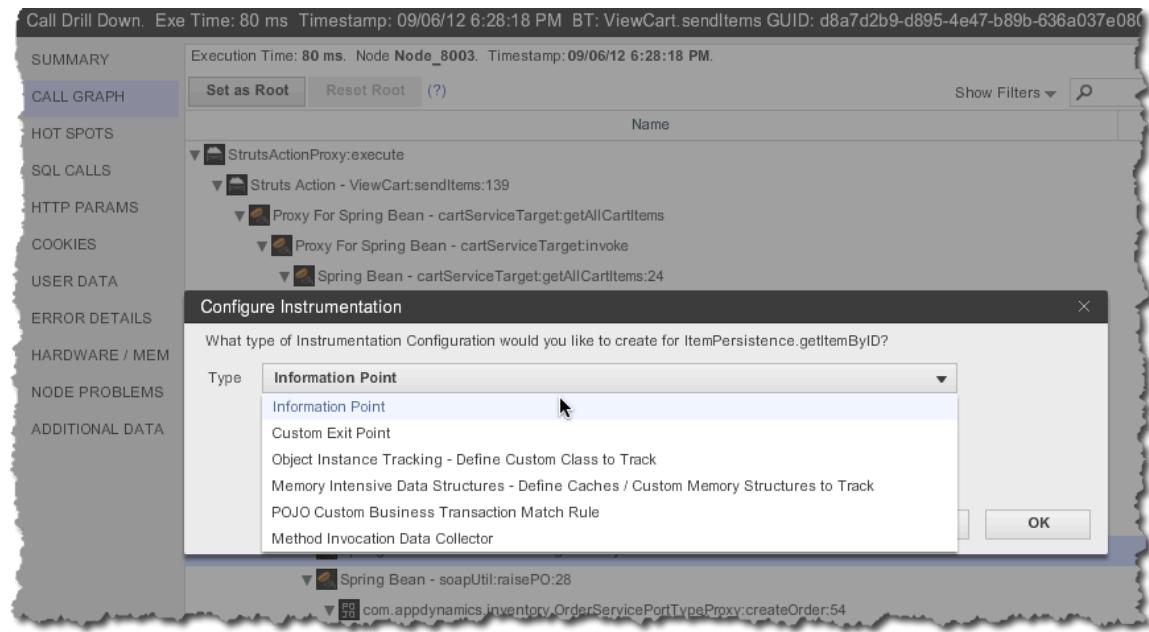
- org.apache.activemq.*
- org.hibernate.transaction.*
- org.apache.struts2.*
- java.lang.*
- org.apache.coyote.*
- org.apache.tomcat.*
- org.apache.catalina.*
- sun.reflect.*
- org.springframework.aop.*
- org.springframework.transaction.*
- org.springframework.orm.*
- org.apache.axis.*
- java.net.*
- net.sf.ehcache.*

(close)

A red arrow points to the message at the bottom of the tooltip: "* Some packages have been excluded".

Configure instrumentation

You can right-click on any item in a call graph and select **Configure Instrumentation for this Class/Method**.



The Configure Instrumentation window presents a drop-down menu form which you can select the type of configuration that you want to create for the method.

Learn More

- Configure Call Graphs
- Configure Code Metric Information Points
- Configure Custom Exit Points (Java)
- Configure Object Instance Tracking (Java)
- Configure Memory Monitoring (Java)
- POJO Entry Points
- Configure Data Collectors
- Tracing Multi-Threaded Transactions (Java)

Configure Call Graphs

- Call Graph Settings
 - To access call graph configuration screens
 - Call Graph Granularity
 - Packages or Namespaces to Exclude from Call Graphs
 - To exclude specific packages or namespaces from call graphs
 - To Include Specific Sub-packages (Sub-namespaces) or Classes from the Excluded Packages
 - SQL Capture Settings
 - To Configure SQL Bind Variables
 - Slow Transaction Snapshot Collection (Java only)
 - To Enable / Disable Aggressive Slow Snapshot Collection
- Learn More

This topic describes how to configure call graphs.

Call Graph Settings

The Call Graph Settings window lets you configure thresholds that affect performance, which packages or namespaces to include in call graphs, and how much detail about SQL statements to capture.

To access call graph configuration screens

1. In the left navigation pane, click **Configure -> Instrumentation**.
2. Click the **Call Graph Settings** tab.
3. Click the **Java Call Graph Settings** or **.NET Call Graph Settings** tab depending on your framework.

Whenever you create or modify a call graph setting in these screens, click the **Save Call Graph Settings** button to save your configuration.

Call Graph Granularity

You can control the granularity for call-graphs using following settings:

- **Control granularity for Methods:** To ensure low performance overhead, choose a threshold in milliseconds for method execution time. Methods taking less than the time specified here will be filtered out of the call graphs.
- **Control granularity for SQL calls:** You can also specify a threshold for SQL queries. SQL queries taking more than the specified time in milliseconds will be filtered out of the call-graphs. Also see [App Agent for Java Performance Tuning](#).

Packages or Namespaces to Exclude from Call Graphs

A call graph can potentially contain hundreds of methods. You can exclude packages (Java) or namespaces (.NET) with classes that you do not want to monitor.

For Java, some packages are excluded by default. These are visible in the Excluded Packages list. The packages that are excluded by default cannot be removed. However, you can include a particular sub-package from an excluded package. See [To Include Specific Sub-packages \(Sub-namespaces\) or Classes from the Excluded Packages](#).

To exclude specific packages or namespaces from call graphs

1. Click **Add Custom Package Exclude** (Java) or **Add Custom Namespace Exclude** (.NET).
2. Enter the name and description of the package or namespace to exclude.
3. Click **Add**.

To Include Specific Sub-packages (Sub-namespaces) or Classes from the Excluded Packages

1. Click **Add Always Show Package/Class** (Java) or **Add Always Show Namespace/Class** (.NET).
2. Specify the subpackage/class or namespace/class and a description to include in the call graph at all times.
3. Click **Add**.

By default, AppDynamics provides support for identifying methods for Jersey REST framework and Apache EJB classes in the call graph.

SQL Capture Settings

Often the SQL Calls section does not display the raw values in a SQL query, as shown in the following query:

```
INSERT INTO ORDERREQUEST ( ITEM_ID, NOTES ) VALUES ( ?, ? )
```

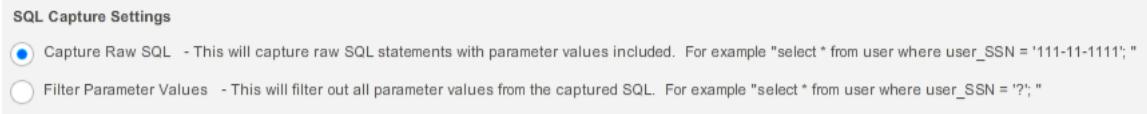
Replacing the literals in a query with parameter markers in this way is called normalizing the query.

Normalizing a query prevents display of sensitive data, such as social security numbers or credit card numbers, which are potential query parameters. Normalizing queries also helps to organize SQL data by flattening the parameter values for the query so that the statistics from different executions of the query can be aggregated and compared against one another.

However, during troubleshooting, you may want to display the values of the bind variables.

To Configure SQL Bind Variables

1. In the **Call Graph Settings** tab, scroll down to the SQL Capture Settings section.
2. Select one of the following:
 - **Capture Raw SQL:** Select this option to see raw SQL data (this captures raw SQL data along with the parameter values). Raw SQL data includes prepared statement bind variables or raw statements. By default, the private SQL data and queries that take less than 10 ms are not captured.
 - **Filter Parameter values:** Select this option to filter certain parameter values from the captured SQL.



3. Click **Save Call Graph Settings**.

Slow Transaction Snapshot Collection (Java only)

Normally AppDynamics captures the full execution path of a request after the business transaction threshold for slow requests has been crossed. Before a request becomes problematic (slow, stalled or error), AppDynamics captures partial call graphs that do not show the invocation stack from before the request started to slow.

You can configure more aggressive snapshot collection to capture the full execution path of requests before thresholds are crossed. This feature is currently supported only for Java frameworks.

By default aggressive snapshot collection is disabled to minimize overhead. You may want to enable it when you are starting to experience performance problems for which you want to find root cause.

This configuration affects snapshot collection at the application level. You can also enable and disable this feature at the node level using the enable-hotspot-snapshots node property. For information about setting node properties see [App Agent Node Properties](#).

To Enable / Disable Aggressive Slow Snapshot Collection

1. Select the application for which you want to enable or disable aggressive snapshot collection.
2. In the left navigation panel click **Configure -> Instrumentation**.
3. Click the **Call Graph Settings** tab, and then the **Java Call Graph Settings** sub-tab.
4. At the bottom of the Call Graph Settings screen, in the Slow Transaction Snapshot Collection section, check the **Enable Aggressive Slow Snapshot Collection** checkbox to enable aggressive collection. Clear the checkbox to disable aggressive collection.
5. Click **Save Call Graph Settings**.

Learn More

- [Call Graphs](#)
- [App Agent for Java Performance Tuning](#)
- [Transaction Snapshots](#)

Diagnostic Sessions

- [Triggering a Diagnostic Session](#)
 - [On-demand Diagnostic Sessions](#)
 - [To Start a Diagnostic Session for a Business Transaction Manually](#)
 - [Automatic Diagnostic Sessions For Slow and Error Transactions](#)
 - [To Configure Diagnostic Session Thresholds](#)
- [Accessing a Diagnostic Session](#)
 - [To View a Diagnostic Session](#)
- [Learn More](#)

This topic explains what a diagnostic session is and how to capture diagnostic session on a transaction.

A diagnostic session captures detailed data about the processing of a transaction as transaction snapshots over a defined period of time. This data includes full call graphs.

Diagnostic sessions can be triggered manually through the user interface or configured to start automatically when thresholds for slow, stalled, or error transactions are reached. If the diagnostic session is triggered manually, the diagnostic session collects snapshots on all the nodes that the selected business transaction passes through. If the diagnostic session is triggered to start automatically, the diagnostics session collects snapshots on the triggering node.

Triggering a Diagnostic Session

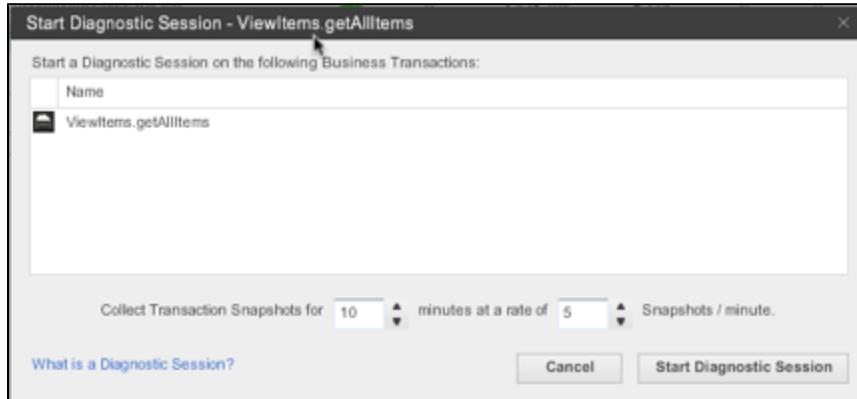
You can manually trigger a diagnostic session when you need one or configure them to start up automatically.

On-demand Diagnostic Sessions

On-demand diagnostic sessions must be started manually.

To Start a Diagnostic Session for a Business Transaction Manually

1. Display the dashboard for the business transaction that you want to analyze. See [Business Transactions List](#).
2. Click **Actions -> Start Diagnostic Session** or right-click on the selected business transaction and then click **Start Diagnostic Session**.
3. Specify the snapshot collection duration at the bottom of the Start Diagnostic Session window.



AppDynamics will start collecting transaction snapshots for that business transaction.

Automatic Diagnostic Sessions For Slow and Error Transactions

AppDynamics provides default thresholds to detect slow, very slow, stalled and error transactions. You can configure settings for triggering diagnostic sessions for these transactions.

To Configure Diagnostic Session Thresholds

1. In the left navigation pane, click **Configure -> Slow Transaction Thresholds**.
2. For configuring thresholds for business transactions click the **User Transaction Thresholds** tab. For thresholds for background tasks, click the **Background Tasks Thresholds** tab.
3. In the thresholds tree list, select the scope of the threshold, either:
 - Default Thresholds
or
 - Individual Transaction Thresholds
4. In the right panel configure thresholds for when diagnostic sessions will be started.

Here you can set a trigger based on the percentage of requests that exceed the Slow Request threshold. For performance reasons you may not want to trigger a diagnostic session each time a threshold is exceeded.

5. Configure diagnostic session duration and collection frequency. This includes:
 - Number of snapshots to collect over a specified time period
 - Number of unsuccessful attempts per minute
 - Wait period between sessions

For performance reasons you want to limit the duration and frequency of diagnostic sessions to the minimum required time to obtain the maximum amount of information for troubleshooting purposes.

When there are ongoing performance problems you do not want a diagnostic session to run continuously. You can set a wait period between sessions and increase the time as needed.

Accessing a Diagnostic Session

You can access a diagnostic session from the transaction snapshot list.

To View a Diagnostic Session

1. Navigate to the transaction snapshot list.
See [To View Transaction Snapshots](#).
2. Select a transaction snapshot from the list and either double-click it or click **View Dashboard**.
3. Click the **Diagnostic Sessions** tab.
4. From the list select a diagnostic sessions and click **View Diagnostic Session**.
From there you can double-click a transaction snapshot to dive deeper.

The screenshot shows a window titled "Diagnostic Session Details". At the top, it displays "Business Transaction: ViewCart.sendItems" and a button "View Dashboard During Diagnostic Session". Below this, it shows "Type: Abnormal Slow Rate" and "Duration: 5 minutes (from 09/19/12 11:09:45 AM to 09/19/12 11:14:45 AM)". A section titled "Snapshots in Session" contains a table with the following data:

	Time	Exe Time (ms)	URL	Tier	Node
	09/19/12 11:12:09 AM	10040	/appdynamicspilot/ViewCart!sendItems.action	ECommerce Server	Node_8003
	09/19/12 11:12:19 AM	10041	/appdynamicspilot/ViewCart!sendItems.action	ECommerce Server	Node_8003
	09/19/12 11:12:54 AM	10038	/appdynamicspilot/ViewCart!sendItems.action	ECommerce Server	Node_8003
	09/19/12 11:13:04 AM	10043	/appdynamicspilot/ViewCart!sendItems.action	ECommerce Server	Node_8003

Learn More

- [Transaction Snapshots](#)
- [Actions](#)
- [Diagnostic Actions](#)

Analyze

AppDynamics provides tools to help you analyze patterns and discover relationships between different metrics and performance data over time. See:

Scalability Analysis

- [Scalability Analysis Comparisons](#)
 - [To Access Scalability Analysis](#)
 - [To Perform Scalability Analysis](#)
- [Learn More](#)

Scalability problems in a distributed environment can cause remote communication overhead. Examples of scalability problems include:

- Increased inter-tier time when tiers are newly separated
- Increased inter-tier time in conjunction with chattiness
- Over-sized payloads with network saturation

Scalability Analysis Comparisons

You can compare:

- Response Time vs Application Load
- CPU Utilization vs Application Load
To compare CPU Utilization vs Application Load, the Machine Agent must be installed on the node machine. See [Install the Machine Agent](#).

CPU Utilization analysis is not available at the business transaction level.

You can perform scalability analysis at the application, business transaction, tier or node level.

The following graph shows Response Time vs Load at the application level for the past three days. The Best fit line is calculated using the Quadratic Least Squares algorithm.



To Access Scalability Analysis

Click **Analyze** -> **Scalability Analysis** in the left navigation pane.

To Perform Scalability Analysis

1. Select the time range to be covered by the analysis from the Time Range dropdown menu.

Note that modifying the time range in this screen does not modify the time range settings in other parts of the UI.

2. Click either the **Response Time vs Load** tab or the **CPU Utilization vs Load** tab, depending on which analysis you want to perform.

3. In the left panel of the scalability viewer either:

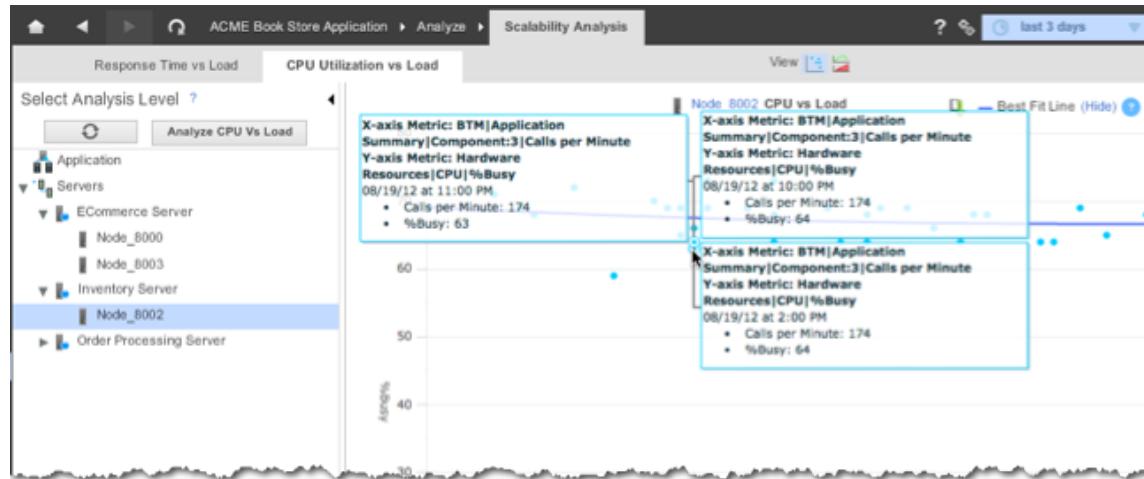
- Select Application for analysis at the application level.

or

- Navigate to the business transaction, tier, or node that you want to analyze.

4. Click **Analyze Response Time Vs Load** or **Analyze CPU vs Load** to update the graph.

You can hover over multiple data points to view their metrics at particular times. The following graph shows CPU Utilization for Node_8002 on the Inventory Server for the last three days with data captured at 2:00 PM, 10:00 PM, 11:00 PM.



Learn More

- Infrastructure Metrics

Correlation Analysis

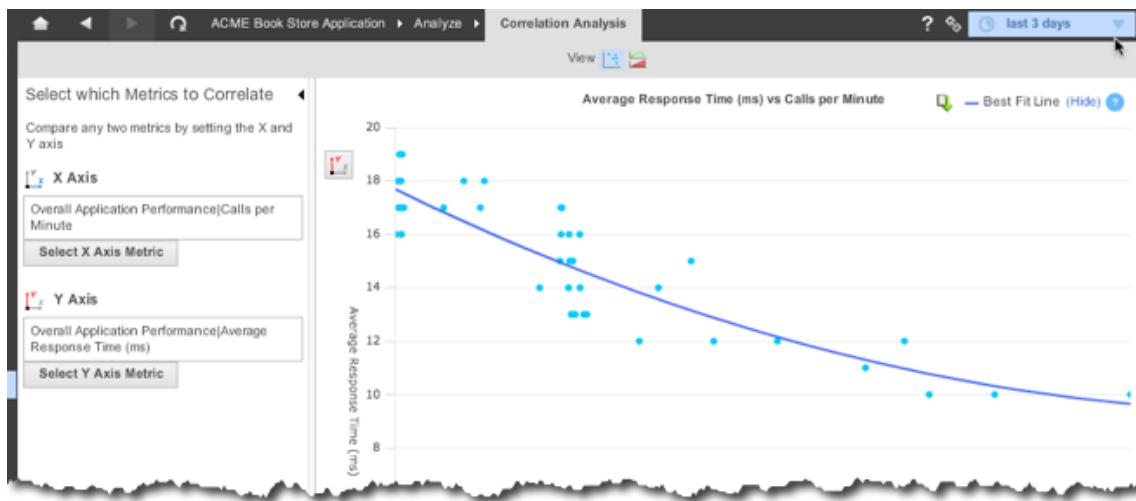
- To perform correlation analysis between two metrics
- Learn More

Correlation analysis lets you compare two metrics on different axes to see how one metric correlates with the other.

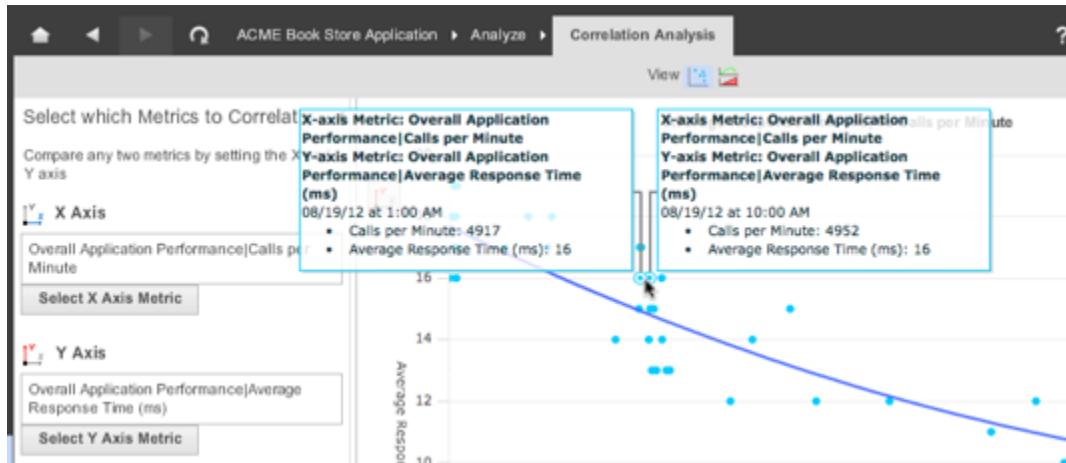
To perform correlation analysis between two metrics

1. In the left navigation pane, click **Analyze** > **Correlation Analysis**.
2. Click **Select X Axis Metric**.
3. From the metric browser navigate to the metric that you want to graph on the X axis and double-click the metric.
4. Click **Select Y Axis Metric**.
5. From the metric browser navigate to the metric that you want to graph on the Y axis and double-click the metric.

The graph is displayed when metrics for both the axes are selected.



Hover over one or more of the data points to view their metrics at particular times.



[Learn More](#)

- Metric Browser

Compare Releases

- Entities to Compare
- Metrics to Compare
 - To Compare Releases
- Learn More

You can compare metrics for two different time periods on a split screen.

This feature is particularly useful for comparing different versions of a release.

Entities to Compare

You can compare summary metrics:

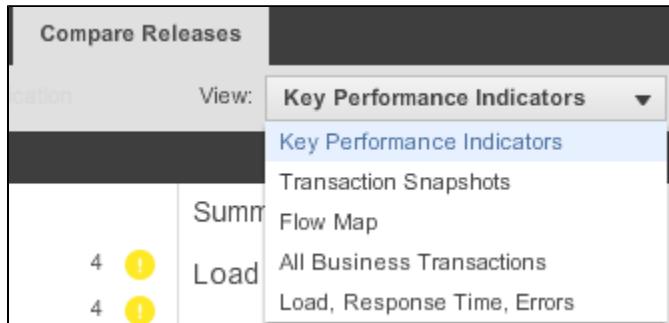
- for the entire application
- for a specific business transaction
- for a specific tier
- for a specific node

Metrics to Compare

The metrics you can compare are:

- Summary Key Performance Indicators (KPIs)
Displays the KPI summaries from the dashboards for the selected entity (application/business transaction/tier/node).
- Transaction Snapshots
Displays the transaction snapshots for the selected entity.
- Graphical Flow
Displays flow graphs for the selected entity.
- KPI Trend Graphs
Displays the KPI summaries from the dashboards for the selected entity (application/business transaction/tier/node).

Select which sets of metrics you want to compare from the View drop-down menu.



To Compare Releases

1. In the left navigation pane, click **Analyze -> Compare Releases**.
2. Set the time ranges to compare from the Time Range drop-down menus in both panes.
3. Select the entities to compare in the Select What to Compare panel.
4. Select the metrics that you want to compare from the View drop-down menu.
5. Click **Compare**.

Whenever you change the entities or the metrics, click **Compare** to refresh the display.

Learn More

- Dashboards
- Transaction Snapshots
- Time Ranges

Information Points

AppDynamics provides information points as a way to monitor:

- **Code metrics:** how a method is performing across the application, from all business transactions
- **Business metrics:** data from a method's parameters or return values, across the application

Information points gather data outside of the context of a business transaction. In this way they are distinct from data collectors. See [Data Collectors Versus Information Points](#).

Business Metrics

- Accessing Business Metrics
- Business Metrics in Call Graphs
- Configuring Business Metrics Using Information Points
- Business Metrics and Health Rules
- Learn More

Business metrics capture data from a method's parameters or return values to report on the performance of the business.

For example:

- What is the average value of the credit card total?
- How many credit cards did my application process in a certain time period, regardless of the business transaction?
- What was the average time spent processing a credit card transaction?
- What is the current rate of rejected credit cards?
- Which items were best-sellers in a certain time period?

AppDynamics gathers business metrics using information points. Information points instrument methods in your application code outside the context of a particular business transaction and extract data from code. When the method configured for the information point is invoked, the information point metric is captured.

Accessing Business Metrics

To access the Information Points List, in the left navigation pane click **Analyze -> Information Points**.

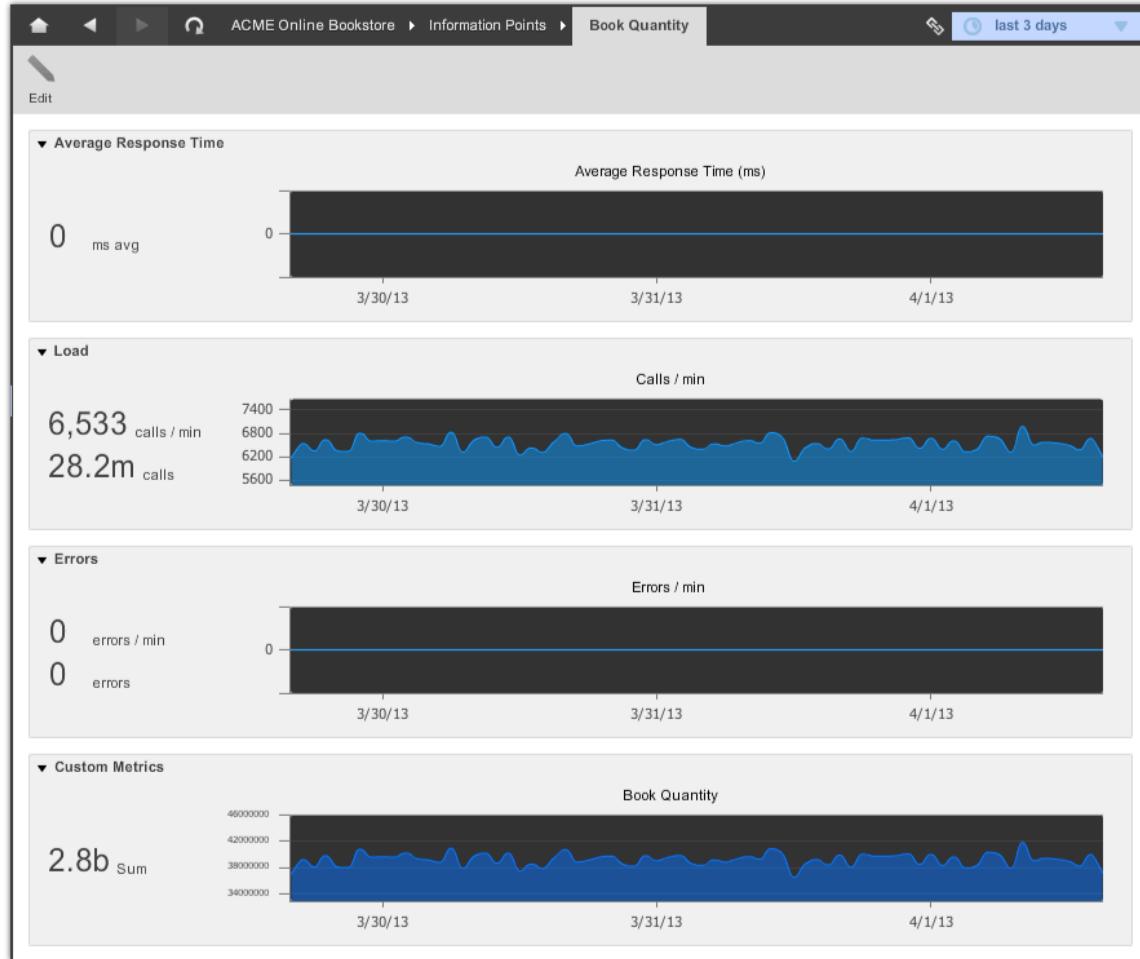
The Information Point List summarizes metrics for all the information points in an application. Business metrics show a value in the # of Custom Metrics column.

Name	Response Time ...	Calls	Calls / min	Errors	# of Custom Metrics
Book Quantity	0	100,186	6,679	11	1
Calls to delete cart items	0	42,411	2,827	11	0
Items in cart checked out	245	20,038	1,336	11	1

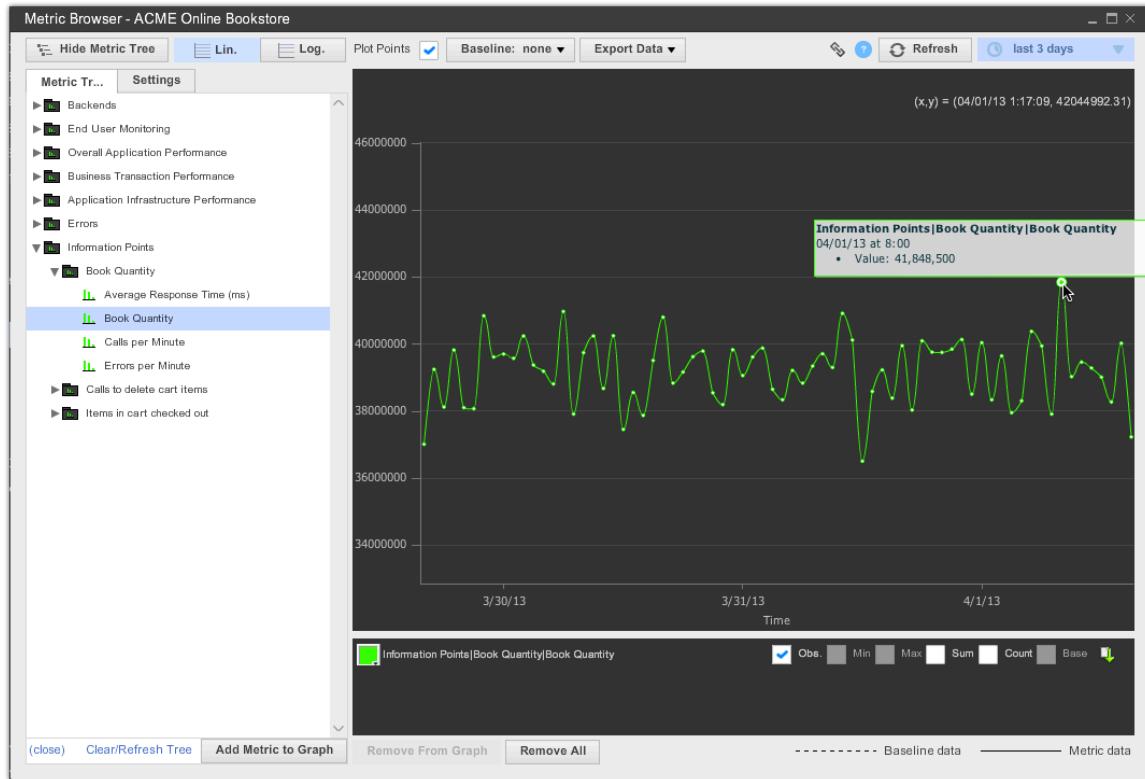
From the Information Points List you can:

- filter the list
- select an information point
- view one of its metrics in the Metrics Browser
- view a graph all its metrics on a dashboard

Each information point has its own dashboard, which reports KPIs (average response time, load, and errors) for the information point, as well as any custom business metrics defined for the information point.



Business metrics appear in the Information Points tree of the Metric Browser.



Business metrics can be accessed from the AppDynamics REST API, just like any other metric. See [Use the AppDynamics REST API](#)#To Copy the REST URL for a Metric.

Business Metrics in Call Graphs

You can configure snapshots to display information point invocations in call graphs by setting the enable-info-point-data-in-snapshots node property to true. By default the property is false. See [App Agent Node Properties](#).

When the enable-info-point-data-in-snapshots node property is set, information point calls appear in the User Data section of the call graph.

Configuring Business Metrics Using Information Points

You define an information point based on the class and method signature of the method being monitored. See [Configure Business Metric Information Points](#).

Business Metrics and Health Rules

You can use any parameter or return value of the method to generate a custom business metric across multiple business transactions. You can then create a custom health rule based on the performance of such a custom metric. See [Health Rules](#).

Learn More

- [Code Metrics](#)
- [Configure Code Metric Information Points](#)
- [Call Graphs](#)
- [Data Collectors Versus Information Points](#)

Configure Business Metric Information Points

- [To create a business metric information point](#)
- [Learn More](#)

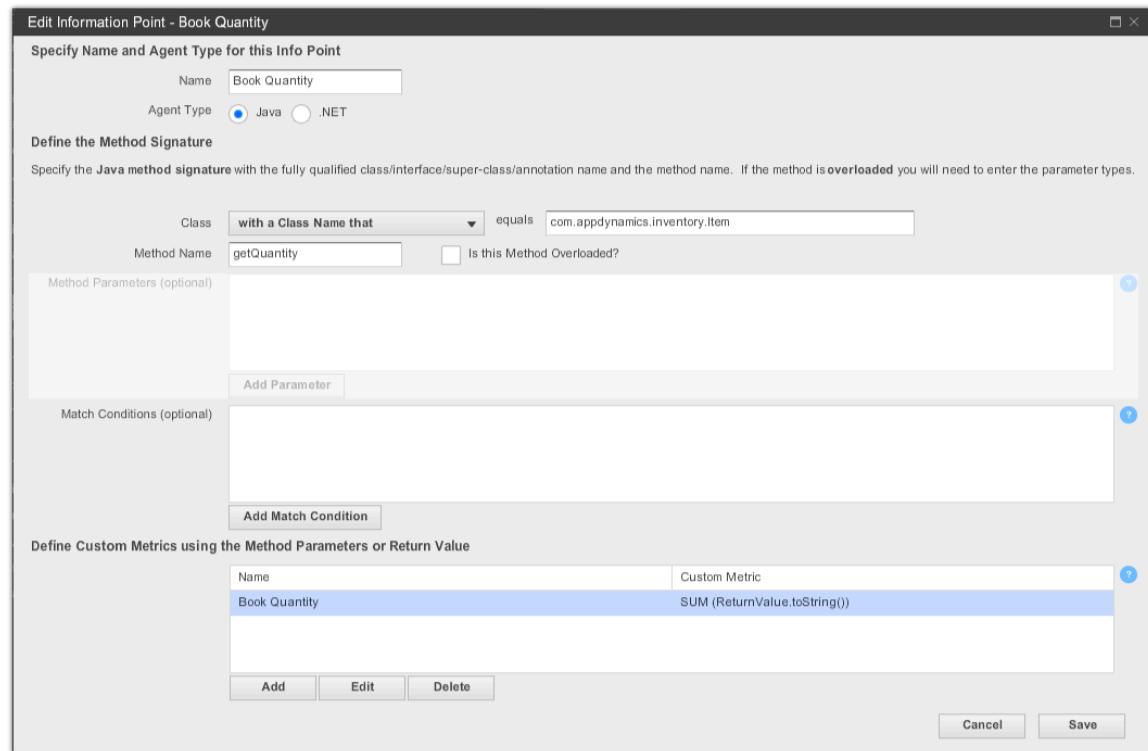
AppDynamics provides information points to specify code and business metrics. See [Code Metrics](#) and [Business Metrics](#).

To create a business metric information point

Configuring business metrics, also called custom metrics, involves:

- Identify the parameter or the return type of a method invocation.
- Derive a value from the method invocation to represent the metric.
- Choose either to generate the sum of the value over a period of time or to average it.

1. Select your application.
2. Click **Analyze -> Information Points**.
3. Click **Add** (the + symbol).
4. Enter a name for the information point.
5. Select the agent type.
6. Define the class and the method on which the information point is based.



7. (Optional) If the method is overloaded, check the **Is this Method Overloaded** option and add the parameters.
8. (Optional) Define match conditions. A match condition ensures that only those invocations that match the configured conditions for the Information Point will be captured.
9. In the Define Custom Metrics section, click **Add**. Choose the parameter or return value from which the metric needs to be generated.
10. Choose the data rollup over time.
11. Click **Save**.
12. **⚠️ Important:** Restart the application server on which this rule will be applied.

For example, this definition counts the sum of books ordered in the Acme Bookstore application.

Edit Information Point - Book Quantity

Specify Name and Agent Type for this Info Point

Name: Book Quantity
Agent Type: Java (.NET)

Define the Method Signature

Specify the Java method signature with the fully qualified class/interface/super-class/annotation name and the method name. If the method is overloaded you will need to enter the parameter types.

Class: with a Class Name that equals com.appdynamics.inventory.item
Method Name: getQuantity Is this Method Overloaded?

Method Parameters (optional)

Match Conditions (optional)

Define Custom Metrics using the Method Parameters or Return Value

Name	Custom Metric
Book Quantity	SUM (ReturnValue.toString())

Information Point Custom Metric Definition

Custom business metrics can be generated from the Java method invocation using the values of the parameters/return values.

To configure custom metrics, pick the parameter index or return value and an optional getter operation on it if it is a complex object.

Name: Book Quantity

Data To Collect

Collect Data From: Method Parameter @ Index: 0 Return Value Invoked Object

Operation on Return Value: Use `toString()` Use Getter Chain
 for example: `getAccount().getBalance()`
[more help](#)

Data Rollup

The result of the data collected from the method invocation has to be an integer value which will either be averaged or added per minute, depending on the choice below.

Data Rollup: SUM (Rollup type cannot be changed once created)

Learn More

- Business Metrics
- Information Points

Code Metrics

- Accessing Code Metrics
- Code Metrics in Call Graphs
- Configuring Code Metrics Using Information Points
- Code Metrics and Health Rules
- Learn More

Code metrics capture how a method is performing. For example:

- How many times was the method executed?
- How long did it take to execute on average?
- How many concurrent users are using my application? (extrapolated by comparing information points on the login and logout methods)

AppDynamics gathers code metrics using information points. Information points instrument methods in your application code outside the context of a particular business transaction. Use them when you need to track data from the same method across multiple business transactions.

Accessing Code Metrics

To access the Information Points List, in the left navigation pane click **Analyze -> Information Points**.

The Information Point List summarizes metrics for all the information points in an application.

Code metrics appear in the Information Points tree of the Metric Browser and can be accessed from the AppDynamics REST API, just like any other metric. See [Use the AppDynamics REST API#To Copy the REST URL for a Metric](#).

From the Information Points List you can:

- filter the list
- select an information point
- view one of its metrics in the Metrics Browser
- view a graph all its metrics on a dashboard

Code Metrics in Call Graphs

You can configure snapshots to display information point invocations in call graphs by setting the enable-info-point-data-in-snapshots node property to true. By default the property is false. See [App Agent Node Properties](#).

When the enable-info-point-data-in-snapshots node property is set, information point calls appear in the User Data section of the call graph.

Configuring Code Metrics Using Information Points

You define an information point based on the class and method signature of the method being monitored. See [Configure Code Metric Information Points](#).

Code Metrics and Health Rules

You can then create a custom health rule based on the performance of such a custom metric. See [Health Rules](#).

Learn More

- Business Metrics
- Configure Code Metric Information Points
- Data Collectors Versus Information Points

Configure Code Metric Information Points

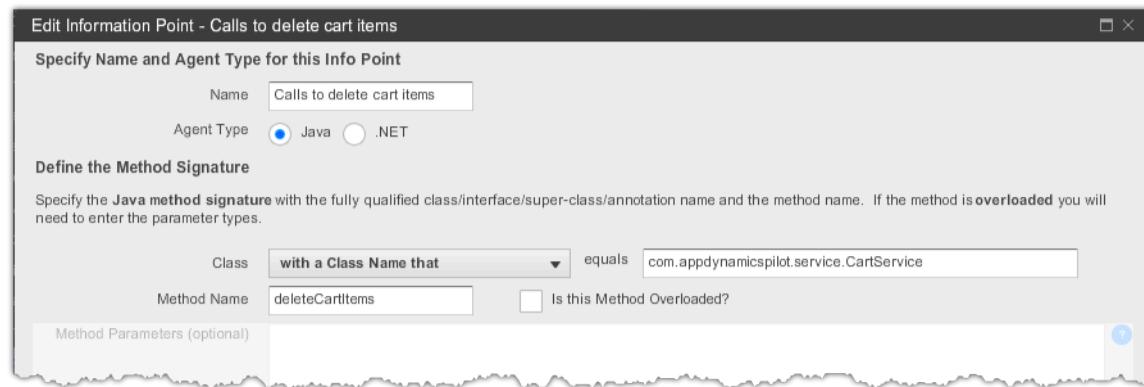
- [To create a code metric information point](#)
- [Learn More](#)

AppDynamics provides information points to specify code and business metrics. See [Code Metrics and Business Metrics](#).

To create a code metric information point

1. Select your application.
2. Click **Analyze -> Information Points**.
3. Click **Add** (the + symbol).
4. Enter a name for the information point.
5. Select the agent type.
6. Define the class and the method on which the information point is based.
7. (Optional) If the method is overloaded, check the **Is this Method Overloaded** option and add the parameters.
8. (Optional) Define match conditions. A match condition ensures that only those invocations that match the configured conditions for the Information Point will be captured. For example, the ACME Bank AccountManager.updateAccount(String customerType, Account newAccountData) method, you can capture only those requests that invoke the parameter "customerType" with value "platinum".

For example, the following configuration counts the number of calls to the method that deletes items from the cart:



Learn More

- [Code Metrics](#)
- [Information Points](#)

System Integrations

See these system integration topics:

Extensions

- Integrate with PagerDuty
 - [Download the PagerDuty extension](#)
- Integrate with Boundary
 - [Blog: AppDynamics and Boundary: The Full-Circle View of Application Health](#)
 - [Download the Boundary extension](#)

For additional extensions see [AppSphere Extensions](#).

Integration Basics

- Add Data to AppDynamics
 - Add Metrics with Custom Monitors
 - Advantages of Adding Metrics
 - Scripts for Custom Monitors
 - Java Monitors (Java only)
 - Performance Counters (.NET only)
 - HTTP Monitors
 - Add Events
- Retrieve Data from AppDynamics
- Integrate AppDynamics Alerting with Third Party Notification Systems
 - Custom Notifications
 - Retrieve Events and Health Rule Violations
- Sample Integrations
- Learn More

AppDynamics is designed to interface with other systems in your organization.

You can add data to AppDynamics, retrieve data from AppDynamics, and integrate alerting of AppDynamics events and health rule violations into your organization's alerting system.

The AppDynamics REST API provides a simple way to accomplish many types of integration. See [Use the AppDynamics REST API](#) for a complete reference.

Add Data to AppDynamics

You can add metrics and events from external systems to AppDynamics.

This allows you to monitor all your application performance data using a single tool. External metrics and events are treated just like standard AppDynamics metrics and events. See [Advantages of Adding Metrics](#).

Add Metrics with Custom Monitors

You can write custom monitors to add metrics to the set that AppDynamics already collects and reports to the Controller. These can include metrics that you collect from other monitoring systems. They can also include metrics that your system extracts from services that are not instrumented by AppDynamics, such as databases, LDAP servers, web servers, C programs, etc.

Advantages of Adding Metrics

Advantages of adding external metrics to AppDynamics include:

- automatic baselines and anomaly detection for external metrics
- availability for display on custom dashboards
- availability for use in policies
- visibility of all metrics in the Metric Browser, where you can display external metrics along with AppDynamics metrics on the same graph

Scripts for Custom Monitors

You can write a shell script (LINUX) or batch file (Windows) to report custom metrics every minute to the machine agent. The machine agent passes these metrics on to the Controller.

See [Add Metrics Using Custom Monitors](#) for information on how integrate a custom monitor into AppDynamics. Download [HardwareMonitor.zip](#) for examples of scripts that implement custom monitors.

Download GroovyAgent.zip for an example of a monitor built with Groovy.

Java Monitors (Java only)

Your custom metrics may be too complicated to collect using a script. For example, you may need to perform complex calculations or call a third party API to get the metrics.

In this case, you can extend the JavaServersMonitor class to collect the metrics and report them to the machine agent. Your Java program extends the JavaServersMonitor class to provide your custom functionality.

The Monitoring section on the AppDynamics Center of Excellence (ACE) website has some sample Java Monitors. See <http://docs.appdynamics.com/display/ACE/Monitoring>.

Performance Counters (.NET only)

You can add additional performance counters to the standard hardware metrics reported by the embedded .NET machine agent by configuring the <perf-counters> element in the machine agent's application.config file.

See [Windows Performance Counters](#).

HTTP Monitors

You can post HTTP requests to the machine agent to send it custom metrics every minute. This is done by starting the machine agent with a jetty HTTP listener.

See [Machine Agent HTTP Listener](#) for information on starting the HTTP listener and sending it metrics.

Add Events

You can create custom events using POST requests in the REST API. These can be events of type APPLICATION_DEPLOYMENT or of type CUSTOM.

You can create application deployment events, in addition to the ones that AppDynamics provides, to notify AppDynamics when you upgrade your application, push new code, etc. This lets you correlate these application deployment activities with other data inside AppDynamics. This is useful for regression analysis, root cause analysis, and performance studies. A useful practice is to include injection of your application deployment event into AppDynamics as part of the build process for deploying a new version of your application.

You can create custom events to be reported in the AppDynamics event viewer and in the events panels on the AppDynamics dashboards. Then you can create alerts triggered by these events as you do for AppDynamics standard events.

See [Create Events](#).

Retrieve Data from AppDynamics

You can retrieve two kinds of data using the AppDynamics REST API:

- Metadata: Metadata describes your managed application architecture: your applications, your tiers, your nodes, your business transactions, which tiers are in which application, which nodes are in which tier, etc.
- Real Data: This includes actual performance data: metrics, events, transaction snapshots, health rule violations, etc. You can retrieve any metric available in the AppDynamics Metric Browser using REST API. See [Retrieve metrics](#).

After you have retrieved data from AppDynamics programmatically, you can use it anywhere in organization, for example:

- integrating it into your alerting system
- adding it (perhaps as a graph) to your organization's unified monitoring portal or to a phone application
- pushing it to a corporate warehouse with other corporate data
- passing it to tool that analyzes systems in your organization

Integrate AppDynamics Alerting with Third Party Notification Systems

You can integrate notification of AppDynamics health rule violations and events with your organization's alerting or ticketing system.

Use a push approach by creating custom notifications that pass the information to your alerting system. See [Custom Notifications](#).

Or use a pull approach by using the REST API to extract events and then pass the data, including the AppDynamics deep link, to your alerting system.

Custom Notifications

A custom notification lets you integrate alerts about AppDynamics health rule violations and events into your own alerting system. This integration requires:

- A custom.xml file that provides information about the custom notification
- An executable script (called a custom action) that accepts parameters from AppDynamics about the events and health rule violations that trigger an alert
- Configuring events or policies to trigger the custom notification

See [Integrate using Custom Action Scripts](#) for details about creating the xml file and the script. See [Alert and Respond](#) for information about configuring events and health rule violations to trigger the custom notifications.

Retrieve Events and Health Rule Violations

See [Retrieve event data](#) and [Retrieve all policy violations in a particular business application](#) for information about retrieving events and policy violations. The responses contain all the information about the violation or event, including a deep link to the details, which your program can pass to your alerting system.

Sample Integrations

Name	Size	Creator	Creation Date	Comment
ZIP Archive HardwareMonitor.zip	59 kB	Admin	Jan 24, 2013 18:28	
ZIP Archive GroovyAgent.zip	5.23 MB	Admin	Jan 24, 2013 18:28	

Learn More

- Use the AppDynamics REST API
- Behavior Learning and Anomaly Detection
- Events
- Add Metrics Using Custom Monitors
- Configure the .NET Machine Agent
- Machine Agent HTTP Listener

Integrate using Custom Action Scripts

- Custom Action Procedure
- Contents of the custom.xml File
- Information Passed to the Custom Action Script from AppDynamics
 - Parameters passed by a health rule violation
 - Parameters passed for an event that is not a health rule violation event
- [Learn More](#)

This topic describes how to create a custom action for integrating with third party systems.

After creating the custom action script and the custom.xml file you can then create a custom action and invoke it in a policy.

Custom Action Procedure

The procedure for creating and installing a custom action is:

1. At the top level of the Controller installation directory, create a directory named "custom" with a sub-directory named "actions".

```
<controller_install_dir>/custom/actions
```

2. In the <controller_install_dir>/custom/actions directory, create a subdirectory for each custom action script that you will install. For example,

```
<controller_install_dir>/custom/actions/jira
```

for an action that interfaces with a Jira system.

3. For each custom action that you want to implement, create an executable script (.bat extension for Windows, .sh extension for Linux) that can accept and process the parameters passed to it by AppDynamics. See [Information Passed to the Custom Action Script from AppDynamics](#) for details on the parameters.

Create this script in the appropriate subdirectory that you created in step 2.

Set correct executable permissions for the shell scripts in a Linux environment. For example: chmod 770 script1.sh.

Ensure that the script file has correct character encoding. This is especially important when creating a Unix shell script on a Windows machine.

4. In the <controller_install_dir>/custom/actions directory, create a custom.xml file that describes the location and name of your custom action script(s).

See [Contents of the custom.xml File](#).

4. Verify the script manually.

To verify the script:

- a. Open a command-line console on the Controller host machine.
- b. Execute the script file from the command line console.

5. Create the custom action in the AppDynamics UI. See [To create a custom action](#).

Contents of the custom.xml File

The custom.xml file has an <actions> element for every custom action on the controller.

The <type> element contains the subdirectory that contains the script file.

The <executable> element contains the name of the script.

Sample custom.xml file

```
<custom-actions>
  <action>
    <type><jira></type>
    <executable><script1.bat></executable>
  </action>
  <action>
    <type><bugzilla></type>
    <executable><script2.sh></executable>
  </action>
</custom-actions>
```

Information Passed to the Custom Action Script from AppDynamics

The custom action script must handle the parameters passed to it from a health rule violation event or any other type of event.

Parameters passed by a health rule violation

For a health rule violation, the custom action script is invoked with the following string parameters:

Health Rule Violation Parameter	Repeated Parameter	Notes
APP_NAME	none	name of the business application
APP_ID	none	application ID number
PVN_ALERT_TIME	none	alert time, such as: Thu 10:00 AM
PRIORITY	none	priority number
SEVERITY	none	allowed values: INFO, WARNING, CRITICAL are called "Info", "Warn", "Crit"
TAG	none	or the empty string if no tags are present

HEALTH_RULE_NAME	none	name of the health rule
HEALTH_RULE_ID	none	health rule ID number
PVN_TIME_PERIOD_IN_MINUTES	none	health rule violation time period in minutes
AFFECTED_ENTITY_TYPE	none	allowed types: APPLICATION, APPLICATION_COMPONENT, APPLICATION_DIAGNOSTIC
AFFECTED_ENTITY_NAME	none	the affected entity name
AFFECTED_ENTITY_ID	none	the affected entity id
NUMBER_OF_EVALUATION_ENTITIES	none	number of entities (BT, application) violating health rule condition
EVALUATION_ENTITY_TYPE	Yes, one for each evaluation entity	allowed types: APPLICATION, APPLICATION_COMPONENT, APPLICATION_DIAGNOSTIC
EVALUATION_ENTITY_NAME	Yes, one for each evaluation entity	the evaluation entity name
EVALUATION_ENTITY_ID	Yes, one for each evaluation entity	the evaluation entity id
NUMBER_OF_TRIGGERED_CONDITIONS_PER_EVALUATION_ENTITY	Yes, one for each evaluation entity	number of times to loop through each evaluation entity
triggered condition	Yes, listed below	if more than one condition is triggered, which triggered condition, when
	SCOPE_TYPE_x	the scope of the parameter node: APPLICATION, APPLICATION_COMPONENT
	SCOPE_NAME_x	the name of the scope,
	SCOPE_ID_x	the scope id
	CONDITION_NAME_x	the health rule condition name
	CONDITION_ID_x	the health rule condition id
	OPERATOR_x	allowed operators: LESS_THAN, GREATER_THAN, GREATEST_THAN_OR_EQUAL, NOT_EQUALS.
	CONDITION_UNIT_TYPE_x	the condition for the threshold: BASELINE_STANDARD, BASELINE_PERCENTAGE
	USE_DEFAULT_BASELINE_x	a Boolean parameter (true) if the unit type is one of the baseline types
	BASELINE_NAME_x	applicable only when the condition unit type and the use default are true
	BASELINE_ID_x	applicable only when the condition unit type and the use default are true
	THRESHOLD_VALUE_x	health rule threshold setting
	OBSERVED_VALUE_x	value that violated the health rule condition
SUMMARY_MESSAGE	none	summary of the notification
INCIDENT_ID	none	the incident identifier number

DEEP_LINK_URL	none	controller deep link URI <a href="http://<controller-host-u">http://<controller-host-u
		Append the incident ID for this policy violation

Parameters passed for an event that is not a health rule violation event

For an event that is not a health rule violation event, the custom action script is invoked with the following string parameters:

Event Notification Parameter	Repeated Parameter	Notes
APP_NAME	none	name of the business application
APP_ID	none	application ID number
EN_TIME	none	event notification time, for example: Wed Jan 04 09:36:56
PRIORITY	none	priority number
SEVERITY	none	Allowed values: INFO, WARN, or ERROR (In the AppDynamics API "Info", "Warning", and "Critical")
TAG	none	or <NULL> if it was not specified by the user
EN_NAME	none	name of the event notification
EN_ID	none	event notification ID number
EN_INTERVAL_IN_MINUTES	none	event notification interval in minutes
NUMBER_OF_EVENT_TYPES	none	determines how many times to loop through the event types
event type	yes, listed below	if there is more than one event type, the parameters repeat "x" increments the number representing the event type
	EVENT_TYPE_x	type of event, such as: ERROR, APPLICATION_ERROR, STALL, BT_SLA_VIOLATION, DEADLOCK, MEMORY_LEAK, MEMORY_LEAK_DIAGNOSTICS, LOW_HEAP_MEMORY, APP_SERVER_RESTART, BT_SLOW, SYSTEM_LOG, INFO_INSTRUMENTATION_VISIBILITY, AGENT_EVENT, AGENT_STATUS, SERIES_SLOW, SERIES_ERROR, A OBJECT_CONTENT_SUMMARY, DIAGNOSTIC_SESSION, HIGH_END_TO_END_LATENCY, APPLICATION_CONFIG, APPLICATION_DEPLOYMENT, AGENT_DIAGNOSTICS
	EVENT_TYPE_NUM_x	number of events of this type
NUMBER_OF_EVENT_SUMMARIES	none	number of event summaries in the notification; determine through the event summary parameters
event summary	yes, listed below	if there is more than one event summary, the following parameters repeat "x" increments the number representing the event summary, where "x" increments the number representing the event type
	EVENT_SUMMARY_ID_x	event summary ID number
	EVENT_SUMMARY_TIME_x	event summary time, for example: Wed Jan 04 09:34:13
	EVENT_SUMMARY_TYPE_x	type of event, such as: APPLICATION_CONFIG_CHANGE, APP_SERVER_RESTART, DIAGNOSTIC_SESSION, STATUS
	EVENT_SUMMARY_SEVERITY_x	event severity, such as: INFO, WARN, or ERROR (In the AppDynamics API "Info", "Warning", and "Critical")
	EVENT_SUMMARY_STRING_x	event summary string, such as: Application Server environment
DEEP_LINK_URL	none	<a href="http://<controller-host-url>/#location=APP_EVENT_VIEW">http://<controller-host-url>/#location=APP_EVENT_VIEW Append each event summary ID to the URL to provide a deep link to the event

The current custom action implementation will most likely pass more than nine parameters to the executing script. In Windows and Linux environments, more than nine command line parameters can be processed only if the SHIFT command is used.

For more information, see [Command Line parameters](#).

Learn More

- Actions
- Custom Actions
- Policies
- System Integrations
- Configure Policies

Use the AppDynamics REST API

- Introduction
- Authentication
- Retrieve all business applications
 - Example - Retrieve list of all business applications for the ACME Book Store.
- Retrieve all Business Transactions in a particular business application
 - Example - Retrieve list of all business transactions for the ACME Book Store.
- Retrieve all tiers in a business application
 - Example - Retrieve the list of all tiers for the ACME Book Store.
- Retrieve machine, agent, and IP information about all nodes in a business application
 - Example - Retrieve machine, agent and IP information about the nodes in application 3
- Retrieve machine, agent, and IP information about a node by node name
 - Example - Retrieve information about Node_8001 in application 3.
- Retrieve machine, agent, and IP information about all the nodes in a tier
 - Example - Retrieve information about the nodes in the E-Commerce tier in the ACME Online Book Store.
- Retrieve information about a tier, including the tier ID and number of nodes, in a tier by tier name
 - Example - Retrieve information about the ECommerce tier in the ACME Online Book Store.
- Retrieve metrics
 - To Copy the REST URL for a Metric
 - To Copy the Metric-Path Parameter
 - Structure of Returned Metrics
 - Retrieve metric hierarchy
 - Example - Retrieve the metric hierarchy for the ACME Book Store.
- Use Wild cards in Metric-Path Parameter
 - Example - Retrieve the average response time for all the tiers in the application using the wild card character for the tier name.
 - Example - Retrieve the CPU %Busy metric for all the nodes in the ECommerce tier using the wild card character for the node name.
 - Example - Retrieve the CPU %Busy metric for all the nodes in all the tiers using wild card characters for the tier and node names.
 - Example - Retrieve the Calls per Minute metric for all the business transactions on the ECommerce tier using the wild card character for the business transaction name
- Retrieve metrics for a time range
 - Example - Retrieve the average response time for the past 15 minutes on the ECommerce server.
 - Example - Retrieve the multiple metrics, for the past 15 minutes, for the ViewCart.sendItems transaction on the ECommerce server.
 - Example - Retrieve snapshots for the 15 minutes after Mon, 13 Aug 2012 08:20:41 for the ACME Book Store Application.
 - Example - Retrieve snapshots for the 15 minutes before Mon, 13 Aug 2012 08:20:41 for the ACME Book Store Application.
 - Example - Retrieve snapshots for the time range between Mon, 13 Aug 2012 08:20:41 and Mon, 13 Aug 2012 08:22:21 GMT for the ACME Book Store Application.
- Retrieve all health rule violations in a particular business application
- Retrieve all policy violations in a particular business application
 - Example - Retrieve the list of all policy violations in the ACME Book Store for the past hour.
- Retrieve event data
 - Example - Retrieve the list of events of type "APPLICATION_ERROR" or "DIAGNOSTIC_SESSION" of any severity that occurred in the specified time range.
- Create Events
 - Application Deployment Event Integration
 - Create a Custom Event
- Retrieve transaction snapshots for a Business Transaction for a time range
 - Example - Retrieve list of transaction snapshots for the ACME Book Store.
 - Example - Retrieve list of transaction snapshots filtered on the value of a data collector.
- Create and modify AppDynamics users

- Include or exclude a business transaction from monitoring
- Retrieve all controller global configuration values
 - Example - Retrieve list of all global configuration values
- Retrieve a single controller global configuration value
 - Example - Retrieve the global metrics buffer size.
- Configure Global Controller Settings
- Mark Nodes as Historical
- Retrieve all policy violations in a particular business application
 - Example - Retrieve the list of all open policy violations in the ACME Book Store for the past 1000 minutes.

Introduction

You use the REST API to retrieve information from AppDynamics programmatically.

The AppDynamics REST API is implemented using Representational State Transfer (REST) Services. The data can be returned in either the JavaScript Object Notation (JSON) or the eXtensible Markup Language (XML) format. The default output format is XML.

The URIs in the Monitor APIs are a set of REST services that open access to Monitor data collected by AppDynamics. Each URI can be found by accessing:

```
http://<Controller_Host>:<Controller_Port>/controller/rest/<REST_URI>
```

You can find more general information at REST on [Wikipedia](#) and in particular RESTful web APIs.

Authentication

To invoke the REST APIs, provide basic HTTP authentication credentials as well as your account information.

If you have installed the Controller on a single-tenant platform, your default account is:

Username: username@customer1
Password: xxxx

If you are using the SaaS Controller, your username is your AppDynamics user name in your AppDynamics account:

<username>@<appdynamicsaccountname>

Retrieve all business applications

URI: /applications

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve list of all business applications for the ACME Book Store.

URI	Sample object output
/controller/rest/applications	XML
/controller/rest/applications?output=JSON	JSON

Retrieve all Business Transactions in a particular business application

URI: /applications/<application-name | application-id>/business-transactions

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
exclude	Query	If false, the query retrieves only the business transactions that are included for monitoring. If true, the query retrieves only the excluded business transactions. Excluded business transactions are those that have been configured to be excluded from monitoring either from the UI or through the REST interface. The default is false.	No
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve list of all business transactions for the ACME Book Store.

URI	Sample object output
/controller/rest/applications/ACME Book Store Application/business-transactions	XML
/controller/rest/applications/ACME Book Store Application/business-transactions?output=JSON	JSON

Retrieve all tiers in a business application

URI: /controller/rest/applications/<application-name|application-id>/tiers

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve the list of all tiers for the ACME Book Store.

URI	Sample object output
/controller/rest/applications/ACME Book Store Application/tiers	XML
/controller/rest/applications/ACME Book Store Application/tiers?output=JSON	JSON

Retrieve machine, agent, and IP information about all nodes in a business application

URI: /controller/rest/applications/<application-name|application-id>/nodes

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve machine, agent and IP information about the nodes in application 3

URI	Sample object output
/controller/rest/applications/3/nodes	XML

Retrieve machine, agent, and IP information about a node by node name

URI: /controller/rest/applications/<application-name | application-id>/nodes/<node-name>

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
<node-name>	URI	Provide the node name	Yes
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve information about Node_8001 in application 3.

URI	Sample object output
/controller/rest/applications/3/nodes/Node_8001	XML

Retrieve machine, agent, and IP information about all the nodes in a tier

URI: /controller/rest/applications/<application-name | application-id>/tiers/<tier-name>/nodes

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
<tier-name>	URI	Provide the tier name.	Yes
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve information about the nodes in the E-Commerce tier in the ACME Online Book Store.

URI	Sample object output
/controller/rest/applications/ACME Online Book Store/tiers/E-Commerce/nodes/	XML

Retrieve information about a tier, including the tier ID and number of nodes, in a tier by tier name

URI: /controller/rest/applications/<application-name | application-id>/tiers/<tier-name>

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
<tier-name>	URI	Provide the tier name.	Yes
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve information about the ECommerce tier in the ACME Online Book Store.

URI	Sample object output
/controller/rest/applications/ACME Online Book Store/tiers/ECommerce	XML
/controller/rest/applications/ACME Online Book Store/tiers/ECommerce?output=JSON	JSON

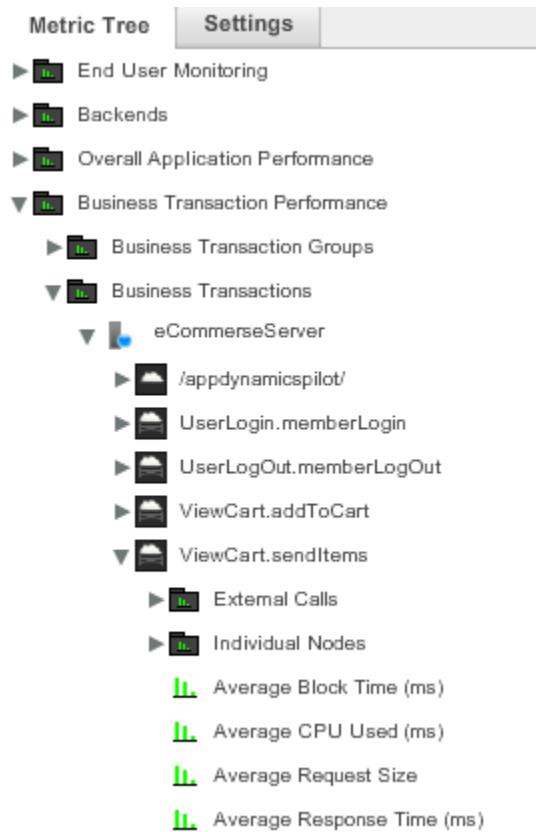
Retrieve metrics

AppDynamics groups the metrics that it collects into following categories:

- Backends
- End User Monitoring
- Overall Application Performance
- Business Transaction Performance
- Application Infrastructure Performance
- Errors
- Information Points

To see the structure of the metric hierarchy of a business application in the AppDynamics UI, expand the nodes in the Metric Browser:

1. In the left navigation pane, select the application for which you are retrieving metrics.
2. Click **Analyze -> Metric Browser**.
3. In left panel expand the nodes in the metric tree.



You can copy the URI for fetching any metric directly from any node in the Metric Browser. This is the easiest way to construct a query to retrieve a particular metric. You can also copy just the metric path portion of the query.

The child elements in the metric path expression are separated by the pipe character (|). The pipe character must not appear at the beginning or end of the metric path expression.

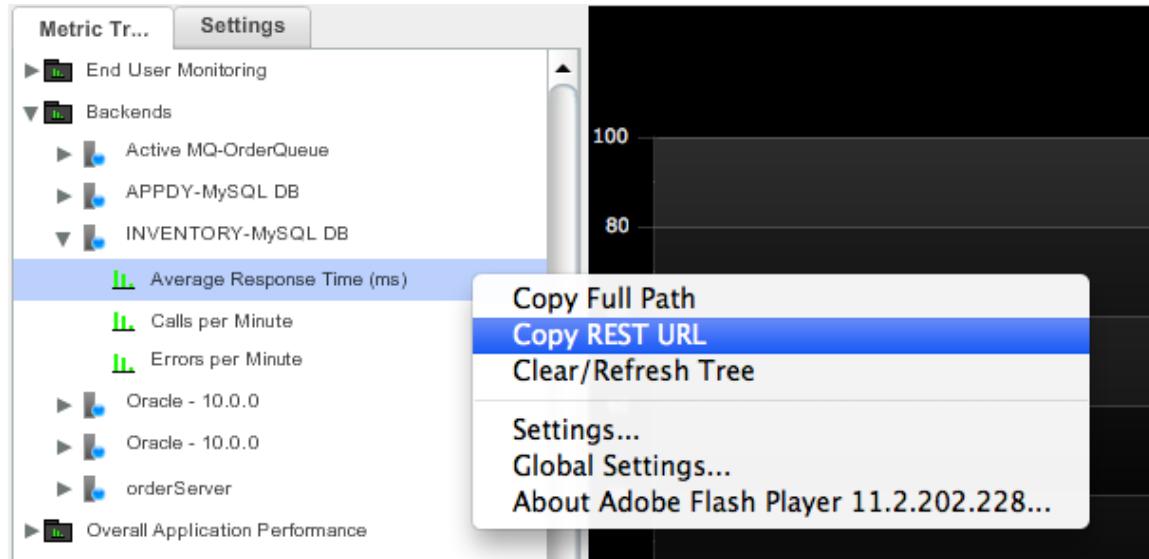
The easiest way to construct a query to retrieve a metric is to select the metric in Metric Browser, copy the REST URL, and paste it into your browser. An alternative is select the metric and copy the metric path, and construct the query by prefacing the metric path with "metric-data?metric-path=".

To Copy the REST URL for a Metric

To copy the REST URL for any metric captured by AppDynamics:

1. Open the Metric Browser as described above.
2. Select the item for which you want to retrieve metrics in the metric tree. You can retrieve the URL at any level of the tree.
3. Right-click the item and select **Copy REST URL** from the drop-down menu.
4. Paste the URL into your Web browser to run the query.

The following example copies the URL for the Average Response Time in the Inventory-MySQL DB for the Acme Online Book Store application for the time period that was selected in the metric browser.



The copied query is

```
http://ec2-23-20-107-243.compute-1.amazonaws.com:8090/controller/rest/applications/AcmeOnlineBook
```

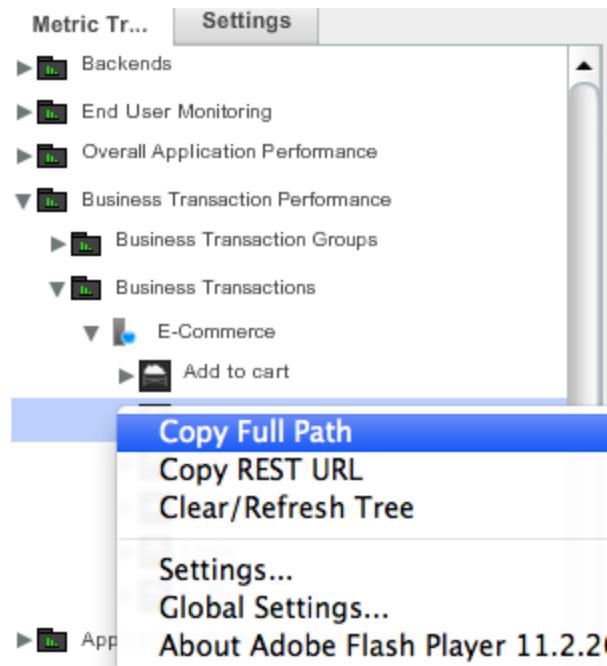
The time range in the request is based on the setting of the time range in the Metric Browser when you copied the URL. You can edit a copied query to change the time range and duration. See [Retrieve metrics for a time range](#) for more information.

To Copy the Metric-Path Parameter

You also copy only the metric path for a specific metric.

To copy a metric path:

1. Open the Metric Browser as described above.
2. Select the item for which you want to retrieve metrics in the metric tree. You can retrieve the metric path at any level of the tree.
3. Right-click the item and select **Copy Full Path** from the drop-down menu.
4. Use the metric path to construct your query.



The copied data is

```
Business Transaction Performance|Business Transactions|E-Commerce|Add to cart
```

Structure of Returned Metrics

The data is returned in a tree structure. If a child element is a container item, its <type> tag is set to "folder". Otherwise the <type> tag for the child element is set to "leaf".

A folder <type> tag indicates that the metric has child elements. The API retrieves the first generation of child elements. You can expand only the children of the folder type.

Retrieve metric hierarchy

URI: /controller/rest/applications/<application-name| application-id>/metrics

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
metric-path	Query	Provide the metric expression to get the metric data. The metric expression can fetch any elements visible in the metric browser. Use the pipe character to separate the parent and child name elements in the tree. Note: The pipe character must not appear at the beginning or at the end of the metric expression.	Yes
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve the metric hierarchy for the ACME Book Store.

URI	Sample object output
/controller/rest/applications/ACME Book Store Application/metrics?metric-path=BusinessTransactionPerformance Business Transactions ECommerce Server ViewCart.sendItems	XML
/controller/rest/applications/ACME Book Store Application/metrics?metric-path=BusinessTransactionPerformance Business Transactions ECommerce Server ViewCart.sendItems&output=JSON	JSON

Use Wild cards in Metric-Path Parameter

You can use the asterisk (*) wild card in the metric data to request metric data for all the instances of AppDynamics entity, such as a business transaction name, tier name or node name, in the metric path.

Example - Retrieve the average response time for all the tiers in the application using the wild card character for the tier name.

URI	Sample object output
<pre>/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Overall Application Performance * Average Response Time (ms)&time-range-type=BEFORE_NOW&duration-in-mins=15</pre>	XML
<pre>/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Overall Application Performance * Average Response Time (ms)&time-range-type=BEFORE_NOW&duration-in-mins=15&output=JSON</pre>	JSON

Example - Retrieve the CPU %Busy metric for all the nodes in the ECommerce tier using the wild card character for the node name.

URI	Sample object output
<pre>/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Application Infrastructure Performance ECommerce Server Individual Nodes * Hardware Resources CPU %Busy&time-range-type=BEFORE_NOW&duration-in-mins=15</pre>	XML

<pre>/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Application Infrastructure Performance ECommerce Server Individual Nodes * Hardware Resources CPU %Busy&time-range-type=BEFORE_NOW&duration-in-mins=15=JSON</pre>	JSON
--	-------------

Example - Retrieve the CPU %Busy metric for all the nodes in all the tiers using wild card characters for the tier and node names.

URI	
<pre>/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Application Infrastructure Performance * Individual Nodes * Hardware Resources CPU %Busy&time-range-type=BEFORE_NOW&duration-in-mins=15</pre>	
<pre>/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Application Infrastructure Performance * Individual Nodes * Hardware Resources CPU %Busy&time-range-type=BEFORE_NOW&duration-in-mins=15&output=JSON</pre>	JSON

Example - Retrieve the Calls per Minute metric for all the business transactions on the ECommerce tier using the wild card character for the business transaction name

URI	Sample object output
<pre>/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Business Transaction Performance Business Transactions ECommerce Server * Calls per Minute&time-range-type=BEFORE_NOW&duration-in-mins=15</pre>	XML
<pre>/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Business Transaction Performance Business Transactions ECommerce Server * Calls per Minute&time-range-type=BEFORE_NOW&duration-in-mins=15&output=JSON</pre>	JSON

Retrieve metrics for a time range

You can fetch the data for any specified metrics for any time range.

Note that metric data REST URIs restrict the amount of data that can be returned. The maximum is 200 metrics.

URI: /controller/rest/applications/<application-name>/metric-data

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
metric-path	Query	The metric expression to get the metric data. Use the pipe character to separate the parent and child name elements in the tree. The Pipe delimiter must not appear at the beginning or at the end of the metric expression.	Yes
time-range-type	Query	Possible values are: BEFORE_NOW To use the "BEFORE_NOW" option, you must also specify the "duration-in-mins" parameter. BEFORE_TIME To use the "BEFORE_TIME" option, you must also specify the "duration-in-mins" and "end-time" parameters. AFTER_TIME To use the "AFTER_TIME" option, you must also specify the "duration-in-mins" and "start-time" parameters. BETWEEN_TIMES To use the "BETWEEN_TIMES" option, you must also specify the "start-time" and "end-time" parameters.	Yes
duration-in-mins	Query	Duration (in minutes) to return the metric data.	If time-range-type is BEFORE_NOW , BEFORE_TIME , or AFTER_TIME
start-time	Query	Start time (in milliseconds) from which the metric data is returned in UNIX epoch time.	If time-range-type is AFTER_TIME or BETWEEN_TIMES
end-time	Query	End time (in milliseconds) until which the metric data is returned in UNIX epoch time.	If time-range-type is BEFORE_TIME or BETWEEN_TIMES
rollup	Query	Default is true. If true, value as single data point is returned. If false, all the values within the specified time range are returned.	No
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve the average response time for the past 15 minutes on the ECommerce server.

URI	Sample object output
/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Overall Application Performance ECommerce Server Average Response Time (ms)&time-range-type=BEFORE_NOW&duration-in-mins=15	XML
/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Overall Application Performance ECommerce Server Average Response Time (ms)&time-range-type=BEFORE_NOW&duration-in-mins=15&output=JSON	JSON

Example - Retrieve the multiple metrics, for the past 15 minutes, for the ViewCart.sendItems transaction on the ECommerce server.

URI	Sample object output
/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Business Transaction Performance Business Transactions ECommerce Server ViewCart.sendItems *&time-range-type=BEFORE_NOW&duration-in-mins=15	XML
/controller/rest/applications/ACME Book Store Application/metric-data?metric-path=Business Transaction Performance Business Transactions ECommerce Server ViewCart.sendItems *&time-range-type=BEFORE_NOW&duration-in-mins=15	JSON

Example - Retrieve snapshots for the 15 minutes after Mon, 13 Aug 2012 08:20:41 for the ACME Book Store Application.

URI
/controller/rest/applications/3/request-snapshots?time-range-type=AFTER_TIME&start-time=1344846041495&duration-in-mins=15
/controller/rest/applications/3/request-snapshots?time-range-type=AFTER_TIME&start-time=1344846041495&duration-in-mins=15&ou

Example - Retrieve snapshots for the 15 minutes before Mon, 13 Aug 2012 08:20:41 for the ACME Book Store Application.

URI
/controller/rest/applications/3/request-snapshots?time-range-type=BEFORE_TIME&end-time=1344846041495&duration-in-mins=15
/controller/rest/applications/3/request-snapshots?time-range-type=BEFORE_TIME&end-time=1344846041495&duration-in-mins=15&o

Example - Retrieve snapshots for the time range between Mon, 13 Aug 2012 08:20:41 and Mon, 13 Aug 2012 08:22:21 GMT for the ACME Book Store Application.

URI
/controller/rest/applications/3/request-snapshots?time-range-type=BEFORE_TIME&end-time=1344846041495&duration-in-mins=15
/controller/rest/applications/3/request-snapshots?time-range-type=BEFORE_TIME&end-time=1344846041495&duration-in-mins=15&o

Retrieve all health rule violations in a particular business application

URI: /controller/rest/applications/<application-name|application-id>/problems/healthrule-violations

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes

time-range-type	Query	Possible values are: BEFORE_NOW To use the "BEFORE_NOW" option, you must also specify the "duration-in-mins" parameter. BEFORE_TIME To use the "BEFORE_TIME" option, you must also specify the "duration-in-mins" and "end-time" parameters. AFTER_TIME To use the "AFTER_TIME" option, you must also specify the "duration-in-mins" and "start-time" parameters. BETWEEN_TIMES To use the "BETWEEN_TIMES" option, you must also specify the "start-time" and "end-time" parameters.	Yes
duration-in-mins	Query	Duration (in minutes) to return the metric data.	If time-range-type is BEFORE_NOW , BEFORE_TIME , or AFTER_TIME
start-time	Query	Start time (in milliseconds) from which the metric data is returned.	If time-range-type is AFTER_TIME or BETWEEN_TIMES
end-time	Query	End time (in milliseconds) until which the metric data is returned.	If time-range-type is BEFORE_TIME or BETWEEN_TIMES
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Retrieve all policy violations in a particular business application

This URI is maintained for compatibility with pre-3.7 versions. Use [Retrieve all health rule violations in a particular business application](#) for 3.7 and later.

URI: /controller/rest/applications/<application-name|application-id>/problems/policy-violations

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
time-range-type	Query	Possible values are: BEFORE_NOW To use the "BEFORE_NOW" option, you must also specify the "duration-in-mins" parameter. BEFORE_TIME To use the "BEFORE_TIME" option, you must also specify the "duration-in-mins" and "end-time" parameters. AFTER_TIME To use the "AFTER_TIME" option, you must also specify the "duration-in-mins" and "start-time" parameters. BETWEEN_TIMES To use the "BETWEEN_TIMES" option, you must also specify the "start-time" and "end-time" parameters.	Yes
duration-in-mins	Query	Duration (in minutes) to return the metric data.	If time-range-type is BEFORE_NOW , BEFORE_TIME , or AFTER_TIME
start-time	Query	Start time (in milliseconds) from which the metric data is returned.	If time-range-type is AFTER_TIME or BETWEEN_TIMES
end-time	Query	End time (in milliseconds) until which the metric data is returned.	If time-range-type is BEFORE_TIME or BETWEEN_TIMES

Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No
--------	-------	---	----

Example - Retrieve the list of all policy violations in the ACME Book Store for the past hour.

URI	Sample object output
/controller/rest/applications/ACME Book Store Application/problems/policy-violations?time-range-type=BEFORE_NOW&duration-in-mins=60&output=XML	XML
/controller/rest/applications/ACME Book Store Application/problems/policy-violations?time-range-type=BEFORE_NOW&duration-in-mins=60&output=JSON	JSON

Retrieve event data

You can capture data for the event types listed in the event-types parameter.

URI: /controller/rest/applications/<application-name|application-id>/events

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
time-range-type	Query	Possible values are: BEFORE_NOW To use the "BEFORE_NOW" option, you must also specify the "duration-in-mins" parameter. BEFORE_TIME To use the "BEFORE_TIME" option, you must also specify the "duration-in-mins" and "end-time" parameters. AFTER_TIME To use the "AFTER_TIME" option, you must also specify the "duration-in-mins" and "start-time" parameters. BETWEEN_TIMES To use the "BETWEEN_TIMES" option, you must also specify the "start-time" and "end-time" parameters.	Yes
duration-in-mins	Query	Specify the duration (in minutes) to return the metric data.	If time-range-type is BEFORE_NOW , BEFORE_TIME , or AFTER_TIME
start-time	Query	Specify the start time (in milliseconds) from which the metric data is returned.	If time-range-type is AFTER_TIME or BETWEEN_TIMES
end-time	Query	Specify the end time (in milliseconds) until which the metric data is returned.	If time-range-type is BEFORE_TIME or BETWEEN_TIMES

event-types	Query	<p>Specify the comma-separated list of event types for which you want to retrieve event information. The event types are:</p> <ul style="list-style-type: none"> • APPLICATION_ERROR • STALL • DEADLOCK • MEMORY_LEAK • MEMORY_LEAK_DIAGNOSTICS • LOW_HEAP_MEMORY • CUSTOM • APP_SERVER_RESTART • SYSTEM_LOG • INFO_INSTRUMENTATION_VISIBILITY • AGENT_EVENT • AGENT_STATUS • ACTIVITY_TRACE • OBJECT_CONTENT_SUMMARY • DIAGNOSTIC_SESSION • HIGH_END_TO_END_LATENCY • APPLICATION_CONFIG_CHANGE • APPLICATION_DEPLOYMENT • AGENT_DIAGNOSTICS • MEMORY • LICENSE • CONTROLLER_AGENT_VERSION_INCOMPATIBILITY • DISK_SPACE • APPDYNAMICS_DATA • APPDYNAMICS_CONFIGURATION_WARNINGS • POLICY_OPEN • POLICY_CLOSE • POLICY_UPGRADED • POLICY_DOWNGRADED • RESOURCE_POOL_LIMIT 	Yes
severities	Query	<p>Specify the comma-separated list of severities for which you want to retrieve event information.</p> <p>The severities are:</p> <ul style="list-style-type: none"> • INFO • WARN • ERROR 	Yes
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve the list of events of type "APPLICATION_ERROR" or "DIAGNOSTIC_SESSION" of any severity that occurred in the specified time range.

URI

```
/controller/rest/applications/ACME%20Book%20Store%20Application/events?time-range-type=BEFORE_NOW&duration-in-mins=30&severity=INFO,ERROR
```

```
/controller/rest/applications/ACME%20Book%20Store%20Application/events?time-range-type=BEFORE_NOW&duration-in-mins=30&severity=WARN,ERROR
```

Create Events

You can create APPLICATION_DEPLOYMENT and CUSTOM events.

Application Deployment Event Integration

The AppDynamics REST API lets you integrate events of type "APPLICATION_DEPLOYMENT" with other systems.

For example, suppose you want to create an event automatically in your AppDynamics monitored system for every new release. To integrate these systems, use the following REST API to create an event of type "APPLICATION_DEPLOYMENT" in your managed environment.

This is a POST request. You should receive the event ID after successful invocation of the request.

URI: /controller/rest/applications/<application-name|application-id>/events

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either application name or application id.	Yes
summary	Query	Provide the summary for the event.	Yes
comment	Query	Provide the comments (if any) for the event.	Yes
eventtype	Query	APPLICATION_DEPLOYMENT	Yes

Create a Custom Event

The AppDynamics REST API lets you create a custom event.

This is a POST request. You should receive the event ID after successful invocation of the request.

URI: /controller/rest/applications/<application-name|application-id>/events?method=POST

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either application name or application id.	Yes
summary	Query	Provide the summary for the event.	No
comment	Query	Provide the comments for the event.	Yes
eventtype	Query	CUSTOM	Yes

Retrieve transaction snapshots for a Business Transaction for a time range

URI: /controller/rest/applications/<application-id>/request-snapshots

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-id>	URI	Provide either the application name or application id.	Yes
time-range-type	Query	Possible values are: BEFORE_NOW To use the "BEFORE_NOW" option, you must also specify the "duration-in-mins" parameter. BEFORE_TIME To use the "BEFORE_TIME" option, you must also specify the "duration-in-mins" and "end-time" parameters. AFTER_TIME To use the "AFTER_TIME" option, you must also specify the "duration-in-mins" and "start-time" parameters. BETWEEN_TIMES To use the "BETWEEN_TIMES" option, you must also specify the "start-time" and "end-time" parameters.	Yes

duration-in-mins	Query	Duration (in minutes) to return the data.	If time-range-type is BEFORE_NOW , BEFORE_TIME , or AFTER_TIME
start-time	Query	Start time (in milliseconds) from which the data is returned.	If time-range-type is AFTER_TIME or BETWEEN_TIMES
end-time	Query	End time (in milliseconds) until which the data is returned.	If time-range-type is BEFORE_TIME or BETWEEN_TIMES
guids	Query	Array of comma-separated guids for the transaction snapshots. If not specified, retrieves all snapshots in the specified time range.	No
archived	Query	True to retrieve archived snapshots. Default is false.	No
deep-dive-policy	Query	Array of comma-separated snapshot policy filters to apply. Valid values are: <ul style="list-style-type: none"> • SLA_FAILURE • TIME_SAMPLING • ERROR_SAMPLING • OCCURRENCE_SAMPLING • ON_DEMAND • HOTSPOT • HOTSPOT_LEARN • APPLICATION_STARTUP • SLOW_DIAGNOSTIC_SESSION • ERROR_DIAGNOSTIC_SESSION • POLICY_FAILURE_DIAGNOSTIC_SESSION • DIAGNOSTIC_SESSION • INFLIGHT_SLOW_SESSION 	No
application-component-ids	Query	Array of comma-separated tier IDs to filter. Default is all the tiers in the application.	No
application-component-node-ids	Query	Array of comma-separated node ID filters. Default is all the nodes in the application	No
business-transaction-ids	Query	Array of comma-separated business transaction ID filters. Default is all the business transactions in the application.	No
user-experience	Query	Array of comma-separated user experiences filters. Valid values are: <ul style="list-style-type: none"> • NORMAL • SLOW • VERY_SLOW • STALL • BUSINESS_ERROR 	No
first-in-chain	Query	If true, retrieve only the first request from the chain. Default is false.	No
need-props	Query	If true, properties are included in the result. Default is false.	No
need-exit-calls	Query	If true, exit calls are included in the result. Default is false.	No
execution-time-in-milis	Query	If set, retrieves only data for requests with execution times greater than this value.	No
session-id	Query	If set, retrieves data only for this session id.	No
user-principal-id	Query	If set, retrieves data only for this user login.	No

error-ids	Query	Array of comma-separated error codes to filter by. Default is to retrieve all error codes.	No
starting-request-id, ending-request-id	Query	If set, retrieves data only for this range of request IDs.	No
error-occurred	Query	If true, retrieves only error requests. Default is false.	No
diagnostic-snapshot	Query	If true, retrieves only diagnostic snapshots. Default is false.	No
bad-request	Query	If true, retrieves only slow and error requests. Default is false.	No
diagnostic-session-guid	Query	Array of comma-separated diagnostic session guids to filter.	No
data-collector-name	Query	Used with data-collector-value to filter snapshot collection based on the value of a data collector. <i>New in 3.6.1.</i>	No
data-collector-value	Query	Used with data-collector-name to filter snapshot collection based on the value of a data collector. <i>New in 3.6.1.</i>	If data-collector-name is set.
data-collector-type	Query	Used with data-collector-name and data-collector-value to filter snapshot collection based on the value of a data collector. <i>New in 3.6.1.</i> Valid values are: <ul style="list-style-type: none"> • ERROR_IDS • STACK_TRACES • ERROR_DETAIL • HTTP_PARAMETER • BUSINESS DATA This type is a method invocation data collector. • COOKIE • HTTP_Header • SESSION_KEY • RESPONSE_HEADER • LOG_MESSAGE • TRANSACTION_PROPERTY • TRANSACTION_EVENT • DOTNET_PROPS • DOTNET_SET 	If data-collector-name is set.
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve list of transaction snapshots for the ACME Book Store.

URI
/controller/rest/applications/AcmeOnlineBookStore/request-snapshots?time-range-type=BEFORE_NOW&duration-in-mins=2
/controller/rest/applications/AcmeOnlineBookStore/request-snapshots?time-range-type=BEFORE_NOW&duration-in-mins=2&output=J

Example - Retrieve list of transaction snapshots filtered on the value of a data collector.

URI
/controller/rest/applications/2/request-snapshots?time-range-type=BEFORE_NOW&duration-in-mins=20&data-collector-type=Http%20f

Create and modify AppDynamics users

This is an HTTP POST operation.

The create and modify user URIs are identical except for the user-id parameter, which is not passed for the create operation. The user-id is generated by the create operation.
A response code of 200 indicates success.

URI: /controller/rest/users

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
user-name	Post	user name	Yes
user-id	Post	user id	No for a create; yes for an update
user-display-name	Post	display name	Yes
user-roles	Post	comma-separated list of roles	No
user-password	Post	user password	Yes for create; optional for update
user-email	Post	user email	Yes

Include or exclude a business transaction from monitoring

This is an HTTP POST operation.

To exclude a business transaction from monitoring, set the exclude parameter to true.
To turn on monitoring for a currently excluded business transaction, set the exclude parameter to false.

URI: /controller/rest/applications/<application-name | application-id>/business-transactions

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
business-transaction-list	Post	List of business transaction IDs to be included or excluded from monitoring	Yes
exclude	Post	true false	Yes

A sample business-transaction-list is:

```
<business-transactions>
  <business-transaction>
    <id>15</id>
  </business-transaction>
  <business-transaction>
    <id>16</id>
  </business-transaction>
</business-transactions>
```

Retrieve all controller global configuration values

These are the values that you set interactively from the Controller Settings screen in the AppDynamics Administration console.

Access requires the "root" password, which was created for the AppDynamics admin user when the controller was installed. See [Access the Administration Console](#).

URI: /configuration

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve list of all global configuration values

URI	Sample object output
/controller/rest/configuration	XML
/controller/rest/configuration?output=JSON	JSON

Retrieve a single controller global configuration value

URI: /configuration?name=<controller_setting_name>

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
name	Query	Name of the Controller setting to retrieve	Yes
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve the global metrics buffer size.

URI	Sample object output
/controller/rest/configuration?name=metrics.buffer.size	XML
/controller/rest/configuration?name=metrics.buffer.size&output=JSON	JSON

Configure Global Controller Settings

This is an HTTP POST operation.

URI: /controller/rest/configuration

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
name	Post	name of a controller setting; to see all the names, retrieve all the values as described in Example - Retrieve list of all global configuration values	Yes
value	Post	value to set	Yes

Mark Nodes as Historical

This is an HTTP POST operation.

Pass a list of one or more node ids of the nodes to be marked as historical as parameters. The parameter serves as both the name and the value of the node. You can specify up to a maximum of 25 node ids to be marked.

The XML response to this operation has the appearance:

```
<nodeId>1</nodeId>
<nodeId>2</nodeId>
...
<nodeId>n</nodeId>
```

AppDynamics stops collecting metrics for a node that is marked as historical.

By default AppDynamics marks as historical (soft deletes) a node that has lost contact with the Controller for the number of hours configured in the node.retention.period controller setting. The default is 500 hours.

Information from a historical node can be retrieved until the node has lost contact with the controller for the number of hours configured in the node.permanent.deletion.period, after which time the historical node is permanently deleted. The default is 720 hours.

URI: /controller/rest/mark-nodes-historical

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
value	Post	node id	Yes

Retrieve all policy violations in a particular business application

URI: /controller/rest/applications/<application-name|application-id>/problems/policy-violations

Input parameters

Parameter Name	Parameter Type	Value	Mandatory
<application-name application-id>	URI	Provide either the application name or application id.	Yes
time-range-type	Query	Possible values are: BEFORE_NOW To use the "BEFORE_NOW" option, you must also specify the "duration-in-mins" parameter. BEFORE_TIME To use the "BEFORE_TIME" option, you must also specify the "duration-in-mins" and "end-time" parameters. AFTER_TIME To use the "AFTER_TIME" option, you must also specify the "duration-in-mins" and "start-time" parameters. BETWEEN_TIMES To use the "BETWEEN_TIMES" option, you must also specify the "start-time" and "end-time" parameters.	Yes
duration-in-mins	Query	Duration (in minutes) to return the metric data.	If time-range-type is BEFORE_NOW , BEFORE_TIME , or AFTER_TIME
start-time	Query	Start time (in milliseconds) from which the metric data is returned.	If time-range-type is AFTER_TIME or BETWEEN_TIMES
end-time	Query	End time (in milliseconds) until which the metric data is returned.	If time-range-type is BEFORE_TIME or BETWEEN_TIMES
incident-status	Query	If OPEN, retrieve only the policy violations that are open. If RESOLVED, retrieve only the policy violations that are resolved.	No
Output	Query	HTTP Request parameter included as part of the URL to change the output format. Valid values are "XML" (default) or "JSON".	No

Example - Retrieve the list of all open policy violations in the ACME Book Store for the past 1000 minutes.

URI

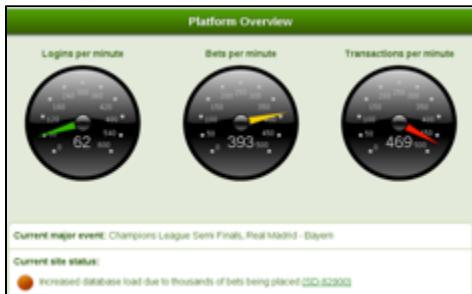
```
/controller/rest/applications/2/problems/policy-violations?time-range-type=BEFORE_NOW&duration-in-mins=1000&incident-status=OP  
/controller/rest/applications/2/problems/policy-violations?time-range-type=BEFORE_NOW&duration-in-mins=1000&incident-status=OP
```

Example Integrations Using the AppDynamics REST API

- Mobile Application
- Jenkins Plugin

Mobile Application

See [Going Mobile with AppDynamics REST API](#) for information about a monitoring application that runs on a smart phone.



The code is available at: <https://github.com/unixunion/appdyngauges>

Courtesy Kegan Holtzhausen, thank you!

Jenkins Plugin

This plugin makes it possible to integrate data from AppDynamics into your Jenkins build.

The plugin is available at: <https://wiki.jenkins-ci.org/display/JENKINS/AppDynamics+Plugin>

Courtesy Miel Donkers, thank you!

Integrate AppDynamics with Splunk

- Prerequisites for Configuring Splunk Integration
 - To Configure AppDynamics to Interface with Splunk
- Launching a Spunk Search from AppDynamics
 - Prerequisites
 - To launch a Splunk search
 - Seeing Performance Data from AppDynamics in Splunk
 - Viewing AppDynamics Notifications in Splunk
 - Learn More

The AppDynamics-Splunk integration enables you to get a 360-degree view of your application performance by leveraging the AppDynamics "inside-the-app" view and the Splunk "outside-the-app" view.

The integration provides three capabilities. You can do the following:

- Launch Splunk searches using auto-populated queries from in-context locations in the AppDynamics Console based on criteria such as time ranges and node IP address.
- Push notifications on policy violations and events from AppDynamics to Splunk so that a Splunk user can use those to launch deep dives in AppDynamics. See [Viewing AppDynamics Notifications in Splunk](#). This enables you to see what was going on in the logs around the time of a specific event as well as navigate to AppDynamics-provided details at that point in time.
- Mine performance data from AppDynamics using the Controller REST API, and push it into Splunk. See [Seeing Performance](#)

Data from AppDynamics in Splunk.

Watch this webinar:<http://info.appdynamics.com/AppDynamicsSplunkWebinarRecording.html> to hear John Martin, Sr. Director of Production Engineering at Edmunds.com, give an overview of how he uses AppDynamics and Splunk together, to gain the visibility he needs to manage the performance of his award-winning consumer car website, Edmunds.com. This video covers the use cases initiated from Splunk using the AppDynamics App for Splunk.

This topic covers the Splunk search capability that is launched from AppDynamics.

Prerequisites for Configuring Splunk Integration

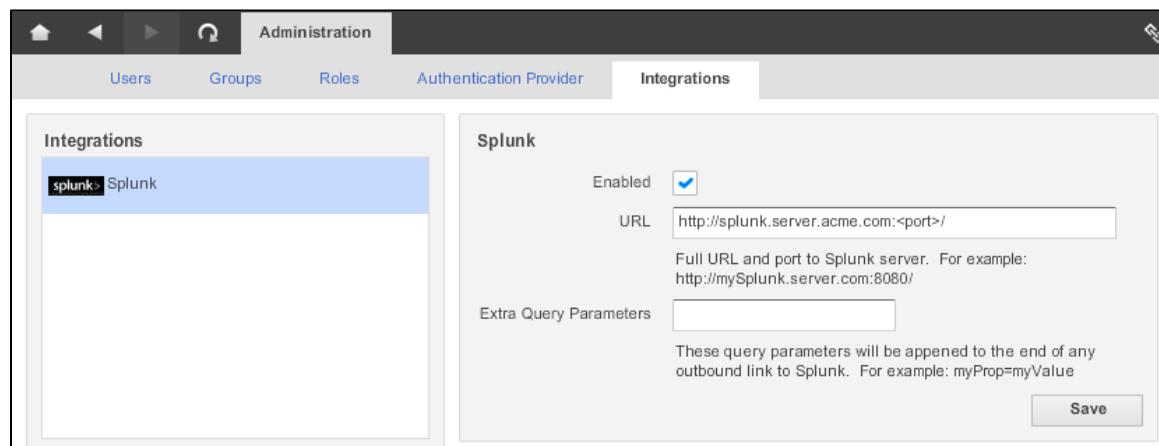
To use the **Search Splunk** capability, you need to configure AppDynamics to communicate with your Splunk Server. To do this you need the following:

- Proper AppDynamics administrator credentials
 - For single tenant installations, login as the AppDynamics Administrator (root administrator)
 - For multi-tenant installations login as the Account Administrator for the specific tenant using Splunk.
- The URL and port number for the Splunk server.

See the [Splunk website](#) for information about Splunk.

To Configure AppDynamics to Interface with Splunk

1. Click the Setup menu in the upper right section of the screen.
2. From the drop-down menu, click **Administration**.
3. Click the **Integration** tab.
4. Click Splunk in the Integrations list.



5. Check the **Enabled** check box to enable the integration.
6. Enter the Splunk URL and port number.
7. (Optional) Enter Extra Query Parameters. These parameters are appended to each Splunk search initiated from AppDynamics.
8. Click **Save**.

Launching a Splunk Search from AppDynamics

You can launch a search of your Splunk logs for a specific time frame associated with a transaction snapshot from several points in AppDynamics.

Prerequisites

- Splunk credentials: The first time that you launch a Splunk search, you are prompted to provide your Splunk credentials. After this, your credentials are cached by the browser.
- Your Splunk Server is up and running.

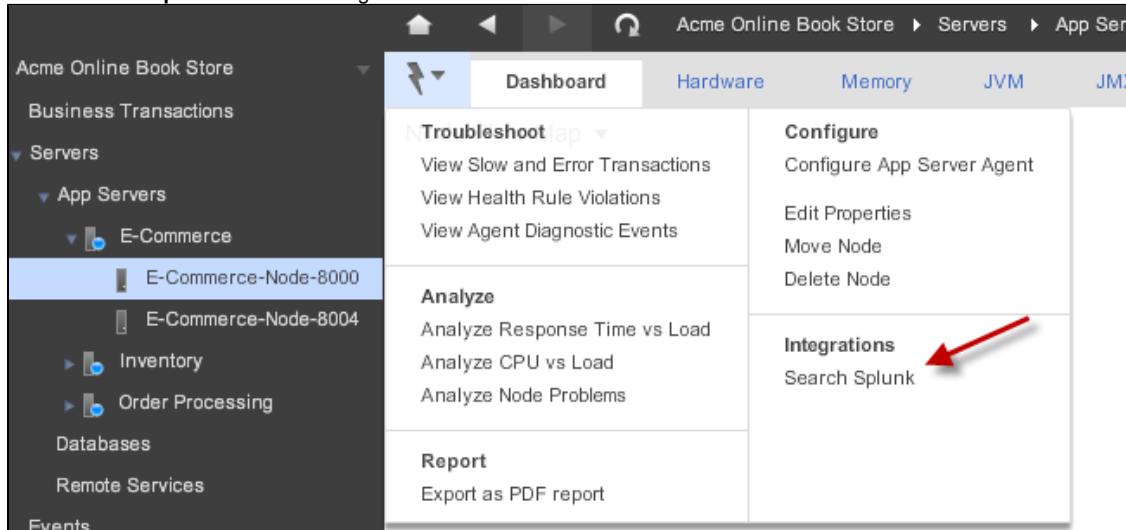
- Your browser is configured to allow pop-ups.

Enable Pop-ups
The first time that you access the Splunk Server, you are prompted to log in. If nothing happens, it is most likely that either your browser is blocking the Splunk login pop-up or the Splunk Server is not running.

To launch a Splunk search

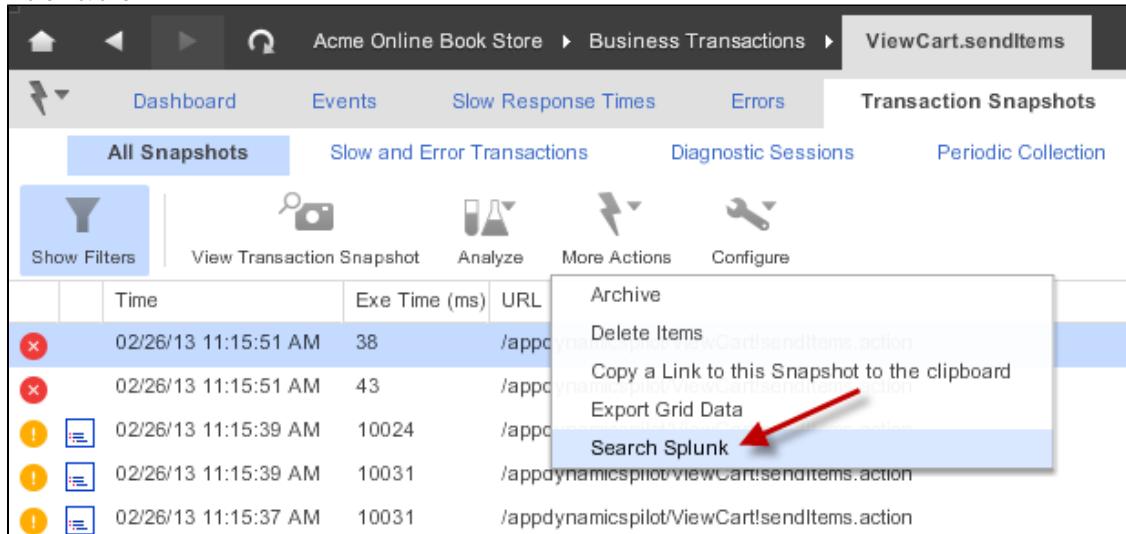
You can access the **Search Splunk** option from the following locations:

1. The node dashboard:
 - a. Navigate to a node dashboard.
 - b. Click **More Actions**.
 - c. Select **Search Splunk** under the Integrations section.



2. The list of transaction snapshots:
 - a. From the business transaction dashboard, select the Transaction Snapshot tab.
 - b. Select a transaction snapshot and right-click to access a list of options or click **More Actions** to see **Search Splunk** in the list of options.

More Actions:



- c. Right-click list:

The screenshot shows the 'Transaction Snapshots' tab selected in the top navigation bar. A context menu is open over a list of transaction snapshots. The menu items include: View Transaction Snapshot, Archive, Delete Items, Compare Snapshots, Identify the most expensive calls / SQL statements in a group of Snapshots, Copy a Link to this Snapshot to the clipboard, **Search Splunk** (highlighted with a red arrow), Settings..., Global Settings..., and About Adobe Flash Player 11.6.602.171... . Below the menu, a table lists three transaction snapshots with columns for Time, Exec Time (ms), and URL.

Time	Exec Time (ms)	URL
02/26/13 11:15:02 AM	71	/appdynamicspilot/ViewCart!sendItems.action
02/26/13 11:14:42 AM	69	/appdynamicspilot/ViewCart!sendItems.action
02/26/13 11:14:42 AM	94	/appdynamicspilot/ViewCart!sendItems.action

- d. Select **Search Splunk**
 3. The Transaction Snapshot Flow Map:
 a. Locate the Search Splunk option at the top of the flow map.

The screenshot shows the 'Transaction Snapshot Flow Map' interface. At the top right, there is a 'Splunk' button and a 'Search Splunk' button (highlighted with a red arrow). Below the buttons, the transaction details are listed: Transaction ID (8971b9df-e95e-4369-97bd-650d2c91a625), USER EXPERIENCE (STALL), EXECUTION TIME (54161 ms), TIMESTAMP (02/26/13 11:15:02 AM), BUSINESS TRANSACTION (ViewCart.addToCart), REQUEST GUID (8971b9df-e95e-4369-97bd-650d2c91a625), and Splunk (8971b9df-e95e-4369-97bd-650d2c91a625).

4. The Call Drill Down Summary pane:

The screenshot shows the 'Call Drill Down Summary' pane. It displays various performance metrics and configuration options. At the bottom right, there is a 'Splunk' button and a 'Search Splunk' button (highlighted with a red arrow).

Metrics shown in the pane:

- User Principal
- No User Principal
- Process ID: 22791
- Thread Name: http-8000-Processor14
- Thread ID: 39
- Transaction Thresholds: Slow: (not found)
Very Slow: (not found)
[Configure](#)
- Request GUID: 8971b9df-e95e-4369-97bd-650d2c91a625

Seeing Performance Data from AppDynamics in Splunk

This feature is initiated using the AppDynamics App for Splunk. See <http://splunk-base.splunk.com/apps/appdynamics>.

Viewing AppDynamics Notifications in Splunk

This feature is initiated using the AppDynamics App for Splunk. See <http://splunk-base.splunk.com/apps/appdynamics>.

Learn More

- [Splunkbase Apps](#)

- Splunk documentation

Integrate AppDynamics with BMC End User Experience Management

- To Integrate AppDynamics with BMC End User Experience Management
 - Setting Up AppDynamics
 - Setting up BMC End User Experience Management
 - Verifying the Integration
 - Check the status of the Application Visibility servers configured
 - Check the reference list for the "Trace (object-level)" custom field
- User Accounts for Controller SaaS Users
- Troubleshooting BMC EUEM-AppDynamics Integration
 - Is the Controller available and can it return its status?
 - Is the BMC EUEM visibility server configuration correct?
 - Is the controller.services.hostName address in the Controller domain.xml file correct?
 - Is the AppDynamics Administration configuration correct?
 - Was the Java App Agent started or restarted AFTER the Controller integration settings changed?
 - Is AppDynamics constructing the correct URLs to link to snapshots?
 - Is the BMC EUEM system is seeing the required response headers from AppDynamics?
 - Is the BMC EUEM system reporting Snapshot status for pages in the Session browser?
 - Why are all the drilldown icons red?
 - Where is the log file containing information about Application Visibility events?
- Learn More

This topic describes how to integrate AppDynamics with the BMC End User Experience Management (EUEM) appliance.

Note that this feature was previously known as Coradiant. See <http://www.bmc.com/products/brand/coradiant.html> for information about the integration with BMC.

To Integrate AppDynamics with BMC End User Experience Management

Setting Up AppDynamics

 An AppDynamics for BMC Software Solutions license is required.

Steps 1 through 3 are for on-premise installations. If you are using the AppDynamics SaaS Controller, skip to Step 4.

1. Install the AppDynamics Controller. For details see [Install the Controller on Linux](#) or [Install the Controller on Windows](#).

2. Stop the Controller. Go to command line console and execute following command:

For Windows:

```
controller.bat stop
```

For Linux:

```
./controller.sh stop
```

3. Open the AppDynamics Controller domain.xml file located at <controller_install>/appdynamics-controller/appserver/domains/domain1/config. Configure the host name using following JVM options:

```
<jvm-options>
-Dappdynamics.controller.services.hostName=<controller_host>
</jvm-options>
```

The controller_host is the IP address or domain (www.host.com) host that you configured during Controller installation. For details see [Install the Controller on Linux](#) or [Install the Controller on Windows](#).

For example, if the controller_host was configured to 192.10.10.1, the -D property will have following value:

```
<jvm-options>
-Dappdynamics.controller.services.hostName=192.10.10.1
</jvm-options>
```

The controller.services.hostName property is used to connect the BMC EUEM User Interface with the AppDynamics Controller. The URL points to a snapshot in the AppDynamics UI. The host name should be an IP or domain name that is accessible to the end users of the EUEM and AppDynamics user interfaces. If you need to change the hostName see [Changing the Host Name](#).

4. In the Controller domain.xml file, change the setting of the appdynamics.controller.ui.deeplink.url property. The property requires a full URL.

```
appdynamics.controller.ui.deeplink.url=http://<controller_host>:<port>/controller
```

5. Save the domain.xml file.

6. Restart the Controller at a command line console:

For Windows:

```
controller.bat start
```

For Linux:

```
./controller.sh start
```

7. Configure the Integration Parameters.

From a browser, log in the AppDynamics Admin UI as the Admin user:

```
http://<controller_host>:<port>/controller/admin.html
```

Note: The UI may not open the Accounts screen if the Controller is not in multi-tenant mode. To work around this, use the URL parameter enableAccounts=true.

```
http://<controller_host>:<port>/controller/admin.html?enableAccounts=true
```

The screenshot shows the AppDynamics Administration interface. On the left, there's a navigation pane with 'Administration' at the top, followed by 'Accounts' (which is selected and highlighted in blue) and 'Controller Settings'. The main content area has a title 'AppDynamics Administration' and two sections: 'Accounts' (with the subtitle 'Create and Manage Accounts') and 'Controller Settings' (with the subtitle 'Configure the Controller').

8. Click **Controller Settings** to open the list of controller settings.

9. Set the vendor identifier to "ad" and click **Save**.

controller.vendor.identifier	Controller vendor id	<input type="text" value="ad"/>	<input type="button" value="Save"/>
------------------------------	----------------------	---------------------------------	-------------------------------------

10. Set the server identifier and click **Save**.

A server identifier is required.

If you are using the SaaS service set this property to "CONTROLLER_1".

controller.server.identifier	Controller server id	<input type="text" value="CONTROLLER_1"/>	<input type="button" value="Save"/>
------------------------------	----------------------	---	-------------------------------------

11. On the left navigation pane, Click **Accounts** to open the Accounts list.

12. Double-click the customer1 account row to open the customer1 account properties editor.

customer1	customer1	SJ5b2m7d1\$354	100000	100000	No expiration date
-----------	-----------	----------------	--------	--------	--------------------

13. Set the BMC EUEM integration properties. An AppDynamics for BMC Software Solutions license is required for these options to be available.

- Click the **BMC End User Experience Management Enabled** checkbox.
- Set the URL to the BMC EUEM Analyzer. Be sure to add the "/" character at the end of the URL.

BMC End User Experience Management Enabled	<input checked="" type="checkbox"/>
BMC End User Experience Management Sever URL	<input type="text" value="http://192.1.22.10/"/>
This is the URL to the BMC Server. For example http://server:port/. Note that you need the last / in there.	

Setting up BMC End User Experience Management

These instructions apply to Real User Analyzer or Real User Monitor.

1. Log in to BMC EUEM as the security officer.
2. Navigate to **Administration->Integration->Application Visibility servers**.
3. Add an Application Visibility server with the following parameters:

```

URI: http://<controller_host>:<port>/controller/rest
Authentication method: None
Username: (not needed)
Password: (not needed)

```

Verifying the Integration

The integration is successful when the response headers from AppDynamics are visible in the BMC EUEM UI. There are two ways to verify the integration:

Check the status of the Application Visibility servers configured

Check the reference list for the "Trace (object-level)" custom field

Check the status of the Application Visibility servers configured

1. In the BMC End User Experience Management UI (Real User Monitor or Real User Analyzer), navigate to the **Administration->Integration->Application Visibility servers** page.
2. Check whether the configured AppDynamics server has a green checkmark under the **Connection** column.
3. If there is a green checkmark, hover the mouse over the icon under the computer icon and verify the settings of the AppDynamics server.

Check the reference list for the "Trace (object-level)" custom field

1. In the BMC End User Experience Management UI (Real User Monitor or Real User Analyzer), open the **Reference lists** menu.
2. Select **Object custom fields** then **Trace (object-level)**.
3. There should be a list of values for this custom field in the form of:

```
vid=ad&sid=CONTROLLER_1&tid=<GUID of a Business Transaction>
```

The screenshot shows the TrueSight Analyzer interface. At the top, there's a navigation bar with links for Dashboards, Watchpoints, Session Browser, Reports, Incidents, Reference lists (which is highlighted in blue), and Administration. Below the navigation bar, there's a header for 'TrueSight Analyzer Device' with icons for Reboot, Shutdown, ON/OFF, and SSH access. The main content area is divided into several sections: General settings, Integration, and Data flow settings. On the right side, there's a sidebar for 'Reference lists' that is currently open. Under 'Object custom fields', the 'Trace (object-level)' option is selected and highlighted with a blue box. Other options listed under Object custom fields include Page ID, Server ID, and Page management.

If the integration is successful, you will see a list of values captured by AppDynamics.

If you do not see any values for the Trace custom field, either the BMC EUEM system is not configured properly or no headers are being sent by the AppDynamics system. For assistance see [Troubleshooting BMC EUEM-AppDynamics Integration](#).

User Accounts for Controller SaaS Users

When integrating BMC EUEM with AppDynamics SaaS Controllers, the username and password must be provided during the

integration.

For a multi-tenant Controller such as the AppDynamics SaaS Controller, the user name format is <user-name>@<account-name>. SaaS customers receive the user-name and account-name in the Welcome email sent by the AppDynamics Support Team.

Troubleshooting BMC EUEM-AppDynamics Integration

The following table summarizes the settings that are required to enable successful BMC EUEM-AppDynamics integration.

Description	What to set	Required by AppDynamics	Required by BMC EUEM
Application Visibility server (AppDynamics Controller) URL	http://<controller_host>:<port>/controller/rest	No	Yes
Controller SaaS User credentials	<user-name>@<account-name>	No	Yes
Controller host name or IP address	controller.services.hostName in the domain.xml file	Yes	Yes
Controller port	default is 8090	Yes	Yes
Controller deep Link URL	appdynamics.controller.ui.deeplink.url in the domain.xml file	Yes	No
Controller vendor ID	controller.server.identifier property in the Controller Settings must be "ad"	Yes	No
Controller server name	controller.vendor.identifier property in the Controller Settings	Yes	No
Turn on EUEM in AppDynamics	the BMC End User Experience Management Enabled option in the Accounts Settings	Yes	No
URL pointing to a BMC EUEM Real User Analyzer or BMC EUEM Real User Monitor	The BMC End User Experience Management Server URL setting in the Account Settings	Yes	No

To determine where a problem may lie, check the following information.

Is the Controller available and can it return its status?

Open the Controller URL in a browser that is in the same network segment as the BMC EUEM Real User Analyzer/Monitor:

```
http://<controller_host>:<port>/controller/rest/serverstatus
```

An XML file with AppDynamics metadata should display. If it does not, then the Controller is not available. If you are using the SaaS server, contact AppDynamics Support. If you are using an on-premise server, you may need to restart it.

Is the BMC EUEM visibility server configuration correct?

In the BMC EUEM UI, the AppDynamics URL should be:

```
http://<controller_host>:<port>/controller/rest
```

Is the controller.services.hostName address in the Controller domain.xml file correct?

The Controller services host name option should be set to your Controller server:

```
<jvm-options>
-Dappdynamics.controller.services.hostName=<controller_host>
</jvm-options>
```

The controller_host is the IP or domain address that accesses the Controller on your network. The Controller uses this address when it provides deep links to snapshots.

Is the AppDynamics Administration configuration correct?

For AppDynamics 3.3.3 and newer, in the **Administration -> Controller Settings** the controller.server.identifier property must be correct. When using the Controller SaaS service, it must be exactly set to:

```
CONTROLLER_1
```

Also the controller.vendor.identifier property must be exactly set to:

```
ad
```

 Remember to **Save** each property!

Also check the **Administration->Accounts->Account Details** settings and confirm that:

- the **BMC End User Experience Management Enabled** option is checked
- The **BMC End User Experience Management Server URL** is the URL to the BMC server, including the slash "/" at the end of the URL

For AppDynamics 3.3.2 and older, use these settings:

```
corariant.integration.url = https://ts02.corariant.com/
```

The last character "/" is required.

```
corariant.integration.enabled = true
```

 Remember to save each property!

Was the Java App Agent started or restarted AFTER the Controller integration settings changed?

When the integration settings change in the Controller Admin UI, the Agent must restart for the changes to take effect.

Is AppDynamics constructing the correct URLs to link to snapshots?

Follow this procedure to get details about snapshots.

1. In AppDynamics, locate a snapshot with a call graph and copy the request GUID. In BMC EUEM this is called the trace ID (tid).
2. In a text file, construct a URL for the snapshot that includes the request tid/GUID:

```
http://<controller_host>:<port>/controller/rest/tracestatus?tid=<requestGuid>
```

3. Run the URL in a browser and see what is returned. For example:

```
<tracelist vendorid="" version="1"><trace  
tid="855a90b1-5698-45c0-bded-22a6374981fc"><available>true</available><displayurl>http://111.0.0.
```

4. Copy the <displayurl> and run it in a browser. You should see the snapshot.

Is the BMC EUEM system is seeing the required response headers from AppDynamics?

The Application Visibility integration works by allowing you to navigate from a page view in the Session browser to a transaction on the

AppDynamics system. The best way to look for pages with an AppDynamics link is to create a Page Watchpoint with the following filter expression:

```
NOT (trace is null)
```

From the Watchpoints page, use the Action button associated with this watchpoint to drill down to these pages and access the Session browser.

Is the BMC EUEM system reporting Snapshot status for pages in the Session browser?

Next to each page there can be four types of icons representing different statuses about the link from the page to the AppDynamics transaction. They are explained as follows.

- A clock icon indicates that the BMC EUEM system is trying to contact the AppDynamics system. If this icon persists, verify the communication between the BMC EUEM system and the AppDynamics system. Contact your IT administrator for assistance.
- A green icon indicates that the BMC EUEM system was able to contact the AppDynamics system and a Snapshot was found for that Business Transaction. Click the details link to navigate to the AppDynamics transaction.
- A yellow icon indicates that the BMC EUEM system was able to contact the AppDynamics system but the captured tid/GUID is no longer available. Contact your AppDynamics administrator for assistance about why the transaction no longer exists. In the BMC EUEM Real User Analyzer software prior to version 1.1 and BMC EUEM Real User Monitor software prior to version 5.1, when the BMC EUEM system contacts the AppDynamics system but the captured tid/GUID is no longer available, a yellow icon is displayed in the Session browser result window. More recent versions of BMC EUEM have suppressed the yellow icon and the behavior is replaced by the clock icon appearing then disappearing. Contact your AppDynamics administrator for assistance about why the transaction no longer exists. To re-enable the yellow icon functionality, please contact BMC Customer Support.
- A red icon indicates that the BMC EUEM system tried to contact the AppDynamics system but did not get a response for the tid/GUID captured. Click the page to see the page detail and click the Custom fields link. Verify that the Trace custom field has a value and try to search the value of the tid/GUID on the AppDynamics system using Advanced Search. If the search does not work, contact your AppDynamics Support representative for assistance.

Note: If you refresh the Session browser result window, all the Snapshot status will be rechecked for the pages shown. This is a good way to quickly check the new status after you've made some changes.

Why are all the drilldown icons red?

If the Session browser result window displays a lot of pages, a timeout may be reached when requesting the status for all of them from the AppDynamics controller. By default this timeout is 12 seconds. If there is a firewall or network device that adds latency to the communication between BMC EUEM and AppDynamics systems then this timeout will need to be increased.

To increase the timeout contact a BMC Support engineer.

Where is the log file containing information about Application Visibility events?

1. Log into EUEM as the security officer and access this URL:

```
http://EUEM-host/tools/logs.do
```

2. Click the link for "Application Visibility events log".

Learn More

- AppDynamics EUM Integration
- BMC End User Experience Management

Integrate with AppDynamics for Databases

- Prerequisites for AppDynamics for Databases Integration
- Configuring AppDynamics to Interface with AppDynamics for Databases
 - To Configure AppDynamics to Interface with AppDynamics for Databases

This topic describes how you can link to AppDynamics for Databases from a monitored relational database that is discovered by

AppDynamics, and supported by AppDynamics for Databases. This integration provides access to the deep database performance metrics provided by AppDynamics for Databases.

Prerequisites for AppDynamics for Databases Integration

To use this integration you must have a AppDynamics for Databases license.

AppDynamics for Databases must be configured to monitor the databases that you want to link to from AppDynamics.

See the AppDynamics for Databases documentation for information about using AppDynamics for Databases.

Configuring AppDynamics to Interface with AppDynamics for Databases

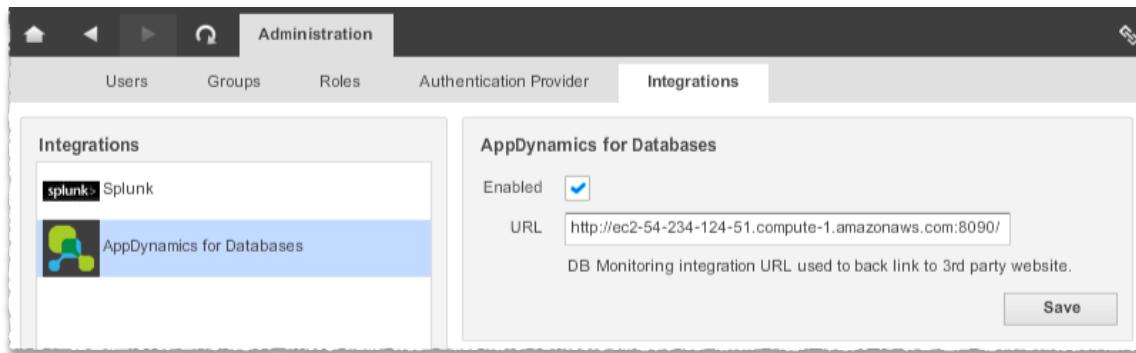
To Configure AppDynamics to Interface with AppDynamics for Databases

1. Click the Setup menu in the upper right section of the screen.

2. From the drop-down menu, click **Administration**.

3. Click the **Integration** tab.

4. Click AppDynamics for Databases in the Integrations list.



5. Check the **Enabled** check box to enable the integration.

6. Enter the URL of your AppDynamics for Databases installation.

7. Click **Save**.

Integrate AppDynamics with DB CAM

- Prerequisites for DB CAM Integration
- Configuring AppDynamics to Interface with DB CAM
 - Configure DB CAM at the Account Level
 - To Configure One AppDynamics Account for DB CAM Integration
 - Configure DB CAM at the Agent Level
- Linking to DB CAM from AppDynamics
 - To Link to DB CAM from a Dashboard
 - To Link to DB CAM from a Transaction Snapshot
- Learn More

You can link to DB CAM for any DB CAM-monitored database that is discovered by AppDynamics. This integration provides access to the database performance metrics provided by DB CAM.

Prerequisites for DB CAM Integration

To use this integration you must have a DB CAM license.

DB CAM must be configured to monitor the databases that you want to link to from AppDynamics.

Configuring AppDynamics to Interface with DB CAM

You configure DB CAM integration at the account level and at the app agent level.

Configure DB CAM at the Account Level

Configure the integration at the account level using the Administration Console at

```
<host>:<port>/controller/admin.html
```

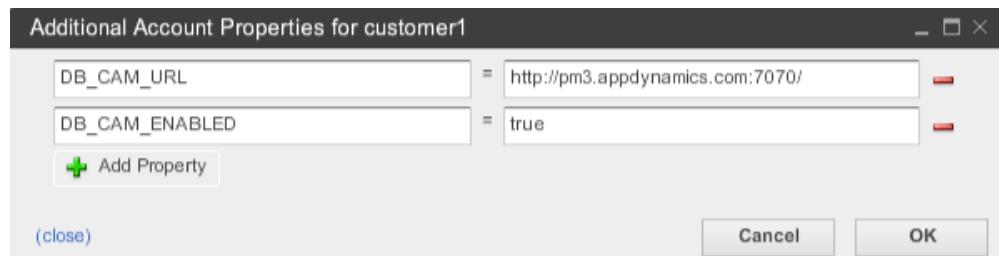
Create two new properties as name-value pairs in each account for which you want to enable DB CAM integration.

To Configure One AppDynamics Account for DB CAM Integration

1. Login to the Administrator Console with the administrator root password.
2. Select Accounts.
3. In the accounts list, double-click the account for which you want to configure DB CAM integration.
4. In the upper right corner of the account screen, click **Additional Account Properties**.
5. In the Additional Account Properties screen, click **Add Property** to add the DB_CAM_URL property.
6. In the left field enter "DB_CAM_URL".
7. In the right field enter the URL of the AppDynamics Controller that you are configuring using the syntax

```
http[s]://<host>:<port>
```

8. Click **Add Property** again to add the DB_CAM_ENABLED property.
9. In the left field enter "DB_CAM_ENABLED".
10. In the right field enter "true".
11. Click **OK** to save the properties.
12. Log out of the Administrator Console.



Configure DB CAM at the Agent Level

For each app agent for which you want to enable access to deep diagnostics from DB CAM:

1. Open the AppServerAgent/conf/app-agent-config.xml file for the app agent.
2. Locate the TransactionMonitoringService element:

```
<agent-service name="TransactionMonitoringService" enabled="true">
```

3. Add the jdbc-dbcam-integration-enabled property for the service:

```
<agent-service name="TransactionMonitoringService" enabled="true">
    <service-dependencies>BCIEngine,SnapshotService</service-dependencies>
    <configuration-properties>
        <property name="jdbc-dbcam-integration-enabled" value="true" />
    </configuration-properties>
</agent-service>
```

4. Save the file.

Linking to DB CAM from AppDynamics

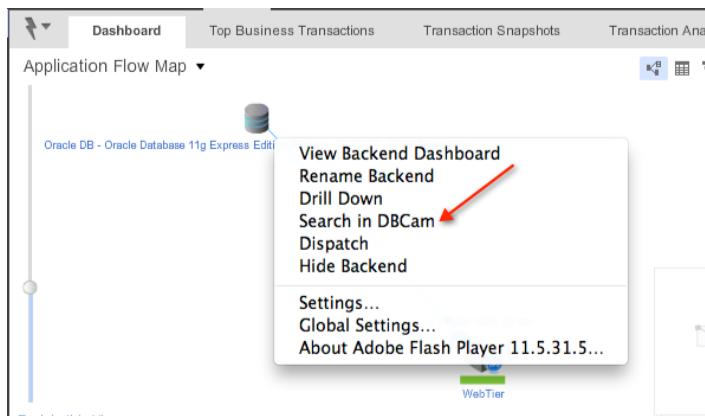
You can link to DB CAM from any AppDynamics flow map that displays a discovered DB CAM-monitored database. The flow map could

be in a dashboard or a transaction snapshot.

If you link to DB CAM from a dashboard you will land in the DB CAM instance dashboard. If you link to be DB CAM from a transaction snapshot, you will land in the DB CAM Session Drill Down screen.

To Link to DB CAM from a Dashboard

1. In the flow map of a dashboard, right-click on the link below a database icon.
2. Click **Search in DBCam**.

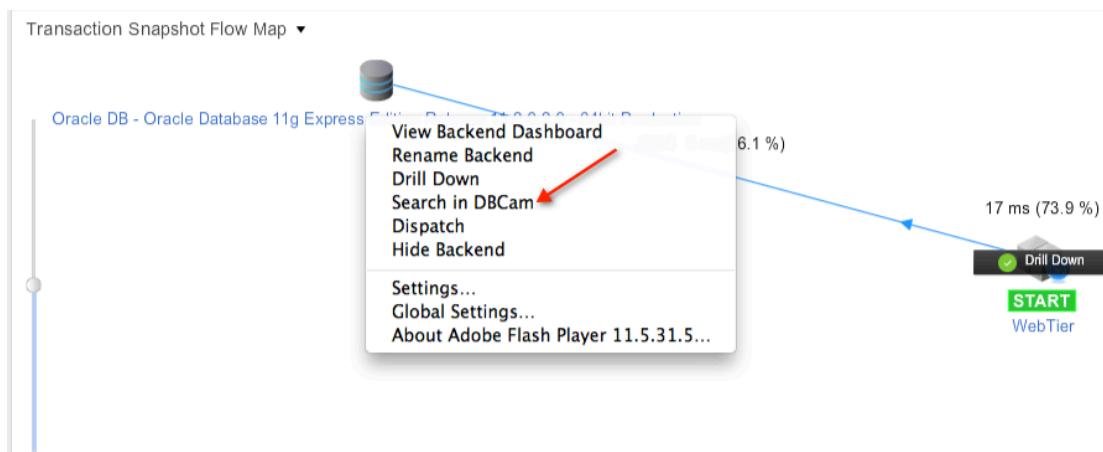


DB CAM launches and displays the instance dashboard for the selected database.



To Link to DB CAM from a Transaction Snapshot

1. In the flow map of a transaction snapshot, right-click on the link below the database icon.
2. Click **Search in DBCam**.



DB CAM launches and displays the session drill down for the selected database.

The screenshot shows the DB CAM application interface. At the top, there's a navigation bar with links for Monitor, DBA, Developer, Analytics, Setup, Help, and Logout. To the right of the navigation is the DB CAM logo. Below the navigation is a toolbar with icons for Session Drill Down, Request, and Session. A search bar labeled "Search for current or historical sessions." is present. Underneath the search bar are input fields for SID, Serial #, and Client ID, with a "Go" button. A table header titled "ASH search" is shown with columns for Inst Id, Sample Time, SID, User, SQL ID, Blocking Session, Event, Program, Module, and Action. There's also a search icon and a refresh icon.

Learn More

- Access the Administration Console
- Dashboards
- Flow Maps
- Transaction Snapshots
- Monitor Databases
- DB CAM

Integrate with Apica

AppDynamics integrates with:

- Apica WebPerformance
- Apica ProxySniffer
- Apica LoadTest

Apica users can drill down from these Apica products into the AppDynamics console to investigate the root cause of performance problems.

For more information about the integration with Apica see [the AppDynamics blog](#) and [the Apica-AppDynamics partner page](#).

AppDynamics for Java

This information covers using AppDynamics for Java applications and environments. For general information see [AppDynamics Essentials](#) and [AppDynamics Features](#).

Tutorials

Monitor Java Applications

Troubleshoot Java Application Problems

Configure AppDynamics for Java

Administer App Agents for Java

Tutorials for Java

This section provides tutorials for tasks in AppDynamics.

Troubleshooting Application Errors

Identifying and troubleshooting errors in your Java application.

Overview Tutorials for Java

Use AppDynamics for the First Time with Java

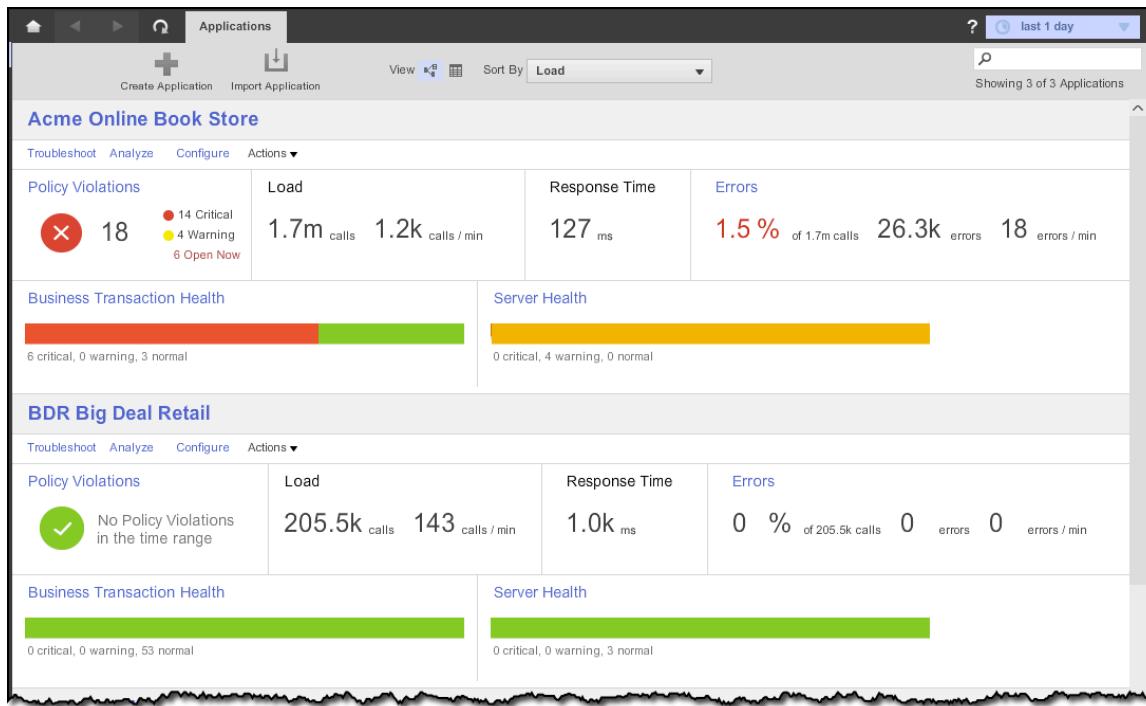
- All Applications Dashboard
- Application Dashboard
 - Time Range
 - Flow Map and KPIs
 - Events
 - Transaction Scorecard
 - Exceptions and Errors
- More Tutorials

This topic assumes that an application is already configured in AppDynamics, and uses the Acme Online application as the example. It also assumes that you have already logged in to AppDynamics.

This topic gives you an overview of how AppDynamics detects actual and potential problems that users may experience in your application - transactions that are slow, stalled or have errors - and helps you easily identify the root causes.

All Applications Dashboard

When you log into the Controller UI you see the All Applications dashboard.



The All Applications dashboard shows high-level performance information about one or more business applications. Load, response time, and errors are standard metrics that AppDynamics calls "key performance indicators" or "KPIs". The others are:

Health Rule Violations and Policies: AppDynamics lets you [define a health rule](#), which consists of a condition or a series of conditions based on metrics exceeding predefined thresholds. You can then use health rules in policies to automate optional remedial actions to take if the conditions exist. AppDynamics also provides default health rules to help you get started.

Business Transaction Health: The health indicators are a visual summary of the extent to which a business transaction is experiencing critical and warning health rule violations. See the [slow transactions tutorial](#).

Server Health: Additional visual indicators that track how well the server infrastructure is performing. See the [server health tutorial](#).

Application Dashboard

Click an application to monitor, one that has some traffic running through it. The Application dashboard gives you a view of how well the application is performing.



You see the dashboard for your application. The flow map on the left gives you an overview of your servers (application servers, databases, remote servers such as message queues, etc.) and metrics for the calls between them. Click, hold and move the icons around to arrange the flow map. Use the scale slider and mini-map to change the view.

Time Range

From the time range drop-down in the upper-right corner select the time range over which to monitor - the last 15 minutes, the last couple of hours, the last couple of days or weeks. Try a few different time ranges and see how the dashboard data changes.

Flow Map and KPIs

In the flow map, click any of the blue lines to see more detail on the aggregated key performance metrics (load, average response time and errors) between the two servers. See the [flow maps tutorial](#).

The graphs at the bottom of the dashboard show the key performance indicators over the selected time range for the entire application.

Events

An event represents a change in application state. The Events pane lists the important events occurring in the application environment. See the [events tutorial](#).

Transaction Scorecard

The Transaction Scorecard panel shows metrics about business transactions within the specified time range, covering the percentage of instances that are normal, slow, very slow, stalled or have errors. Slow and very slow transactions have completed. Stalled transactions never completed or timed out. Configurable thresholds define the level of performance for the slow, very slow and stalled categories. See the [Transaction Scorecard tutorial](#).

Exceptions and Errors

An exception is a code-logged message outside the context of a business transaction. An error is a departure from the expected behavior of a business transaction, which prevents the transaction from working properly. See the [exceptions tutorial](#).

More Tutorials

Monitoring Tutorials for Java

Tutorial for Java - Events

- Monitoring Events
- Filtering Events

Monitoring Events

1. From an application, tier or node dashboard, look at the Events panel:

The screenshot shows the AppDynamics Events panel. On the left, there's a sidebar with navigation links like MyApp, Business Transactions, Servers, Events, Troubleshoot, Errors, Policy Violations, Analyze, Configure, and Policies. A callout bubble points to the 'Events' link in the sidebar with the text 'Click here to get to the Events Panel'. The main area has tabs for 'Events', 'Tiers', 'View Event Details', 'Delete', and 'Archive'. Below these are buttons for 'Register Application Change Event', 'Clear Criteria', and 'Search'. A status bar at the top right says 'Viewing 8 events' and 'last 30 minutes'. The main content area displays a table of events with columns: Type, Summary, Time, Business Transaction, Tier, and Node. The table contains several entries, mostly 'Code Deadlock' events, with some 'Slow Requests - Very Slow' entries. A large green oval highlights the table area, and a callout bubble below it explains: 'The Events Panel shows the type of event, a synopsis, the timestamp when the event occurred, what business transaction the event is associated with (if any), the source of the event by tier and node.'

Type	Summary	Time	Business Transaction	Tier	Node
Code Deadlock	JVM deadlock det...	09/24/12 9:44:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:39:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:34:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:29:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:24:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:19:36 AM		2Tier	node2
Slow Requests - Very Slow	/processor/elect...	09/24/12 9:17:08 AM	/processor/elect...	1Tier	node1
Slow Requests - Very Slow	/product/outdoorM...	09/24/12 9:17:08 AM	/product/outdoor	1Tier	node1

2. The Events panel shows all the events that are monitored by AppDynamics. There are several types of events:

Health Rule Violation Events include business transaction health rule events such as average response time, and server health events such as Java VM Heap Utilization Thresholds. To change what triggers a health rule event, see [Health Rules](#).

Slow Transaction Events fire when slow, very slow or stalled transactions occur. See [Configure Thresholds](#).

Error Events trigger when application exceptions are thrown or HTTP Errors are returned. See [Configure Error Detection](#).

Code Problem Events throw when a code deadlock is detected or a resource pool is completely utilized. For example this event fires when a JDBC connection pool is completely utilized, or a java.lang.Thread is deadlocked.

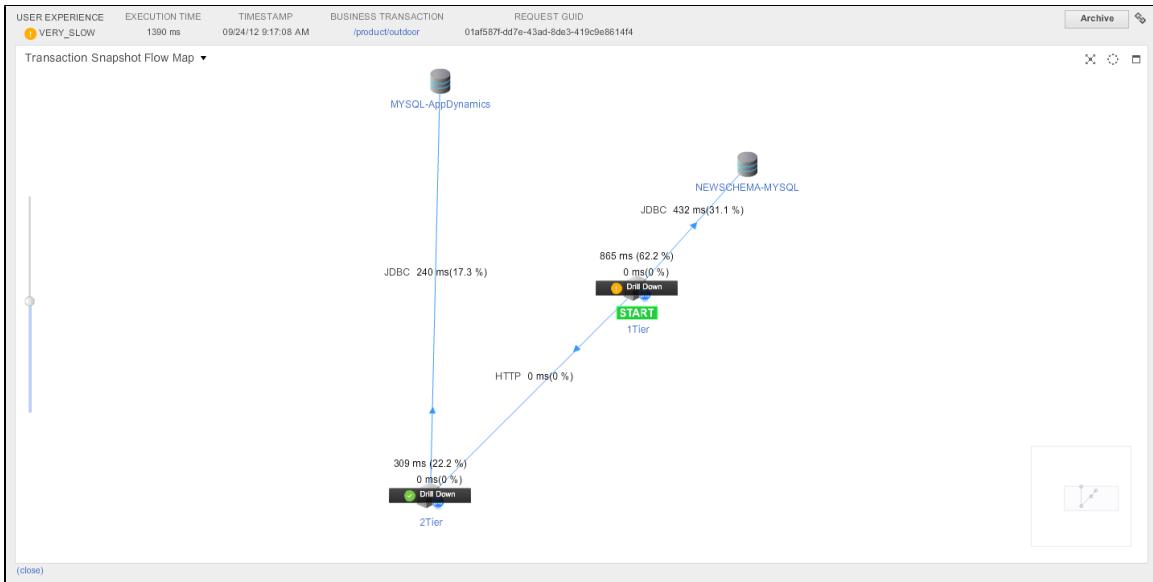
Application Change Events fire when administrative changes are made to an application tier. For example, this event is triggered when an application server instance is restarted, or when a Java VM option is modified on an application server. See [Monitor Application Change Events](#).

AppDynamics Config Warnings occur when the AppDynamics infrastructure needs attention. For example, when the Controller detects low disk space conditions on its host this event type will fire. See [AppDynamics Administration](#).

AppDynamics Internal Diagnostics help troubleshoot AppDynamics errors when working with AppDynamics Support. See [AppDynamics Support](#).

Custom events are user-defined events. See [Events](#).

To get details click on the event in the list. For example, click on a slow request event to see the details of the request in the transaction flow map so you can start troubleshooting the slow request.



Filtering Events

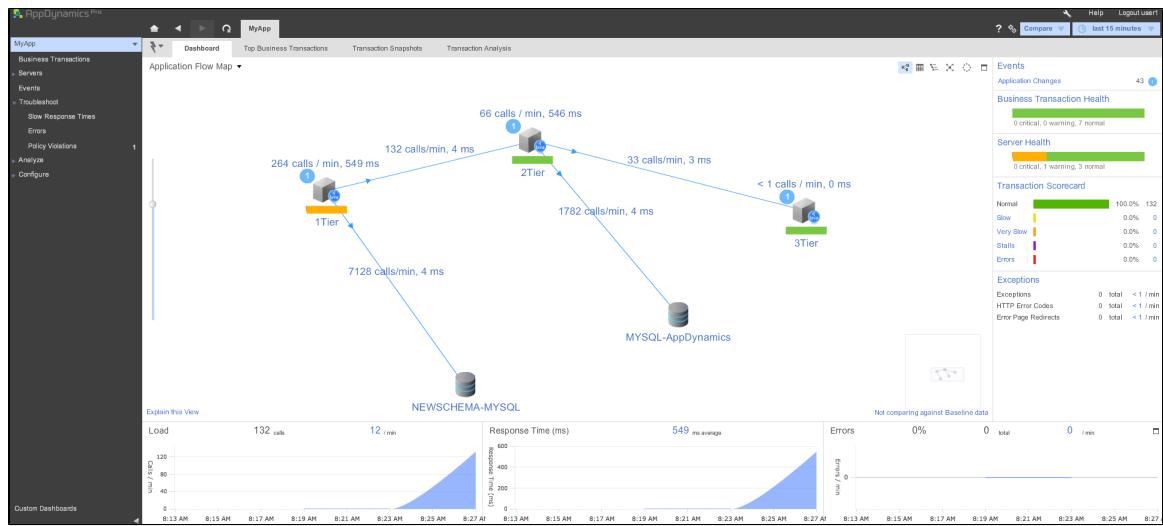
You can filter events by type in the Events panel. Click on the type of event you want to see and click search. For example, to see only Deadlocks and Resource Pool Exhaustion events select the Code Problem Event and click Search.

The screenshot shows the Events panel in the AppDynamics interface. On the left, a sidebar navigation includes 'Business Transactions', 'Events' (selected), 'Troubleshoot', 'Analyze', 'Configure', and 'Custom Dashboards'. The main area has tabs for 'Events' (selected) and 'View Event Details'. It includes 'Hide Filters', 'Search', and 'Register Application Change Event' buttons. A 'Clear Criteria' button is highlighted with a red arrow. The 'FILTER BY EVENT TYPE' section contains several checkboxes, with 'Code Problems' checked and highlighted with a red arrow. The 'FILTER BY OBJECT' section lists 'Business Transactions' and 'Tiers / Nodes'. To the right, a table titled 'viewing 6 events' displays event details. A green callout bubble points to the table with the text 'The list just shows selected event types'. The table columns are Type, Summary, Time, Business Transaction, Tier, and Node. The data rows show various code deadlock events:

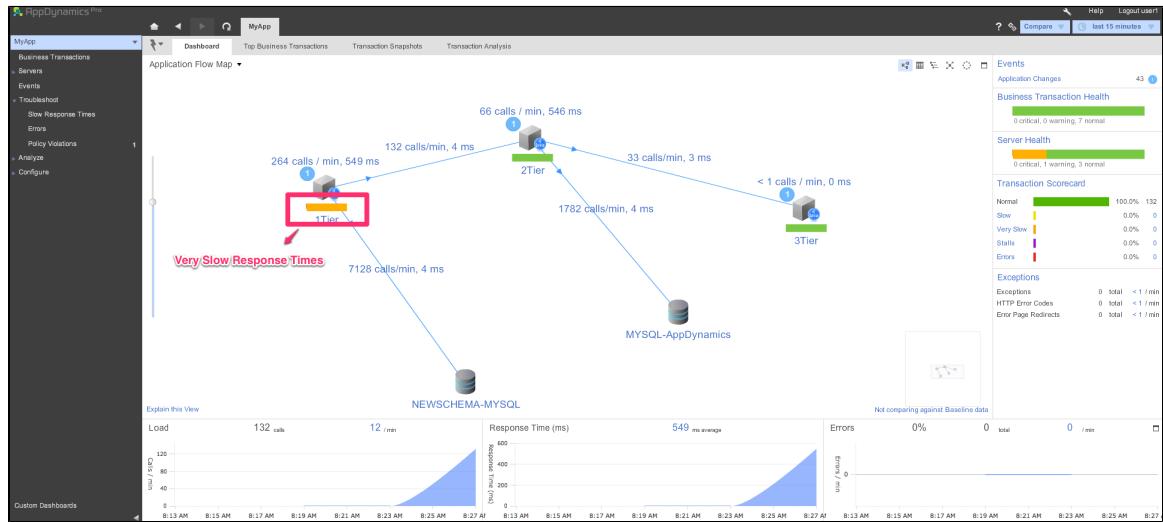
Type	Summary	Time	Business Transaction	Tier	Node
Code Deadlock	JVM deadlock det...	09/24/12 9:54:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:49:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:44:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:39:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:34:36 AM		2Tier	node2
Code Deadlock	JVM deadlock det...	09/24/12 9:29:36 AM		2Tier	node2

Tutorial for Java - Flow Maps

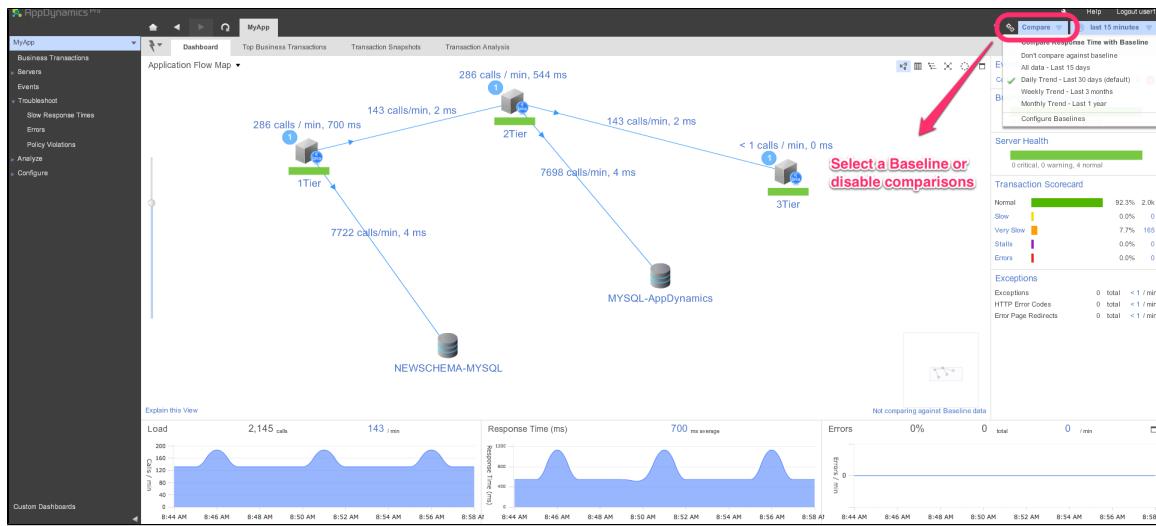
Flow maps show the health of your application, all tiers, and the communication between the tiers. It includes summary indicators such as call rates and error rates:



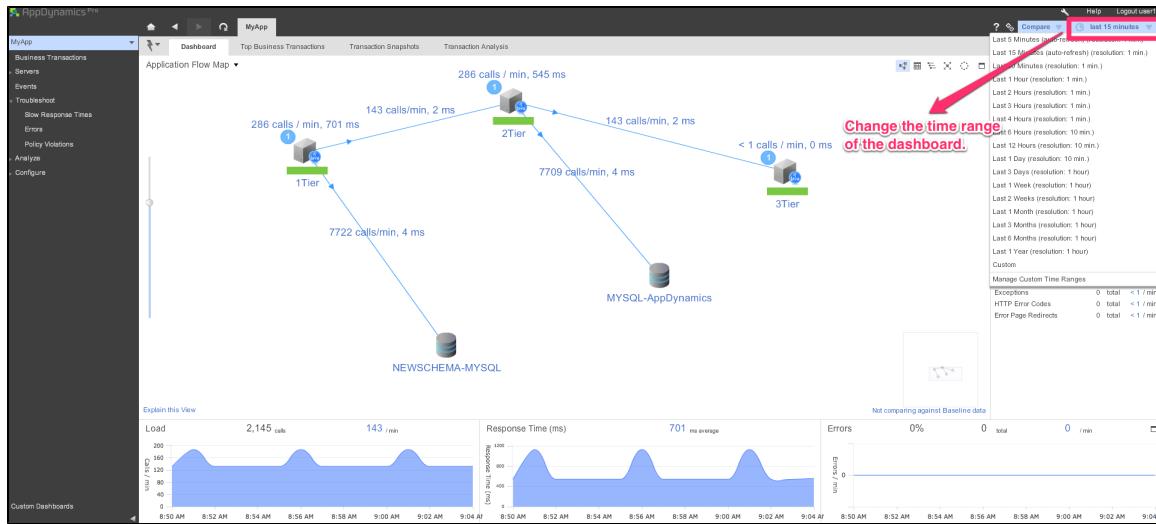
Visual indicators quickly show you problem tiers and healthy tiers. Below you can see tier 1 is experiencing very slow response times, while tier 2 and tier 3 are healthy:



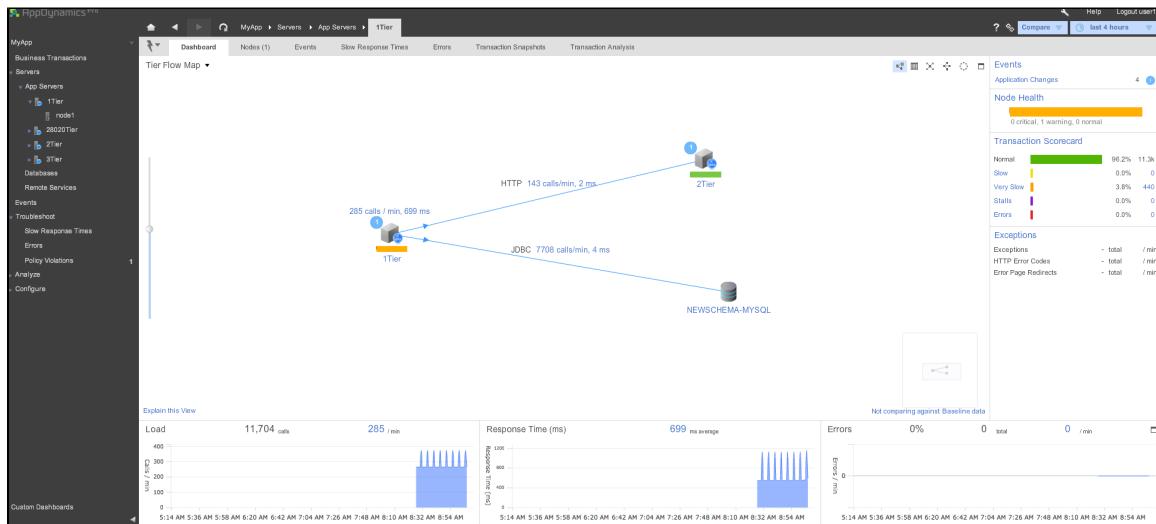
By default, the flow map computes tier health by comparing the state of the tier averaged over the last 15 minutes against the daily trend (the 30 day rolling average). You can change the time window for baseline comparison using the time window pull down menu. You can also disable baseline comparisons:



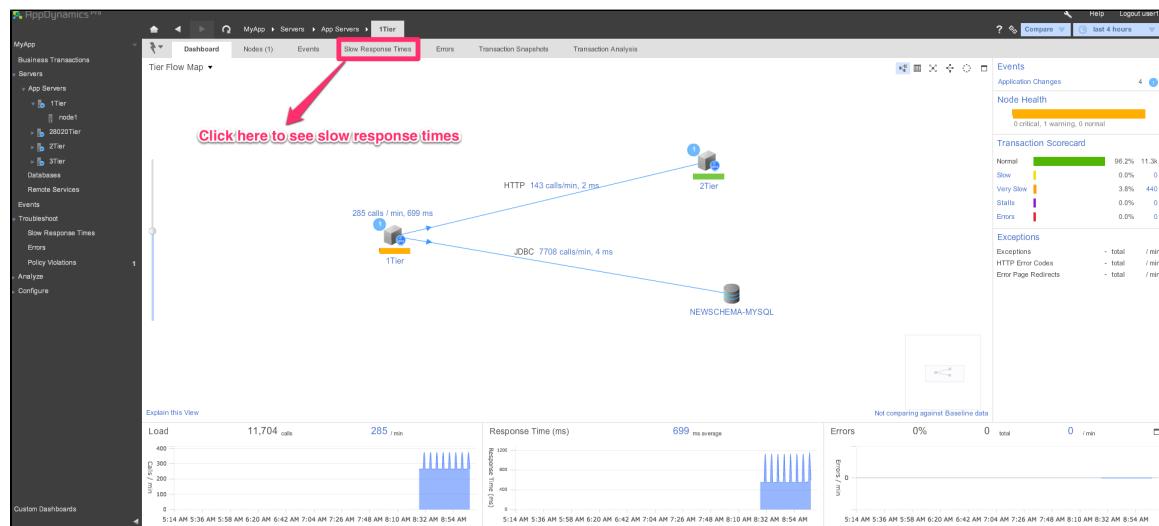
You can change the time range displayed in the flow map by changing the time window using the time window pull down menu. Changing the time range effects the entire dashboard:



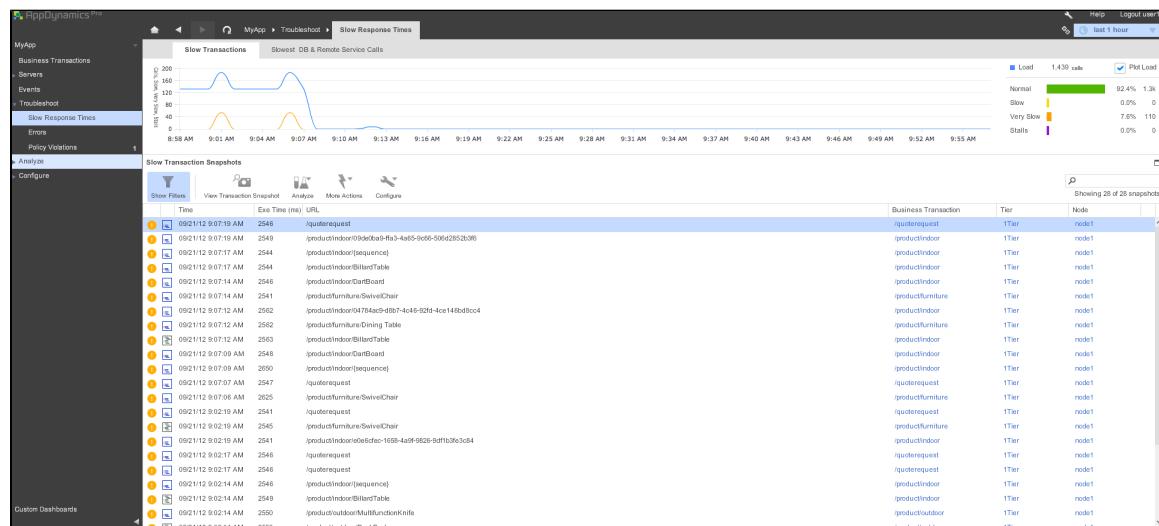
You can troubleshoot a problem system call by clicking on a tier's name to drill down into a subset of the system involving the tier:



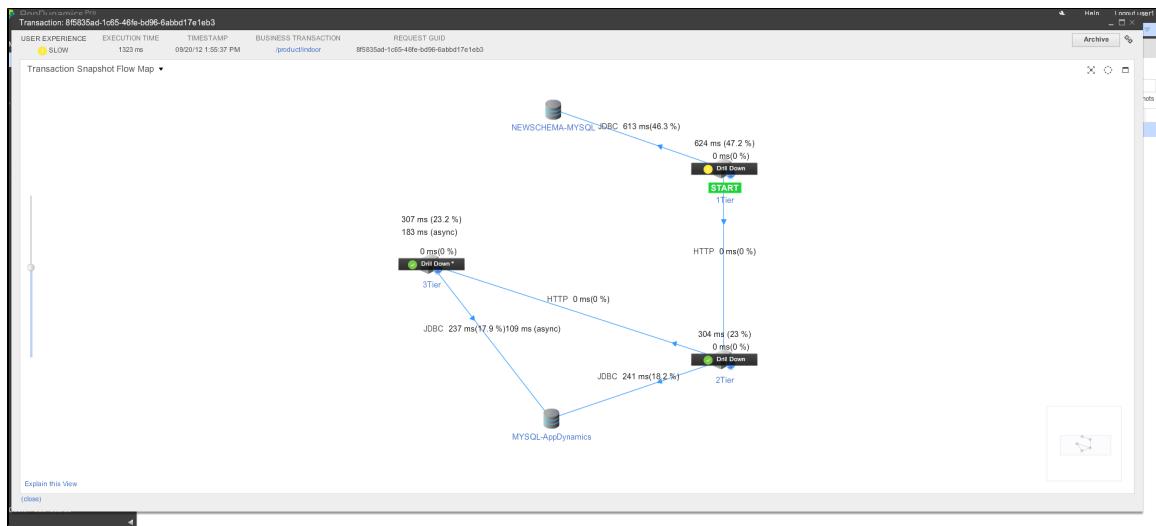
To view the slow response times in detail click on the slow response time menu:



From the slow response time pick a transaction to see a snapshot of the slow transaction:



From the transaction snapshot you can troubleshoot the slow transaction:



Tutorial for Java - Server Health



Node health is driven by node health policies. For example, node 1 is experiencing a JVM heap health rule violation --all of the heap is consumed.

Out of the box, AppDynamics provides default policies for CPU utilization, physical memory utilization, JVM heap utilization, and CLR heap utilization. For example the default health rule for CPU utilization triggers a warning when a node exceeds 75% CPU utilization and a critical event fires when CPU utilization is 90% or above.

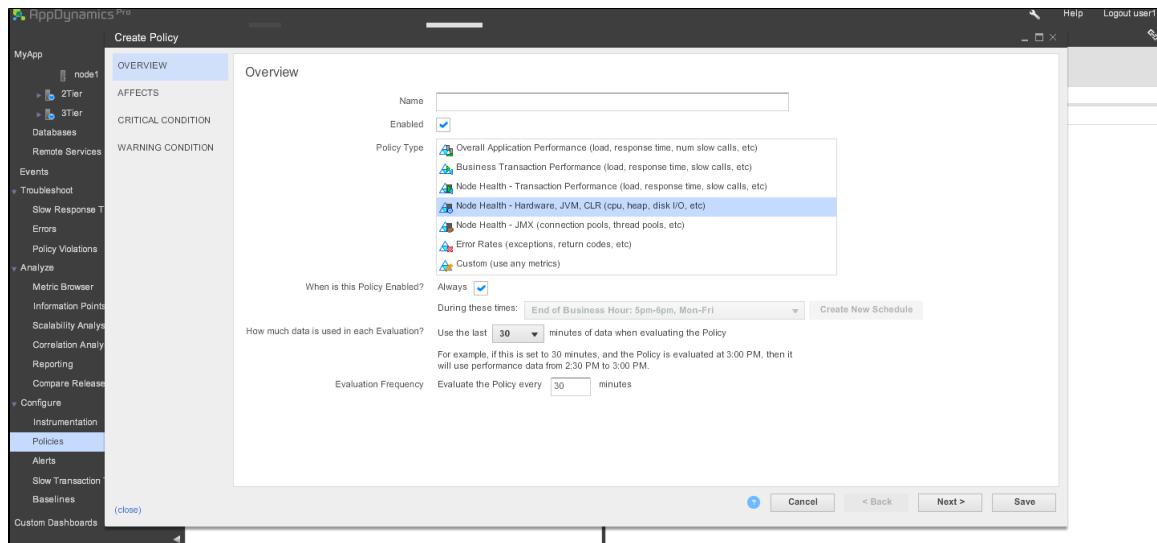
The screenshot shows the AppDynamics Pro interface for the application 'MyApp'. The left sidebar navigation includes 'Business Transactions', 'Servers' (selected), 'Databases', 'Remote Services', 'Events', 'Troubleshoot', 'Slow Response Times', 'Errors', 'Policy Violations' (selected), 'Analyze', and 'Configure'. The main content area is titled 'App Servers' and displays three tiers: 1Tier, 2Tier, and 3Tier. Each tier has a summary row and individual node details. Node 1 in the 1Tier tier is highlighted with a red status bar, while other nodes in all tiers have green status bars. The 'Memory' tab is selected, showing metrics like JVM % Heap, Max Heap, JVM CPU Burnt (ms), GC Time Spent (ms), Major Collections, Minor Collections, and Minor Col time per min.

You can view the health rule violation details and the status of the violation:

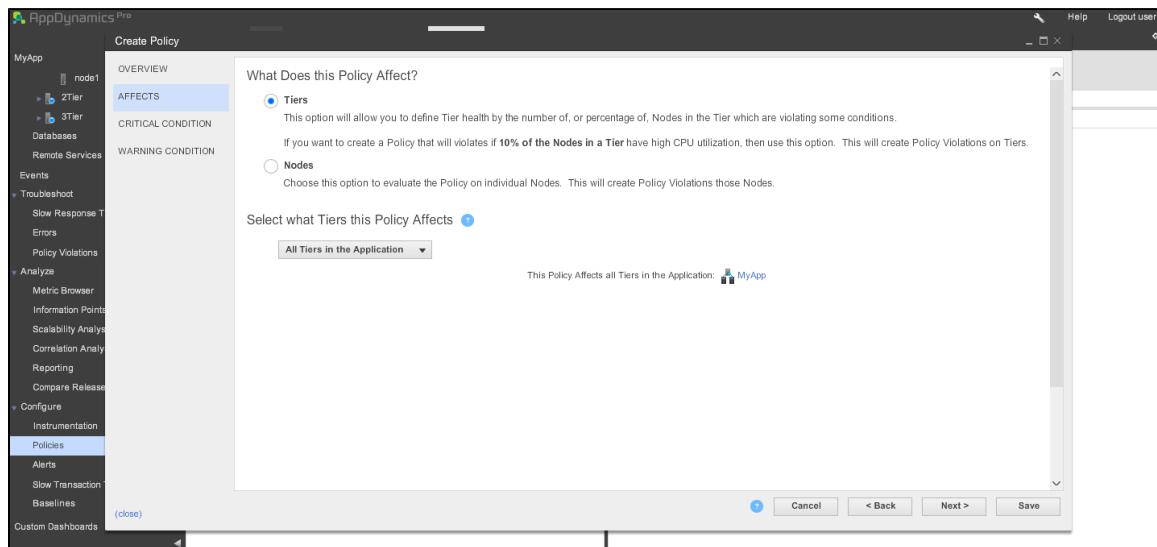
The screenshot shows the 'Policy Violations' section of the AppDynamics Pro interface. The left sidebar navigation includes 'Policy Violations' (selected). The main content area shows a single policy violation for 'node1' in the '1Tier' tier. The violation details are as follows:

- Status:** Resolved
- Policy:** JVM Heap utilization is too high
- Description:** Evaluation Entity: node1 Node
Violated Condition: JVM[Memory:Heap]Used % Condition
Violation Period: 30 min(s) between 09/20/2012 01:52:55 PM and 09/20/2012 02:22:55 PM
Observed Value: 97
Violation: Observed value (97) greater than threshold (90)
- View:** View button
- Configuration:** Configure Policies button

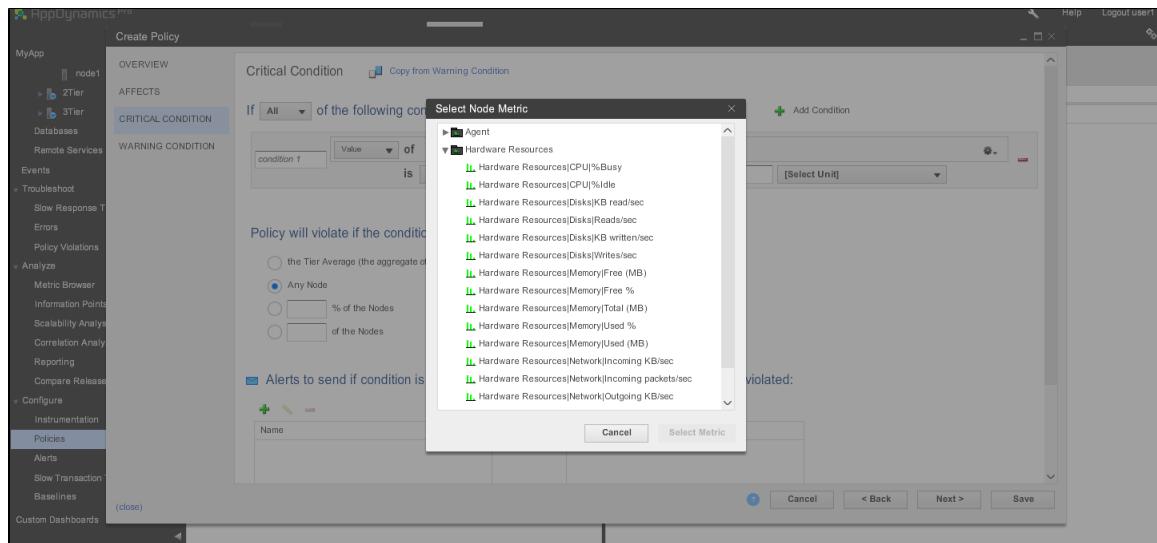
To change or add a new health rule see [Configure Health Rules](#).



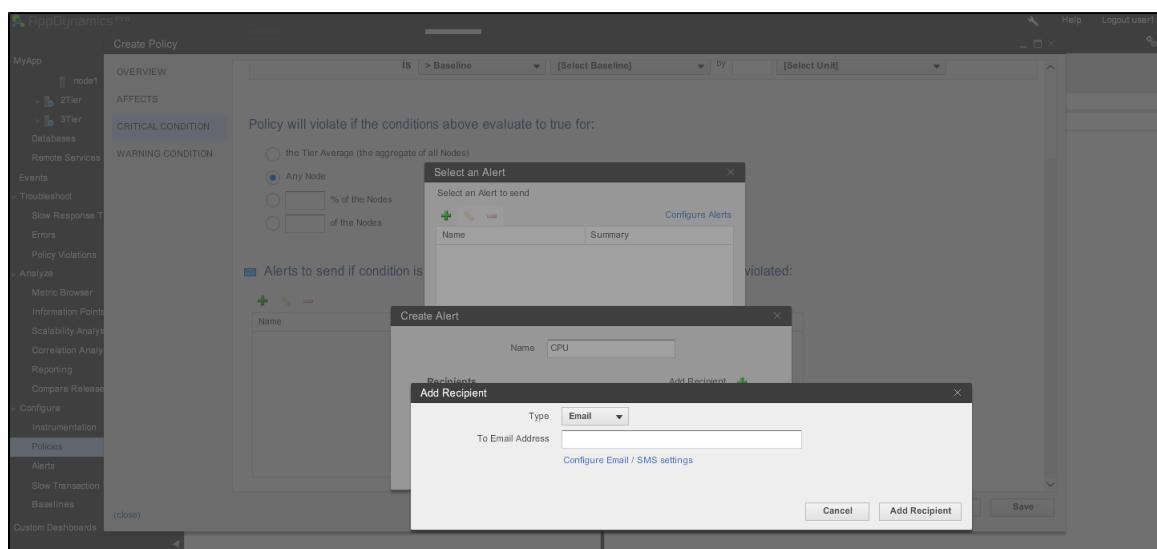
You can control the scope of a health rule. You can choose all nodes, or if you have a large cluster you may want this health rule to apply to a percentage of the nodes in a tier.



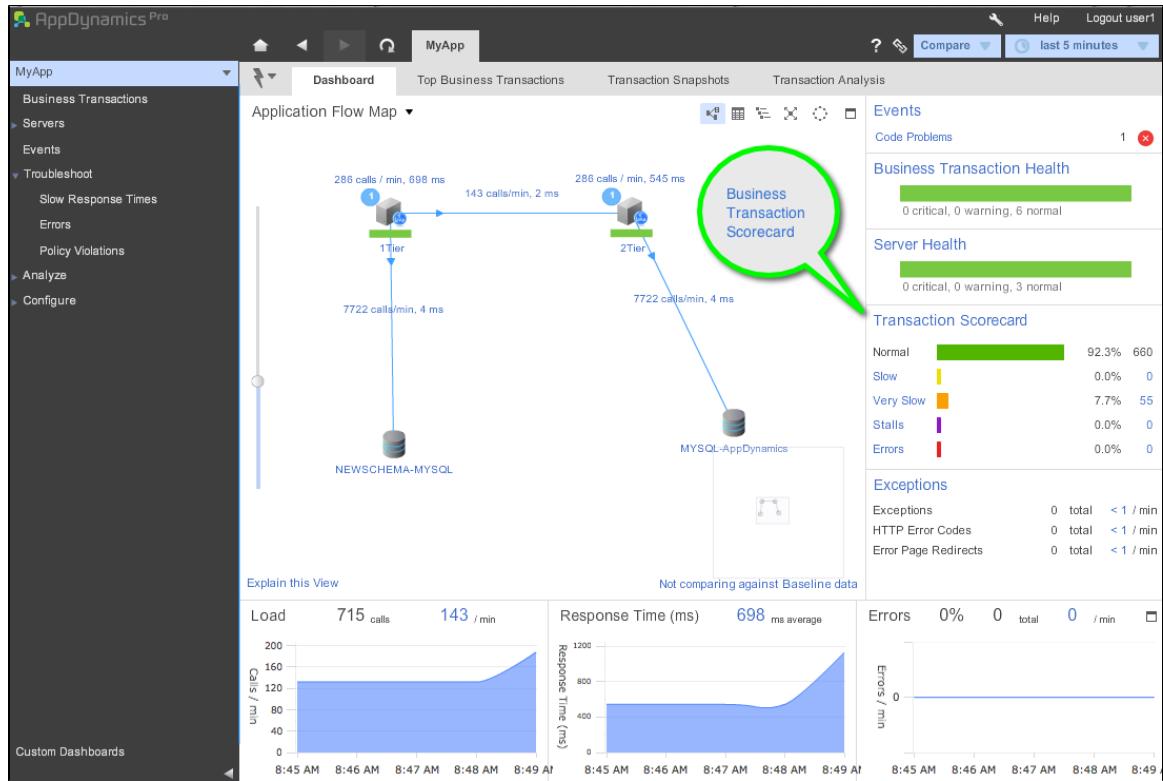
If you do not see a metric provided out of the box, you can [Add Metrics Using Custom Monitors](#) or [create a JMX metric from MBeans](#).



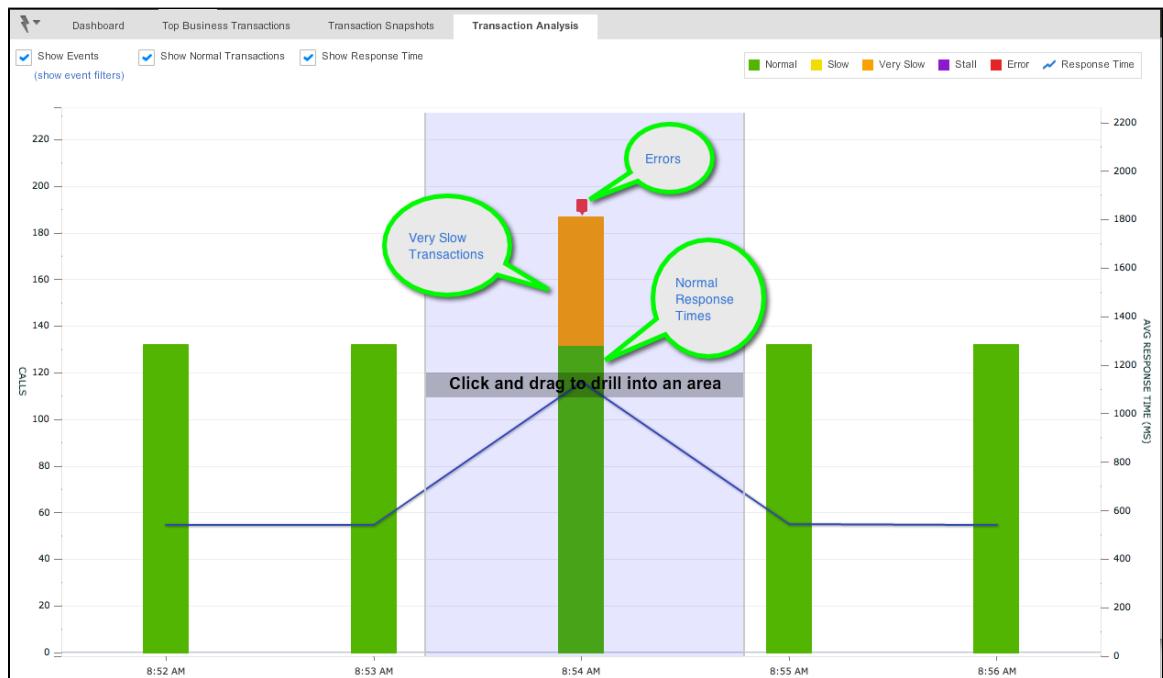
Health rule violations can be used in policies to trigger alerts that can notify Administrators via email or SMS systems.



Tutorial for Java - Transaction Scorecards



Transactions are categorized as Normal, Slow, Very Slow, Stalled, or Errors, which are determined by thresholds and the AppDynamics error detection subsystem. Thresholds can be static or dynamic; dynamic thresholds are based on historical data. The Transaction Analysis Histogram shows the distribution over time.



Default Transaction Baselines can be viewed here:

AppDynamics Pro

MyApp > Configure > Slow Transaction Thresholds

User Transaction Thresholds

This section lets you configure Slow Transaction Thresholds, and when to trigger Diagnostic Sessions.

Slow Transactions Thresholds

Every Transaction that is processed by the Application will be categorized as normal, slow, very slow, or stalled based on these thresholds.

Slow Transaction Threshold

- More than % slower than the average of the last hours
- Greater than Milliseconds
- Greater than **3** Standard Deviations for the last hours

Very Slow Transaction Threshold

- More than % slower than the average of the last hours
- Greater than Milliseconds
- Greater than **4** Standard Deviations for the last hours

Stall Threshold

- Disable Stall detection
- Stall occurs when a transaction takes more than **45** seconds.
- Stall occurs when a transaction's response time is deviations above the average for the last 2 hours.

Apply to all Existing Business Transactions

Diagnostic Session Settings

Diagnostic Sessions are started to capture detailed Transaction Snapshots including full call graphs. This section configures when Diagnostic Sessions are started and how they run.

Configure thresholds for when Diagnostic Sessions will be started

Diagnostic sessions are started after a series of slow or error Transactions.

Start Diagnostic Session if more than **10** % of the requests in a minute are slower than the Slow Transaction Threshold (configured above).

Custom Dashboards

Refresh Tree

If you want to change the thresholds for all or individual transactions see the transaction threshold policy configurations (see below). See [Thresholds](#).

Set Individual Business Transaction Thresholds - /http/to3d

You can configure Slow Transaction Thresholds, and when to trigger Diagnostic Sessions.

Slow Transactions Thresholds

Every Transaction that is processed by the Application will be categorized as normal, slow, very slow, or stalled based on these thresholds.

Slow Transaction Threshold

- More than % slower than the average of the last hours
- Greater than Milliseconds
- Greater than **3** Standard Deviations for the last hours

Very Slow Transaction Threshold

- More than % slower than the average of the last hours
- Greater than Milliseconds
- Greater than **4** Standard Deviations for the last hours

Stall Threshold

- Disable Stall detection
- Stall occurs when a transaction takes more than **45** seconds.
- Stall occurs when a transaction's response time is deviations above the average for the last 2 hours.

Diagnostic Session Settings

Diagnostic Sessions are started to capture detailed Transaction Snapshots including full call graphs. This section configures when Diagnostic Sessions are started and how they run.

Configure thresholds for when Diagnostic Sessions will be started

Diagnostic sessions are started after a series of slow or error Transactions.

Refresh Tree

To troubleshoot slow transactions, see [Tutorial for Java - Slow Transactions](#) and [Troubleshoot Slow Response Times](#).

By Default, errors are determined when HTTP Error Codes are returned and by default AppDynamics instruments Java error and warning methods such as logger.warn and logger.error. AppDynamics captures the exception stack trace and automatically correlates it with the request. To learn how to change this, for example to turn down the "noise" or add redirect error pages, see [Configure Error Detection](#).

Troubleshooting Tutorials for Java

Super-Simple Java Troubleshooting using Events

- Super-Simple Troubleshooting using the Events List
 - How to Set up the Events List
 - How to Know Something is Not Quite Right
 - How to Investigate
 - Drill down to an Error
 - Drill down to a stalled Business Transaction
 - Drill down to a slow or very slow Business Transactions
 - Drill down to an Application Server Exception
 - Drill down to a Code Deadlock
 - Drill down to Application Change Events

Super-Simple Troubleshooting using the Events List

How to Set up the Events List

1. From an application dashboard, click Events (either the menu item or the Events pane label).
2. In the Events window, use the filter criteria to pick which events you want to monitor. Click Search.
3. Set the time range.
4. Watch and look.

How to Know Something is Not Quite Right

You see:

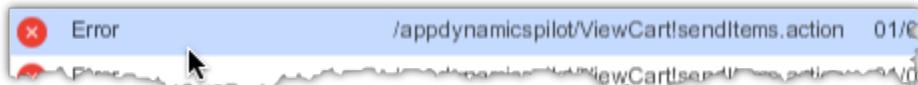
Red (critical, policy violation)
Purple (warning, stall)
Orange (warning, very slow)
Yellow (warning, slow)

How to Investigate

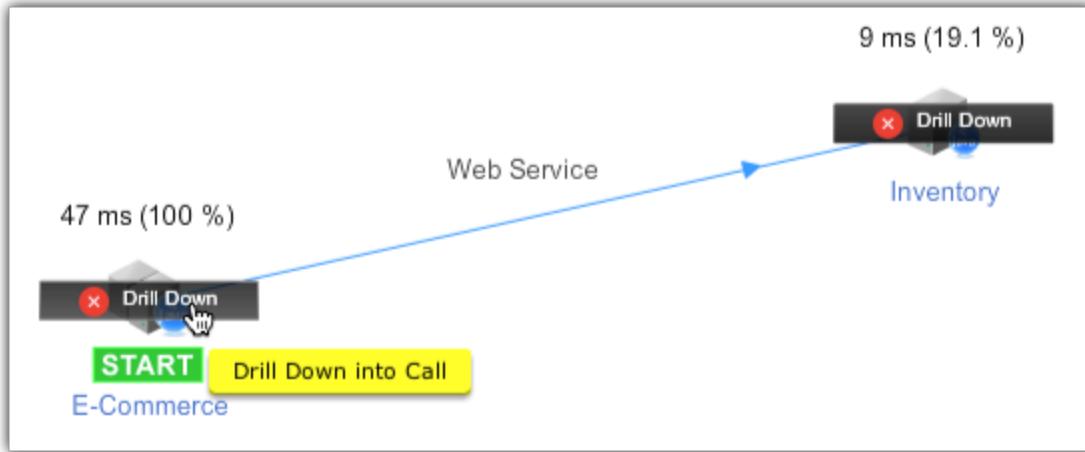
You drill down to the root cause of the problem in different ways depending on the type of event.

Drill down to an Error

1. In the Events window click an Error.



2. In the Transaction Flow Map click the Drill Down icon. If there are multiple drill down icons, select the one with the transaction that takes the most time.



3. In the Call Drill Down window click the Summary tab.

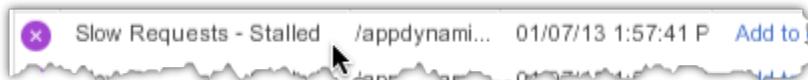
Call Drill Down. Exe Time: 47 ms Timestamp: 01/07/13 1:58:32 PM BT: Checkout GUID: 0cdcf690-e6a1-4c4d-8d2f-e53b69

SUMMARY	Value
User Experience	ERROR
Execution Time	47 ms
CPU Time	0 ms 0 %
Transaction Timestamp	01/07/13 1:58:32 PM (server) 01/07/13 1:58:32 PM (agent)
Summary	[Error] - com.appdynamicspilot.webserviceclient.SoapUtils::There was an error invoke service method createOrder http://localhost:8002/cart/services/OrderService?wsdl#com.appdynamicspilot.webserviceclient.OrderService createOrder -
Error	Exception Message: Exception occurred while trying to invoke service method createOrder
Tier	E-Commerce
Node	E-Commerce-Node-8000
Business Transaction	Checkout
URL	/appdynamicspilot/ViewCart!sendItems.action
Session ID	6F9E4F18355CB2B4958E27CBF5A87DE0
User Principal	No User Principal
Process ID	7366
Thread Name	http-8000-Processor19
Thread ID	46
Transaction Thresholds	Slow: 350 ms. Very Slow: 700 ms. Configure
Request GUID	0cdcf690-e6a1-4c4d-8d2f-e53b693d0556

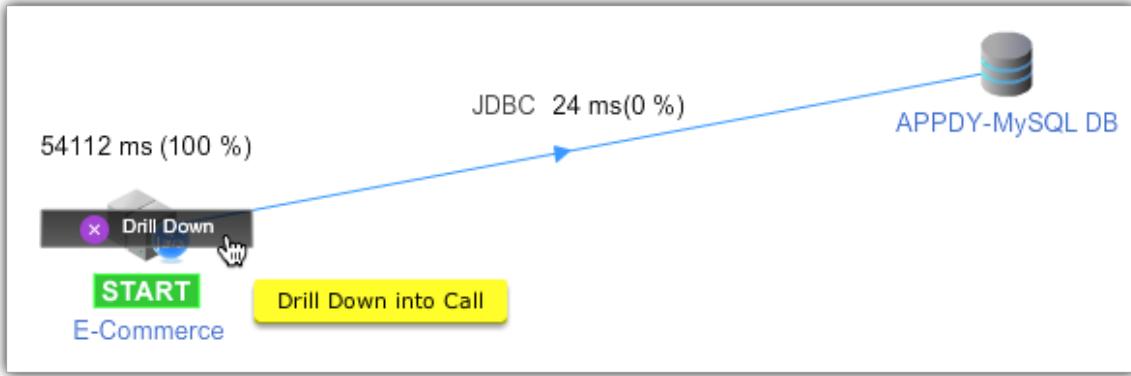
4. Use the Export to PDF button to save and email the information to your colleagues.

Drill down to a stalled Business Transaction

1. In the Events window click a Slow Requests - Stalled row.



2. In the Transaction Flow Map click the Drill Down icon.



3. In the Call Drill Down window click the Summary tab.

User Experience: STALL
 Execution Time: 54136 ms
 CPU Time: 0 ms 0 %
 Transaction Timestamp: 01/07/13 1:57:41 PM (server) 01/07/13 1:57:41 PM (agent)
 Summary: Request took higher than the stall threshold of [45000] ms -
 Tier: E-Commerce
 Node: E-Commerce-Node-8000
 Business Transaction: Add to cart
 Stack Dump:
 Thread Name:http-8000-Processor24
 ID:51
 Time:Mon Jan 07 21:58:26 UTC 2013
 State:TIMED_WAITING
 Priority:5
 java.lang.Thread.sleep(Native Method)
 com.appdynamicspilot.action.CartAction.addToCart(CartAction.java:109)
 sun.reflect.GeneratedMethodAccessor163.invoke(Unknown Source)
 sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
 java.lang.reflect.Method.invoke(Method.java:597)
 com.opensymphony.xwork2.DefaultActionInvocation.invokeAction(DefaultActionInvocation.java:40)
 com.opensymphony.xwork2.DefaultActionInvocation.invokeActionOnly(DefaultActionInvocation.java:229)
 com.opensymphony.xwork2.interceptor.DefaultWorkflowInterceptor.doIntercept(DefaultWorkflowInterceptor.java:59)
 com.opensymphony.xwork2.interceptor.MethodFilterInterceptor.intercept(MethodFilterInterceptor.java:91)

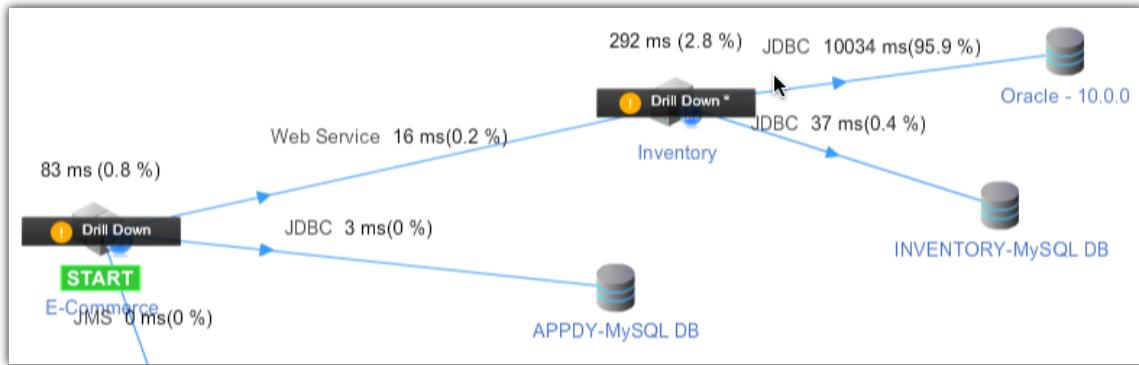
4. Use the Export to PDF button to save and email the information to your colleagues.

Drill down to a slow or very slow Business Transactions

1. In the Events window click a Slow Requests - Very Slow or Slow row.

Slow Requests - Very Slow /appdynamicspilot/ViewCart!sendItems.action
 Slow Requests - Very Slow /appdynamicspilot/ViewCart!sendItems.action - 0

2. In the Transaction Flow Map click the Drill Down icon. If there are multiple drill down icons, select the one with the transaction that takes the most time.



3. In the Select a Call windows click the slowest call.

Select a Call to Drill Down into

Multiple calls were made to this Tier as part of this Transaction.

	Exe Time (ms)	Summary	Exit Calls
!	10032 ms	[Web Service] call from E-Commerce	7 JDBC calls (10003 ms. max, 1429.0 ms. avg.)
✓	299 ms	[Web Service] call from E-Commerce	7 JDBC calls (17 ms. max, 2.4 ms. avg.)
✓	32 ms	[Web Service] call from E-Commerce	7 JDBC calls (14 ms. max, 2.0 ms. avg.)

4. In the Call Drill Down window click the Hot Spots tab to see the slowest methods.

This screen displays all of the method calls in the call graph sorted by time

Name	Method Time (ms)	External Calls
com.appdynamics.jdbc.MPreparedStatement:executeQuery:45	10005 ms (self)	99.7 % JDBC
Spring Bean - transactionManager:doCommit:578	23 ms (self)	0.2 % JDBC

Invocation Trace

```

AxisServlet.doPost:unknown (3ms self time, 10031 ms total time)
AbstractInOutSyncMessageReceiver.receive:39 (0ms self time, 10028 ms total time)
RPCMessageReceiver.invokeBusinessLogic:116 (0ms self time, 10028 ms total time)
OrderWebservices.createOrder:16 (0ms self time, 10028 ms total time)
OrderService$$EnhancerByCGLIB$$1ee8c32e.createOrder:unknown (0ms self time, 10028 ms total time)
OrderService$$FastClassByCGLIB$$e49d675f.invoke:unknown (0ms self time, 10005 ms total time)
OrderService.createOrder:22 (0ms self time, 10005 ms total time)
OrderDaoImpl.createOrder:33 (0ms self time, 10005 ms total time)
QueryExecutor.executeSimplePS:61 (0ms self time, 10005 ms total time)
MPreparedStatement.executeQuery:45 (10005ms self time, 10005 ms total time)
    
```

5. In this example, since the slow call is a database call you can click the SQL Calls tab to see the slowest SQL.

Query Ty	Query	Avg. Time	Count
Insert	Insert into OrderRequest (item_id, notes) values (?, ?)	10003	1
COMMIT	DB Transaction Commit	22	1

6. Use the Export to PDF button to save and email the information to your colleagues.

Drill down to an Application Server Exception

In the Events window click an Application Server Exception.



In the Application Server Exception window click the Details tab.

The screenshot shows the 'Application Server Exception' details window. The 'Details' tab is selected. The content area displays a stack trace starting with 'ConfigurationException: There is no Action mapped for namespace / and action name addToCart. - [unknown location] at com.opensymphony.xwork2.DefaultActionProxy.prepare(DefaultActionProxy.java:186) at org.apache.struts2.impl.StrutsActionProxyFactory.createActionProxy(StrutsActionProxyFactory.java:41)' and a note 'Error capture limit has been reached, this stack trace is truncated.'

Use the Copy to Clipboard button to save and email the information to your colleagues.

Drill down to a Code Deadlock

1. In the Events window click a Code Deadlock.



2. In the Code Deadlock window click the Details tab.

Code Deadlock

Summary Details Comments (0)

Copy to Clipboard

pool-1-thread-1 Name[pool-1-thread-1]Thread ID[64]
Deadlocked on Lock[java.lang.Object@35f626a6] held by thread [pool-1-thread-2] Thread ID[65]
Thread stack [
 com.appdynamicspilot.action.DeadLockAction.lock12(DeadLockAction.java:63)
 com.appdynamicspilot.action.DeadLockAction.access\$000(DeadLockAction.java:8)
 com.appdynamicspilot.action.DeadLockAction\$1.call(DeadLockAction.java:29)
 java.util.concurrent.FutureTask\$Sync.innerRun(FutureTask.java:303)
 java.util.concurrent.FutureTask.run(FutureTask.java:138)
 java.util.concurrent.ThreadPoolExecutor\$Worker.runTask(ThreadPoolExecutor.java:886)
 java.util.concurrent.ThreadPoolExecutor\$Worker.run(ThreadPoolExecutor.java:908)
 java.lang.Thread.run(Thread.java:662)
]

pool-1-thread-2 Name[pool-1-thread-2]Thread ID[65]
Deadlocked on Lock[java.lang.Object@2eb10475] held by thread [pool-1-thread-1] Thread ID[64]
Thread stack [
 com.appdynamicspilot.action.DeadLockAction.lock21(DeadLockAction.java:72)
 com.appdynamicspilot.action.DeadLockAction.access\$100(DeadLockAction.java:8)
 com.appdynamicspilot.action.DeadLockAction\$2.call(DeadLockAction.java:36)
 java.util.concurrent.FutureTask\$Sync.innerRun(FutureTask.java:303)
 java.util.concurrent.FutureTask.run(FutureTask.java:138)
 java.util.concurrent.ThreadPoolExecutor\$Worker.runTask(ThreadPoolExecutor.java:886)
 java.util.concurrent.ThreadPoolExecutor\$Worker.run(ThreadPoolExecutor.java:908)
 java.lang.Thread.run(Thread.java:662)
]

3. Use the Copy to Clipboard button to save and email the information to your colleagues.

Drill down to Application Change Events

By default AppDynamics reports events when applications are deployed, app servers restarted, and configuration parameters changed. Since these are not problems, they are indicated by a blue icon.

1 App Server Restart Application Server JVM was re-started Node: Inv... 01

1 Application Configuration Change Application Server environment variables changed 01

1 Application Configuration Change Application Server VM system properties changed 01

Click a change event to see a summary and details, for example:

Application Configuration Change

Summary Details Comments (0)

Copy to Clipboard

Modified Variable 1: ACTION=start (changed to) ACTION=stop
 Modified Variable 2: _EXECJAVA=start "Tomcat" "C:\Program Files\Java\jdk1.6.0_33\bin\java" (changed to) _EXECJAVA="C:\Prog

Tutorial for Java - Business Transaction Health Drilldown



Business Transaction Drilldown

Download MP4 version: BTHealthDrilldown.mp4
 Download QuickTime version: BTHealthDrilldown.mov

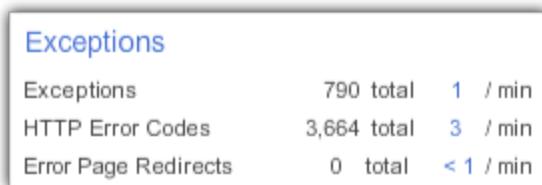
Tutorial for Java - Exceptions

- The Exceptions
- Drill Down into the HTTP Error Code Exception
- Drill Down into the AxisFault Exception
- Drill Down into the Logger Exception
- See How Exceptions are Configured
- Learn More

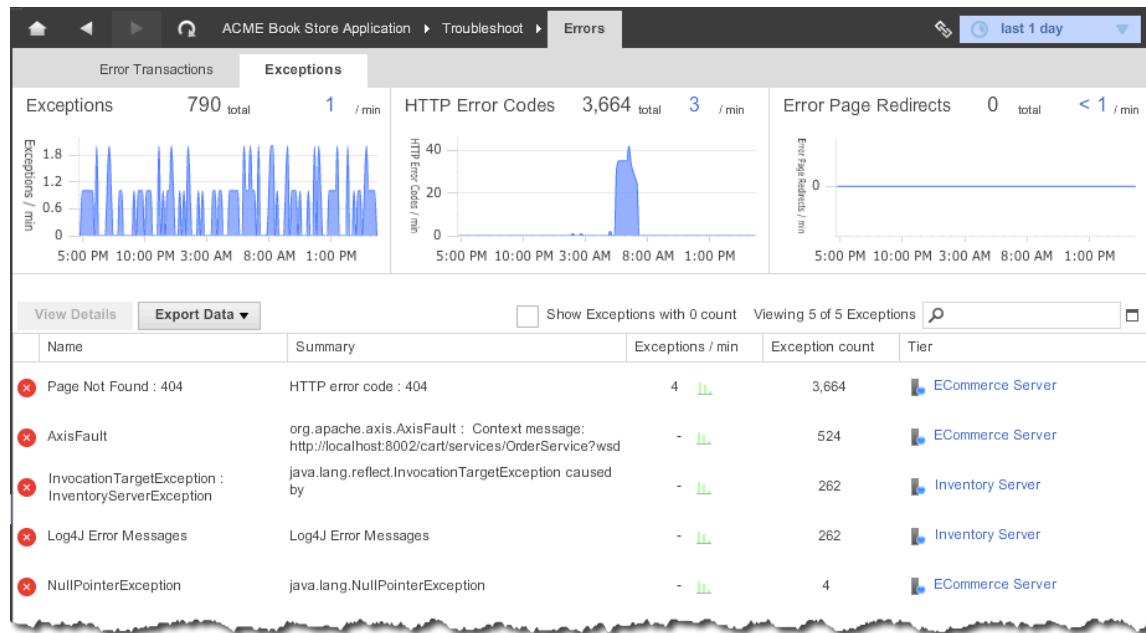
The Exceptions

An exception is a code-logged message outside the context of a business transaction. Common exceptions include code exceptions or logged errors, HTTP error codes, and error page redirects.

Exceptions display in the Exceptions pane of many dashboards.



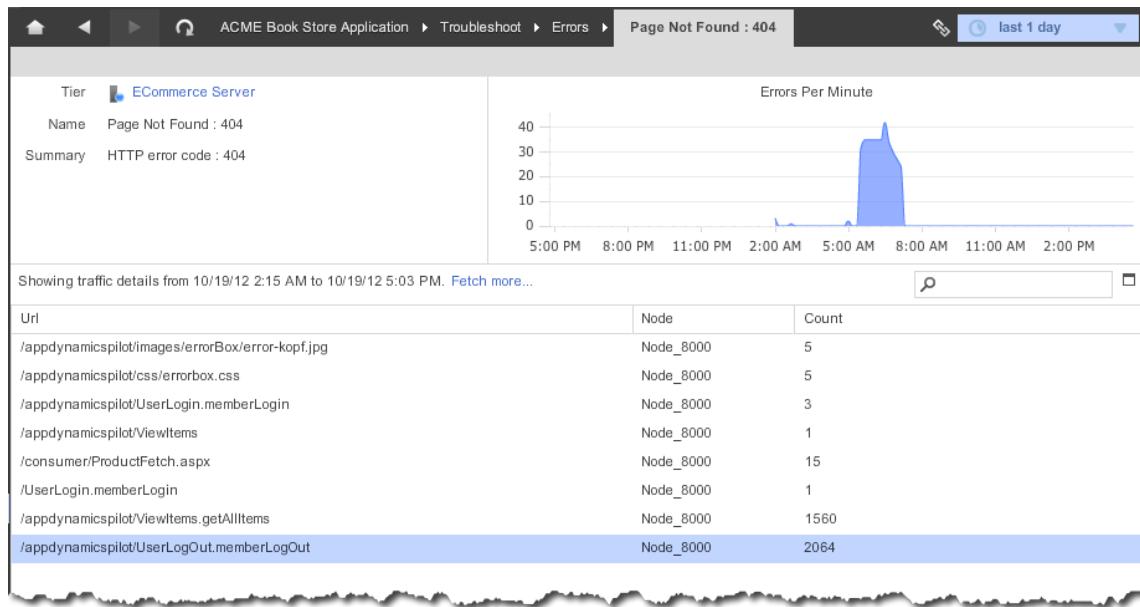
Click Exceptions to quickly see a list, ordered by frequency.



Drill Down into the HTTP Error Code Exception

Notice the spike in the HTTP Error Codes graph, and that the "Page Not Found: 404 error" is the most frequent.

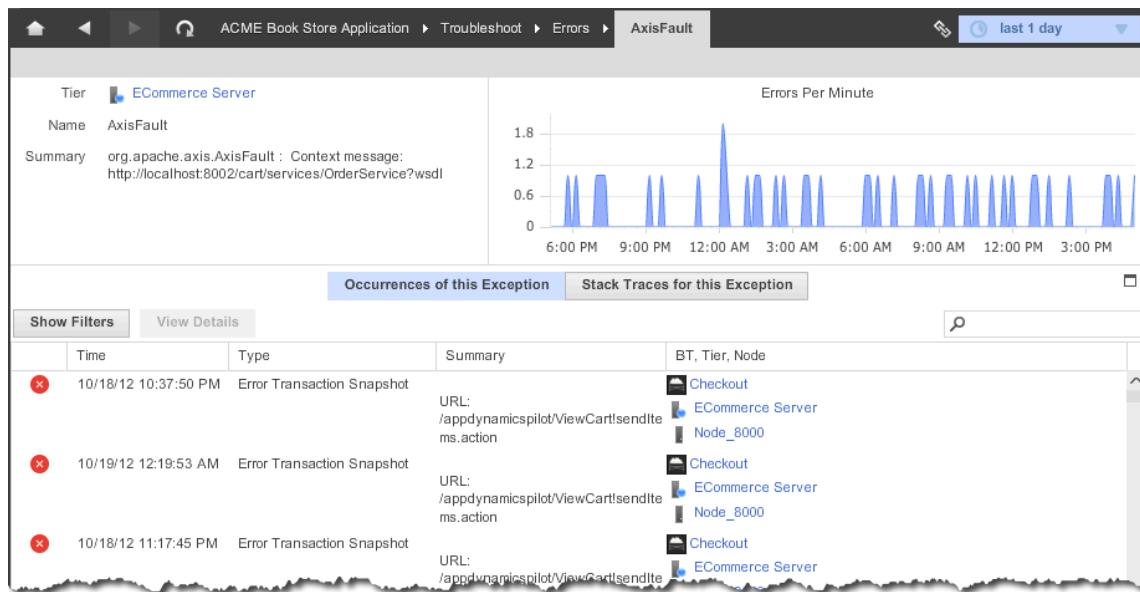
To find out more about the 404 error, click the row.



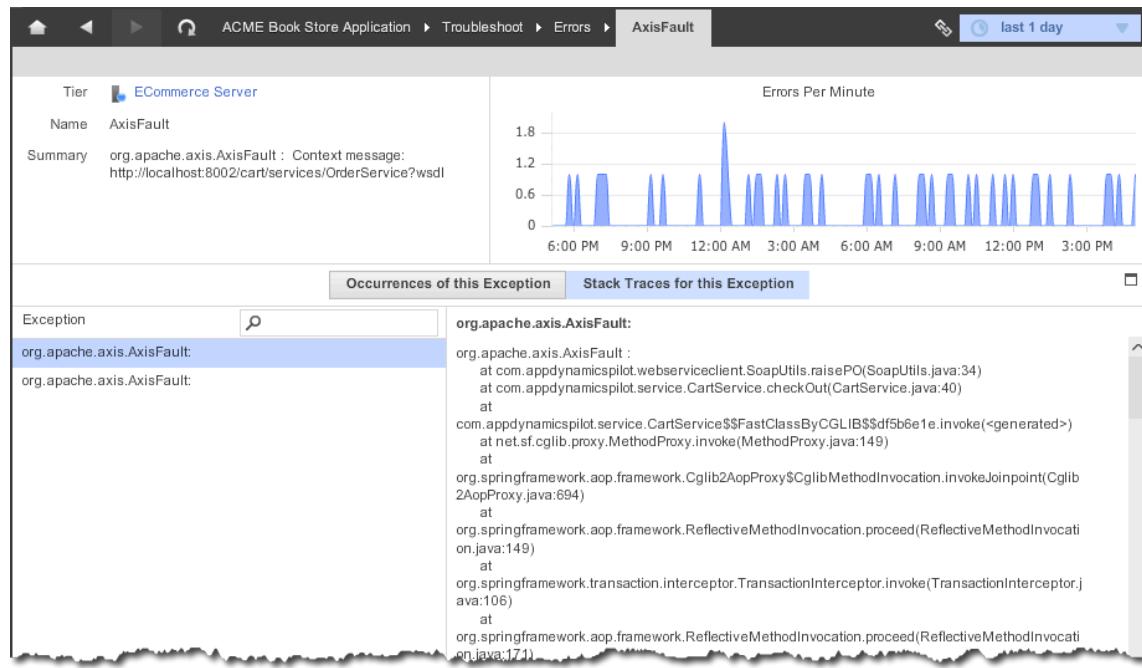
The list of URLs shows pages that have 404 errors. The memberLogOut and getAllItems URLs have the most 404 errors. You can provide this information to the web team to determine why those pages have so many 404 errors.

Drill Down into the AxisFault Exception

In the Exceptions tab, click the AxisFault row. A list of error snapshots shows the affected URL, tier, and node.



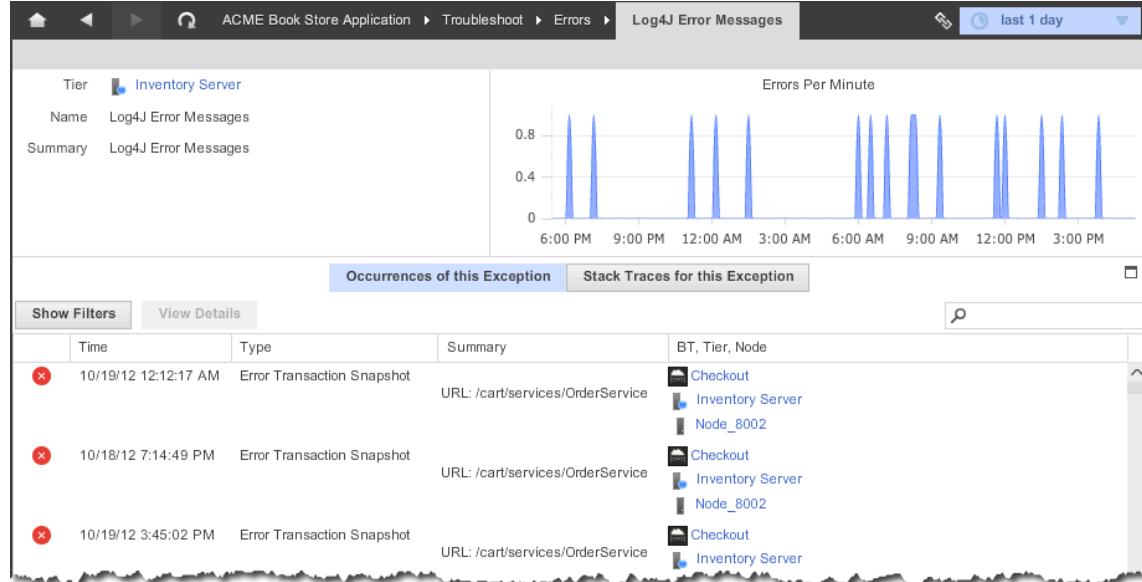
Click a row and then click the Stack Traces for This Exception tab to drill down further. Then click on one of the exceptions to see the stack trace.



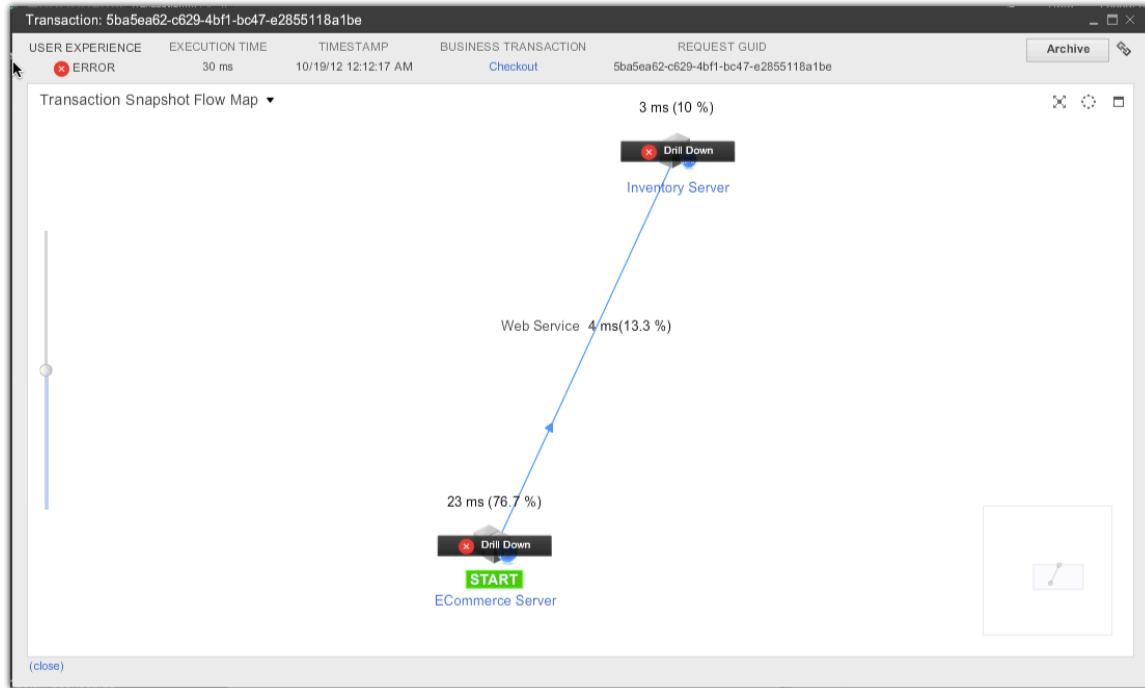
Share the stack trace with the development team to solve the problem.

Drill Down into the Logger Exception

In the Exceptions tab, click the Log4J Error Messages row. A list of error transaction snapshots shows the affected URL, business transaction, tier, and node. You can see a graph of the errors-per-minute data.



Click on a row to see the flow map for the error transaction snapshot.



The icons for both tiers have a Drill Down button. Click the Drill Down button on Ecommerce tier; it also says "Start", indicating that the transaction started on this tier.

The Call Drill Down shows the summary of the error message.

The screenshot shows the 'Call Drill Down' window for the transaction. The title bar indicates the transaction details: Exe Time: 30 ms, Timestamp: 10/19/12 12:12:17 AM, BT: Checkout GUID: 5ba5ea62-c629-4bf1-bc47-e2855118a1be. The left sidebar lists various data categories: SUMMARY, SQL CALLS, HTTP PARAMS, COOKIES, USER DATA, ERROR DETAILS, HARDWARE / MEM, NODE PROBLEMS, and ADDITIONAL DATA. The main pane displays the error message for the 'Checkout' business transaction:

```

User Experience: ERROR
Execution Time: 30 ms
CPU Time: 0 ms 0 %
Transaction Timestamp: 10/19/12 12:12:17 AM (server) 10/19/12 12:12:17 AM (agent) ⓘ
Summary: [Error] - com.appdynamicspilot.webserviceclient.SoapUtils::There was an exception in checking out 5 : AxisFault: Exception occurred while trying to invoke service method createOrder http://localhost:8002/cart/services/OrderService?wsdl : AxisFault: Exception occurred while trying to invoke service method createOrder -
Error: Exception Message: Exception occurred while trying to invoke service method createOrder
Tier: ECommerce Server
Node: Node_8000
Business Transaction: Checkout
URL: /appdynamicspilot/ViewCart!sendItems.action
Session ID: C1EF6D0085AA5CCB44A7BBA9878A43382
User Principal: No User Principal
Process ID: 8117
Thread Name: http-8000-Processor17
Thread ID: 42
Transaction Thresholds: Slow: 3.0x of standard deviation [1792.2704] for moving average [585.3495] for the last [120] minutes.
Very Slow: 4.0x of standard deviation [1792.2704] for moving average [585.3495] for the last [120] minutes.
Configure
Request GUID: 5ba5ea62-c629-4bf1-bc47-e2855118a1be
  
```

At the bottom left, there is an 'Export to PDF' button.

You can use the Export to PDF button at the lower left to send this information to your colleagues.

Go back to the flow map and click the Inventory tier **Drill Down** button. You see the Call Drill Down of the Inventory tier error message.

Call Drill Down. Exe Time: 3 ms Timestamp: 10/19/12 12:12:17 AM BT: Checkout GUID: 5ba5ea62-c629-4bf1-bc47-e2855118a1be

SUMMARY

User Experience ERROR
 Execution Time 3 ms
 CPU Time 0 ms 0 %
 Transaction Timestamp 10/19/12 12:12:17 AM (server) 10/19/12 12:12:17 AM (agent)

Summary [Error] - org.apache.axis2.rpc.receivers.RPCMessageReceiver::Exception occurred while trying to invoke service method createOrder : InvocationTargetException com.appdynamics.inventory.OrderService : Error in creating order5 -
Error Exception Message: null com.appdynamics.inventory.OrderService : Error in creating order5
Tier Inventory Server
Node Node_8002
Business Transaction Checkout
URL /cart/services/OrderService
Session ID (not found)
User Principal No User Principal
Process ID 8153
Thread Name http-8002-Processor18
Thread ID 46
Transaction Thresholds Slow: 3.0x of standard deviation [1392.9971] for moving average [296.60364] for the last [120] minutes.
 Very Slow: 4.0x of standard deviation [1392.9971] for moving average [296.60364] for the last [120] minutes.
[Configure](#)
Request GUID 5ba5ea62-c629-4bf1-bc47-e2855118a1be

Export to PDF

(close)

Compare the two error messages.

See How Exceptions are Configured

AppDynamics provides application-level default configurations for detecting exceptions. In the left navigation pane click **Configure -> Instrumentation -> Error Detection**.

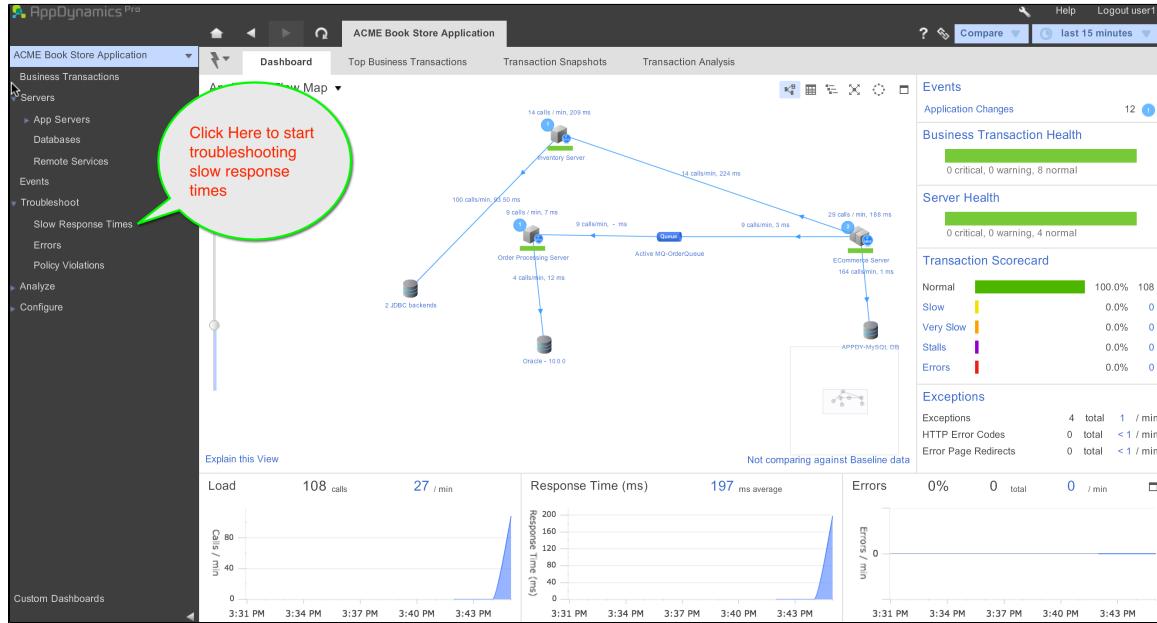
The screenshot shows the AppDynamics configuration interface for the "ACME Book Store Application". The left sidebar shows various monitoring categories like Business Transactions, Servers, Databases, and Configure. Under Configure, "Instrumentation" is selected. The main panel is titled "Error Detection" and is divided into sections for Java and .NET. The Java section is active. It includes a "Save Error Configuration" button and a "Java - Error Detection" tab. A prominent section is "Error Detection Using Logged Exceptions or Messages", which contains a list of detection rules with checkboxes. One rule is checked: "Detect errors logged using java.util.logging (Java 1.6 is required)". Another rule is checked: "Detect errors logged using Log4j". A third rule is checked: "Detect errors by looking at messages logged with ERROR or higher". A fourth rule is checked: "Mark Business Transaction as error". To the right of these rules is a note about configuring AppDynamics to look for logged errors or method invocation, and a "Name" input field with a "Add Custom Logger Definition" button. Below this is a section for "Ignored Exceptions" and "Ignored Messages".

Learn More

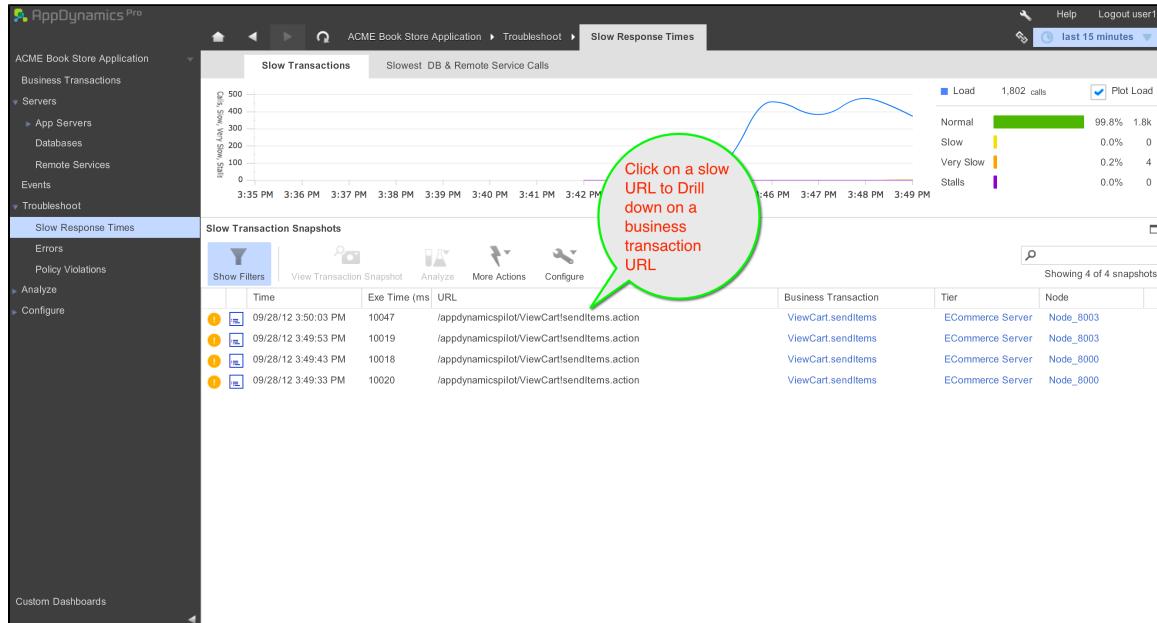
- Troubleshoot Errors
- Configure Error Detection

Tutorial for Java - Slow Transactions

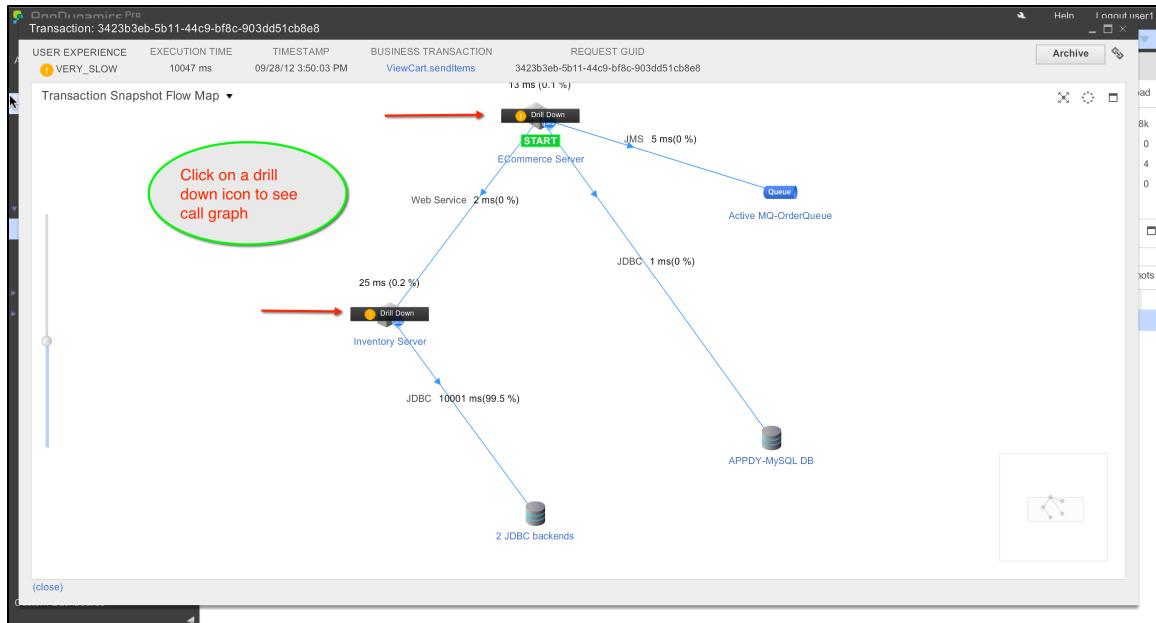
To begin troubleshooting navigate using the menu on the right:



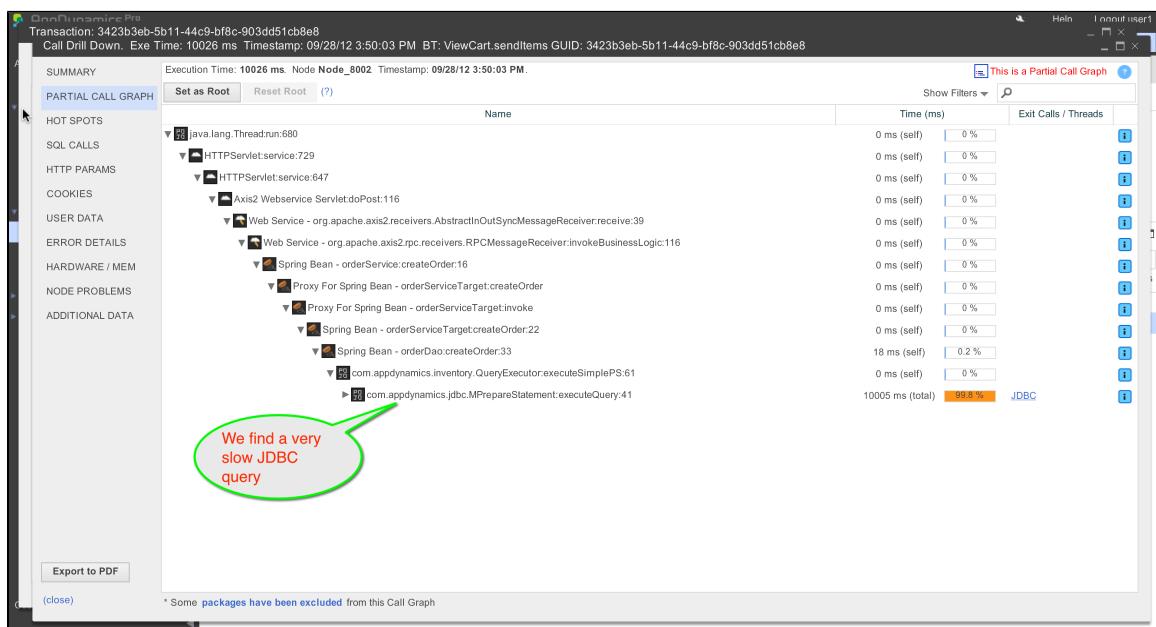
To Troubleshoot slow URLs use the Slow URL browser:



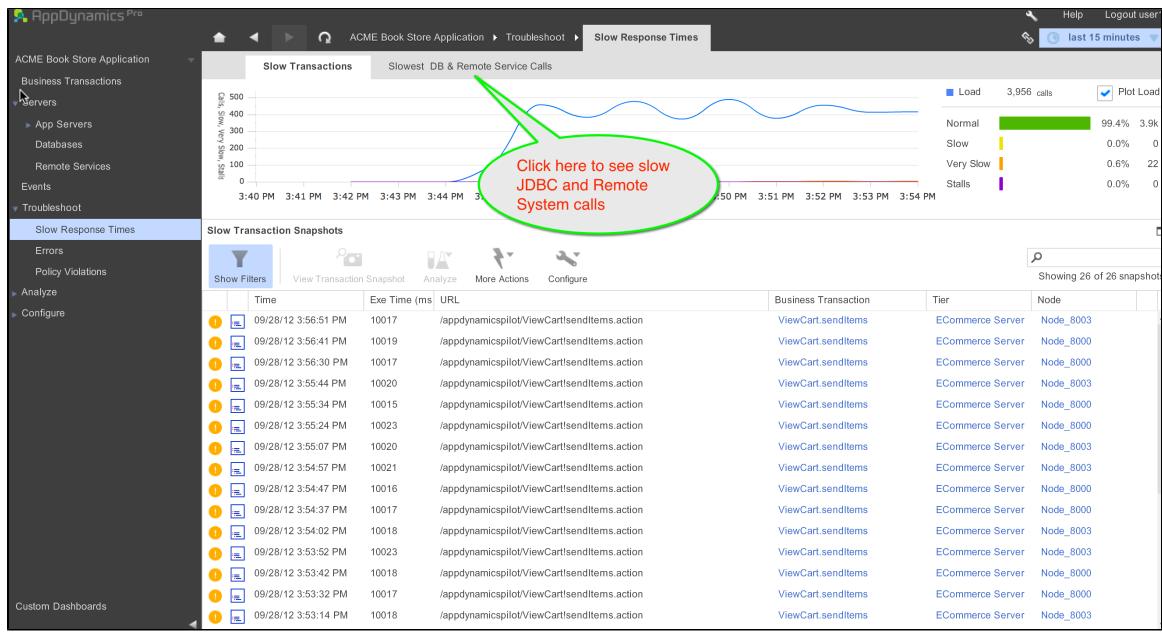
Once you select the URL you will see a visualization of the transaction. You can drill into a callgraph by clicking on the drill-down icon.



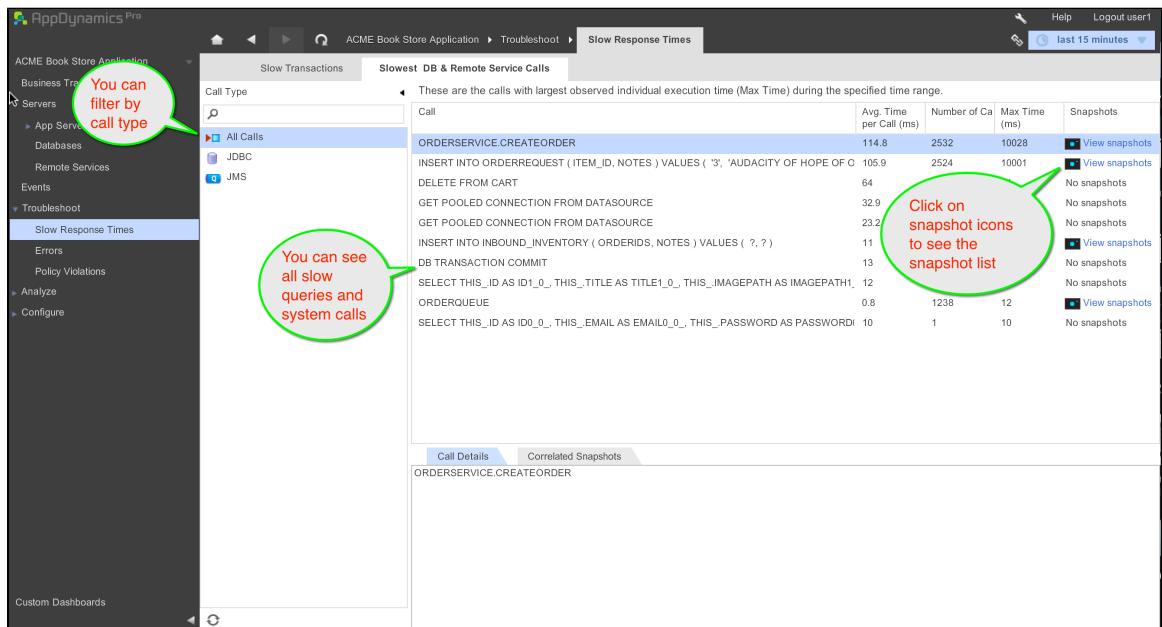
Once you are in the call graph you can look for methods that have a significant response time. For example, the executeQuery method is responsible for 99% of response time:



From the Troubleshoot menu you can navigate to the list of slow DB and Remote Calls.



You can drill into the transaction snapshots from the slow query and remote calls page to see the snapshot view:



Install the App Agent for Java

- Overview of the App Agent for Java Installation Process
- Planning for Agent Installation
 - Important Files
- To Install the Java App Server Agent
 - 1. Download and unzip the App Agent for Java
 - 2. Add the agent properties as a 'javaagent' argument to your JVM
 - 3. Configure how the agent connects to the Controller
 - 4. (Only for Multi-tenant mode or SaaS Installations): Configure Agent account information
 - 5. Configure how the agent identifies the AppDynamics business application, tier, and node.
 - Automatic Naming for Application, Tier, and Node
 - Additional Installation Scenarios
 - 6. Verify agent configuration

- 7. Verify successful installation and reporting
 - a. Verify agent installation
 - b. Verify that the agent is reporting to the Controller
- Example Configuration: App Agent for Java Deployment on a Single JVM
- Learn More

The AppDynamics App Agent for Java identifies and tracks business transactions, captures statistics and diagnostic data, and analyzes and reports data to the Controller. The App Agent for Java uses [dynamic bytecode injection](#) to instrument a JVM and it runs as a part of the JVM process.

Overview of the App Agent for Java Installation Process

Installing the App Agent for Java involves adding it as a javaagent ([Java Programming Language Agent](#)) on your JVM and setting up connection and identifying parameters for it to report data to the Controller.

Install the App Agent for Java as the same user or administrator of the JVM. Otherwise the agent may not have the correct write permissions for the system. The agent directories must have write permission so that AppDynamics can update the logs and other agent files.

Planning for Agent Installation

Before installing the App Agent for Java, be prepared with the following information.

	Planning Item	Description
	Where is the startup script for the JVM? If using a Java service wrapper, you need to know where is the wrapper configuration.	This is where you can add startup arguments in the script file and system properties, if needed.
	What host and port is the Controller running on?	For SaaS customers, AppDynamics provides this information to you. For on-premise Controllers, this information is configured during Controller installation. See (Install the Controller on Linux or Install the Controller on Windows).
	To what AppDynamics business application does this JVM belong?	Usually, all JVMs in your distributed application infrastructure belong to the same AppDynamics business application. You assign a name to the business application. For details see Logical Model .
	To what AppDynamics tier does this JVM belong?	You assign a name to the tier. For details see Logical Model .

Important Files

In addition to the JVM startup script file, two other files are important during installation:

- The -javaagent argument uses the fully-qualified path of the javaagent.jar file. No separate classpath arguments need to be added.
- The <agent_home>/conf/controller-info.xml file is where you add the configuration mentioned in the planning list.

To Install the Java App Server Agent

1. Download and unzip the App Agent for Java

- Download the App Agent for Java ZIP file from [AppDynamics Download Center](#).
- Extract the ZIP file to the destination directory as the same user or administrator of the JVM.

2. Add the agent properties as a 'javaagent' argument to your JVM

This step adds the agent to the startup script of your application server. Use the server-specific instructions below to add this argument for different Application Server JVMs:

3. Configure how the agent connects to the Controller

- Configure properties for the Controller host name and its port number.
- You can configure these two properties using either the controller-info.xml file or the JVM startup script:

Configure using controller-info.xml	Configure using System Properties	Required	Default
<controller-host>	-Dappdynamics.controller.hostName	Yes	None
<controller-port>	-Dappdynamics.controller.port	Yes	<p>For On-premise Controller installations: By default, port 8090 is used for HTTP and 8181 is used for HTTPS communication.</p> <p>For SaaS Controller service: By default, port 80 is used for HTTP and 443 is used for HTTPS communication.</p>

Optional settings for Agent-Controller communication

- To configure the Java Agent to use SSL, see [App Agent for Java Configuration Properties](#)
- To configure the Java Agent to use proxy settings see [App Agent for Java Configuration Properties](#)

4. (Only for Multi-tenant mode or SaaS Installations): Configure Agent account information

- This step is required only when the AppDynamics Controller is configured in [Controller Tenant Mode](#) or when you [Use a SaaS Controller](#).
-  Skip this step if you are using single-tenant mode, which is the default in an on-premise installation.
- Specify the properties for Account Name and Account Key. This information is provided in the Welcome email from the AppDynamics Support Team.

Configure using controller-info.xml	Configure using System Properties	Required	Default
<account-name>	-Dappdynamics.agent.accountName	Required only if your Controller is configured for multi-tenant mode or your controller is hosted.	None.
<account-access-key>	-Dappdynamics.agent.accountAccessKey	Required only if your Controller is configured for multi-tenant mode or your controller is hosted.	None.

5. Configure how the agent identifies the AppDynamics business application, tier, and node.

To better understand agents and how they relate to business applications, tiers, and nodes see [Logical Model](#) and [Name Business Applications, Tiers, and Nodes](#).

You can configure these properties using either the controller-info.xml file or JVM startup script options. Use these guidelines when configuring agents:

- Configure items that are common for all the nodes in the controller-info.xml file.
- Configure information that is unique to a node in the startup script.

Configure using controller-info.xml	Configure using System Properties	Required	Default
<application-name>	-Dappdynamics.agent.applicationName	Yes, unless you use automatic naming	None
<tier-name>	-Dappdynamics.agent.tierName	Yes, unless you use automatic naming	None
<node-name>	-Dappdynamics.agent.nodeName	Yes, unless you use automatic naming	None

Automatic Naming for Application, Tier, and Node

Starting with release 3.7.0, the App Agent for Java javaagent command has a new uniqueID argument that AppDynamics uses to automatically name the node and its tier. For example, using this command argument AppDynamics will name the node "my-app-jvm1" and the tier "my-app-jvm1":

```
-javaagent:<agent_home>/javaagent.jar=uniqueID=<my-app-jvm1>
```

When uniqueID is used, AND when an application name is not provided either through the system property or in the controller-info.xml, AppDynamics creates a new business application called "MyApp".

The new naming mechanism is used by the new Self-Service process. See [Install Agents for 5 or fewer JVMs or CLRs \(Self-Service Installations\)](#).

Additional Installation Scenarios

Refer to the links below for typical installation scenarios, especially for cases where there are multiple JVMs on the same machine:

- [Configure App Agent for Java on Multiple JVMs on the Same Machine that Serves the Same Tier](#)
- [Configure App Agent for Java on Multiple JVMs on the Same Machine that Serve Different Tiers](#)
- [Configure App Agent for Java to Use Existing System Properties](#)
- [App Agent for Java on z-OS or Mainframe Environments Configuration](#)
- [Configure App Agent for Java for Batch Processes](#)

6. Verify agent configuration

- Ensure that you have added -javaagent argument in your JVM startup script. This is not a -D system property but a different standard argument for all JVMs v1.5 and higher.
- Ensure that you have added all mandatory items either in the Agent controller-info.xml file or in the JVM startup script file.
- The user running the JVM process/application server process is the user accessing the Java Agent installation.

7. Verify successful installation and reporting

a. Verify agent installation

After a successful install, your agent logs, located at <agent_home>/logs, should contain following message:

```
Started AppDynamics Java Agent Successfully
```

 If the agent log file is not present, the App Agent for Java may not be accessing the javaagent command properties. The application server log file where STDOUT is logged will have the fallback log messages, for further troubleshooting.

b. Verify that the agent is reporting to the Controller

Use the AppDynamics UI, to verify that the Java Agent is able to connect to the Controller:

- Point your browser to: <http://<controller-host>:<controller-port>/controller>
- Provide the admin credentials to log into the AppDynamics UI.
- Select the application. In the left navigation pane, click **Servers -> App Servers -> <tier> -> <node>**. Click the Agents tab and App Server Agent subtab. An agent successfully reporting to the Controller will be listed and the Reporting property shows an "up" arrow symbol. For more details see [Verify App Agent - Controller Communication](#).
- When deploying multiple agents for the same tier, see if you get the exact number of nodes reporting in the same tier.

Example Configuration: App Agent for Java Deployment on a Single JVM

The following example shows a sample deployment of the App Agent for Java for the ACME Bookstore.

- Add the javaagent argument to the start-up script of the JVM:

```
java -javaagent:/home/appdynamics/AppServerAgent/javaagent.jar
```

- Define the five mandatory items for agent configuration in the Agent controller-info.xml file:

```
<controller-info>
    <controller-host>192.168.1.20</controller-host>
    <controller-port>8090</controller-port>
    <application-name>ACMEOnline</application-name>
    <tier-name>InventoryTier</tier-name>
    <node-name>Inventory1</node-name>

</controller-info>
```

Learn More

- App Agent for Java Configuration Properties
- Uninstall the App Agent for Java
- Logical Model

Multi-Agent Deployment for Java

- Deployment Procedure
 - To Deploy Java App Agents
 - To Deploy Java Machine Agents
- Sample Deployment Solutions
- Learn More

This topic describes the high-level procedures for deploying multiple AppDynamics app agents and machine agents on Java platforms.

Deployment Procedure

To Deploy Java App Agents

1. Download the latest agent ZIP file from <http://download.appdynamics.com/>.
2. Update deployment artifacts to use the downloaded agent.
3. Unzip the downloaded app agent file on the destination machine in the desired app agent directories.
4. Modify the app-agent-config.xml file with any custom settings for the node, tier or app.
5. Do one of the following for each application server:
 - Set the application name, tier name, node name, controller host and controller port properties in the <Agent_Installation_Directory>/conf/controller-info.xml file.
OR
 - Set these properties as system startup properties in the application server startup script using the -D option.
See [App Agent for Java Configuration Properties](#) for more information about these properties.
6. Restart the application servers to make the changes take effect.

See [Install the App Agent for Java](#) for detailed instructions on installing the app agent.

To Deploy Java Machine Agents

1. Download the latest AppDynamics Machine Agent ZIP file from <http://download.appdynamics.com/>.

2. Unzip the downloaded machine agent file on the destination machine in the desired machine agent directories.
3. Modify the <Machine_Agent_Installation_Directory>/conf/controller-info.xml files to set the application name, tier name, node name, controller host and controller port properties. Note that there are no -D settings allowed for machine agents, unlike app agents.
4. Configure the startup script for the machine to start the machine agent every time the machine reboots. For example, you could add the machine startup command to .bashrc.

To handle large values for metrics, run the machine agent using a 64-bit JDK.

Sample Deployment Solutions

You can download some sample solutions that our customers have created to perform multi-agent AppDynamics rollouts.

Use these samples for ideas on how to automate AppDynamics agent deployment for your environment. All of these samples deploy the agents independently of the application deployment.

- ChefExample1 and ChefExample2 use Opscode Chef recipes to automate deployment on Java platforms. See <http://www.opscode.com/chef/> for information about Chef.
- JavaExample1 uses a script, configuration file and package repository.

Click below to download the samples.

- [ChefExample1.tar](#)
- [ChefExample2.tar](#)
- [JavaExample1.tar](#)

Learn More

- <https://github.com/edmunds/cookbook-appdynamics>

Java Server-Specific Installation Settings

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
-javaagent:<agent_home>/javaagent.jar=uniqueID=<my-app-jvm1>
```

See [Name Business Applications, Tiers, and Nodes](#).

Apache Cassandra Startup Settings

- [To add the javaagent command in a Windows environment](#)
- [To add the javaagent command in a Linux environment](#)

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option. Add this option to the cassandra (Linux) or cassandra.bat (Windows) file.

To add the javaagent command in a Windows environment

1. Open the apache-cassandra-x.x\x\bin\cassandra.bat file.
2. Add the AppDynamics javaagent to the JAVA_OPTS variable. Make sure to include the drive in the full path to the App Server agent directory.

```
-javaagent:<agent_home>\javaagent.jar
```

For example:

```
set JAVA_OPTS=-ea  
-javaagent:C:\appdynamics\agent\javaagent.jar  
-javaagent:%CASSANDRA_HOME%\lib\jamm-0.2.5.jar  
. . .  
. . .
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
-javaagent:<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>
```

3. Restart the Cassandra server. The Cassandra server must be restarted for the changes to take effect.

To add the javaagent command in a Linux environment

1. Open the apache-cassandra-x.x.x/bin/cassandra.in.sh file.
2. Add the javaagent argument at the top of the file:

```
JVM_OPTS=-javaagent:<agent_home>/javaagent.jar
```

For example:

```
JVM_OPTS=-javaagent:/home/software/appdynamics/agent/javaagent.jar
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

3. Restart the Cassandra server for the changes to take effect.

Apache Tomcat Startup Settings

- To add the javaagent command in a Windows environment
- To add the javaagent command in a Linux environment

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option. Add this option to your Tomcat catalina.sh or catalina.bat file.

If you are using Tomcat as a Windows service, see [Tomcat as a Windows Service Configuration](#).

To add the javaagent command in a Windows environment

1. Open the **catalina.bat** file, located at <apache_version_tomcat_install_dir>\bin.
2. Add following javaagent argument to the beginning of your application server start script.

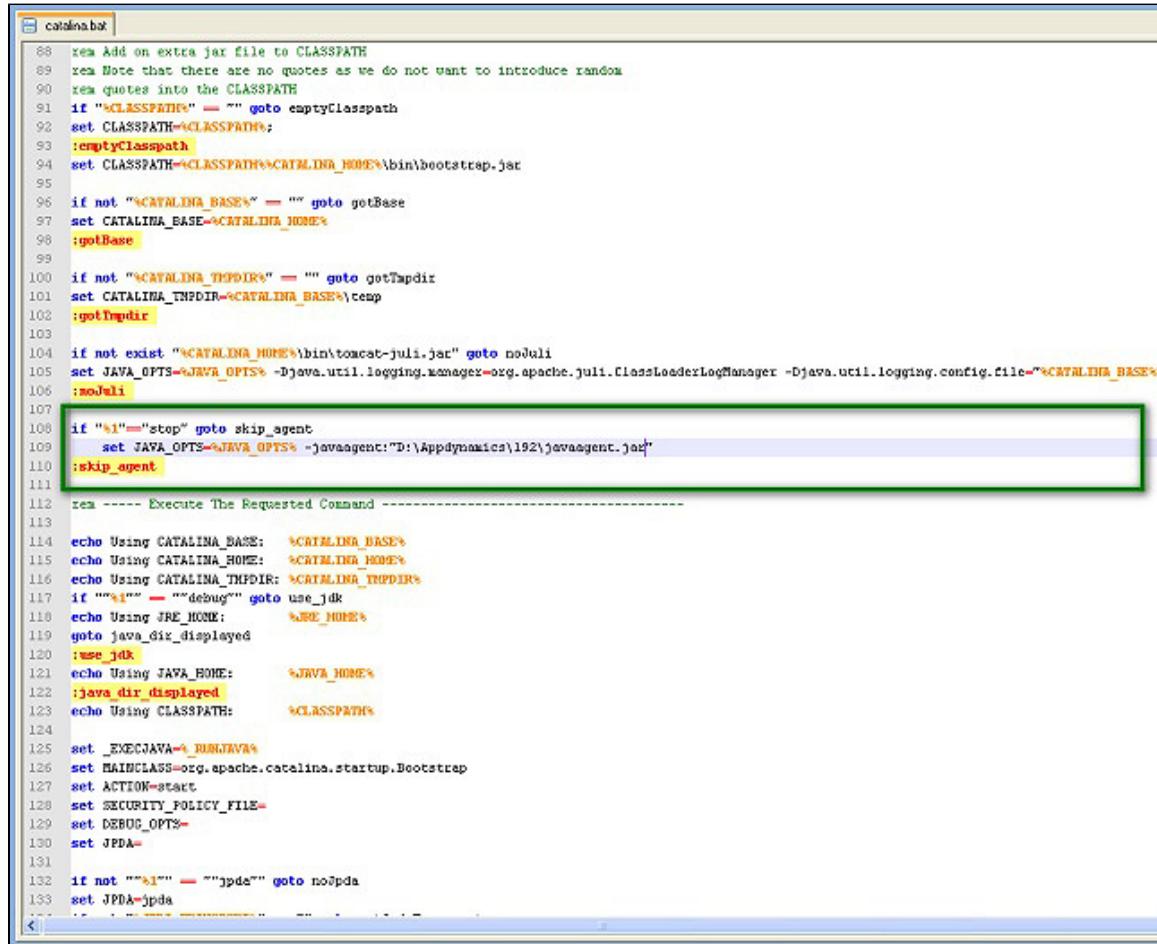
```
if "%1"=="stop" goto skip_agent  
set JAVA_OPTS=%JAVA_OPTS% -javaagent:"Drive:<agent_home>\javaagent.jar"  
:skip_agent
```

! If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
-javaagent:"Drive:<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>"
```

The javaagent argument references the full path to the App Server Agent installation directory, including the drive. For details see the screen captures.

2a. Sample Tomcat 5.x catalina.bat file



```
REM Add an extra jar file to CLASSPATH
REM Note that there are no quotes as we do not want to introduce random
REM quotes into the CLASSPATH
if "%CLASSPATH%" == "" goto emptyClasspath
set CLASSPATH=%CLASSPATH%;%CATALINA_HOME%\bin\bootstrap.jar
:emptyClasspath
set CLASSPATH=%CLASSPATH%;%CATALINA_HOME%\bin\bootstrap.jar
if not "%CATALINA_BASE%" == "" goto getBase
set CATALINA_BASE=%CATALINA_HOME%
:gotBase
if not "%CATALINA_TMPDIR%" == "" goto gotTmpdir
set CATALINA_TMPDIR=%CATALINA_BASE%\temp
:gotTmpdir
if not exist "%CATALINA_HOME%\bin\tomcat-juli.jar" goto noJuli
set JAVA_OPTS=%JAVA_OPTS% -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djava.util.logging.config.file=%CATALINA_BASE%\conf\logging.properties
:noJuli
if "%1"=="stop" goto skip_agent
set JAVA_OPTS=%JAVA_OPTS% -javaagent:"D:\Appdynamics\192\javaagent.jar"
:skip_agent
rem ----- Execute The Requested Command -----
echo Using CATALINA_BASE: %CATALINA_BASE%
echo Using CATALINA_HOME: %CATALINA_HOME%
echo Using CATALINA_TMPDIR: %CATALINA_TMPDIR%
if "%1" == "debug" goto use_jdk
echo Using JRE_HOME: %JRE_HOME%
goto java_dir_displayed
:use_jdk
echo Using JAVA_HOME: %JAVA_HOME%
:java_dir_displayed
echo Using CLASSPATH: %CLASSPATH%
:_EXECJAVA=%_JUBLJAVA%
set MAINCLASS=org.apache.catalina.startup.Bootstrap
set ACTION=start
set SECURITY_POLICY_FILE=
set DEBUG_OPTS=
set JPDA=
if not "%1" == "jpda" goto noJPDA
set JPDA=jpda
```

2b. Sample Tomcat 6.x catalina.bat file

```

catalina.bat
118
119 if not "%CATALINA_BASE%" == "" goto gotBase
120 set CATALINA_BASE=%CATALINA_HOME%
121 :gotBase
122
123 if not "%CATALINA_TMPDIR%" == "" goto gotTmpdir
124 set CATALINA_TMPDIR=%CATALINA_BASE%\temp
125 :gotTmpdir
126
127 if not "%LOGGING_CONFIG%" == "" goto noJuliConfig
128 set LOGGING_CONFIG=Dnprop
129 if not exist "%CATALINA_BASE%\conf\logging.properties" goto noJuliConfig
130 set LOGGING_CONFIG=Djava.util.logging.config.file="%CATALINA_BASE%\conf\logging.properties"
131 :noJuliConfig
132 set JAVA_OPTS=%JAVA_OPTS% %LOGGING_CONFIG%
133
134 if not "%LOGGING_MANAGER%" == "" goto noJuliManager
135 set LOGGING_MANAGER=Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
136 :noJuliManager
137 set JAVA_OPTS=%JAVA_OPTS% %LOGGING_MANAGER%
138
139 if "%1"=="stop" goto skip_agent
140     set JAVA_OPTS=%JAVA_OPTS% -javaagent:"D:\Appdynamics\192\javaagent.jar"
141 :skip_agent
142
143
144 rem ----- Execute The Requested Command -----
145
146 echo Using CATALINA_BASE: %CATALINA_BASE%
147 echo Using CATALINA_HOME: %CATALINA_HOME%
148 echo Using CATALINA_TMPDIR: %CATALINA_TMPDIR%
149 if "%1"=="debug" goto use_jdk
150 echo Using JRE_HOME: %JRE_HOME%
151 goto java_dir_displayed
152 :use_jdk
153 echo Using JAVA_HOME: %JAVA_HOME%
154 :java_dir_displayed
155

```

3. Restart the application server. The application server must be restarted for the changes to take effect.

To add the javaagent command in a Linux environment

1. Open the catalina.sh file located at <apache_version_tomcat_install_dir>/bin).
2. Add the following commands at the beginning of your application server start script.

```

if [ "$1" = "start" -o "$1" = "run" ]; then
    export JAVA_OPTS="$JAVA_OPTS -javaagent:agent_install_dir/javaagent.jar"
fi

```

The javaagent argument references the full path to the App Server Agent installation directory.

! If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

For details see the screen captures.

2a. Sample Tomcat 5.x catalina.sh file

```

148 have_tty=0
149 if [ "$tty" != "not a tty" ]; then
150     have_tty=1
151 fi
152
153 # For Cygwin, switch paths to Windows format before running java
154 if $cygwin; then
155     JAVA_HOME=`cygpath --absolute --windows "$JAVA_HOME"`
156     JRE_HOME=`cygpath --absolute --windows "$JRE_HOME"`
157     CATALINA_HOME=`cygpath --absolute --windows "$CATALINA_HOME"`
158     CATALINA_BASE=`cygpath --absolute --windows "$CATALINA_BASE"`
159     CATALINA_TMPDIR=`cygpath --absolute --windows "$CATALINA_TMPDIR"`
160     CLASSPATH=`cygpath --path --windows "$CLASSPATH"`
161     JAVA_ENDORSED_DIRS=`cygpath --path --windows "$JAVA_ENDORSED_DIRS"`
162 fi
163
164 # Set juli LogManager if it is present
165 if [ -r "$CATALINA_HOME/bin/tomcat-juli.jar" ]; then
166     JAVA_OPTS="$JAVA_OPTS -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager"
167     LOGGING_CONFIG="-Djava.util.logging.config.file=$CATALINA_BASE/conf/logging.properties"
168 else
169     # Bugzilla 45585
170     LOGGING_CONFIG="-Dnop"
171 fi
172
173 if [ "$1" = "start" -o "$1" = "run" ] ; then
174     export JAVA_OPTS="$JAVA_OPTS -javaagent:/mnt/agenttest/agent192-2/javaagent.jar"
175 fi
176
177 # ----- Execute The Requested Command -----
178
179 # Bugzilla 37848: only output this if we have a TTY
180 if [ $have_tty -eq 1 ]; then
181     echo "Using CATALINA_BASE: $CATALINA_BASE"
182     echo "Using CATALINA_HOME: $CATALINA_HOME"
183     echo "Using CATALINA_TMPDIR: $CATALINA_TMPDIR"
184     if [ "$1" = "debug" ] ; then
185         echo "Using JAVA_HOME: $JAVA_HOME"
186     else
187         echo "Using JRE_HOME: $JRE_HOME"
188     fi
189 fi

```

2b. Sample Tomcat 6.x catalina.sh file

```

199 fi
200
201 # Set juli LogManager config file if it is present and an override has not been issued
202 if [ -z "$LOGGING_CONFIG" ]; then
203     if [ -r "$CATALINA_BASE/conf/logging.properties" ]; then
204         LOGGING_CONFIG="-Djava.util.logging.config.file=$CATALINA_BASE/conf/logging.properties"
205     else
206         # Bugzilla 45585
207         LOGGING_CONFIG="-Dnop"
208     fi
209 fi
210
211 if [ -z "$LOGGING_MANAGER" ]; then
212     JAVA_OPTS="$JAVA_OPTS -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager"
213 else
214     JAVA_OPTS="$JAVA_OPTS $LOGGING_MANAGER"
215 fi
216
217 if [ "$1" = "start" -o "$1" = "run" ] ; then
218     export JAVA_OPTS="$JAVA_OPTS -javaagent:/mnt/agenttest/agent192/javaagent.jar"
219 fi
220
221 # ----- Execute The Requested Command -----
222
223 # Bugzilla 37848: only output this if we have a TTY
224 if [ $have_tty -eq 1 ]; then
225     echo "Using CATALINA_BASE: $CATALINA_BASE"
226     echo "Using CATALINA_HOME: $CATALINA_HOME"
227     echo "Using CATALINA_TMPDIR: $CATALINA_TMPDIR"
228     if [ "$1" = "debug" ] ; then
229         echo "Using JAVA_HOME: $JAVA_HOME"
230     else
231         echo "Using JRE_HOME: $JRE_HOME"
232     fi
233     echo "Using CLASSPATH: $CLASSPATH"
234 fi
235
236 if [ "$1" = "jpdas" ] ; then
237     if [ -z "$JPDA_TRANSPORT" ]; then
238         JPDA_TRANSPORT="socket,address=8000"
239     fi
240 fi

```

3. Restart the application server. The application server must be restarted for the changes to take effect.

Tomcat as a Windows Service Configuration

- To install the javaagent as a Tomcat Windows service

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option. Add this option to your Tomcat properties.

If you are not running Tomcat as a Windows service, see [Apache Tomcat Startup Settings](#).

To install the javaagent as a Tomcat Windows service

These instructions apply to Apache Tomcat 6.x or later versions.

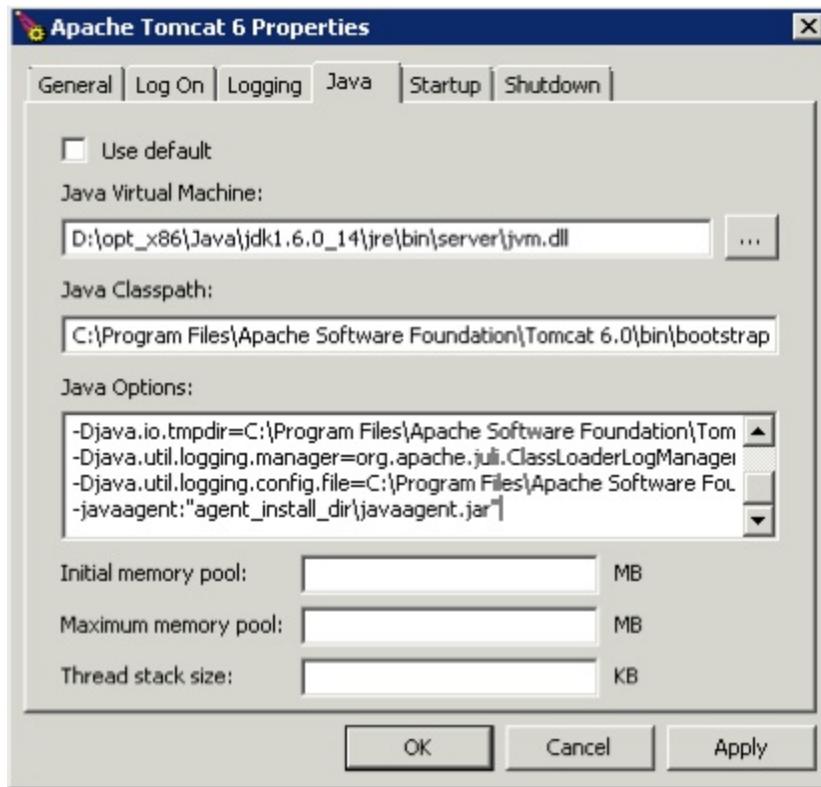
1. Ensure that you are using administrator privileges.
2. Click **Programs -> Apache Tomcat**.
3. Run **Configure Tomcat**.
4. Click the **Java** tab.
5. In the **Java Options** add:

```
-javaagent:<agent_home>\javaagent.jar
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
-javaagent:<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>
```

For details see the following screenshot.



6. Restart the Tomcat service. The application server must be restarted for the changes to take effect.

Glassfish Startup Settings

- To add the javaagent command in a GlassFish environment
- To verify the Agent configuration
- About AppServer Management Extensions

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option.

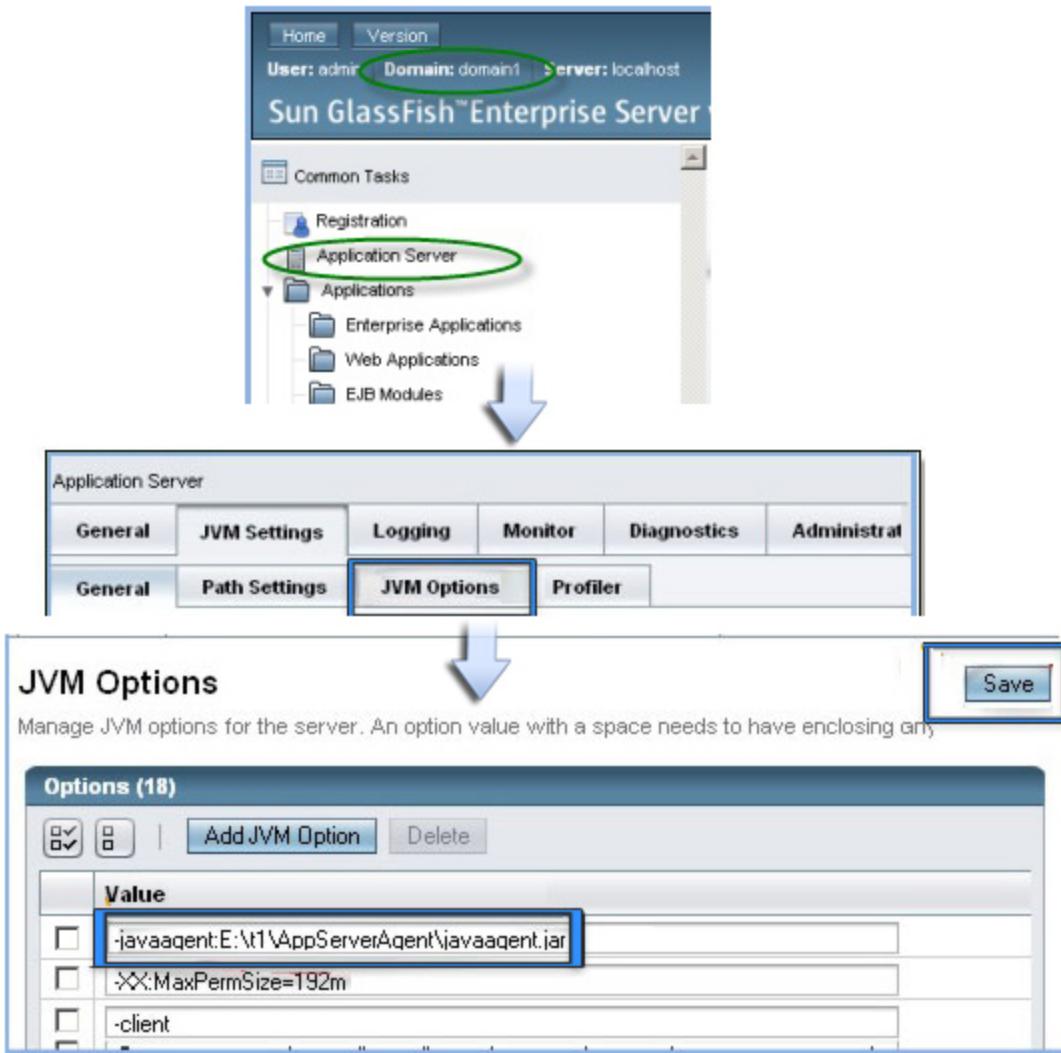
To add the javaagent command in a GlassFish environment

1. If you are using **GlassFish v3.x**, first configure the OSGi containers. For details see [OSGi Infrastructure Configuration](#).
 2. Log into the **GlassFish domain** where you want to install the App Server Agent.
 3. In the left navigation tree **Common Tasks** section, click **Application Server**. The Application Server Settings dialog opens.
 4. In the **JVM Settings** tab, click **JVM Options**.
5. Click **Add JVM Option** and add an entry for the javaagent argument. The javaagent argument contains the full path, including the drive, of the App Server Agent installation directory.

```
-javaagent:<drive>:\<agent_home>\javaagent.jar
```

! If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

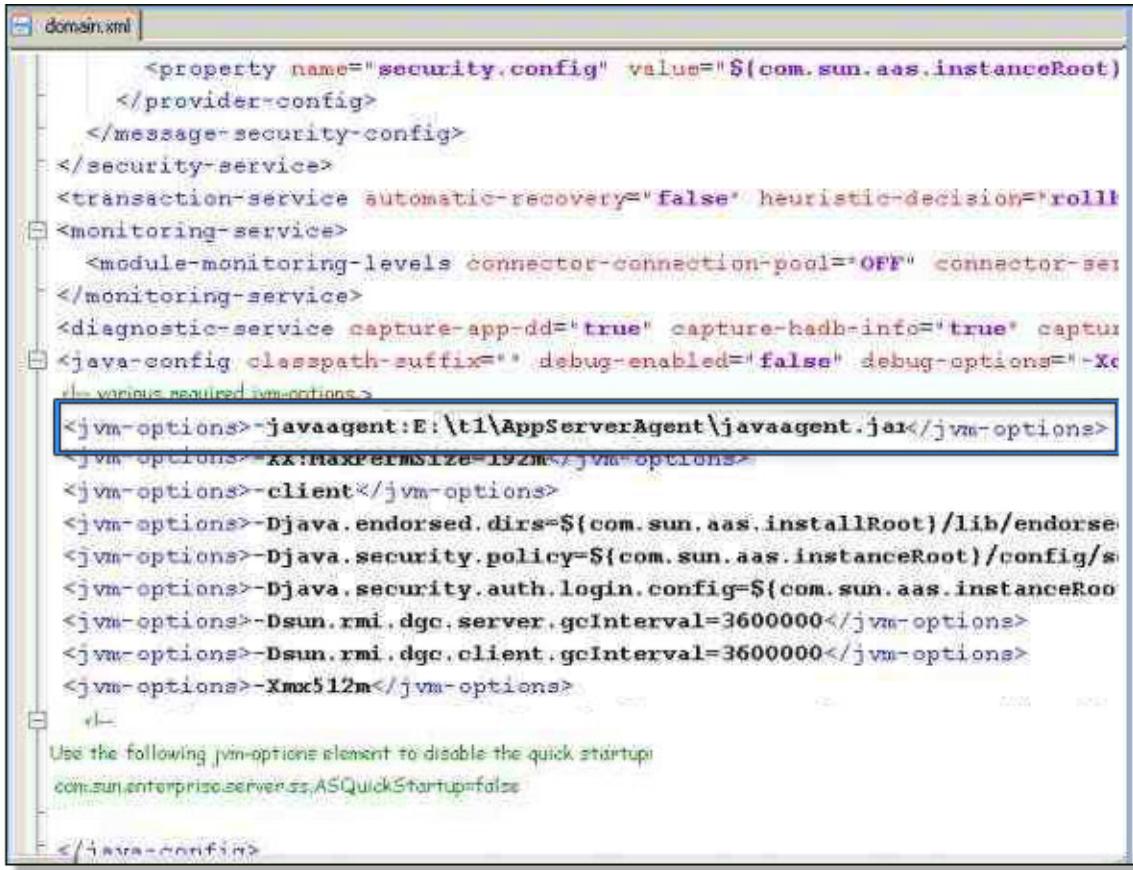
```
-javaagent:<drive>:\<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>
```



6. Restart the application server. The application server must be restarted for the changes to take effect.

To verify the Agent configuration

To verify this configuration, look at the domain.xml file located at <glassfish_install_dir>\domains\<domain_name>. The domain.xml file should have an entry as shown in the following screenshot.



```

<domain.xml>
    <provider-config>
        </provider-config>
    </message-security-config>
</security-service>
<transaction-service automatic-recovery="false" heuristic-decision="rollb
<monitoring-service>
    <module-monitoring-levels connector-connection-pool="OFF" connector-se
</monitoring-service>
<diagnostic-service capture-app-dd="true" capture-hadb-info="true" captur
<java-config classpath-suffix="" debug-enabled="false" debug-options="-Xc
    <!-- various required configurations -->
<jvm-options>-javaagent:E:\t1\AppServerAgent\javaagent.jar</jvm-options>
<jvm-options>-XX:MaxPermSize=192M</jvm-options>
<jvm-options>-client</jvm-options>
<jvm-options>-Djava.endorsed.dirs=${com.sun.aas.instanceRoot}/lib/endorse
<jvm-options>-Djava.security.policy=${com.sun.aas.instanceRoot}/config/s
<jvm-options>-Djava.security.auth.login.config=${com.sun.aas.instanceRoo
<jvm-options>-Dsun.rmi.dgc.server.gcInterval=3600000</jvm-options>
<jvm-options>-Dsun.rmi.dgc.client.gcInterval=3600000</jvm-options>
<jvm-options>-Xmx512m</jvm-options>
<!--
Use the following jvm-options element to disable the quick startup
com.sun.enterprise.server.ss.ASQuickStartup=false
-->
</java-config>

```

About AppServer Management Extensions

AppDynamics does not support Glassfish AMX. AMX MBeans do not appear in the MBean Browser.

IBM WebSphere Startup Settings

- To add the javaagent command in a WebSphere environment
- To verify the Agent configuration
- Security permissions requirement

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option.

To add the javaagent command in a WebSphere environment

1. Log in to the **Administrator** console of the WebSphere node where you want to install the App Server Agent.
2. In the left navigation tree, click **Servers -> Application servers**.
3. Click the name of your server in the list of servers.

For quick access, place your bookmarks here in the bookmarks bar.

Integrated Solutions Console Welcome jpowar

Help | Logout IBM

View: All tasks

- Welcome
- Guided Activities**
- Servers
 - Application servers
 - Web servers
 - WebSphere MQ servers
- Applications
 - Enterprise Applications
 - Install New Application
- Resources
- Security
- Environment

Application servers

Application servers

Use this page to view a list of the application servers in your environment. You can also use this page to change the status of a specific server.

Preferences

Name	Node	Version
server1	ww-84f6cf8ea82aNode01	Base 6.1.0.0

Total 1



4. In the "Configuration" tab, click **Java and Process Management**.

Application servers > server1

Use this page to configure an application server. An application server is a server that provides services required to run enterprise applications.

Runtime Configuration

General Properties

Name: server1
Node name: rajendraNode01
 Run in development mode
 Parallel start
 Start components as needed
Access to internal server classes: Allow
Server-specific Application Settings
Classloader policy: Multiple
Class loading mode: Classes loaded with parent class loader first

Container Settings

- [Session management](#)
- + [SIP Container Settings](#)
- + [Web Container Settings](#)
- + [Portlet Container Settings](#)
- + [EJB Container Settings](#)
- + [Container Services](#)
- + [Business Process Services](#)

Applications

- [Installed applications](#)

Server messaging

- [Messaging engines](#)
- [Messaging engine inbound transports](#)
- [WebSphere MQ link inbound transports](#)
- [SIB service](#)

Server Infrastructure

- + Java and Process Management
- + Administration

Communications

- + Ports

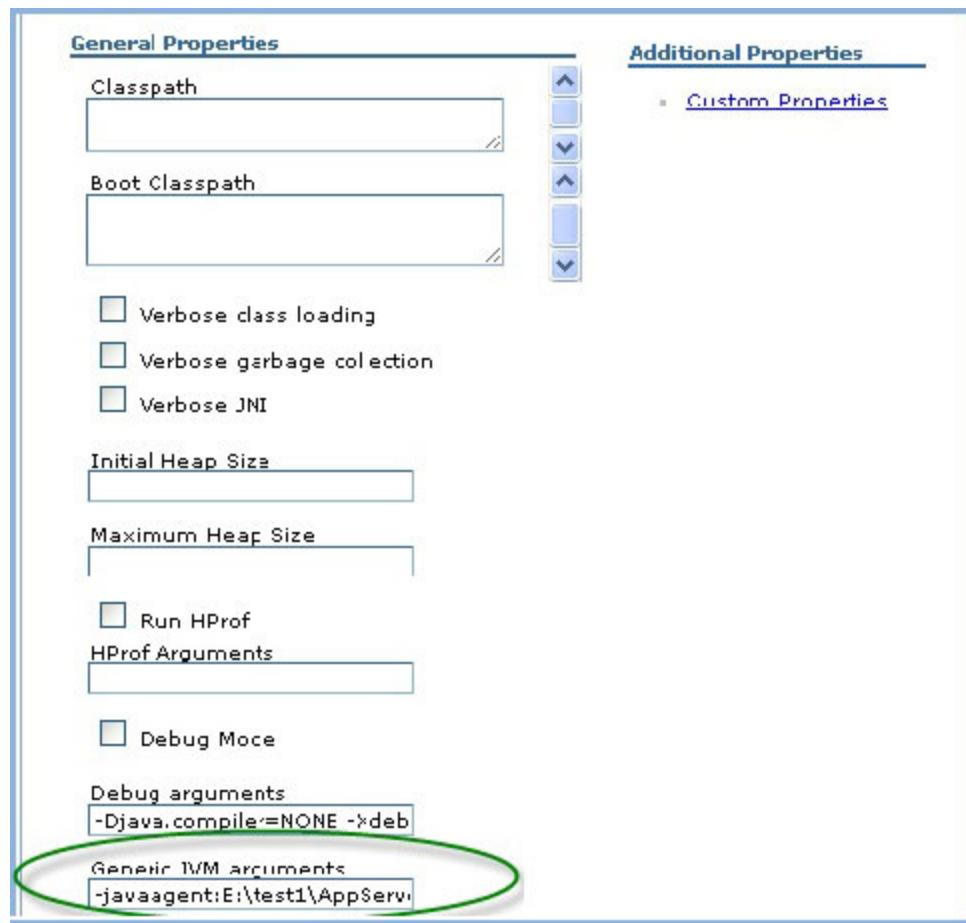
5. Enter the javaagent option with the full path to the AppDynamics javaagent.jar file in the **Generic JVM arguments** field.

-javaagent:<drive>:\<agent_home>\javaagent.jar

⚠ If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

-javaagent:<drive>:\<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>

6. Click OK.



To verify the Agent configuration

Verify the configuration settings by checking the server.xml file of the WebSphere node where you installed the App Server Agent. The server.xml file should have this entry:

```
<jvmEntries ...
genericJvmArguments='"-javaagent:E:\test1\AppServerAgent\javaagent.jar"
cisableJIT="false"/>
```

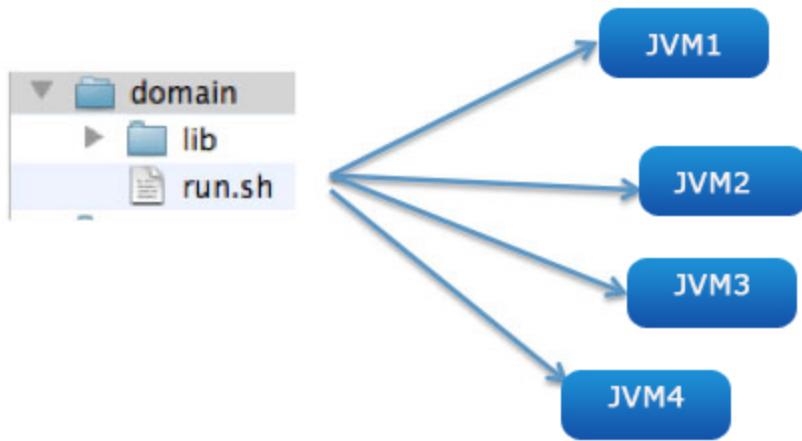
Security permissions requirement

Full permissions are required for the agent to function correctly with WebSphere. Grant all permissions on both the server level and the profile level.

App Agent for Java on z-OS or Mainframe Environments Configuration

- To configure the Agent to automatically generate node prefixes
- To remove the nodes that are not alive

In some environments JVMs have transient identity, such as when a single script spawns multiple JVMs.



For example, an environment may consist of WebSphere on IBM Mainframes, using a dynamic workload management feature that spawns new JVMs for an existing application server (called a servant). These JVMs are exact clones of an existing JVM, but each of them has a different process ID. Based on load, any number of additional JVMs may be created.

The App Server Agent can monitor each of these dynamically generated JVMs using the `appdynamics.agent.auto.node.prefix` option. You specify a node name prefix in your JVM startup script.

```
-Dappdynamics.agent.auto.node.prefix=<node name prefix>
```

To configure the Agent to automatically generate node prefixes

1. Add the application and tier name to "controller-info.xml" file for each JVM.

i These JVMs belong to the **same** tier.

2. Add the `javaagent` argument and system properties (-D options) to the startup script of your JVM processes or to your Java Runtime Environment (JRE):

```
java -javaagent:<Agent-Installation-Directory>/javaagent.jar
-Dappdynamics.agent.reuse.nodeName=true
-Dappdynamics.agent.auto.node.prefix=$nodePrefix
```

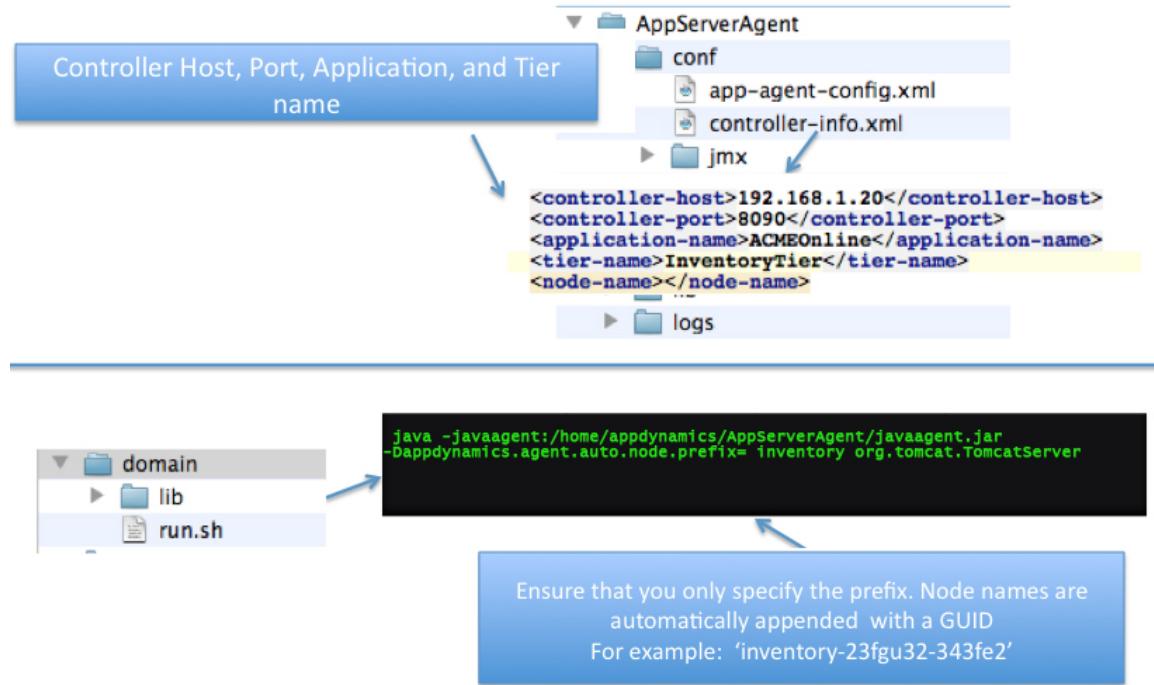
By default, the node names will be based on a serial number maintained by the Controller with a prefix of the tier name. If you need to change the prefix, specify the property `-Dappdynamics.agent.auto.node.prefix=$nodePrefix`.

For example, if no prefix is specified for node on tier ECommerceTier, the node name will be `ECommerceTier-1`, `ECommerceTier-2`, etc. If a prefix is provided as in the example, the node names will be `customPrefix-1`, `customPrefix-2`, etc.

IMPORTANT:

- Ensure that you have not specified the node name anywhere (controller-info.xml file or as a system property in your JVMs start-up script).
- Ensure that all these system properties are separated by a white space character.
- These JVMs may or may not be short-lived.

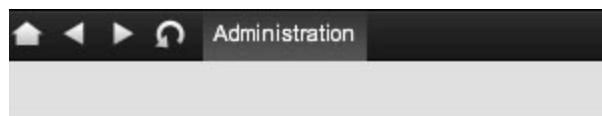
The following illustration shows the sample configuration for ACME Bookstore. This configuration will create unique node names for every instance of the virtual machine starting up in the ACME Bookstore environment.



To remove the nodes that are not alive

In a typical environment, the nodes are recycled and new nodes get generated. AppDynamics strongly recommends that you remove the dead nodes both from the AppDynamics user interface (UI) as well as from the system.

1. Log in to the Controller administration console.
2. Go to "Controller Settings" section to see the advanced properties for the Controller.



AppDynamics Administration

Accounts

Create and Manage Accounts

Controller Settings

Configure the Controller

3. Set the retention and deletion properties, based on the requirements for your environment. AppDynamics recommends that you set the permanent deletion period at least an hour more than the retention period.

- **node.permanent.deletion.period:** Time (in hours) after which a node that has lost contact with the Controller is deleted permanently from the system.
- **node.retention.period:** Time (in hours) after which a node that has lost contact with the Controller is deleted. In this case, the

AppDynamics UI will not display the node, however the system will continue to retain it.

metrics.write.thread.count	The count of parallel threads to be i	<input type="text" value="1"/>
multitenant.controller	Is the controller running in multi-tier	<input type="text" value="false"/>
node.permanent.deletion.period	Time (in hours) after which a node t	<input type="text" value="720"/>
node.retention.period	Time (in hours) after which a node t	<input type="text" value="500"/>

JBoss Startup Settings

- To add the javaagent command in a Windows environment
- To add the javaagent command in a Linux environment for JBoss 5.x
- To add the javaagent command in a Linux environment for JBoss AS 6.x
- To add the javaagent command in a Linux environment for JBoss AS 7.x
- To add the javaagent command in a Windows environment for JBoss AS 7.x

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option. Add this option to your JBoss server run.sh or run.bat file.

To add the javaagent command in a Windows environment

1. Open the server run.bat file, located at <jboss_version_install_directory>\bin.
2. Add the following javaagent argument at the beginning of your app server start script.

```
set JAVA_OPTS=%JAVA_OPTS% -javaagent:<drive>:\<agent_home>\javaagent.jar
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
-javaagent:<drive>:\<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>
```

The javaagent argument references the full path of the App Server Agent installation directory, including the drive. For details see the screen captures.

2a. Sample JBoss 4.x run.bat file

```
run.bat
62
63 rem If JBOSS_CLASSPATH is empty, don't include it, as this will
64 rem result in including the local directory, which makes error tracking
65 rem harder.
66 if "%JBOSS_CLASSPATH%" == "" (
67     set JBOSS_CLASSPATH=%JAVAC_JAR%;%RUNJAR%
68 ) else (
69     set JBOSS_CLASSPATH=%JBOSS_CLASSPATH%;%JAVAC_JAR%;%RUNJAR%
70 )
71
72 rem Setup JBoss specific properties
73 set JAVA_OPTS=%JAVA_OPTS% -Dprogram.name=%PROGNAME%
74 set JBOSS_HOME=%DIRNAME%..
75
76 rem Add -server to the JVM options, if supported
77 "%JAVA%" -version 2>&1 | findstr /I hotspot > nul
78 if not errorlevel == 1 (set JAVA_OPTS=%JAVA_OPTS% -server)
79
80 rem JVM memory allocation pool parameters. Modify as appropriate.
81 set JAVA_OPTS=%JAVA_OPTS% -Xms128m -Xmx512m
82
83 rem With Sun JVMs reduce the RMI GCs to once per hour
84 set JAVA_OPTS=%JAVA_OPTS% -Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=360000
85
86 set JAVA_OPTS=%JAVA_OPTS% -javaagent:"E:\t1\AppServerAgent\javaagent.jar"
87 rem JDB options. Uncomment and modify as appropriate to enable remote debugging.
88 rem set JAVA_OPTS=-Xdebug -Xrunjdwp:transport=dt_socket,address=8787,server=y,suspend=y %JAVA_OPTS%
89
90 rem Setup the java endorsed dirs
91 set JBOSS_ENDORSED_DIRS=%JBOSS_HOME%\lib\endorsed
92
93 echo -----
94 echo.
95 echo JBoss Bootstrap Environment
96 echo.
97 echo JBOSS_HOME: %JBOSS_HOME%
98 echo.
99 echo JAVA: %JAVA%
echo.
101 echo JAVA_OPTS: %JAVA_OPTS%
```

2b. Sample JBoss 5.x run.bat file

```

97
98 rem If JBOSS_CLASSPATH empty, don't include it, as this will
99 rem result in including the local directory in the classpath, which makes
100 rem error tracking harder.
101 if "%JBOSS_CLASSPATH%" == "" (
102     set "RUN_CLASSPATH=%RUNJAR%"
103 ) else (
104     set "RUN_CLASSPATH=%JBOSS_CLASSPATH%;%RUNJAR%"
105 )
106
107 set JBOSS_CLASSPATH=%RUN_CLASSPATH%
108
109 rem Setup JBoss specific properties
110 rem JVM memory allocation pool parameters. Modify as appropriate.
111 set JAVA_OPTS=%JAVA_OPTS% -Xms128m -Xmx512m -XX:MaxPermSize=256m
112
113 rem Warn when resolving remote XML dtd/schemas
114 set JAVA_OPTS=%JAVA_OPTS% -Dorg.jboss.resolver.warning=true
115
116 rem With Sun JVMs reduce the RMI GCs to once per hour
117 set JAVA_OPTS=%JAVA_OPTS% -Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000
118
119 set JAVA_OPTS=%JAVA_OPTS% -javaagent:"E:\t1\AppServerAgent\javaagent.jar"
120 rem JPA options. Uncomment and modify as appropriate to enable remote debugging.
121 rem set JAVA_OPTS=%JAVA_OPTS% -Xdebug -Xrunjdwp:transport=dt_socket,address=8787,server=y,suspend=y
122
123 rem Setup the java endorsed dirs
124 set JBOSS_ENDORSED_DIRS=%JBOSS_HOME%\lib\endorsed
125
126 echo =====
127 echo.
128 echo JBoss Bootstrap Environment
129 echo.
130 echo JBOSS_HOME: %JBOSS_HOME%
131 echo.
132 echo JAVA: %JAVA%
133 echo.
134 echo JAVA_OPTS: %JAVA_OPTS%
135 echo.
136 echo CLASSPATH: %JBOSS_CLASSPATH%

```

3. Restart the application server. The application server must be restarted for the changes to take effect.

To add the javaagent command in a Linux environment for JBoss 5.x

1. Open the server run.sh file, located at <jboss_version_install_dir>/bin.
2. Add the following javaagent argument to the server start script.

```
export JAVA_OPTS="$JAVA_OPTS -javaagent:<agent_home>/javaagent.jar"
```

! If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

2a. Sample JBoss 5.x run.sh file

```

run.sh

130     fi
131
132     # Enable -server if we have Hotspot, unless we can't
133     if [ "x$HAS_HOTSPOT" != "x" ]; then
134         # MacOS does not support -server flag
135         if [ "$darwin" != "true" ]; then
136             JAVA_OPTS="-server $JAVA_OPTS"
137         fi
138         fi
139     fi
140
141     # Setup JBoss specific properties
142     JAVA_OPTS="-Dprogram.name=$PROGNAME $JAVA_OPTS"
143
144     # Setup the java endorsed dirs
145     JBOSS_ENDORSED_DIRS="$JBOSS_HOME/lib/endorsed"
146
147     # For Cygwin, switch paths to Windows format before running java
148     if $cygwin; then
149         JBOSS_HOME=`cygpath --path --windows "$JBOSS_HOME"`
150         JAVA_HOME=`cygpath --path --windows "$JAVA_HOME"`
151         JBOSS_CLASSPATH=`cygpath --path --windows "$JBOSS_CLASSPATH"`
152         JBOSS_ENDORSED_DIRS=`cygpath --path --windows "$JBOSS_ENDORSED_DIRS"`
153     fi
154     export JAVA_OPTS="$JAVA_OPTS -javaagent:/home/AppServerAgent/javaagent.jar"
155     # Display our environment
156     echo ====
157     echo ""
158     echo "  JBoss Bootstrap Environment"
159     echo ""
160     echo "  JBOSS_HOME: $JBOSS_HOME"

```

3. Restart the application server. The application server must be restarted for the changes to take effect.

To add the javaagent command in a Linux environment for JBoss AS 6.x

1. Open the server run.sh file, located at <jboss_version_install_dir>/bin.
2. Add the following Java environment variables to the server start script.

```

JAVA_OPTS="$JAVA_OPTS
-Djava.util.logging.manager=org.jboss.logmanager.LogManager"
JAVA_ARGS="$JAVA_OPTS
-Dorg.jboss.logging.Logger.pluginClass=org.jboss.logging.logmanager.LoggerPluginIt

```

3. Add the following javaagent argument to the server start script.

```
export JAVA_OPTS="$JAVA_OPTS -javaagent:/agent_install_dir/javaagent.jar"
```

4. Restart the application server. The application server must be restarted for the changes to take effect.

To add the javaagent command in a Linux environment for JBoss AS 7.x

1. Open the standalone.conf file.
2. Search for the following line in standalone.conf.

```
JBOSS_MODULES_SYSTEM_PKGS="org.jboss.byteman"
```

Add the com.singularity and org.jboss.logmanager packages to that line as follows:

```
JBOSS_MODULES_SYSTEM_PKGS="org.jboss.byteman,com.singularity,org.jboss.logmanager"
```

For JBoss 7.1.1, add the additional packages as shown here:

```
JBOSS_MODULES_SYSTEM_PKGS="org.jboss.byteman,com.appdynamics,com.appdynamics.,com
```

3. Add the following to the end of the standalone.conf file in the JAVA_OPTION section.

```
-Djava.util.logging.manager=org.jboss.logmanager.LogManager  
-Xbootclasspath/p:<JBoss-DIR>/modules/org/jboss/logmanager/main/jboss-logmanager-
```

Note: The path for the necessary JAR files may differ for different versions. Provide the correct path of these JAR files for your version. If any of the packages are not available with the JBoss ZIP, download the missing package and add it to the path.

4. In the standalone.sh file, add the following javaagent argument.

```
export JAVA_OPTS="$JAVA_OPTS -javaagent:/agent_install_dir/javaagent.jar"
```

above the following section of standalone.sh

```
...  
while true;do  
if [ "$LAUNCH_JBOSS_IN_BACKGROUND" = "X" ]; then  
# Execute the JVM in the foreground  
eval \"$JAVA\" -D\"[Standalone]\"$JAVA_OPTS \  
\"-Dorg.jboss.boot.log.file=$JBOSS_LOG_DIR/boot.log\" \  
\"-Dlogging.configuration=file:$JBOSS_CONFIG_DIR/logging.properties\" \  
-jar \"$JBOSS_HOME/jboss-modules.jar\" \  
...
```

The revised section of your startup script file should look similar to the following image:

```

173  echo ""
174
175  export JAVA_OPTS="$JAVA_OPTS -javaagent:/agent_install_dir/javaagent.jar"
176
177  while true; do
178      if [ "x$LAUNCH_JBOSS_IN_BACKGROUND" = "x" ]; then
179          # Execute the JVM in the foreground
180          eval \'$JAVA\' -D\"[Standalone]\" $JAVA_OPTS \
181              \\"-Dorg.jboss.boot.log.file=$JBOSS_LOG_DIR/boot.log\" \
182              \\"-Dlogging.configuration=file:$JBOSS_CONFIG_DIR/logging.properties\" \
183              \\"-jar \"$JBOSS_HOME/jboss-modules.jar\" \
184              \\"-mp \"$JBOSS_MODULEPATH\" \
185              \\"-jaxpmodule \"javax.xml.jaxp-provider\" \
186              org.jboss.as.standalone \
187              \\"-Djboss.home.dir=\"$JBOSS_HOME\" \
188              \"\$@\"
189          JBOSS_STATUS=\$?
190      else

```

5. End User Monitoring (EUM) is not supported on JBoss 7 and you must disable it using the eum-disable-filter-injection agent configuration property. This property is available in AppDynamics App Agent for Java, version 3.5.2 and newer.

- In the Controller UI left navigation pane, select **Servers > App Servers<tier> -> <node>**.
- Click the **Agents** tab and then the **App Server Agent** subtab.
- Click **Configure**.
- In the App Server Agent Configuration, select the node.
- Click Add (the plus symbol) and enter the eum-disable-filter-injection property. Its type is Boolean and its value is "true".
- Click **Save**.

6. Restart the application server. The application server must be restarted for the changes to take effect.

To add the javaagent command in a Windows environment for JBoss AS 7.x

- Open the bin\standalone.conf file.
- Search for the line JBOSS_MODULES_SYSTEM_PKGS="org.jboss.byteman" and the com.singularity ad org.jboss.logmanager packages to that line as follows:

```
JBOSS_MODULES_SYSTEM_PKGS="org.jboss.byteman,com.singularity,org.jboss.logmanager"
```

- Open the bin\standalone.conf.bat file.

4. Search for the following line:
[code]
set "JAVA_OPTS=%JAVA_OPTS% -Djboss.modules.system.pkgs=org.jboss.byteman

```
set "JAVA_OPTS=%JAVA_OPTS%
-Djboss.modules.system.pkgs=org.jboss.byteman,com.singularity"
```

- Save the file.
- Open the standalone.bat file.
- Add the following javaagent argument to the standalone.bat file.

```

:RESTART
"%JAVA%" -javaagent:<AGENT-DIR>javaagent.jar %JAVA_OPTS% ^
"-Dorg.jboss.boot.log.file=%JBOSS_HOME%\standalone\log\boot.log" ^
"-Dlogging.configuration=file:%JBOSS_HOME%\standalone\configuration\logging.properties"
^
-jar "%JBOSS_HOME%\jboss-modules.jar" ^

```

8. Save the file.
9. End User Monitoring (EUM) is not supported on JBoss 7 and you must disable it using the eum-disable-filter-injection agent configuration property. This property is available in AppDynamics App Agent for Java, version 3.5.2 and newer.
 - a. In the Controller UI left navigation pane, select **Servers > App Servers<tier> -> <node>**.
 - b. Click the **Agents** tab and then the **App Server Agent** subtab.
 - c. Select the node.
 - d. Click Add (the plus symbol) and enter the eum-disable-filter-injection property. Its type is Boolean and its value is "true".
 - e. Click **Save**.

10. Restart the application server. The application server must be restarted for the changes to take effect.

Jetty Startup Settings

- To add the javaagent command in a Jetty environment

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option. Add this option to your jetty.sh file.

To add the javaagent command in a Jetty environment

1. Open the jetty.sh start script file.
2. Add the following javaagent argument to the beginning of the script.

```
java -javaagent:/<agent_home>/javaagent.jar
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

3. Save the script file.
4. Restart the application server for the changes to take effect.

Oracle WebLogic Startup Settings

- To add the javaagent command in a Windows environment
- To add the javaagent command in a Linux environment

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option. Add this option to your startWebLogic.sh or startWebLogic.cmd file.

To add the javaagent command in a Windows environment

1. Open the startWebLogic.cmd file, located at <weblogic_version_install_dir>\user_projects\domains\<domain_name>\bin.
2. Add following javaagent argument to the beginning of your application server start script.

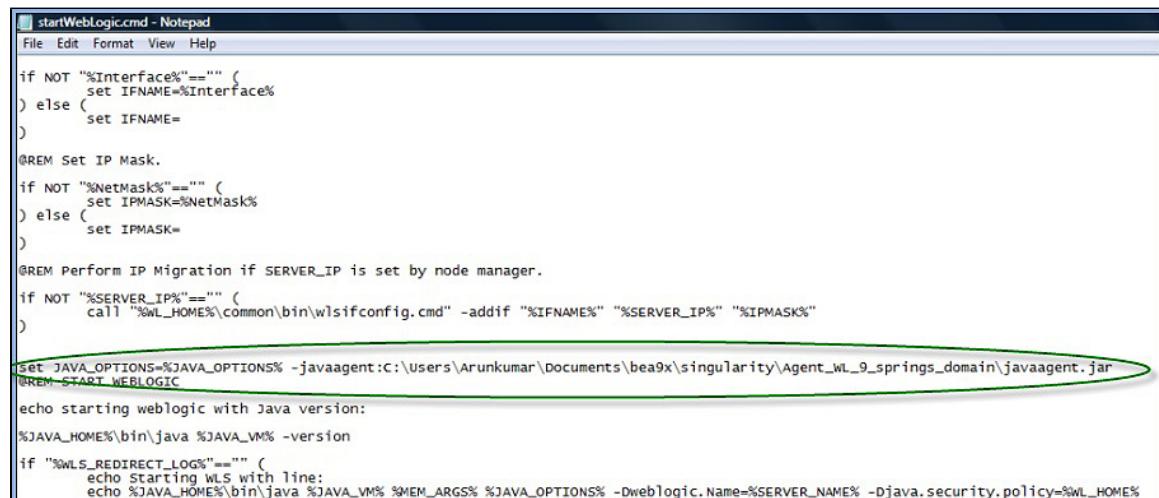
```
set JAVA_OPTIONS=% JAVA_OPTIONS%
-javaagent:"<drive>:\<agent_home>\javaagent.jar"
```

⚠ If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
-javaagent:<drive>:\<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>"
```

The javaagent argument references the full path of the App Server Agent installation directory, including the drive.

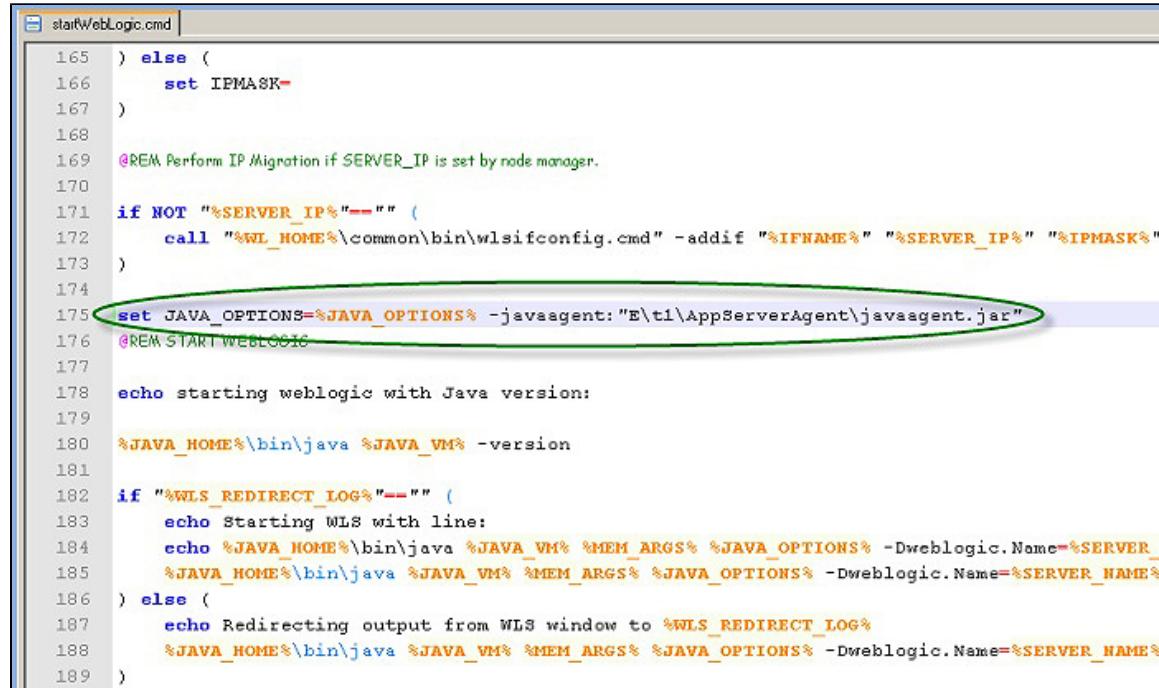
2a. Sample WebLogic v9.x startWebLogic.cmd file



```
startWebLogic.cmd - Notepad
File Edit Format View Help
if NOT "%Interface%"==""
) else (
    set IFNAME=%Interface%
)
@REM Set IP Mask.
if NOT "%NetMask%"==""
) else (
    set IPMASK=%NetMask%
)
@REM Perform IP Migration if SERVER_IP is set by node manager.
if NOT "%SERVER_IP%"==""
    call "%WL_HOME%\common\bin\wlsifconfig.cmd" -addif "%IFNAME%" "%SERVER_IP%" "%IPMASK%"
)

set JAVA_OPTIONS=%JAVA_OPTIONS% -javaagent:c:\users\Arunkumar\Documents\bea9x\singularity\Agent_WL_9_springs_domain\javaagent.jar
@REM START WEBLOGIC
echo starting weblogic with Java version:
%JAVA_HOME%\bin\java %JAVA_VM% -version
if "%WLS_REDIRECT_LOG%"==""
    echo Starting WLS with line:
    echo %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME% -Djava.security.policy=%WL_HOME%
```

2b. Sample WebLogic v10.x startWebLogic.cmd file



```
startWebLogic.cmd
165 ) else (
166     set IPMASK=
167 )
168
169 @REM Perform IP Migration if SERVER_IP is set by node manager.
170
171 if NOT "%SERVER_IP%"==""
172     call "%WL_HOME%\common\bin\wlsifconfig.cmd" -addif "%IFNAME%" "%SERVER_IP%" "%IPMASK%"
173
174
175 set JAVA_OPTIONS=%JAVA_OPTIONS% -javaagent:"E:\ti\AppServerAgent\javaagent.jar"
176 @REM START WEBLOGIC
177
178 echo starting weblogic with Java version:
179
180 %JAVA_HOME%\bin\java %JAVA_VM% -version
181
182 if "%WLS_REDIRECT_LOG%"==""
183     echo Starting WLS with line:
184     echo %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
185     %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
186 ) else (
187     echo Redirecting output from WLS window to %WLS_REDIRECT_LOG%
188     %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
189 )
```

3. Restart the application server. The application server must be restarted for the changes to take effect.

To add the javaagent command in a Linux environment

1. Open the startWebLogic.sh file, located at <weblogic_<version#>_install_dir>/user_projects/domains/<domain_name>/bin.

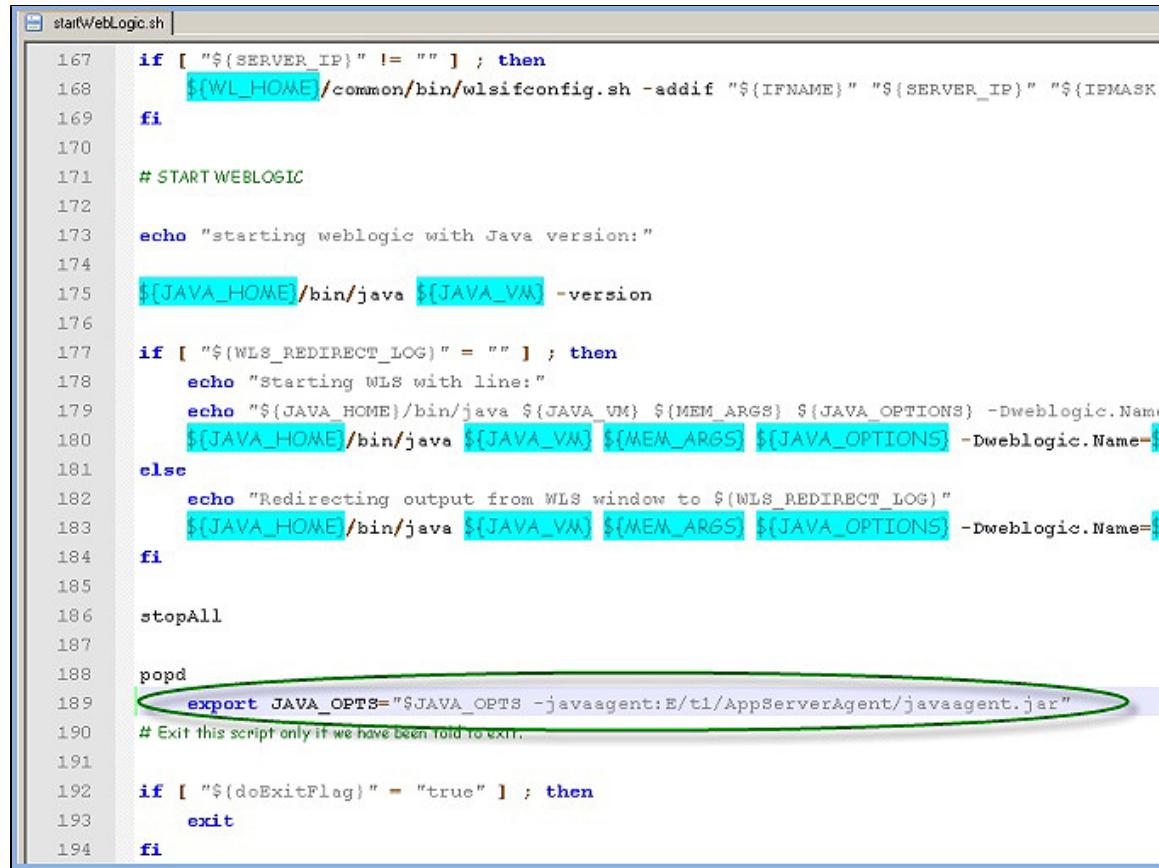
2. Add the following lines of code to the beginning of your application server start script.

```
export JAVA_OPTIONS="$JAVA_OPTIONS -javaagent:/agent_home/javaagent.jar"
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

The javaagent argument references the full path of the App Server Agent installation directory. For details see the screen captures.

2a. Sample WebLogic v9.x startWebLogic.sh file



```
startWebLogic.sh
167 if [ "${SERVER_IP}" != "" ] ; then
168     ${WL_HOME}/common/bin/wlsifconfig.sh -addif "${IFNAME}" "${SERVER_IP}" "${IPMASK}"
169 fi
170
171 # START WEBLOGIC
172
173 echo "starting weblogic with Java version:"
174
175 ${JAVA_HOME}/bin/java ${JAVA_VM} -version
176
177 if [ "${WLS_REDIRECT_LOG}" = "" ] ; then
178     echo "Starting WLS with line:"
179     echo "${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} ${JAVA_OPTIONS} -Dweblogic.Name=${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} ${JAVA_OPTIONS} -Dweblogic.Name="
180 else
181     echo "Redirecting output from WLS window to ${WLS_REDIRECT_LOG}"
182     ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} ${JAVA_OPTIONS} -Dweblogic.Name=
183 fi
184
185 stopAll
186
187 popd
188
189 export JAVA_OPTS="$JAVA_OPTS -javaagent:/agent_home/javaagent.jar"
190 # Exit this script only if we have been told to exit.
191
192 if [ "${doExitFlag}" = "true" ] ; then
193     exit
194 fi
```

2b. Sample WebLogic v10.x startWebLogic.sh file

```

startWebLogic.sh
167 if [ "${SERVER_IP}" != "" ] ; then
168     ${WL_HOME}/common/bin/wlsifconfig.sh -addif "${IFNAME}" "${SERVER_IP}" "${IPMASK}"
169 fi
170
171 # START WEBLOGIC
172
173 echo "starting weblogic with Java version:"
174
175 ${JAVA_HOME}/bin/java ${JAVA_VM} -version
176
177 if [ "${WLS_REDIRECT_LOG}" = "" ] ; then
178     echo "Starting WLS with line:"
179     echo "${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} ${JAVA_OPTIONS} -Dweblogic.Name=${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} ${JAVA_OPTIONS} -Dweblogic.Name=${}
180 else
181     echo "Redirecting output from WLS window to ${WLS_REDIRECT_LOG}"
182     ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} ${JAVA_OPTIONS} -Dweblogic.Name=${}
183 fi
184
185 stopAll
186
187 popd
188
189 export JAVA_OPTS="$JAVA_OPTS -javaagent:B/tl/AppServerAgent/javaagent.jar"
190 # Exit this script only if we have been told to exit.
191
192 if [ "${doExitFlag}" = "true" ] ; then
193     exit
194 fi

```

3. Restart the application server. The application server must be restarted for the changes to take effect.

OSGi Infrastructure Configuration

- Configuring OSGi Containers
 - To configure Equinox
 - To configure Apache Sling
 - To configure Felix for GlassFish
 - For GlassFish 3.1.2
 - To configure Felix for Jira or Confluence
 - Jira 5.x and newer
 - To configure other OSGi-based containers

Configuring OSGi Containers

The GlassFish application server versions 3.x and later uses OSGi architecture. By default, OSGi containers follow a specific model for bootstrap class delegation. Classes that are not specified in the container's CLASSPATH are not delegated to the bootstrap classloader; therefore you must configure the OSGi containers for the App Server Agent classes.

For more information see [GlassFish OSGi Configuration per Domain](#).

To ensure that the OSGi container identifies the agent, specify the following package prefix:

org.osgi.framework.bootdelegation=com.singularity.*

This prefix follows the regular boot delegation model so that the App Server Agent classes are visible.

If you already have existing boot delegations, add "com.singularity.*" to the existing path separated by a comma. For example:

org.osgi.framework.bootdelegation=com.sun.btrace., com.singularity.

To configure Equinox

1. Open the config.ini file located at <glassfish-install>/glassfish/osgi/equinox/configuration.
2. Add following package prefix to the config.ini file:

```
org.osgi.framework.bootdelegation=com.singularity.*
```

For more information see [Getting Started with Equinox](#).

To configure Apache Sling

1. Open the sling.properties file. The location of the sling.properties varies depending on the Java platform. In the Sun/Oracle implementation, the sling.properties file is located at <java.home>/lib.
2. Add following package prefix to the sling.properties file.

```
org.osgi.framework.bootdelegation=com.singularity.*
```

To configure Felix for GlassFish

1. Open the config.properties file, located at <glassfish-install>/glassfish/osgi/felix/conf.
2. Add following package prefix to the config.properties file.

```
org.osgi.framework.bootdelegation=com.singularity.*
```

For GlassFish 3.1.2

Add:

```
com.singularity.*
```

to the boot delegation list in the <GlassFish_Home_Directory>\glassfish\config\osgi.properties file. For example:

```
org.osgi.framework.bootdelegation=${eclipselink.bootdelegation}, com.sun.btrace,  
com.singularity.*
```

To configure Felix for Jira or Confluence

1. From <atlassian-install>/bin (Stand-alone) or <Tomcat-home>/bin (EAR-WAR installation), open:
 - For Linux: setenv.sh
 - For Windows: setenv.bat
2. Update the Java options:
 - For Linux: JAVA_OPTS=
 - For Windows: set JAVA_OPTS=%JAVA_OPTS%

```
-javaagent:<path>/javaagent.jar  
-Datlassian.org.osgi.framework.bootdelegation=<other_services>,com.singularity,  
com.singularity.*,<other_services>
```

Jira 5.x and newer

```
-Datlassian.org.osgi.framework.bootdelegation=META-INF.services,com.yourkit,com.yo
```

To configure other OSGi-based containers

For other OSGI-based runtime containers, add the following package prefix to the appropriate OSGi configuration.

```
file.org.osgi.framework.bootdelegation=com.singularity.*
```

Resin Startup Settings

- To Configure Resin 1.x - 3.x
 - To add the javaagent command in a Windows environment
 - To add the javaagent command in a Linux environment
- To Configure Resin 4.x

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option. Add this option to the resin.sh or resin.bat file.

To Configure Resin 1.x - 3.x

To add the javaagent command in a Windows environment

1. Open the resin.bat file, located at <Resin_installation_directory>/bin.
2. Add following javaagent argument to the beginning of your application server start script.

```
exec JAVA_EXE -javaagent:"<drive>:\<agent_home>\javaagent.jar"
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
-javaagent:"<drive>:\<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>"
```

The javaagent argument references the full path of the App Server Agent installation directory, including the drive.

3. Restart the application server for changes to take effect.

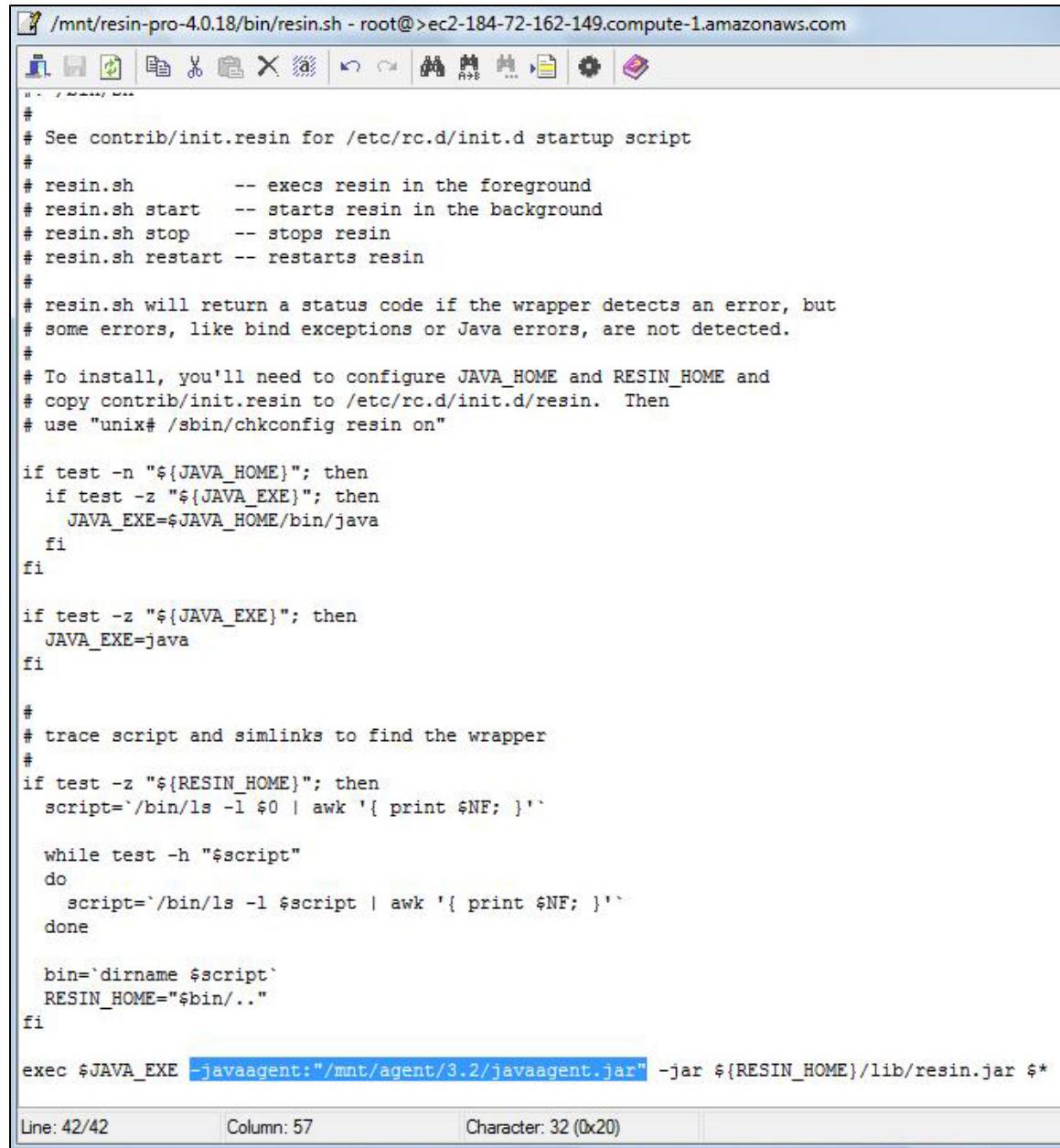
To add the javaagent command in a Linux environment

1. Open the resin.sh file, located at <Resin_Installation_Directory>/bin.
2. Add the following javaagent argument to the beginning of your application server start script.

```
exec $JAVA_EXE -javaagent:<agent_home>/javaagent.jar"
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

The javaagent argument references the full path of the App Server Agent installation directory. See the following screenshot.



```
/mnt/resin-pro-4.0.18/bin/resin.sh - root@>ec2-184-72-162-149.compute-1.amazonaws.com
# 
# See contrib/init.resin for /etc/rc.d/init.d startup script
#
# resin.sh      -- execs resin in the foreground
# resin.sh start -- starts resin in the background
# resin.sh stop   -- stops resin
# resin.sh restart -- restarts resin
#
# resin.sh will return a status code if the wrapper detects an error, but
# some errors, like bind exceptions or Java errors, are not detected.
#
# To install, you'll need to configure JAVA_HOME and RESIN_HOME and
# copy contrib/init.resin to /etc/rc.d/init.d/resin. Then
# use "unix# /sbin/chkconfig resin on"

if test -n "${JAVA_HOME}"; then
  if test -z "${JAVA_EXE}"; then
    JAVA_EXE=${JAVA_HOME}/bin/java
  fi
fi

if test -z "${JAVA_EXE}"; then
  JAVA_EXE=java
fi

#
# trace script and simlinks to find the wrapper
#
if test -z "${RESIN_HOME}"; then
  script=`/bin/ls -l $0 | awk '{ print $NF; }'`

  while test -h "$script"
  do
    script=`/bin/ls -l $script | awk '{ print $NF; }'`
  done

  bin='dirname $script'
  RESIN_HOME="$bin/.."
fi

exec $JAVA_EXE -javaagent:"/mnt/agent/3.2/javaagent.jar" -jar ${RESIN_HOME}/lib/resin.jar $*
```

Line: 42/42 Column: 57 Character: 32 (0x20)

3. Restart the application server. The application server must be restarted for the changes to take effect.

To Configure Resin 4.x

1. To install the App Server Agent into Resin 4.X or later, edit the ./conf/resin.xml file and add:

```
<jvm-arg>-Xmx512m</jvm-arg>
<jvm-arg>-javaagent:<$appagent_location>/javaagent.jar</jvm-arg>
```

2. Restart the application server. The application server must be restarted for the changes to take effect.

Solr Startup Settings

- To add the javaagent command in a Windows environment
- To add the javaagent command in a Linux environment

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option. Add this option to your Solr server.

To add the javaagent command in a Windows environment

1. Open the Windows command line utility.
2. Execute the following commands to add the javaagent argument to the Solr server:

```
>cd $Solr_Installation_Directory
>java -javaagent:"<drive>:\<agent_home>\javaagent.jar" -jar start.jar
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
-javaagent:"<drive>:\<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>"
```

The javaagent argument references the full path to the App Server Agent installation directory, including the drive. For details see the screenshots.

```
cmd Select C:\Windows\System32\cmd.exe - java -javaagent:"F:\lab\3.3\t\AppServerAgent-3.2.0.1GA1\javaagent.jar" -jar start.jar
F:\lab\myinstallations\apache-solr-3.2.0\example>java -javaagent:"F:\lab\3.3\t\AppServerAgent-3.2.0.1GA1\javaagent.jar" -jar start.jar
Install Directory resolved to F:\lab\3.3\t\AppServerAgent-3.2.0.1GA1
[INFO]: JavAgent - Using Agent Version Iserver Agent v3.2.0.1 GH Build Date 2011-05-10 16:58]
[INFO]: JavAgent - Using Agent Edition Iserver
[INFO]: AgentInstallManager - AppDynamics Agent edition Iserver
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver is running
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver using application name [solrtest]
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver using tier name [solr1]
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver using node name [solr1]
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver finished running
[INFO]: AgentInstallManager - Agent runtime directory set to [F:\lab\3.3\t\AppServerAgent-3.2.0.1GA1]
[INFO]: AgentInstallManager - Agent node directory set to [solr1]
[INFO]: JavAgent - Agent Directory [F:\lab\3.3\t\AppServerAgent-3.2.0.1GA1]
Agent logging directory [F:\lab\3.3\t\AppServerAgent-3.2.0.1GA1\logs\solr1]
Registering obfuscated agent
Registered app server agent with Node ID[27] Component ID[30] Application ID [24]
Started AppDynamics Java Agent Successfully.
```

To add the javaagent command in a Linux environment

1. Open the terminal.
2. Execute the following commands to add the javaagent argument to the Solr server:

```
>cd $Solr_Installation_Directory
>java -javaagent:"<agent_home>/javaagent.jar" -jar start.jar
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

The javaagent argument references the full path to the App Server Agent installation directory. For details see the screenshot.

```

root@domU-12-31-39-05-75-42:/mnt/solr/apache-solr-3.2.0/example
root@domU-12-31-39-05-75-42:/mnt/solr/apache-solr-3.2.0# ls
CHANGES.txt LICENSE.txt NOTICE.txt README.txt client contrib dist docs example
root@domU-12-31-39-05-75-42:/mnt/solr/apache-solr-3.2.0# cd example/
root@domU-12-31-39-05-75-42:/mnt/solr/apache-solr-3.2.0/example# java -javaagent:"/mnt/agent/3.2/test/javaagent.jar" -jar start.jar
[INFO]: JavAgent - Using Agent Version [Server Agent v3.2.1.0 GA Build Date 2011-06-03 20:53]
[INFO]: JavAgent - Running IBM Agent [No]
[INFO]: AgentInstallManager - AppDynamics Agent edition [server]
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver is running
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver using application name [casstest]
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver using tier name [test1]
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver using node name [nodextestcassandra]
[INFO]: AgentInstallManager - Full Agent Registration Info Resolver finished running
[INFO]: AgentInstallManager - Agent runtime directory set to [/mnt/agent/3.2/test]
[INFO]: AgentInstallManager - Agent node directory set to [nodextestcassandra]
[INFO]: JavAgent - Agent Directory [/mnt/agent/3.2/test]
Agent Logging Directory [/mnt/agent/3.2/test/logs/nodextestcassandra]
Running obfuscated agent
Registered app server agent with Node ID[28] Component ID[31] Application ID [25]
Started AppDynamics Java Agent Successfully.
2011-06-08 12:24:06.068:INFO::Logging to STDERR via org.mortbay.log.StdErrLog
2011-06-08 12:24:06.359:INFO::jetty-6.1-SNAPSHOT

```

Standalone JVM Startup Settings

- To add the javaagent command in a Windows environment
- To add the javaagent command in a Linux environment

AppDynamics works just as well with JVMs that are not application servers or containers.

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option, a standard Java option and can be used with any JVM. Add this option to your standalone JVM.

To add the javaagent command in a Windows environment

1. Open the command line utility for Windows.

2. Add javaagent argument to the standalone JVM:

```
>java -javaagent:"Drive:\agent_home\javaagent.jar"
<fully_qualified_class_name_with_main_method>
```

For example:

```
>java -javaagent:"C:\AppDynamics\agentDir\javaagent.jar" com.main.HelloWorld
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
-javaagent:<drive>:\<agent_home>\javaagent.jar=uniqueID=<my-app-jvm1>"
```

The javaagent argument references the full path to the App Server Agent installation directory, including the drive.

To add the javaagent command in a Linux environment

1. Open the terminal.

2. Add the javaagent argument to the standalone JVM:

```
>java -javaagent:"/agent_install_dir/javaagent.jar"  
<fully_qualified_class_name_with_main_method>
```

For example:

```
>java -javaagent:"/mnt/AppDynamics/agentDir/javaagent.jar" com.main.HelloWorld
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

The javaagent argument references the full path to the App Server Agent installation directory.

Tanuki Service Wrapper Configuration

- To configure the Tanuki Service Wrapper

The AppDynamics Java App Server Agent bootstraps using the javaagent command line option. Add this option to the Tanuki Service wrapper.conf file.

To configure the Tanuki Service Wrapper

1. Open the wrapper.conf file.
2. Use the wrapper.java.additional.<n> property to add the javaagent option.

```
wrapper.java.additional.6=-javaagent:/C:/agent/javaagent.jar
```

 If you are a Self-Service Trial user, add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

For more information see:

- [Tanuki Service Wrapper Properties](#)
- [Example Configuration](#)
- [More Help On Tanuki Service Wrapper](#)

Tibco BusinessWorks Configuration

There are typically two scripts associated with the Tibco BusinessWorks Services engine.

- my_application.sh
- my_application.tra

The JVM that runs the services start with a command line tool called bwengine(.exe).

Add the following to the .tra file:

```
Memory  
java.extended.properties -javaagent:/opt/appagent/javaagent.jar
```

See also: <https://ssl.tibcommunity.com/thread/14856>

Upgrade the App Agent for Java

This topic provides instructions for upgrading the App Agent for Java.

Upgrade Sequence

If you are upgrading both the Controller and agents, first upgrade the Controller and then upgrade the Agents.

 **Upgrade Note:** If you upgrade the Java App Server Agent to 3.3.3 and your environment has a JDBC URL greater than 500 characters, you must upgrade your Controller to version 3.3.3.

To upgrade the App Agent for Java

1. Shut down the application server where the App Agent for Java and the optional machine agent is installed.
2. Create a backup copy of the current agent installation directory and move the backup directory to a new location.
3. Download the latest release from [AppDynamics Download Center](#).
4. Extract the agent binaries. Where you extract them depends on the version you are upgrading from.
 - a. **From 3.1.x or earlier:** extract the Agent binaries to a new directory and rename old directory. Then rename the new directory the same name as the original one. Copy the controller-info.xml from the old Agent directory to the new Agent directory. At the end, you should have the new files using the same directory path as the previous one.
 - b. **From 3.2.x or later:** extract in the same directory. AppDynamics takes care of potential naming conflicts.
5. If you previously made changes to the <App_Server_Agent_Installation_Directory>/conf/app-agent-config.xml file, copy those changes to the new file.

Using the same directory path avoids the task of manually changing the agent-related configurations in your JVM startup script.

6. Restart the application server.

Uninstall the App Agent for Java

- [To uninstall the Java Agent](#)
- [Learn More](#)

This topic describes how to uninstall the App Agent for Java.

When a JVM is no longer instrumented by the App Agent for Java, it no longer uses a license.

To uninstall the Java Agent

1. Stop the application server on which the App Agent for Java is configured.
2. Remove the -javaagent argument in startup script of the JVM.
3. Remove any system properties configured for the App Agent for Java from the startup script of your JVM.
4. Restart the application server.

Learn More

- [Manage App Agents](#)
- [Install the App Agent for Java](#)
- [App Agent for Java Configuration Properties](#)

Administer App Agents for Java

App Agent for Java Configuration Properties

- Where to Configure App Agent Properties
 - Creating and Registering Tiers
- Example Java App Agent controller-info.xml File
- Example Startup-up Using System Properties
- Java App Server Agent Properties
 - Agent-Controller Communication Properties
 - Controller Host Property
 - Controller Port Property
 - Agent Identification Properties
 - Application Name Property
 - Tier Name Property
 - Node Name Property
 - Multi-Tenant Mode Properties
 - Account Name Property
 - Account Access Key Property
 - Proxy Properties for the Controller
 - Proxy Host Property
 - Proxy Port Property
 - Other Properties
 - Controller SSL Enabled Property
 - Enable Orchestration Property
 - Agent Runtime Directory Property
 - Redirect Logfiles Property
 - Force Agent Registration Property
 - Reuse Node Name Property
 - Auto Node Name Prefix Property
 - Chron/Batch JVM Property
 - Unique Host ID Property
- Learn More

Where to Configure App Agent Properties

You can configure the App Server Agent properties:

- in the controller-info.xml file in the <Agent_Installation_Directory>/conf directory
- in the system properties (-D options) in the JVM startup script

The system properties override the settings in the controller-info.xml file.

For shared binaries among multiple JVM instances, AppDynamics recommends using a combination of the xml file and the start-up properties to configure the app agent. Configure all the properties common to all the JVMs in the controller-info.xml file. Configure the properties unique to a JVM using the system properties in the start-up script.

For example:

- For multiple JVMs belonging to the same application serving different tiers, configure the application name in the controller-info.xml file and the tier name and node name using the system properties.
- For multiple JVMs belonging to the same application and the same tier, configure the application name and the tier name in the controller-info.xml file and the node name using the system properties.

After you configure agent properties, confirm that the javaagent argument has been added to the JVM startup script. For more information, see [Java Server-Specific Installation Settings](#).

For some properties, you can use system properties already defined in the start-up script as the App Server Agent property values. For more information, see [Configure App Agent for Java to Use Existing System Properties](#).

Creating and Registering Tiers

You can create a tier in the Controller prior to setting up any agents. Alternatively, an agent can register its tier with the Controller the first time, and only the first time, that it connects with the Controller. If a tier with the name used to connect already exists, the agent is associated with the existing tier.

Example Java App Agent controller-info.xml File

```

<?xml version="1.0" encoding="UTF-8"?>
<controller-info>

<controller-host>192.168.1.20</controller-host>

<controller-port>8090</controller-port>

<controller-ssl-enabled>false</controller-ssl-enabled>

<application-name>ACMEOnline</application-name>

<tier-name>InventoryTier</tier-name>

<node-name>Inventory1</node-name>

<agent-runtime-dir></agent-runtime-dir>

<enable-orchestration>false</enable-orchestration>

<account-name></account-name>
<account-access-key></account-access-key>

<force-agent-registration>false</force-agent-registration>

</controller-info>

```

Example Startup-up Using System Properties

The following command uses the system properties to start the agent that monitors the ACME Online sample application's Inventory tier.

```

java -javaagent:/home/appdynamics/AppServerAgent/
-DappDynamics.controller.hostName=192.168.1.20 -DappDynamics.controller.port=8090
-DappDynamics.agent.applicationName=ACMEOnline -DappDynamics.agent.tierName=Inventory
-DappDynamics.agent.nodeName=Inventory1 SampleApplication

```

Java App Server Agent Properties

This section describes the Java App Agent configuration properties, including their controller-info-xml elements and their system property options.

Agent-Controller Communication Properties

Controller Host Property

Description: This is the host name or the IP address of the AppDynamics Controller. Example values are 192.168.1.22 or myhost or myhost.abc.com. This is the same host that you use to access the AppDynamics browser-based user interface. For an on-premise Controller, use the value for Application Server Host Name that was configured when the Controller was installed. If you are using the AppDynamics SaaS Controller service, see the Welcome email from AppDynamics.

Element in controller-info.xml: <controller-host>

System Property: -Dappdynamics.controller.hostName

Type: String

Default: None

Required: Yes, if the Enable Orchestration property is false.

If Enable Orchestration is true, and if the app agent is deployed in a compute cloud instance created by an AppDynamics workflow, do not set the Controller host unless you want to override the auto-detected value. See [Enable Orchestration Property](#).

Controller Port Property

Description: This is the HTTP(S) port of the AppDynamics Controller. This is the same port that you use to access the AppDynamics browser-based user interface.

If the Controller SSL Enabled property is set to true, specify the HTTPS port of the Controller; otherwise specify the HTTP port. See [Controller SSL Enabled Property](#).

Element in controller-info.xml: <controller-port>

System Property: -Dappdynamics.controller.port

Type: Positive Integer

Default: For On-premise installations, port 8090 for HTTP and port 8181 for HTTPS are the defaults.
For the SaaS Controller Service, port 80 for HTTP and port 443 for HTTPS are the defaults.

Required: Yes, if the Enable Orchestration property is false.

If Enable Orchestration is true, and if the app agent is deployed in a compute cloud instance created by an AppDynamics workflow, do not set the Controller port unless you want to override auto-detected value. See [Enable Orchestration Property](#).

Agent Identification Properties

Application Name Property

Description: This is the name of the logical business application that this JVM node belongs to. Note that this is not the deployment name(ear/war/jar) on the application server.

If a business application of the configured name does not exist, it is created automatically.

Element in controller-info.xml: <application-name>

System Property: -Dappdynamics.agent.applicationName

Type: String

Default: None

Required: Yes

Tier Name Property

Description: This is the name of the logical tier that this JVM node belongs to. Note that this is not the deployment name (ear/war/jar) on the application server.

If the JVM / AppServer start-up script already has a system property that references the tier, such as -Dserver.tier, you could use \${server.tier} as the tier name. For more information, see [Configure App Agent for Java to Use Existing System Properties](#).

See [Name Business Applications, Tiers, and Nodes](#).

Element in controller-info.xml: <tier-name>

System Property: -Dappdynamics.agent.tierName

Type: String

Default: None

Required: Yes

Node Name Property

Description: This is the name of JVM node.

Where JVMs are dynamically created, use the system property to set the node name.

If your JVM / AppServer start-up script already has a system property that can be used as a node name, such as -Dserver.name, you

could use \${server.name} as the node name. You could also use expressions such as \${server.name}_\${host.name}.MyNode to define the node name. See [Configure App Agent for Java to Use Existing System Properties](#) for more information.

In general, the node name must be unique within the business application and physical host. If you want to use the same node name for multiple nodes on the same physical machine, create multiple virtual hosts using the Unique Host ID property. See [Unique Host ID Property](#).

See [Name Business Applications, Tiers, and Nodes](#).

Element in controller-info.xml: <node-name>

System Property: -Dappdynamics.agent.nodeName

Type: String

Default: None

Required: Yes

Multi-Tenant Mode Properties

Description: If the AppDynamics Controller is running in multi-tenant mode or if you are using the AppDynamics SaaS Controller, specify the account name and account access key for this agent to authenticate with the Controller. If you are using the AppDynamics SaaS Controller, the account name is provided in the Welcome email sent by AppDynamics.

If the Controller is running in single-tenant mode (the default) there is no need to configure these values.

Account Name Property

Description: This is the account name used to authenticate with the Controller.

Element in controller-info.xml: <account-name>

System Properties: -Dappdynamics.agent.accountName

Type: String

Default: None

Required: Yes for AppDynamics SaaS Controller and other multi-tenant users; no for single-tenant users.

Account Access Key Property

Description: This is the account access key used to authenticate with the Controller.

Element in controller-info.xml: <account-access-key>

System Properties: -Dappdynamics.agent.accountAccessKey

Type: String

Default: None

Required: Yes for AppDynamics SaaS Controller and other multi-tenant users; no for single-tenant users.

Proxy Properties for the Controller

These properties route data to the Controller through a proxy.

Proxy Host Property

Description: This is the proxy host name or IP address.

Element in controller-info.xml: Not applicable

System Property: -Dappdynamics.http.proxyHost

Type: String

Default: None

Required: No

Proxy Port Property

Description: This is the proxy HTTP(S) port.

Element in controller-info.xml: Not applicable

System Property: -Dappdynamics.http.proxyPort

Type: Positive Integer

Default: None

Required: No

Other Properties

Controller SSL Enabled Property

Description: When set to true, this property specifies that the agent should use SSL (HTTPS) to connect to the Controller. If SSL Enabled is true, set the Controller Port property to the HTTPs port of the Controller. See [Controller Port Property](#).

Element in controller-info.xml: <controller-ssl-enabled>

System Property: -Dappdynamics.controller.ssl.enabled

Type: Boolean

Default: False

Required: No

Enable Orchestration Property

Description: When set to true, enables auto-detection of the controller host and port when the app server is a compute cloud instance created by an AppDynamics orchestration workflow. See [Controller Host Property](#) and [Controller Port Property](#).

In a cloud compute environment, auto-detection is necessary for the Create Machine tasks in the workflow to run correctly.

If the host machine on which this agent resides is not created through AppDynamics workflow orchestration, this property should be set to false.

Element in controller-info.xml: <enable-orchestration>

System Property: Not applicable

Type: Boolean

Default: False

Required: No

Agent Runtime Directory Property

Description: This property sets the runtime directory for all runtime files (logs, transaction configuration) for nodes that use this agent installation. If this property is specified, all agent logs are written to <Agent-Runtime-Directory>/logs/node-name and transaction configuration is written to the <Agent-Runtime-Directory>/conf/node-name directory.

Element in controller-info.xml: <agent-runtime-dir>

System Property: -Dappdynamics.agent.runtime.dir

Type: String

Default: <Agent_Installation_Directory>/nodes

Required: No

Redirect Logfiles Property

Description: This property sets the destination directory to which to redirect log files for a node.

Element in controller-info.xml: Not applicable

System Property: -Dappdynamics.agent.logs.dir

Type: String

Default: <Agent_Installation_Directory>/logs/<Node_Name>

Required: No

Force Agent Registration Property

Description: Set to true only under the following conditions:

- The Agent has been moved to a new application and/or tier from the UI and
- You want to override that move by specifying a new application name and/or tier name in the agent configuration.

Element in controller-info.xml: <force-agent-registration>

System Property: Not applicable

Type: Boolean

Default: False

Required: No

Reuse Node Name Property

Description: Set this property if you want the Controller to generate unique node names automatically using a prefix.

You can specify the prefix in the [Auto Node Prefix Property](#). If you do not provide a prefix but set the reuse.nodeName property to true, the Controller uses the tier name as a prefix.

This property is useful for dynamic multi-tier clustered applications with many JVMs that have short life spans. It allows AppDynamics to reuse node names and to capture historical data for these short-lived nodes after they become historical or are deleted.

Element in controller-info.xml: Not applicable

System Property: -Dappdynamics.agent.reuse.nodeName

Type: Boolean

Default: None

Required: No

Auto Node Name Prefix Property

Description: Set this property if you want the Controller to generate node names automatically using a prefix that you provide.

The Controller generates node names based on the prefix concatenated with a number, which is incremented sequentially. For example, if you assign a value of "mynode" to this property, the Controller generates node names "mynode-1", "mynode-2" and so on.

If one of the nodes is deleted and the [Reuse Node Name Property](#) is true, the Controller will re-use the deleted node name.

Element in controller-info.xml: Not applicable

System Property: -Dappdynamics.agent.auto.node.prefix=<your_prefix>

Type: String

Default: Serial number maintained by the Controller appended to the tier name

Required: No

Chron/Batch JVM Property

Description: Set this property to true if the JVM is a batch/chron process or if you are instrumenting the main() method.

Element in controller-info.xml: Not applicable

System Property: -Dappdynamics.cron.vm

Type: Boolean

Default: False

Required: No

Unique Host ID Property

Description: This property logically partitions a single physical host or virtual machine.

You can use the unique host id when you want to use the same node name for multiple nodes on the same physical machine.

Set the value to a string that is unique across the entire managed infrastructure. The string may not contain any spaces.

If this property is set on the app agent, it must be set on the machine agent as well.

System Property: -Dappdynamics.agent.uniqueHostId

Type: String

Default: None

Required: No

Learn More

- Name Business Applications, Tiers, and Nodes
- Configure App Agent for Java for JVMs that are Dynamically Identified
- Configure App Agent for Java to Use Existing System Properties
- Java Agent on z-OS or Mainframe Environments Configuration

App Agent for Java Diagnostic Data

- To view agent diagnostic data
- To view agent diagnostic stats
- Learn More

To view agent diagnostic data

1. From an Application Dashboard, click **Actions -> View Agent Diagnostics**.

The Agent Diagnostic Data window opens and displays various metrics related to the application, tiers, and nodes.

Agent Diagnostic Data																		
		View Agent Diagnostic Events for selected Node																
Name	Overflow Calls	Transactions Successfully Registered	Transactions Registration Failed	Discovered Backends	Discovered Backends Success	Metrics Upload Requests Time Skew	Metrics Upload Invalid Metrics	Metrics Uploaded	Metrics Upload Requests Exceeding Limit	Metrics Upload Request License Errors	Snapshots Time Skew Errors	Snapshots With Invalid Data	Snapshots Uploaded	Snapshots Exceeding Limit	Events Time Skew Errors	Events Uploaded	Events Exceeding Limit	Application Infrastructure Changes
ACME Book Store Application	-	9	-	3	-	-	-	572	-	-	-	-	46	-	-	5	-	6
	-	9	-	3	-	-	-	333	-	-	-	-	15	-	-	2	-	3
	-	5	-	2	-	-	-	174	-	-	-	-	11	-	-	1	-	2
	-	3	-	1	-	-	-	159	-	-	-	-	4	-	-	1	-	2
	-	-	-	-	-	-	-	121	-	-	-	-	20	-	-	1	-	2
	-	1	-	-	-	-	-	118	-	-	-	-	11	-	-	1	-	2

2. Select a node and click **View Agent Diagnostic Events for selected Node**.

The Agent Diagnostics Event window opens.

To view agent diagnostic stats

1. From the left navigation pane, click *Servers -> App Servers -> <Tier> -> <Node>.
2. In the Node Dashboard, click the Agents tab.
3. Click the Agent Diagnostic Stats subtab.

Metric	Count
Overflow Calls	-
Transactions Successfully Registered	5
Transactions Registration Failed	-
Discovered Backends Successfully Registered	2
Discovered Backends Registration Failed	-
Metrics Upload Requests Time Skew Errors	-
Metrics Upload Invalid Metrics	-
Metrics Uploaded	174
Metrics Upload Requests Exceeding Limit	-
Metrics Upload Request License Errors	-
Snapshots Time Skew Errors	-
Snapshots With Invalid Data	-
Snapshots Uploaded	11
Snapshots Exceeding Limit	-
Events Time Skew Errors	-
Events Uploaded	1
Events Exceeding Limit	-
Application Infrastructure Changes Sent	2

[Learn More](#)

- [Troubleshoot Node Problems](#)

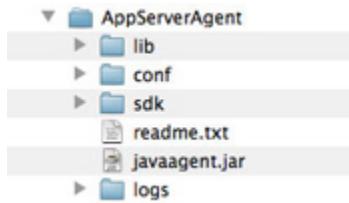
App Agent for Java Directory Structure

- [App Agent for Java Directory Structure](#)
- [The conf Directory](#)

App Agent for Java Directory Structure

The AppServerAgent.zip folder contains files for installing the App Agent for Java.

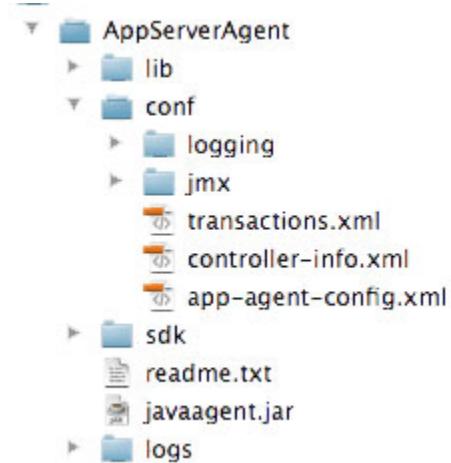
The directory structure for the agent is illustrated in the following screen shot.



- **lib:** The lib folder has the core agent libraries and the third-party libraries.
- **conf:** The conf folder is the configuration directory that has local configuration backup.
- **sdk:** This folder contains the Javadocs and APIs to extend AppDynamics' monitoring capabilities.
- **javaagent.jar:** This JAR file has the argument to bootstrap the App Agent for Java. To enable the agent, the --javaagent argument for JVM startup is required. See [Install the App Agent for Java](#).
- **logs:** All logs written by the agent in this directory.

The conf Directory

The files located in this directory are commonly used for agent configuration and deployment.



- **transactions.xml:** This XML file has configurations for business transaction identified by the agent.
- **controller-info.xml:** This XML file has controller connectivity and identification.

The following agent settings are configured using this XML file:

- Agent communication settings (specified using <controller-host> and <controller-port> properties).
- Agent identification settings (specified using <application-name>, <tier-name>, and <node-name> properties).
- **app-agent-config.xml:** This XML file contains agent configuration. Any changes to the agent's local settings are specified in this file, and these changes override the global configuration. This file is typically used for short-term properties settings or for

debugging agent issues.

- **jmx:** The jmx folder contains files for configuring the JMX and Websphere PMI metrics.
- **logging/log4j.xml:** This file contains flags to control logging levels for the agent. It is highly recommended not to change the default logging levels.

To specify a different log directory, use the following system property:

```
-Dappdynamics.agent.logs.dir
```

App Agent for Java Performance Tuning

- Business Transaction Thresholds
- Snapshot Collection Thresholds
 - Disable Scheduled Snapshots
 - Suggested Scheduling Settings
 - Suggested Diagnostic Session Settings
- Tuning Call Graph Settings
 - Suggested SQL Query Time and Parameters
- Memory Monitoring
- Learn More

This topic discusses how to get the best performance from App Agents for Java.

Business Transaction Thresholds

AppDynamics determines whether transactions are slow, very slow, or stalled based on the thresholds for Business Transactions. AppDynamics recommends using standard deviation based dynamic thresholds. See [Configure Thresholds](#).

Snapshot Collection Thresholds

Snapshot collection thresholds determine when snapshots are collected for a Business Transaction. Too many snapshots can affect performance and therefore snapshot collection thresholds should be considered carefully in production or load testing scenarios. See [Configure Thresholds](#).

Disable Scheduled Snapshots

For more information see [Transaction Snapshots](#).

Suggested Scheduling Settings

- 10 minutes for small deployments < 10 BTs
- 20 minutes for medium deployments < 10 - 50 BTs
- 30 minutes for > 50 - 100 BTs
- 60 minutes > 100 BTs

Suggested Diagnostic Session Settings

- Settings for Slow requests (%value): 20 - 30
- Settings for Error requests (%value): 10 - 20
- Click **Apply to all Business Transactions**.

Tuning Call Graph Settings

To configure call graph settings, click **Configure -> Instrumentation** and the **Call Graph Settings** tab. See [Configure Call Graphs](#).

Suggested SQL Query Time and Parameters

- Minimum SQL Query Time : 100 ms (Default is 10)
- Enable Filter SQL Parameter Values.

Memory Monitoring

Memory monitoring features such as leak detection, object instance tracking, and custom memory should be enabled only for a specific node or nodes when debugging a memory problem. Automatic leak detection is on-demand, and therefore, the leak detection will execute only for the specified duration.

When you observe periods of growth in the heap utilization (%), you should enable on-demand memory leak capture. See [Troubleshoot Java Memory Leaks](#).

Learn More

- Configure Thresholds
- Configure Call Graphs
- Troubleshoot Java Memory Leaks

Configure App Agent for Java for Batch Processes

- To configure the App Agent for Java
- To use the script name as the node name
- Learn More

You can configure the App Agent for Java for those JVMs that run as cron or batch jobs where the JVM runs only for the duration of the job. AppDynamics monitors the main method of the Java program.

To configure the App Agent for Java

1. Add the application and tier name to the controller-info.xml file.
2. Add the appdynamics.cron.vm property to the AppDynamics javaagent command in the startup script of your JVM process:

```
-javaagent:<agent_install_dir>/javaagent.jar
-Dappdynamics.agent.nodeName=${NODE_NAME} -Dappdynamics.cron.vm=true
```

The agent_install_dir is the full path of the App Agent for Java installation directory.

The appdynamics.cron.vm property creates a delay between the end of the main method and the JVM exit so that the Agent has time to upload metrics to the Controller.

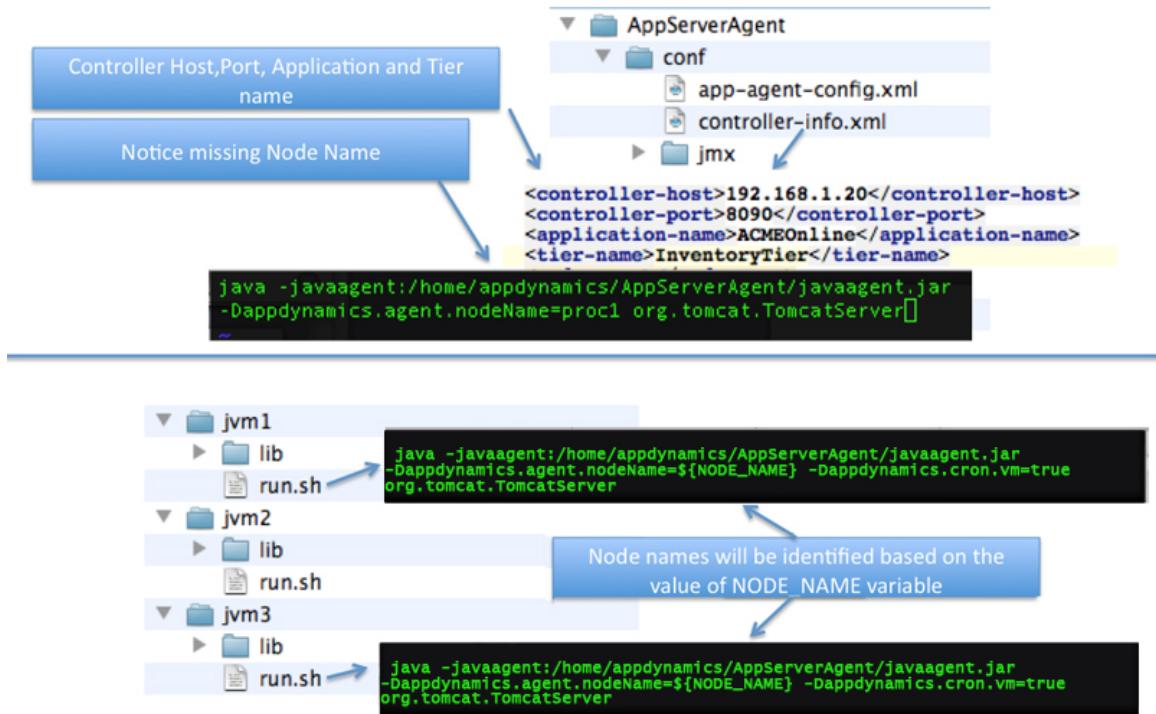
To use the script name as the node name

You can use the script name that executes the cron or batch job as the node name.

The following commands set the value of variable NODE_NAME using the combination of the script and host name. Add these commands to the startup script of the JVM.

```
# Use the name of the script (no path, no extension) as the name of the node.
NODE_NAME=sample
NODE_NAME="\${NODE_NAME%.*}"
echo $NODE_NAME
# Localize the script to the host.
NODE_NAME="\$NODE_NAME@\${HOSTNAME}"
```

The following illustration shows the sample configuration for controller-info.xml and the startup script of the JVM.



Learn More

- Configure Background Tasks (Java)

Configure App Agent for Java for JVMs that are Dynamically Identified

- To configure the node name of the App Agent for Java
- Configuration notes

This topic describes how to configure the App Agent for Java in environments where the JVMs are dynamic.

To configure the node name of the App Agent for Java

- Add the **application** and **tier name** to the `controller-info.xml` file.
- Add the `javaagent` argument and the following system properties (-D options) to the startup script of the JVMs:

```
java -javaagent:<agent-install-dir>/javaagent.jar -Dappdynamics.agent.nodeName=${NODE_NAME}
```

Configuration notes

The system properties are separated by a white space character.

The `<agent-install-dir>` references the full path of the App Agent for Java installation directory.

The token `${NODE_NAME}` identifies the JVMs dynamically and names these JVMs based on the parameter value passed during the execution of the startup script for your JVM process.

The application and tier names can also be configured using the system properties. For more details see [App Agent for Java Configuration Properties](#).

Some application server management consoles allow you to specify startup arguments using a web interface. For details see [Java Server-Specific Installation Settings](#).

Configure App Agent for Java in Restricted Environments

- To write the "startup hook" agent program

Some restricted environments do not allow any changes to the JVM startup script. For these environments AppDynamics provides the appdynamics.agent.startup.hook property. This "startup hook" allows a single point of deployment for the agent. You create a Java main method that is invoked programmatically, before your startup script is executed.

To write the "startup hook" agent program

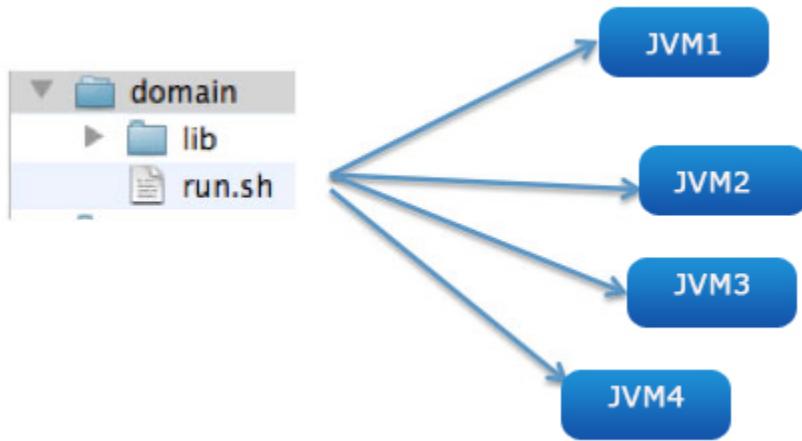
1. Implement a class with Java main method.
2. Create a JAR file for this class.
3. In the manifest of the JAR file, specify the class created in step 1.
4. Add the following javaagent argument and system properties (-D options) to your startup script:

```
-javaagent:<agent_install_dir>/javaagent.jar  
-Dappdynamics.agent.startup.hook=<JAR-file>
```

Configure App Agent for Java in z-OS or Mainframe Environments

- To name nodes automatically
- To remove dead nodes
- Learn More

In some environments JVMs have transient identity, such as when a single script spawns multiple JVMs.



For example, an environment may consist of WebSphere on IBM Mainframes, using a dynamic workload management feature that spawns new JVMs for an existing application server (called a servant). These JVMs are exact clones of an existing JVM, but each of them has a different process ID. Based on load, any number of additional JVMs may be created.

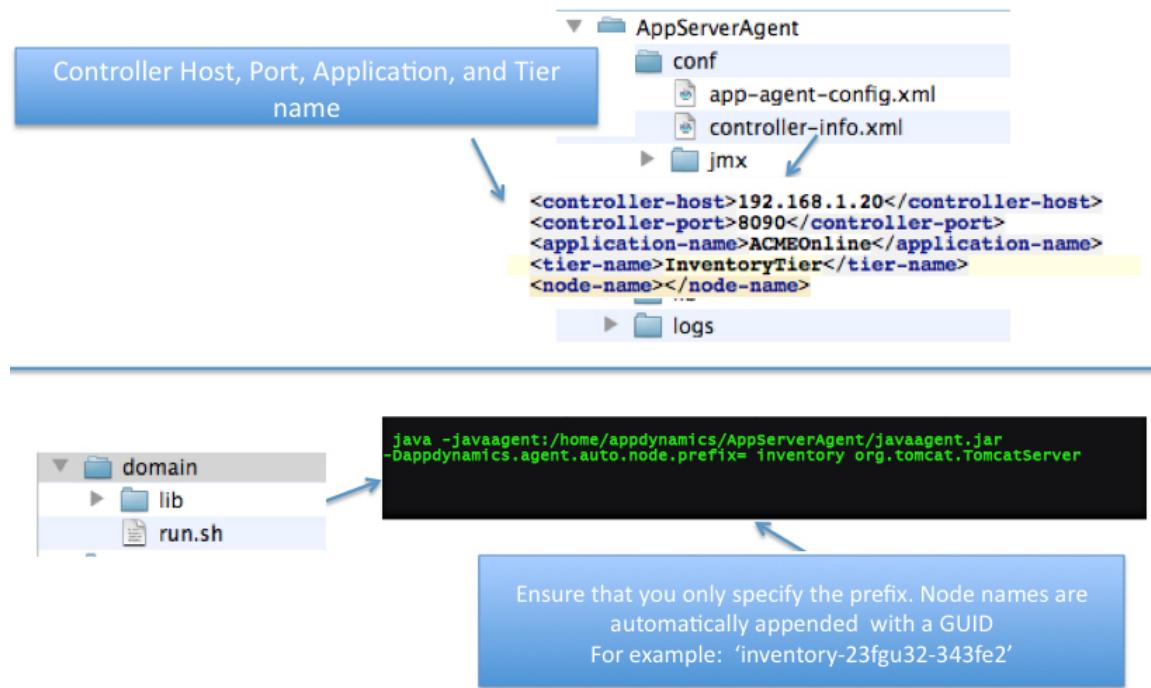
To name nodes automatically

The App Server Agent can automatically name the dynamically generated JVMs using the appdynamics.agent.auto.node.prefix property. See [Reuse Node Name Property](#) to enable re-use of node names and [Auto Node Name Prefix Property](#) to set the prefix used for automatically-named nodes. You used these properties in your startup script.

```
-Dappdynamics.agent.auto.node.prefix=<node name prefix>
-Dappdynamics.agent.agent.reuse.nodeName=true
```

If you are using these properties, ensure that you have not specified the node name anywhere (controller-info.xml file or as a system property) in your JVMs start-up script.

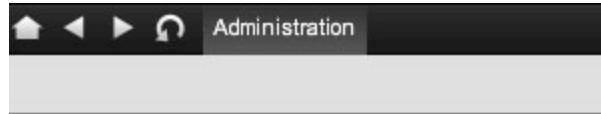
The following illustration shows the sample configuration for ACME Bookstore. This configuration will create unique node names for every instance of the virtual machine starting up in the ACME Bookstore environment.



To remove dead nodes

In a typical environment, the nodes are recycled and new nodes get generated. AppDynamics strongly recommends that you remove the dead nodes both from the AppDynamics user interface (UI) as well as from the system.

1. Log in to the Controller administration console.
2. Go to "Controller Settings" section to see the advanced properties for the Controller.



AppDynamics Administration

Accounts

Create and Manage Accounts

Controller Settings

Configure the Controller

3. Set the retention and deletion properties, based on the requirements for your environment. AppDynamics recommends that you set the permanent deletion period at least an hour more than the retention period.

- **node.permanent.deletion.period:** Time (in hours) after which a node that has lost contact with the Controller is deleted permanently from the system.
- **node.retention.period:** Time (in hours) after which a node that has lost contact with the Controller is deleted. In this case, the AppDynamics UI will not display the node, however the system will continue to retain it.

metrics.write.thread.count	The count of parallel threads to be i	<input type="text" value="1"/>
multitenant.controller	Is the controller running in multi-tier	<input type="text" value="false"/>
node.permanent.deletion.period	Time (in hours) after which a node t	<input type="text" value="720"/>
node.retention.period	Time (in hours) after which a node t	<input type="text" value="500"/>

Learn More

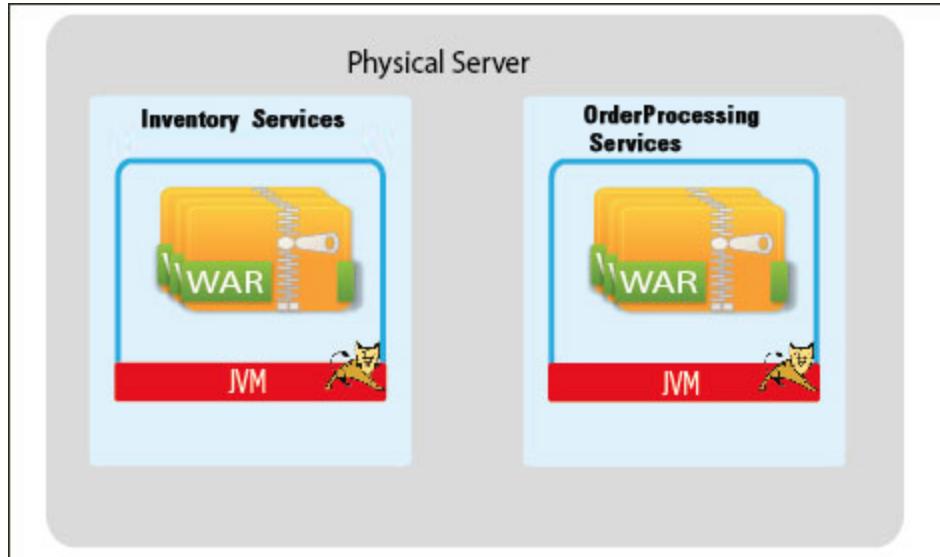
- [App Agent for Java Configuration Properties](#)

Configure App Agent for Java on Multiple JVMs on the Same Machine that Serve Different Tiers

- [To configure the App Agent for Java](#)

This section describes how you can configure the App Agent for Java for multiple JVMs that are located on a single machine and are serving two tiers.

For example, the ACME Bookstore has two JVMs on the same physical server. These two JVMs are bound to two different virtual IP (one JVM is used for Order Processing Services and the other JVM is used for Inventory Services).



For such cases, follow these rules:

- All of the common information should be configured using controller-info.xml file.
- All of the information unique to a JVM should be configured using the system properties in the JVM startup script.
- Information in the startup scripts always overrides the information in the controller-info.xml file.

To configure the App Agent for Java

1. Add **application name** to controller-info.xml file.
2. Add **javaagent** argument and following system properties (-D options) to the start-up script to each of your JVM:

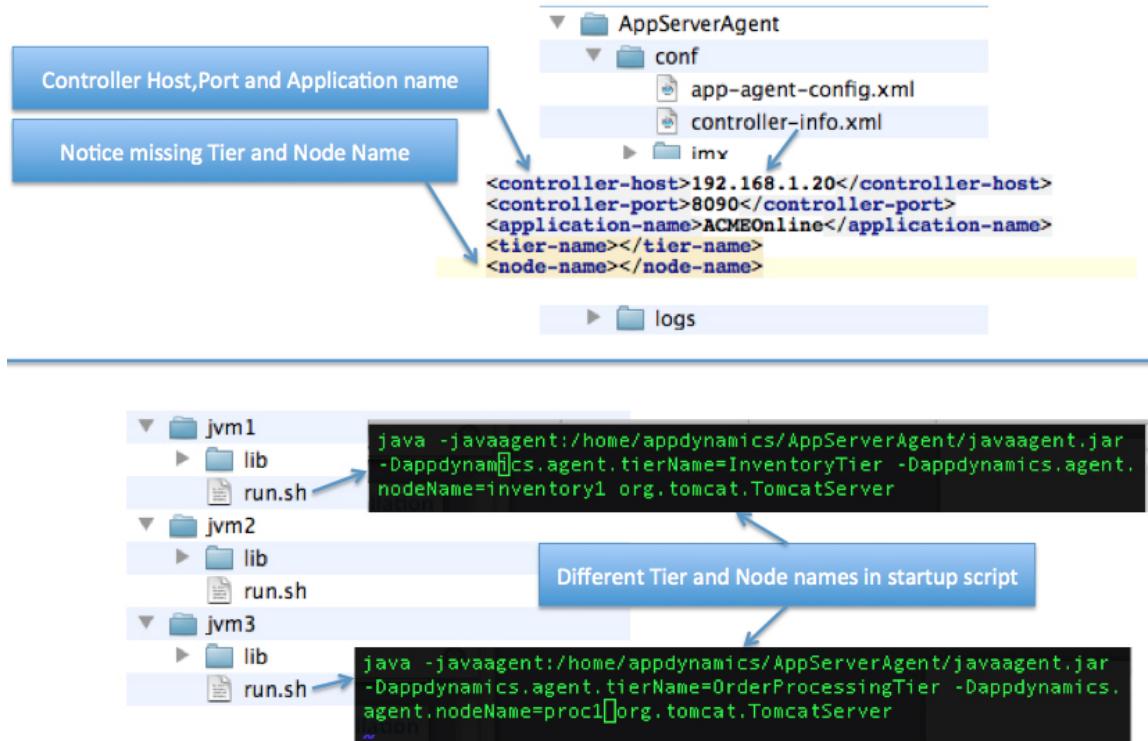
```
java -javaagent:<Agent-Installation-Directory>/javaagent.jar
-Dappdynamics.agent.tierName=$tierName -Dappdynamics.agent.nodeName=$nodeName
```

Separate the system properties with a white space character.

All agents in shared mode need a unique node name so that they can be differentiated from one another. See [Configure App Agent for Java to Use Existing System Properties](#).

The following illustration displays how this configuration is applied to ACME Bookstore:

Add Controller Host, Port, and Application Name in controller-info.xml file
and add rest of the properties in the start-up script.



Tips:

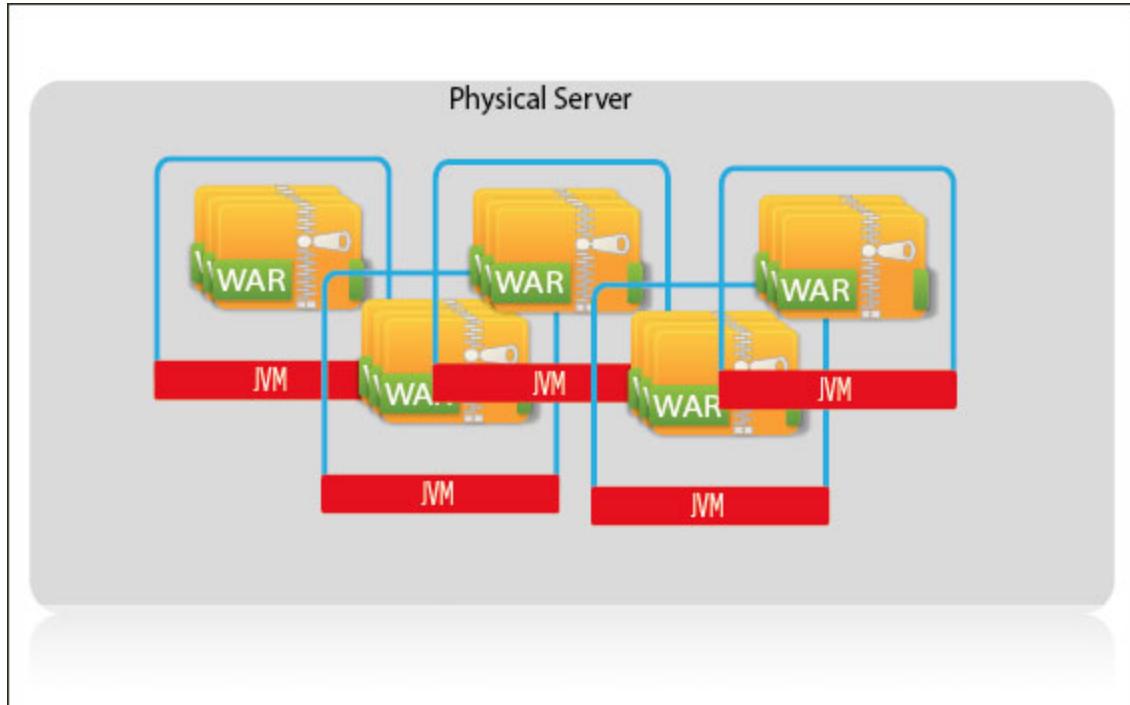
- The application and tier names can also be configured using the system properties. For more details see [App Agent for Java Configuration Properties](#).
- Some application server management consoles allow you to specify startup arguments using a web interface. See [Java Server-Specific Installation Settings](#).

Configure App Agent for Java on Multiple JVMs on the Same Machine that Serves the Same Tier

- To configure the App Agent for Java properties

This topic describes how to configure the App Agent for Java for multiple JVMs that are located on a single machine and are serving the same tier.

For example, ACME Bookstore has a physical server with five JVMs installed on it. All of these JVMs are used for the Inventory Services.



For such cases, follow these rules:

- All of the common information should be configured using controller-info.xml.
- All of the information unique to a JVM should be configured using the system properties in the start-up script.
- Information in the startup scripts always overrides the information in the controller-info.xml file.

To configure the App Agent for Java properties

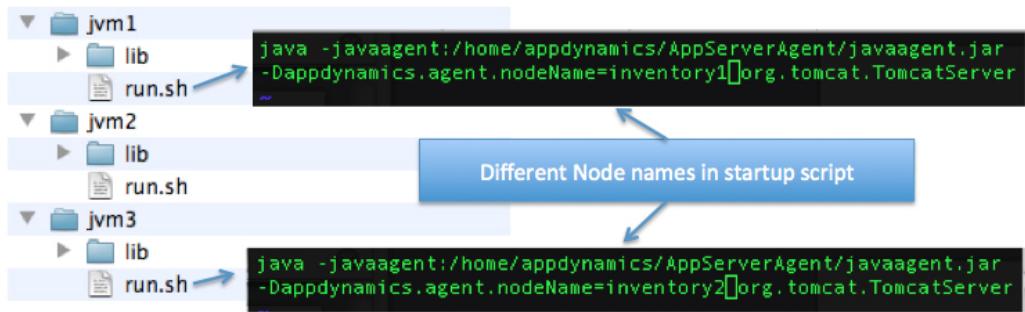
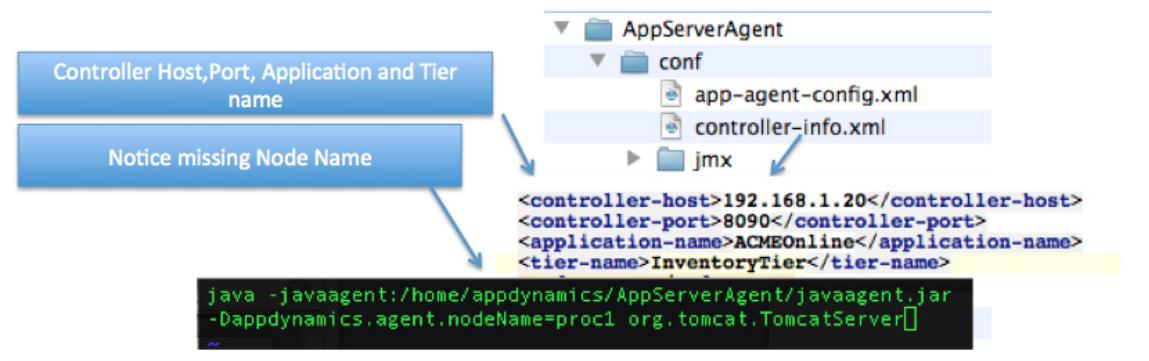
1. Provide the application and tier name in the controller-info.xml file.
2. Add the javaagent argument and system property (-D option) for the node name to the batch file or startup script of each JVM.

```
java -javaagent:<Agent-Installation-Directory>/javaagent.jar
-Dappdynamics.agent.nodeName=$nodeName
```

Separate the system properties with a white space character.

The following illustration displays how this configuration is applied to the ACME Bookstore.

Add Controller Host, Port, Application, and Tier Name in controller-info.xml file and add Node Name in the start-up script.



The application and tier names can also be configured using the system properties. See App Agent for Java Configuration Properties.

Some application server management consoles allow you to specify start-up arguments using a web interface. See Java Server-Specific Installation Settings.

Configure App Agent for Java to Use Existing System Properties

- System Properties
 - Using System Properties
 - To identify nodes
 - To identify tiers
 - Sample Agent Configuration Using System Properties
 - Learn More

This topic explains how to configure the App Agent for Java using the existing system property values.

System Properties

AppDynamics recommends that you use the existing system properties to configure the Agent when your environment consists of multiple JVMs on the same machine. Once you have these variables configured, you can complete Agent installation for all JVMs by simply adding the javaagent argument to each JVM startup script. Then add the rest of the information to the controller-info.xml file.

AppDynamics recommends that you use the system properties if the same startup script is starting all the JVMs in your environment.

You can identify the node name based on the value of -Dserver.name and the tier name based on the value of -Dcluster.name.

Also, you can combine two or more system properties to identify the node or tier name. You can use -Dhost.name and -Dserver.name to identify similarly named nodes on different machines even when they belong to the same tier.

Using System Properties

Use the following syntax to represent the value of the system property in the controller-info.xml file.

```
 ${system.property.name}
```

You can combine multiple system properties.

```
 ${host.name}${server.name}
```

You can combine system properties with literals. In the following example '_' and 'inventory' are literals.

```
 ${host.name}_${server.name}.inventory
```

To identify nodes

```
 ${host.name}
```

or

```
 ${server.name}
```

or

```
 ${host.name}${server.name}
```

To identify tiers

```
 ${cluster.name}
```

Sample Agent Configuration Using System Properties

Consider a JVM with a script file named startserver.sh. This script file has following system property:

```
 -Dserver.name=$1
```

If you execute:

```
 startserver.sh ecommerce01
```

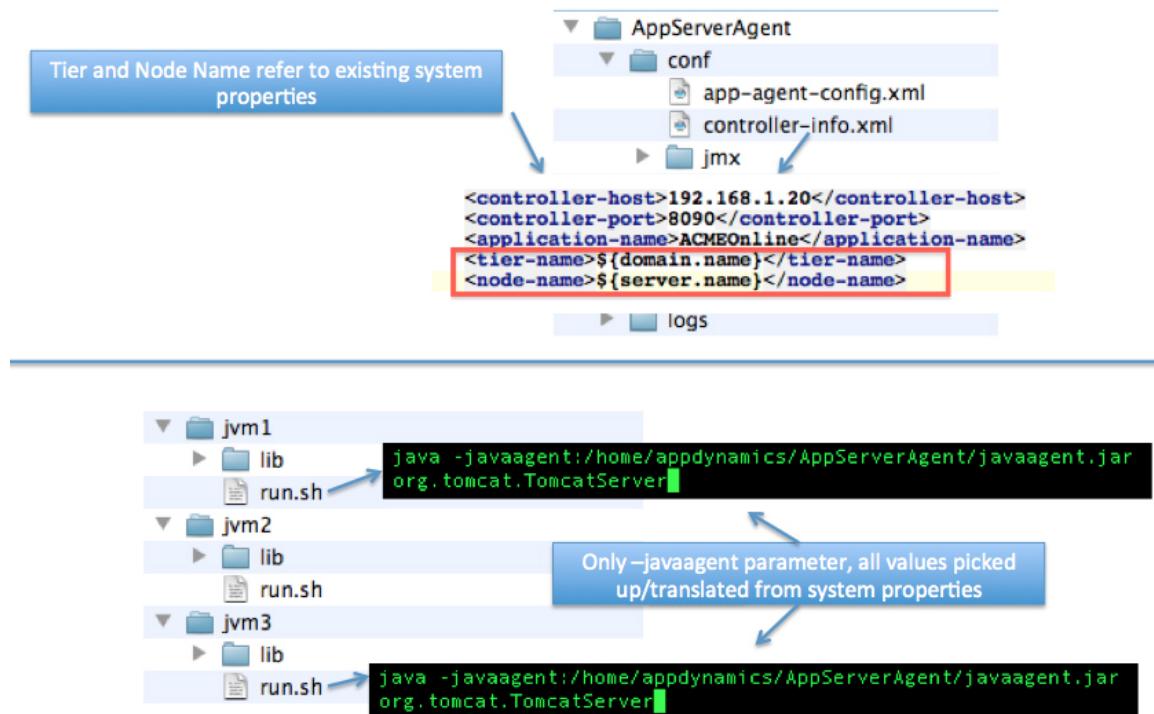
The script creates a new server named ecommerce01.

To use this system property for Agent configuration, add the appdynamics.nodeName property to startserver.sh file.

```
-Dappdynamics.nodeName=$server.name
```

When the script creates the new server named ecommerce01, it will be identified by AppDynamics as both a server and as a node.

The following screenshot shows a sample configuration for the controller-info.xml file and the startup script.



Learn More

- Logical Model
- Install the App Agent for Java

IBM App Agent for Java

- Supported JVMs
- Instrumenting the IBM App Agent for Java

Under most circumstances, the IBM App Agent for Java works the same as the App Agent for Java. This topic gathers information specific to the IBM App Agent for Java.

Supported JVMs

IBM JVM 1.5.x, 1.6.x

Instrumenting the IBM App Agent for Java

To change instrumentation for the IBM App Agent for Java, the IBM JVM must be restarted. By default the IBM App Agent for Java does not apply BCI changes without restarting the JVM. This is because in the IBM VM (J9 1.6.0) the implementation of re-transformation affects performance (changes the JIT behavior such that less optimization occurs).

The following changes require that you restart the IBM JVM.

- Automatic leak detection
- Custom memory structures
- Information points
- Aggressive snapshot policy (also called hotspot instrumentation)
- Custom POJO rules for transaction detection
- Custom exit point rules
- End user experience monitor (EUM), when you enable it and/or disable it after first enabling it

Move an App Agent for Java Node to a New Application or Tier

- **Moving a Node to a New Application or Tier**
 - To change the Java Agent associations from the UI
 - Optionally update the controller-info.xml file
 - Forcing node re-registration using the controller-info.xml file
 - Learn More

If your JVM machine has both an App Agent for Java and a Machine Agent, you cannot change the associations in the Machine Agent controller-info.xml file. You can only change these associations either through the UI or by modifying the App Agent for Java controller-info.xml file.

Moving a Node to a New Application or Tier

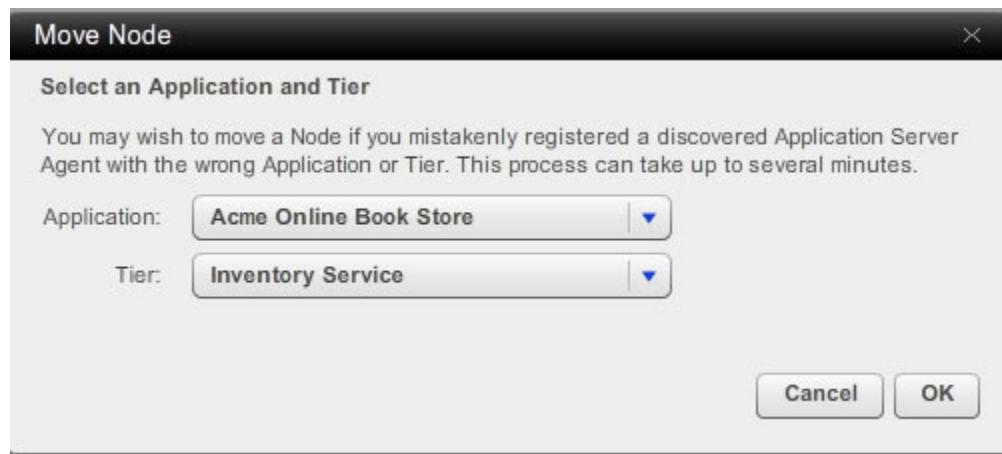
You can assign an App Agent for Java node to a new application or tier using the AppDynamics UI without restarting the JVM.

You can optionally update the controller-info.xml file and restart the JVM. You must restart the JVM when you change the associations in the controller-info.xml file.

Moving the node using the AppDynamics UI cannot be overridden by an agent configuration unless you set the force-agent-registration flag to true in the controller-info.xml file.

To change the Java Agent associations from the UI

1. Select the node that you want to move by clicking **Servers -> App Servers -> <Tier> -><Node>**.
2. Click **Actions -> Move Node**.
4. In the Move Node window select the new application and/or tier from the drop-down menus.
5. Click **OK** to confirm the reassignment. It may take several minutes to complete.



i When you change the associations for an App Agent for Java, AppDynamics registers an Application Changes event. You can review the details of the event in the [Events](#) view.

Optionally update the controller-info.xml file

The UI does not update the controller-info.xml file. However if you restart the JVM the Controller will remember and keep the change you made from the UI.

You may want to maintain consistency with the controller-info.xml file, just for neatness' sake. Perform these two additional steps:

6. Update these configuration changes in the App Agent for Java controller-info.xml file.
7. Restart the JVM.

If it is not feasible to restart the JVM at this time, remember the change and update the file the next time you restart the JVM.

Forcing node re-registration using the controller-info.xml file

If you've moved a node in the UI and you want to move it again elsewhere using controller-info.xml, then when you restart the JVM you set the force-agent-registration property to 'true'. See [App Agent for Java Configuration Properties#Force Agent Registration Property](#).

Learn More

- [Logical Model](#)

Troubleshoot App Agent for Java

- [Troubleshooting App Agent for Java Startup](#)
 - [Locating the App Agent for Java Log Files](#)
 - [Troubleshooting Incomplete Agent Configuration](#)
 - [Troubleshooting the Controller Port](#)
 - [Troubleshooting Incorrect File Permissions](#)
- [Learn More](#)

This topic discusses techniques for troubleshooting and interpreting the information in the App Agent for Java log files.

Troubleshooting App Agent for Java Startup

After sending a request to your web application, data should appear in the UI. If no data appears, check the following:

1. You have re-started the application server.
2. Verify that the javaagent argument has been added to the startup script of your JVM.
3. Verify that you configured the agent-controller communication properties and agent identification properties in the controller-info.xml file or as system properties in the startup script of your JVM. See [App Agent for Java Configuration Properties](#).
4. Check the Agent logs directory located at <Agent_Installation_Directory>/logs/<Node_Name> for the agent.log file.
5. Verify that the Agent is compatible with the Controller. For details see [Agent - Controller Compatibility Matrix](#).

Locating the App Agent for Java Log Files

Agent log files are located in the <Agent_Installation_Directory>/logs/<Node_Name> folder.

The agent.log file is the recommended file to help you with troubleshooting. This log can indicate the following:

- [Incomplete information in your Agent configuration.](#)
- [Indicates if the Controller port is blocked.](#)
- [Incorrect permissions.](#)

Error messages related to starting the App Agent for Java use this format:

```
ERROR com.singularity.JavaAgent - Could Not Start Java Agent
```

Troubleshooting Incomplete Agent Configuration

The following table lists the typical error messages for incomplete Agent configuration:

Error Message	Solution
---------------	----------

Cannot connect to the Agent - ERROR com.singularity.XMLConfigManager - Incomplete Agent Identity data, Invalid Controller Port Value []	<p>This indicates that the value for the controller port in controller-info.xml is missing. Add the port value, along with the host value (<your-host-name>), to fix this error.</p> <p></p> <ul style="list-style-type: none"> • For on-premise Controller installations: Default port value is 8090 for HTTP and 8181 for HTTPS. • For Controller SaaS service: Default port value is 80 for HTTP and 443 for HTTPS.
Caused by: com.singularity.ee.agent.configuration.a: Could not resolve agent-controller basic configuration	<p>This is usually caused because of incorrect configuration in the Controller-info.xml file. Ensure that the information for agent communication (Controller host and port) and agent identification (application, tier and node names) is correctly configured. Alternatively, you can also use the system properties (-D options) or environment variables to configure these settings.</p>

For more information about agent properties see [App Agent for Java Configuration Properties](#).

Troubleshooting the Controller Port

The following table lists the typical error message when the Controller port is blocked in your network:

Error Message	Solution
<p>ERROR com.singularity.CONFIG.ConfigurationChannel - Fatal transport error: Connection refused</p> <p>WARN com.singularity.CONFIG.ConfigurationChannel - Could not connect to the controller/invalid response from controller, cannot get initialization information, controller host \x.x.x.x\, port 8090, exception Fatal transport error: Connection refused</p>	<p>Try to ping <your-host-name> from the machine where you have configured the Application Server Agent. If it works, then confirm if the Controller port is not blocked in your network.</p> <p> To check if a port was blocked in the network; use command: netstat -an for Windows and nmap for Linux.</p> <p> * For on-premise Controller installations: Default port value is 8090 for HTTP and 8181 for HTTPS.</p> <ul style="list-style-type: none"> • For Controller SaaS service: Default port value is 80 for HTTP and 443 for HTTPS.

Troubleshooting Incorrect File Permissions

Following table lists the typical error message when the file permissions are not correct:

Error Message	Solution
ERROR com.singularity.JavaAgent - Could Not Start Java Agent com.singularity.ee.agent.appagent.kernel.spi.c: Could not start services"	<p>This is usually caused because of incorrect permissions for log files. Confirm if the user who is running the server, has read and write permission on the agent directories. If the user has chmod a-r equivalent permission, change the permission to chmod a+r "<agent_directory>"</p>

Learn More

- [Install the App Agent for Java](#)
- [App Agent for Java Configuration Properties](#)

Administer App Agent for Java FAQ

- [App Agent for Java Administration FAQ](#)
 - Q. Why am I seeing "WARN AsyncHandoffIdentificationInterceptor - Reached maximum limit 500 of async handoff call graph samples. No more samples will be taken" message in the agent log files?

App Agent for Java Administration FAQ

Q. Why am I seeing "WARN AsyncHandoffIdentificationInterceptor - Reached maximum limit 500 of async handoff call graph samples. No more samples will be taken" message in the agent log files?

In AppDynamics 3.6 and 3.7, all Runnables, Callables and Threads are instrumented by default except for the ones that are excluded by the agent configuration in app-agent-config.xml. In some environments, this could result in too many classes being instrumented, or cause common classes in a framework that implements the Runnable interface to be mistaken for asynchronous activity when it is not, for example Groovy application using Closures.

To debug the cause of the message, check the call graph to see if so many asynchronous activities are legitimate. If they are not, then exclude the packages that are not really asynchronous activities. See [Configure Multi-Threaded Transactions \(Java\)](#).

Configure AppDynamics for Java

Configure Custom Exit Points (Java)

- Default Backends Discovered by the App Agent for Java
- Configure Custom Exit Points for Java Backends
- To create a custom exit point
 - To split an exit point
 - To group an exit point
- To define custom metrics for a custom exit point
- To define transaction snapshot data collected
- Learn More

AppDynamics provides default automatic discovery for commonly-used backends. If a backend used in your environment is not discovered, first compare the list of default backends to determine whether you need to modify the default configuration. If it is not on the list then configure a custom exit point according to these instructions.

Default Backends Discovered by the App Agent for Java

The default backends for Java are:

- HTTP
- JDBC
- JMS
- MQ
- RMI
- Thrift (not currently configurable)
- Web Service

To configure a default backend see [Configure Backend Detection](#).

Configure Custom Exit Points for Java Backends

Configure Custom Exit Points

Custom exit points provide identification for backend types that are not automatically detected, such as file systems, mainframes etc. For example, you can define a custom exit call to monitor the file system read method. Custom exit points appear as unresolved backends in the flow maps. Unresolved backends are shown on flow maps with this icon



You define a custom exit point by specifying the class and method used to identify the backend. If the method is overloaded, you need to add the parameters to identify the method uniquely.

You can restrict the method invocations for which you want AppDynamics to collect metrics by specifying match conditions for the method. The match conditions can be based on a parameter or the invoked object.

You can also optionally split the exit point based on a method parameter, the return value, or the invoked object.

You can also configure custom metrics and transaction snapshot data to collect for the backend.

See [Configurations for Custom Exit Points](#) for suggested custom configurations for some common backends.

To create a custom exit point

1. In the left navigation panel, click **Configure -> Instrumentation**.
2. Click the **Backend Detection** tab.
3. Click the tab corresponding to the backend platform.
4. Select the application or tier for which you are configuring the custom exit point. Backend detection configuration is applied on a hierarchical inheritance model. See [Hierarchical Configuration Model](#).
5. Scroll down to Custom Exit Points.
6. Click **Add** (the + icon).
7. In the Create Custom Exit Point window, click the **Identification** tab if it is not selected.
8. Enter a name for the exit point. This is the name that identifies the backend.
9. Select the type of backend from the Type drop-down menu or check Use Custom if the type is not listed.
10. Configure the class and method name that identify the custom exit point.
If the method is overloaded, check the Overloaded check box and add the parameters.
11. If you want to restrict metric collection based on a set of criteria that are evaluated at runtime, click **Add Match Condition** and define the match condition(s).
For example, you may want to collect data only if the value of a specific method parameter contains a certain value.
12. Click **Save**.

The following screenshot shows a custom exit point of Type Cache. This exit point is defined on the `getAll()` method of the specified class. The exit point appears in flow maps as an unresolved backend named CoherenceGetAll.

Edit Custom Exit Point

Name: <input type="text" value="CoherenceGetAll"/>	Type: <input type="text" value="Cache"/>				
Identification Custom Metrics Snapshot Data					
Define the class and method name which, when called, will be identified as a Custom Exit Point					
Class <input type="text" value="that implements an Interface which"/> <input type="button" value="▼"/>	equals <input type="text" value="com.tangosol.net.NamedCache"/>				
Method Name <input type="text" value="getAll"/>	<input type="checkbox"/> Is this Method Overloaded?				
Method Parameters (optional)					
<input type="button" value="Add Parameter"/>					
Match Conditions (optional)					
<input type="button" value="Add Match Condition"/>					
Calls to the specified class and method name can be further split based by a combination of match conditions.					
<table border="1"><thead><tr><th>Name</th><th>Description</th></tr></thead><tbody><tr><td>Cachename</td><td>Collect data from the invoked object and capture the result of the operation.</td></tr></tbody></table>		Name	Description	Cachename	Collect data from the invoked object and capture the result of the operation.
Name	Description				
Cachename	Collect data from the invoked object and capture the result of the operation.				
<input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/>					
<input type="button" value="Cancel"/> <input type="button" value="Save"/>					

To split an exit point

1. Click **Add**.
2. Enter a display name for the split exit point.
3. Specify the source of the data (parameter, return value, or invoked object).
4. Specify the operation to invoke on the source of the data (`toString()` or getter chain for complex objects).
5. Click **Save**.

The following example shows a split configuration of the previously created CoherenceGetAll exit point based on the getCacheName() method of the invoked object.

Edit Custom Exit Point Identifier

Specify the parameter index or indicate if it the return value of the diagnostic data to be collected.
Simple getters without parameters can be used on the parameter or the return value to be displayed against the display name specified here.

Display Name

Create your own name for the data collected. This will be the display name for the data in Request Snapshots

Collect Data From Method Parameter @ Index:
 Return Value
 Invoked Object

Operation on Invoked Object Use `toString()`
 Use Getter Chain
for example: `getAccount().getBalance()`

To group an exit point

You can group methods as a single exit point. The only requirement is that these methods point to the same key.

For example, ACME Online has an exit point for NamedCache.getAll. This exit point has a split configuration of getCacheName() on the invoked object as illustrated in the previous screenshot.

Suppose we also define another exit point for NamedCache.entrySet. This is another exit point, but it has the split configuration that has getCacheName() method of the invoked object.

Edit Custom Exit Point

Name:	CoherenceCacheAccess	Type:	Cache				
<input checked="" type="radio"/> Identification <input type="radio"/> Custom Metrics <input type="radio"/> Snapshot Data							
Define the class and method name which, when called, will be identified as a Custom Exit Point:							
Class	<input style="width: 200px;" type="text" value="that implements an Interface which"/> <input type="button" value="▼"/> equals <input style="width: 200px;" type="text" value="com.tangosol.net.NamedCache"/>						
Method Name	<input type="text" value="entrySet"/>	<input type="checkbox"/> Is this Method Overloaded?					
Method Parameters (optional)							
<input type="button" value="Add Parameter"/>							
Match Conditions (optional)							
<input type="button" value="Add Match Condition"/>							
Calls to the specified class and method name can be further split based by a combination of method parameters and results.							
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>CacheName</td> <td>Collect data from the invoked object and capture the result of getCacheName().</td> </tr> </tbody> </table>				Name	Description	CacheName	Collect data from the invoked object and capture the result of getCacheName().
Name	Description						
CacheName	Collect data from the invoked object and capture the result of getCacheName().						
<input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/>							

If the getAll() and the entrySet() methods point to the same cache name, they will point to the same backend.

Matching name-value pairs identify the back-end. In this case, there is only one key, i.e. cache name, has to be matched. So, here both exit points have the same name for the cache and they resolve to the same backend.

To define custom metrics for a custom exit point

Custom metrics are collected in addition to the standard metrics.

The result of the data collected from the method invocation must be an integer value, which will either be averaged or added per minute, depending on your selection of data roll-up.

To configure custom business metrics that can be generated from the Java method invocation:

1. Click the **Custom Metrics** tab.
2. Click **Add**.
3. In the Add Custom Metric window, enter a name for the metric.
4. Select the Collect Data From radio button to specify the source of the metric data.
5. Select the Operation on Method Parameter to specify how the metric data is processed.
6. Select how the data should be rolled up (average or sum) from the Data Rollup drop-down menu.
7. Click **Create Custom Metric**.

To define transaction snapshot data collected

1. Click the **Snapshot Data** tab.
2. Click **Add**.
3. In the Add Snapshot Data window, enter a display name for the snapshot data.
4. Select the Collect Data From radio button to specify the source of the snapshot data.
5. Select the Operation on Method Parameter to specify how the snapshot data is processed.
5. Click **Save**.

Learn More

- Configurations for Custom Exit Points

Configurations for Custom Exit Points

- Configurations for Coherence Exit Points
- Configurations For Memcached Exit Points
- Configurations for DangaMemcached Exit Points
- Configurations for SAP Exit Points
- Configurations for EhCache Exit Points
- Configurations for Mail Exit Points
- Configurations for LDAP Exit Points
- Configurations for MongoDB Exit Points
- Learn More

This topic suggests custom exit point configurations that you can create for specific backends.

Configurations for Coherence Exit Points

Name of the Exit Point	Type	Method Name	Match Criteria value for the Class	Class/Interface/Superclass/Annotation Name	Splitting Configuration
Coherence.Put	Cache	put	that implements an interface which	com.tangosol.net.NamedCache	getCacheName()
Coherence.PutAll	Cache	putAll	that implements an interface which	com.tangosol.net.NamedCache	getCacheName()
Coherence.EntrySet	Cache	entrySet	that implements an interface which	com.tangosol.net.NamedCache	getCacheName()
Coherence.KeySet	Cache	keySet	that implements an interface which	com.tangosol.net.NamedCache	getCacheName()
Coherence.Get	Cache	get	that implements an interface which	com.tangosol.net.NamedCache	getCacheName()
Coherence.Remove	Cache	remove	that implements an interface which	com.tangosol.net.NamedCache	getCacheName()

Configurations For Memcached Exit Points

Name of the Exit Point	Type	Method Name	Match Criteria value for the Class	Class/Interface/Superclass/Annotation Name
Memcached.Add	Cache	add	With a class name that	net.spy.memcached.MemcachedClient
Memcached.Set	Cache	set	With a class name that	net.spy.memcached.MemcachedClient
Memcached.Replace	Cache	replace	With a class name that	net.spy.memcached.MemcachedClient

Memcached.CompareAndSwap	Cache	cas	With a class name that	net.spy.memcached.MemcachedClient
Memcached.Get	Cache	get	With a class name that	net.spy.memcached.MemcachedClient
Memcached.Remove	Cache	remove	With a class name that	net.spy.memcached.MemcachedClient

Configurations for DangaMemcached Exit Points

Name of the Exit Point	Type	Method Name	Match Criteria value for the Class	Class/Interface/Superclass/Annotation Name
Memcached.Add	Cache	add	With a class name that	com.danga.MemCached.MemCachedClient
Memcached.Set	Cache	set	With a class name that	com.danga.MemCached.MemCachedClient
Memcached.Replace	Cache	replace	With a class name that	com.danga.MemCached.MemCachedClient
Memcached.Delete	Cache	delete	With a class name that	com.danga.MemCached.MemCachedClient
Memcached.Get	Cache	get	With a class name that	com.danga.MemCached.MemCachedClient
Memcached.GetBulk	Cache	getBulk	With a class name that	com.danga.MemCached.MemCachedClient
Memcached.GetMultiArray	Cache	getMultiArray	With a class name that	com.danga.MemCached.MemCachedClient
Memcached.GetMulti	Cache	getMulti	With a class name that	com.danga.MemCached.MemCachedClient

Configurations for SAP Exit Points

Name of the Exit Point	Type	Method Name	Match Criteria value for the Class	Class/Interface/Superclass/Annotation Name
SAP.Execute	SAP	execute	With a class name that	com.sap.mw.jco.rfc.MiddlewareRFC\$Client
SAP.Connect	SAP	connect	With a class name that	com.sap.mw.jco.rfc.MiddlewareRFC\$Client
SAP.Disconnect	SAP	disconnect	With a class name that	com.sap.mw.jco.rfc.MiddlewareRFC\$Client
SAP.Reset	SAP	reset	With a class name that	com.sap.mw.jco.rfc.MiddlewareRFC\$Client
SAP.CreateTID	SAP	createTID	With a class name that	com.sap.mw.jco.rfc.MiddlewareRFC\$Client
SAP.ConfirmTID	SAP	confirmTID	With a class name that	com.sap.mw.jco.rfc.MiddlewareRFC\$Client

Configurations for EhCache Exit Points

Name of the Exit Point	Type	Method Name	Match Criteria value for the Class	Class/Interface/Superclass/Annotation Name	Splitting Configuration
EhCache.Get	Cache	get	With a class name that	net.sf.ehcache.Cache	getName()
EhCache.Put	Cache	put	With a class name that	net.sf.ehcache.Cache	getName()
EhCache.PutIfAbsent	Cache	putIfAbsent	With a class name that	net.sf.ehcache.Cache	getName()
EhCache.PutQuiet	Cache	putQuiet	With a class name that	net.sf.ehcache.Cache	getName()
EhCache.Remove	Cache	remove	With a class name that	net.sf.ehcache.Cache	getName()
EhCache.RemoveAll	Cache	removeAll	With a class name that	net.sf.ehcache.Cache	getName()
EhCache.RemoveQuiet	Cache	removeQuiet	With a class name that	net.sf.ehcache.Cache	getName()
EhCache.Replace	Cache	replace	With a class name that	net.sf.ehcache.Cache	getName()

Configurations for Mail Exit Points

Name of the Exit Point	Type	Method Name	Match Criteria value for the Class	Class/Interface/Superclass/Annotation Name
MailExitPoint.Send	Mail Server	send	With a class name that	javax.mail.Transport
MailExitPoint.SendMessage	Mail Server	sendMessage	With a class name that	javax.mail.Transport

Configurations for LDAP Exit Points

Name of the Exit Point	Type	Method Name	Match Criteria value for the Class	Class/Interface/Superclass/Annotation Name
LDAPExitPoint.Bind	LDAP	bind	With a class name that	javax.naming.directory.InitialDirContext
LDAPExitPoint.Rebind	LDAP	rebind	With a class name that	javax.naming.directory.InitialDirContext
LDAPExitPoint.Search	LDAP	search	With a class name that	javax.naming.directory.InitialDirContext
LDAPExitPoint.ModifyAttributes	LDAP	modifyAttributes	With a class name that	javax.naming.directory.InitialDirContext
LDAPExitPoint.GetNextBatch	LDAP	getNextBatch	With a class name that	com.sun.jndi.ldap.LdapNamingEnum
LDAPExitPoint.NextAux	LDAP	nextAux	With a class name that	com.sun.jndi.ldap.LdapNamingEnum
LDAPExitPoint.CreatePooledConnection	LDAP	createPooledConnection	With a class name that	com.sun.jndi.ldap.LdapClientFactory
LDAPExitPoint.Search	LDAP	search	With a class name that	com.sun.jndi.ldap.LdapClientFactory
LDAPExitPoint.Modify	LDAP	modify	With a class name that	com.sun.jndi.ldap.LdapClientFactory

Configurations for MongoDB Exit Points

Name of the Exit Point	Type	Method Name	Match Criteria value for the Class	Class/Interface/Superclass/Annotation Name	Splitting configuration	Snapshot data

MongoDB.Insert	JDBC	insert	With a class name that	com.mongodb.DBCollection	Invoked_Object.getDB(). getName()	Parameter_0.toString()
MongoDB.Find	JDBC	find	With a class name that	com.mongodb.DBCollection	Invoked_Object.getDB(). getName()	Parameter_0.toString()
MongoDB.Update	JDBC	update	With a class name that	com.mongodb.DBCollection	Invoked_Object.getDB(). getName()	Parameter_0.toString()
MongoDB.Remove	JDBC	remove	With a class name that	com.mongodb.DBCollection	Invoked_Object.getDB(). getName()	Parameter_0.toString()
MongoDB.Apply	JDBC	apply	With a class name that	com.mongodb.DBCollection	Invoked_Object.getDB(). getName()	Parameter_0.toString()

Learn More

- Configure Backend Detection
- Configure Custom Exit Points

Code Metric Information Points (Java)

- Code Metric Information Points for Java System Classes
 - To instrument a Java system class
 - Learn More

Code Metric Information Points for Java System Classes

System classes like `java.lang.*` are by default excluded by AppDynamics. To enable instrumentation for a system class, use code metric information points.

The overhead of instrumenting Java system classes is based on the number of calls. AppDynamics recommends that you instrument only a small number of nodes and monitor the performance for these nodes before adding configuring all the nodes in your system.

To instrument a Java system class

1. Open the `<agent_home>/conf/app-agent-config.xml` file for the node where you want to enable the metric.
2. Add the fully-qualified system class name to the override exclude section in the XML file. For example, to configure the `java.lang.Socket` class connect method, modify following element:

```
<override-system-exclude filter-type="equals" filter-value="java.lang.Socket" />
```

3. Restart those JVMs for which you have modified the XML file.

Learn More

- Code Metrics
- Configure Code Metric Information Points

Configure JMX Metrics from MBeans

- JMX Metric Rules and Metrics

- Using the JMX Metric Rules Configuration Panel
 - To create one or more JMX metrics in the JMX Metric Rules panel
- Using the MBean Browser
 - To create a metric from an MBean attribute in the MBean Browser
- Learn More

This topic describes how to create persistent JMX metrics from MBean attributes.

For background information about creating JMX metrics see [Monitor JVMs](#) and [Monitor JMX MBeans](#).

JMX Metric Rules and Metrics

A JMX Metric Rule maps a set of MBean attributes from one or more MBeans into AppDynamics persistent metrics. You configure a metric rule that creates one or more metrics in the AppDynamics system. You may want to create new metrics if the preconfigured metrics do not provide sufficient visibility into the health of your system.

After the MBean attribute is configured to provide a persistent metric in AppDynamics, you can use it to configure health rules. For details see [Health Rules](#).

To view the MBeans that are reporting currently in your managed environment use the [Metric Browser](#).

You can use the [JMX Metrics Rules Panel](#) or the [Using the MBean Browser](#) to create new metrics.

Using the JMX Metric Rules Configuration Panel

The JMX Metric Rules Panel is the best way to create metrics for multiple attributes based on the same MBean or for complex matching patterns.

To create one or more JMX metrics in the JMX Metric Rules panel

1. In the left navigation pane, click **Configure -> Instrumentation**.
2. Click the **JMX** tab.
3. In the JMX Metric Configurations pane, click the Java platform for which you are configuring metrics.

The screenshot shows the JMX Metric Rules Configuration Panel interface. It consists of two main panes:

- JMX Metric Configurations** (Left Pane):

Name	Enabled
ActiveMQ	✓
Cassandra	✓
Glassfish	✓
HornetQ	✓
JBoss	✓
Platform	✓
Solr	✓
Tomcat	✓
WebLogic	✓
WebSpherePMI	✓
- Tomcat JMX Metric Rules** (Right Pane):

Name	Enabled
Tomcat_GlobalRequestPi	✓
Tomcat_HttpThreadPools	✓
Tomcat_JDBCConnector	✓
Tomcat_Sessions	✓

4. Click the + icon. A panel called "New Rule" and the next incremented number opens.
5. Provide the name and settings for this rule:
 - The **Name** is the identifier you want to display in the UI.

- An **Exclude Rule** is used for excluding existing rules, so leave the default **No** option.
- **Enabled** means that you want this rule to run, so leave it selected.
- The **Metric Path** is the category as shown in the Metric Browser where the metrics will be displayed. A metric path groups the metrics and is relative to the Metric Browser node.

For example, the following screenshot displays how the JMX Metric Rule "Tomcat_HttpThreadPools" is defined for the ACME Online demo. The metric path is "Web Container Runtime", the category on Metric Browser where all metrics configured under the "Tomcat_HttpThreadPools" Metric Rule will be available.

The screenshot shows the 'Metric Tree' interface with the 'Settings' tab selected. On the left, a 'New Rule 4' dialog box is open, containing the following fields:

- Name:** Tomcat_HttpThreadPools
- Exclude Rule:** No (radio button selected)
- Enabled:** Yes (checkbox checked)
- Metric Path:** Web Container Runtime

A large blue arrow points from the 'Metric Path' field in the dialog to the 'Web Container Runtime' node in the Metric Tree on the right. A callout box with the text 'This is the path in the metric browser where this metric will be created relative to the JMX metric browser node.' is positioned near the arrow.

The Metric Tree structure is as follows:

- Overall Application Performance
- Business Transaction Performance
- Application Infrastructure Performance
 - E-Commerce
 - Agent
 - Hardware Resources
 - Individual Nodes
 - JMX
 - Sessions
 - Web Container Runtime
 - http-8000
 - JVM
 - Inventory
 - Order Processing

6. Click + icon. An panel called "New Rule" and the next incremented number opens. Specify details for the MBeans that you want to monitor.

- The **Domain name** is the Java domain. This property must be the exact name; no wildcard characters are supported.
- The **Object Name Match Pattern** is the full object name pattern. The property may contain wildcard characters, such as the asterisk for matching all the name/value pairs.
- The **Advanced MBean Matching Criteria** is optional for more complex matching. Use one of the following:
 - any-substring
 - final-substring
 - equals
 - initial-substring

For example, the following screenshot displays the MBean matching criteria for the "Tomcat_HTTPThreadPools" rule.

MBean Matching Criteria

Domain	Catalina
Object Name Match Pattern	Catalina:type=ThreadPool,*

▼ Advanced MBean Matching

Find MBeans where **All** of the following conditions apply:

name **any-substring** http
 -

Add Condition **+**

For all MBeans that match the preceding criteria, you can define one or more metrics for the attributes of those MBeans.

7. In the Attributes panel click **Add Attribute** to specify the MBean attributes.

- Provide the name of the attribute and the metric name.
The metric name will be used to represent the metric in the AppDynamics metric browser.
- Specify any of the following "Advanced Properties" for the attribute:
 - Metric Time Rollup** determines how the metric will be aggregated over a period of time. You can choose to either average or sum the data points, or use the latest data point in the time interval.
 - Metric Cluster Rollup** defines how the metric will be aggregated for a tier, using the performance data for all the nodes in that tier. You can either average or sum the data.
 - Metric Aggregator Rollup** defines how the Agent rolls up multiple individual measurements (observations) into the observation that it reports once a one minute. For performance reasons, Agents report data to the Controller at one minute intervals. Some metrics, such as Average Response Time, are measured (observed) many times in a minute. The Metric Aggregator Rollup setting determines how the Agent aggregates these metrics. You can average or sum observations on the data points or use the current observation. Alternatively you can use the delta between the current and previous observation (*new in 3.4*).

The following screenshot shows how the MBean attributes configured for the Tomcat_HttpThreadPools rule will be displayed in the Metric Browser.

Attributes

Define Metrics from MBean Attribute(s)

MBean Attribute	maxThreads
Metric Name	Maximum Threads
▶ Advanced	
MBean Attribute	currentThreadsBusy
Metric Name	Busy Threads
▶ Advanced	
MBean Attribute	currentThreadCount
Metric Name	Current Threads In Pool
▶ Advanced	
Add Attribute	+

Metric Tr... Settings

- Overall Application Performance
- Business Transaction Performance
- Application Infrastructure Performance
 - E-Commerce
 - Agent
 - Hardware Resources
 - Individual Nodes
 - JMX
 - Sessions
 - Web Container Runtime
 - http-8000
 - Busy Threads
 - Current Threads In Pool
 - Error Count
 - Maximum Threads
 - Request Count
 - Total Bytes Received
 - Total Bytes Sent

Using the MBean Browser

Sometimes you know exactly which particular MBean attribute you want to monitor. You can select the attribute in the MBean Browser and create a JMX Metric Rule for it.

To create a metric from an MBean attribute in the MBean Browser

1. Navigate to the attribute and select it in the MBean Browser.

The screenshot shows the AppDynamics interface with the 'MBean Browser' tab selected. On the left, a tree view shows nodes like 'JMImplementation', 'Users', 'Catalina' (expanded), 'Cache', 'Connector' (expanded), and various sub-components under 'Connector'. The 'Connector' node '8000' is selected. On the right, the 'Attributes' section displays a table of attributes for the selected MBean. The table has columns for 'Name' and 'Type'. The attributes listed are:

	Name	Type
acceptCount	int	
bufferSize	int	
compression	java.lang.S	
connectionLinger	int	
connectionTimeout	int	
connectionUploadTimeout	int	

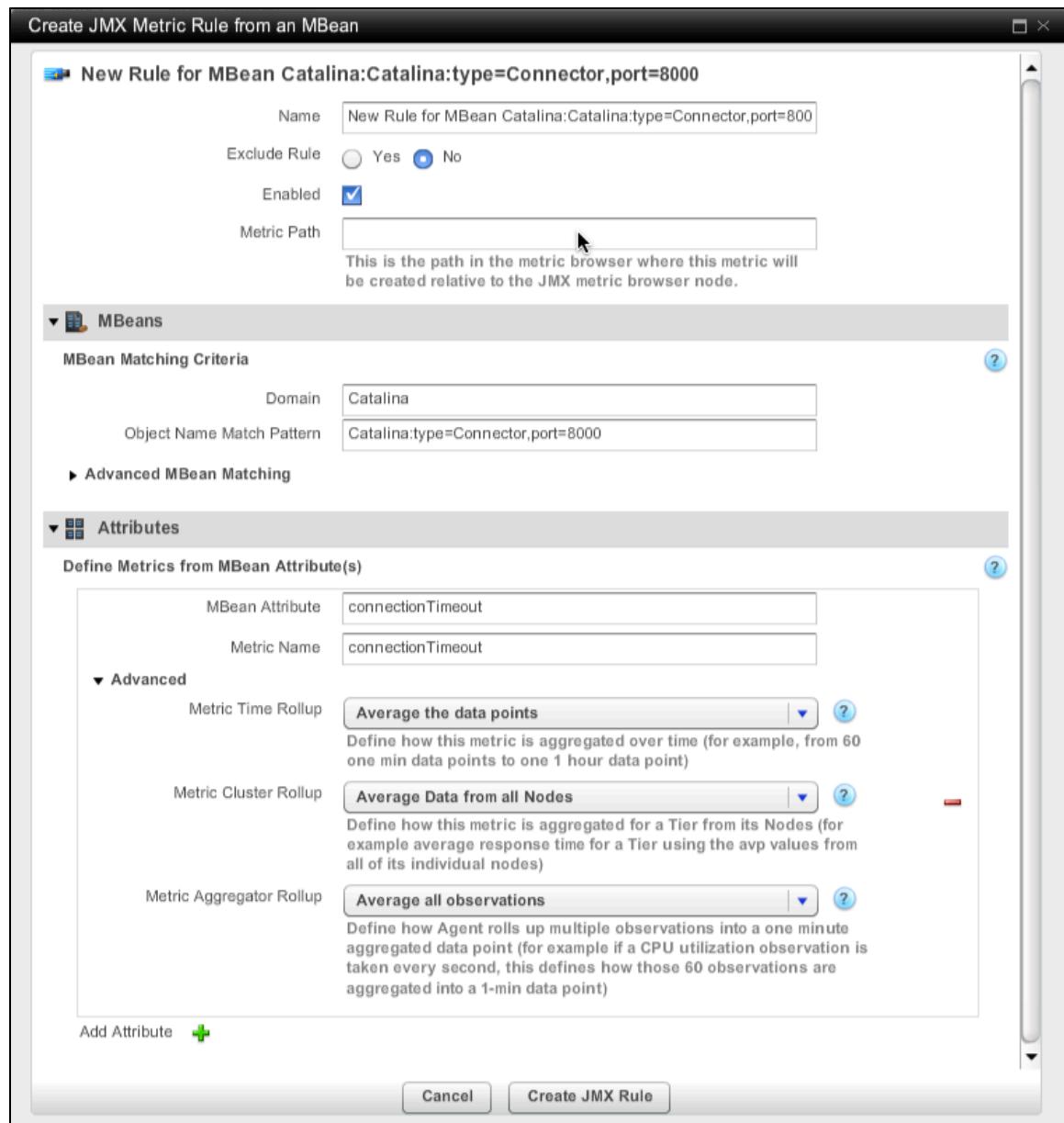
2. Click **Create Metric**.

3. In the **Select JMX Configuration for the Metric Rule** window, select the group in which to create the rule from the **Select JMX Configuration** pulldown. The categories that already exist on your system are listed.



Alternatively, you can create a new group with an name of your choosing. Click **Create New JMX Configuration** to make a new category. This is useful if you want to separate out the custom metrics from the out-of-the-box metrics.

The Create JMX Metric Rule from an MBean window opens.



4. Supply the **Metric Path**, the category as shown in the Metric Browser where the metrics will be displayed. For more discussion of the Metric Path see Step 5 of the previous section.
5. Review the **MBean Matching Criteria** and modify it as needed. Since this is the MBean you selected, you probably do not need to change it.
6. Review the **MBean Attribute** and **Metric Name**. This is the MBean attribute you originally selected. By default the name is the same as the attribute. You can change it if you want to be more specific about its use.
7. Review the **Advanced** panel rollup criteria and update as needed. For more description of these options see Step 7 of the previous section.
8. Click **Create JMX Rule**. The new metric displays in the JMX Metric Browser.

Learn More

- Monitor JVMs
- Monitor JMX MBeans
- Exclude JMX Metrics

Exclude JMX Metrics

- Tuning What Metrics are Gathered
 - To exclude a metric
- Learn More

This topic describes how to exclude MBean attributes from being monitored as JMX metrics.

For background information about JMX metrics see [Monitor JVMs](#) and [Monitor JMX MBeans](#).

Tuning What Metrics are Gathered

AppDynamics provides a default configuration for certain JMX metrics. However, in situations where an environment has many resources, there may be too many metrics gathered. AppDynamics lets you exclude resources and particular operations on resources.

To exclude a metric

For example, suppose you want to exclude monitoring for HTTP Thread Pools. Create a new JMX Metrics Rule with the following criteria:

1. Set the Exclude Rule option to **Yes**.
2. Provide the **Object Name Match Pattern**:

```
Catalina:type=ThreadPool,*
```

2. Provide the Advanced MBean Pattern Matching value:

```
http
```

This configuration causes AppDynamics to stop monitoring metrics for HTTP Thread Pools.

Later, if you want to see all HTTP Thread Pool metrics, clear the Enabled checkbox to disable the rule.

Learn More

- [Monitor JVMs](#)
- [Monitor JMX MBeans](#)
- [Configure JMX Metrics from MBeans](#)

Exclude MBean Attributes

- Excluding MBean Attributes from the MBean Browser
 - To exclude an MBean attribute
- Learn More

Excluding MBean Attributes from the MBean Browser

Some MBean attributes contain sensitive information that you do not want the Java Agent to report. You can configure the Java Agent to exclude these attributes using the <exclude object-name> setting in the app-agent-config.xml file.

To exclude an MBean attribute

1. Open the AppServerAgent/conf/app-agent-config.xml file.
2. Locate the JMXService section:

```
<agent-service name="JMXService" enabled="true">
```

3. In the JMXService <configuration> section add the <jmx-metric-browser-excludes> section and the <exclude object-name> property as per the instructions in the comment.

```

<configuration>
    <!--
        Use the below configuration sample to create rules to exclude MBean attributes
        from MBean Browser.
        <exclude object-name=<MBean name pattern> attributes=< * |comma separated list
        of attribute names> >
            The example below will exclude all attributes of MBeans that match
        "Catalina:*".
        <jmx-metric-browser-excludes>
            <exclude object-name="Catalina:/" attributes="*"/>
        </jmx-metric-browser-excludes>
    -->
</configuration>

```

4. Save the file.

The new configuration takes effect immediately if the agent-overwrite property is set to true in the app-agent-config.xml. If agent-overwrite is false, which is the default, then the new configuration will be ignored and you have to restart the agent.

Learn More

- [App Agent for Java Directory Structure](#)

Configure JMX Without Transaction Monitoring

- Collect Metrics without Transaction Monitoring
 - To turn off transaction detection
- [Learn More](#)

Collect Metrics without Transaction Monitoring

In some circumstances, such as for monitoring caches and message buses, you want to collect JMX metrics without the overhead of transaction monitoring.

You can do this by turning off transaction detection at the entry point.

To turn off transaction detection

1. In the left navigation panel, click **Configure -> Instrumentation**.
2. In the Select Application or Tier panel, select the application.
3. In the right panel, click **Java Transaction Detection**.
4. Expand the Entry Points list if it is not already expanded.
5. Clear all the relevant **Enabled** checkboxes in the Transaction Monitoring column.

Transaction monitoring on all selected entry points in the application is disabled. Exit point detection remains enabled.

Learn More

- [Configure Business Transaction Detection](#)

Resolve JMX Configuration Issues

- Unable to browse MBeans on WebSphere Application Server (WAS).
- Unable to get metrics from the Java App Server Agent on GlassFish
 - Unable to get JMX metrics for database connections on GlassFish
- [Learn More](#)

This topic describes how to resolve issues that may prevent AppDynamics from properly reporting JMX MBean metrics.

Unable to browse MBeans on WebSphere Application Server (WAS).

In certain situations, you may encounter the following exception in the agent.log file for a Java App Server Agent deployed on the WebSphere Application Server (WAS).

```
[AD Thread-Transient Event Channel Poller0] 17 Aug 2011 08:14:08,031
ERROR JMXTransientOperationsHandler - Error trying to lookup clz -
java.lang.ClassNotFoundException: com.ibm.ws.security.core.SecurityContext
```

To resolve this issue:

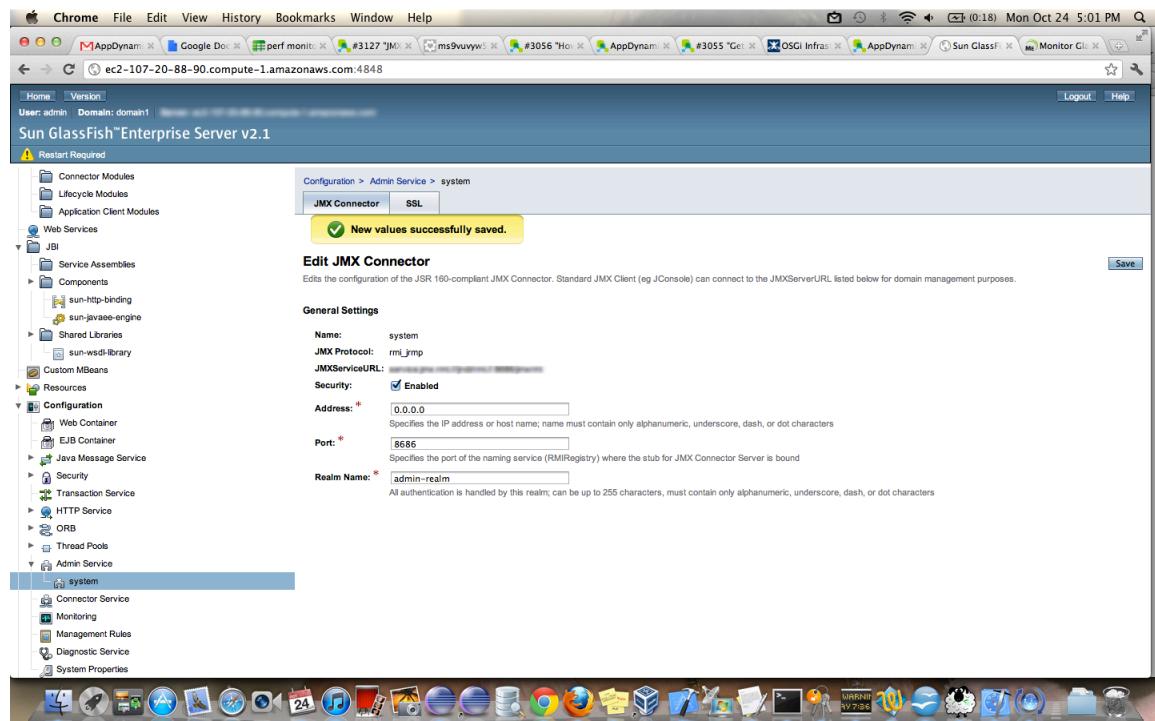
1. From the WAS administration console, navigate to the JVM settings for the server of interest: **Application servers -> <server> -> Process Definition -> Java Virtual Machine.**
2. Remove the following setting from the generic JVM settings:

```
-Djavax.management.builder.initial = -Dcom.sun.management.jmxremote
```

Unable to get metrics from the Java App Server Agent on GlassFish

Under some situations JMX metrics from GlassFish are not reported. Also some metrics may not be enabled by default. Try these solutions:

- 1) Confirm that JMX monitoring is enabled in the GlassFish server. Refer to the following screenshot:



- 2) Copy the text below into an mbean-servers.xml file in the following directory:

```
<App_Agent_Dir>/conf/jmx/
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--<!DOCTYPE servers SYSTEM "mbean-servers.dtd"> -->

<servers xmlns="http://www.appdynamics.com"
```

```

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.appdynamics.com
mbean-servers.xsd">
    <!--
        <server mbean-server-name="WebSphere"
mbean-name-pattern="WebSphere:*,type=Server,j2eeType=J2EEServer"
version-attribute="platformVersion" version-startsWith="7"
config-file="servers/websphere-7-jmx-config.xml" />

        <server mbean-server-name="WebSphere"
mbean-name-pattern="WebSphere:*,type=Server,j2eeType=J2EEServer"
version-attribute="platformVersion" version-startsWith="6"
config-file="servers/websphere-7-jmx-config.xml" />
    -->
        <server mbean-server-name="WebSphere"
mbean-name-pattern="WebSphere:*,type=Server"
config-file="servers/websphere-7-jmx-config.xml" />
        <server mbean-server-name="JBoss_4"
mbean-name-pattern="jboss.management.local:j2eeType=J2EEServer,name=Local"
version-attribute="serverVersion" version-startsWith="4"
config-file="servers/jboss-4-jmx-config.xml" />
        <server mbean-server-name="JBoss_5"
mbean-name-pattern="jboss.management.local:j2eeType=J2EEServer,name=Local"
version-attribute="serverVersion" version-startsWith="5"
config-file="servers/jboss-5-jmx-config.xml" />
        <server mbean-server-name="JBoss_6"
mbean-name-pattern="jboss.management.local:j2eeType=J2EEServer,name=Local"
version-attribute="serverVersion" version-startsWith="6"
config-file="servers/jboss-5-jmx-config.xml" />
        <server mbean-server-name="Tomcat_5.5"
mbean-name-pattern="Catalina:type=Server" version-attribute="serverInfo"
version-startsWith="Apache Tomcat/5.5"
config-file="servers/tomcat-5-jmx-config.xml" />
        <server mbean-server-name="Tomcat_6.0"
mbean-name-pattern="Catalina:type=Server" version-attribute="serverInfo"
version-startsWith="Apache Tomcat/6.0"
config-file="servers/tomcat-6-jmx-config.xml" />
        <server mbean-server-name="Tomcat_7"
mbean-name-pattern="Catalina:type=Server" version-attribute="serverInfo"
version-startsWith="Apache Tomcat/7"
config-file="servers/tomcat-7-jmx-config.xml" />
        <server mbean-server-name="Sun GlassFish_2.1"
mbean-name-pattern="com.sun.appserv:j2eeType=J2EEServer,name=server,category=runt
config-file="servers/glassfish-v2-jmx-config.xml" />
        <server mbean-server-name="WebLogic_10"
mbean-server-lookup-string="java:comp/jmx/runtime"
mbean-name-pattern="com.bea:*,Type=ServerRuntime"
version-attribute="WeblogicVersion" version-startsWith="WebLogic Server 10"
config-file="servers/weblogic-10-jmx-config.xml" />
        <server mbean-server-name="WebLogic_9"
mbean-server-lookup-string="java:comp/jmx/runtime"
mbean-name-pattern="com.bea:*,Type=ServerRuntime"
version-attribute="WeblogicVersion" version-startsWith="WebLogic Server 9"
config-file="servers/weblogic-9-jmx-config.xml" />
        <server mbean-server-name="ActiveMQ_5.3.2"
mbean-name-pattern="org.apache.activemq:*

```

```

config-file="servers/activemq-5.3.2-jmx-config.xml" />
    <server mbean-server-name="Apache Solr 1.4.1" mbean-name-pattern="solr:*" config-file="servers\solr-1.4.1-jmx-config.xml" />
        <server mbean-server-name="Apache Cassandra 0.7.0" mbean-name-pattern="org.apache.cassandra.net:*" config-file="servers\cassandra-0.7.0-jmx-config.xml" />
            <server mbean-server-name="Apache Cassandra 0.7.0" mbean-name-pattern="org.apache.cassandra.db:*" config-file="servers\cassandra-0.7.0-jmx-config.xml" />
                <server mbean-server-name="Apache Cassandra 0.7.0" mbean-name-pattern="org.apache.cassandra.request:*" config-file="servers\cassandra-0.7.0-jmx-config.xml" />
                    <server mbean-server-name="Apache Cassandra 0.7.0" mbean-name-pattern="org.apache.cassandra.internal:*" config-file="servers\cassandra-0.7.0-jmx-config.xml" />
                    <!-- If you are using Platform MBean server to report activemq metrics then you may uncomment the following line.
-->
<!--
<server mbean-server-name="Platform" mbean-name-pattern="org.apache.activemq:*" config-file="servers\activemq-5.3.2-jmx-config.xml" />
-->

<!-- If your app publishes custom jmx metrics to platform jmx server then you may modify the platform-jmx-config.xml and update the mbean-name-pattern in the following line to start recording your metrics by appdynamics agent
-->

<!--
<server mbean-server-name="Platform" mbean-name-pattern="com.foo.myjmx:*" config-file="servers\platform-jmx-config.xml" />
-->

```

```
</servers>
```

You should see a new JMX node in the metrics tree.

Unable to get JMX metrics for database connections on GlassFish

JDBC connection pool metrics are not configured out-of-the-box for GlassFish. To configure them, uncomment the JDBC connection pool section and provide the relevant information in the following file:

```
<app_agent_install>/conf/jmx/servers/glassfish-v2-jmx-config.xml
```

Uncomment the following section and follow the instructions provided in the file.

```

<!-- The following config can be uncommented to monitor glassfish JDBC
connection pool. Please set the name of the connection
pool (not the datasource name) and enable monitoring for the JDBC Pools on
glassfish admin console. -->
<!--
<metric
mbean-name-pattern="com.sun.appserv:type=jdbc-connection-pool,category=monitor,nat
the name of pool,<*>
category="JDBC Connection Pools">
<attribute-counter-mappings>
<attribute-counter-mapping>
<attribute-name>numconnused-current</attribute-name>
<counter-name>Connections In Use</counter-name>
<counter-type>average</counter-type>
<time-rollup-type>average</time-rollup-type>
<cluster-rollup-type>individual</cluster-rollup-type>
</attribute-counter-mapping>
<attribute-counter-mapping>
<attribute-name>numconnused-highwatermark</attribute-name>
<counter-name>Max Connections Used</counter-name>
<counter-type>observation</counter-type>
<time-rollup-type>average</time-rollup-type>
<cluster-rollup-type>individual</cluster-rollup-type>
</attribute-counter-mapping>
<attribute-counter-mapping>
<attribute-name>numpotentialconnleak-count</attribute-name>
<counter-name>Potential Leaks</counter-name>
<counter-type>observation</counter-type>
<time-rollup-type>average</time-rollup-type>
<cluster-rollup-type>individual</cluster-rollup-type>
</attribute-counter-mapping>
<attribute-counter-mapping>
<attribute-name>averageconnwaittime-count</attribute-name>
<counter-name>Avg Wait Time Millis</counter-name>
<counter-type>observation</counter-type>
<time-rollup-type>average</time-rollup-type>
<cluster-rollup-type>individual</cluster-rollup-type>
</attribute-counter-mapping>
<attribute-counter-mapping>
<attribute-name>waitqueuelength-count</attribute-name>
<counter-name>Current Wait Queue Length</counter-name>
<counter-type>observation</counter-type>
<time-rollup-type>average</time-rollup-type>
<cluster-rollup-type>individual</cluster-rollup-type>
</attribute-counter-mapping>
</attribute-counter-mappings>
</metric>

```

Learn More

- IBM WebSphere Startup Settings
- Glassfish Startup Settings

Import or Export JMX Metric Configurations

- To import JMX metrics configuration
- To export JMX metrics configuration
- To use a pre-3.3 configuration file for JMX metrics

This topic describes how to import or export your existing JMX configurations.

To import JMX metrics configuration

1. Click **Configure -> Instrumentation**.
2. Click the **JMX** tab.
3. Click the **Import JMX Configuration** icon.



4. In the JMX Configuration Import screen, click **Select JMX Config. File** and select the XML configuration file for your JMX metrics.



5. Click **Import**.

To export JMX metrics configuration

1. Click **Configure -> Instrumentation**.
2. Click the **JMX** tab.
3. Click the **Export JMX Configuration** icon.



The configuration is downloaded as an XML file.

To use a pre-3.3 configuration file for JMX metrics

In AppDynamics Pro Version 3.3 the structure of the XML-based configuration file for JMX metrics changed. As a result, if you use a pre-3.3 version, you must provide additional attributes in the configuration file.

The following screenshot shows a sample XML configuration file for the JMX metrics for pre-3.3 versions of AppDynamics:

```

tomcat-7-jmx-config.xml
1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE jmx-configuration SYSTEM "jmx-config.dtd">
3
4  <jmx-configuration instance-identifier="http://www.appdynamics.com" version="1.0" >
5    <!-- JMX Configuration For Tomcat 6.* -->
6    <server>
7      <metric mbean-name-pattern="Catalina:type=GlobalRequestProcessor,*" category="Web Container Runtime" >
8        <attribute-counter-mappings>
9        </attribute-counter-mappings>
10     </metric>
11     <metric mbean-name-pattern="Catalina:type=ThreadPool,*" category="Web Container Runtime" >
12       <query-expression-type>initial-substring</query-expression-type>
13       <query-attribute>name</query-attribute>
14       <query-value>http</query-value>
15       <bean-name>WebThreadPool</bean-name>
16     </metric>
17     <metric mbean-name-pattern="Catalina:type=DataSource,*" category="JDBC Connection Pools" >
18       <bean-name>JDBCConnectionPool</bean-name>
19     </metric>
20     <metric mbean-name-pattern="Catalina:type=Manager,*" category="Sessions" >
21       <instance-identifier>path</instance-identifier>
22   </server>
23 </jmx-configuration>

```

For previous versions, only following attributes are mandatory for "<metric>" element:
- "mbean-name-pattern".
- "category".

Beginning with AppDynamics version 3.3, the structure for this XML file is the following:

```

Start Page jmx-config.xsd JMX_1314140862295.xml
1  <jmx-configuration>
2    <server description="Default Config" enabled="true" name="Tomcat">
3      <metric category="Web Container Runtime" domain-name="Catalina" >
4        <enabled>true</enabled>
5        <mbean-name-pattern>Catalina:type=GlobalRequestProcessor,*</mbean-name-pattern>
6        <name>Tomcat_GlobalRequestProcessor</name>
7        <attribute-counter-mappings>
8      </metric>
9    </server>
10   <metric category="Web Container Runtime" domain-name="Catalina" >
11     <name>Tomcat_GlobalRequestProcessor</name>
12     <domain-name>Catalina</domain-name>
13     <mbean-name-pattern>Catalina:type=GlobalRequestProcessor,*</mbean-name-pattern>
14     <category>Web Container Runtime</category>
15     <enabled>true</enabled>
16     <exclude>false</exclude>
17   </metric>
18   <metric category="JDBC Connection Pools" domain-name="Catalina" >
19     <name>Tomcat_JDBCConnectionPools</name>
20     <domain-name>Catalina</domain-name>
21     <mbean-name-pattern>Catalina:type=DataSource,*</mbean-name-pattern>
22     <category>JDBC Connection Pools</category>
23     <enabled>true</enabled>
24     <exclude>false</exclude>
25   </metric>
26   <metric category="Sessions" domain-name="Catalina" >
27     <name>Tomcat_Sessions</name>
28     <domain-name>Catalina</domain-name>
29     <mbean-name-pattern>Catalina:type=Manager,*</mbean-name-pattern>
30     <category>Sessions</category>
31     <enabled>true</enabled>
32     <instance-identifier>path</instance-identifier>
33   </metric>
34 </jmx-configuration>

```

To be able to import your existing configurations for JMX metrics, add the following attributes for <server> element and **each** of the <metric> elements in your XML file.

The attributes for each element are listed below:

Attributes for the <server> element

Attribute Name	Attribute Type	Allowed Values	Mandatory/Optional
description	String		Mandatory
enabled	Boolean	true/false	Mandatory
name	String		Mandatory

Attributes for each <metric> element

For each <metric> element, add the mandatory attributes from the following list to your existing configuration file:

Attribute Name	Attribute Type	Allowed Values	Mandatory/Optional
name	String		Mandatory
domain-name	String	true/false	Mandatory
mbean-name-pattern	String		Mandatory
category	String		Mandatory <i>All those <metric> elements that have same value for this attribute will be grouped together on the metric browser.</i>
enabled	Boolean	true/false	Mandatory
exclude	Boolean	true/false	Mandatory
bean-name	String		Optional
query-attribute	String		Optional
query-expression-type		Use any one from the following values: <ul style="list-style-type: none">• any-substring• final-substring• equals• initial-substring	Optional
query-value	String		Optional
instance-identifier	String		Optional
instance-name	String		Optional

Configure Custom Memory Structures (Java)

- Custom Memory Structures and Memory Leaks
 - Using Automatic Leak Detection vs Monitoring Custom Memory Structures
 - Supported JVMs
 - To identify custom memory structures
 - To Add a Custom Memory Structure
- Identifying Potential Memory Leaks
 - Diagnosing memory leaks
 - Isolating a leaking collection
 - Access Tracking
- [Learn More](#)

This topic describes how to configure custom memory structures and monitor the potential leaks in production environments.

Custom Memory Structures and Memory Leaks

Typically custom memory structures are used as caching solutions. In a distributed environment, caching can easily become a source of memory leaks. AppDynamics helps you to manage and track memory statistics for these memory structures.

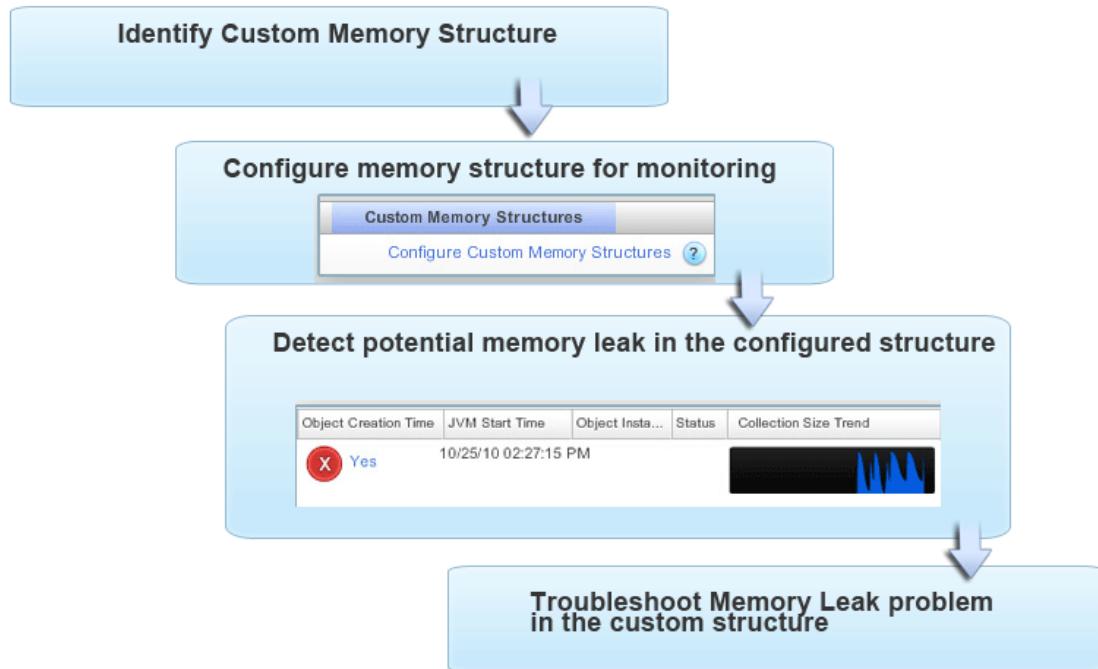
AppDynamics provide visibility into:

- Cache access for slow, very slow, and stalled business transactions
- Usage statistics, rolled up to the Business Transaction level
- Keys being accessed
- Deep size of internal cache structures

Using Automatic Leak Detection vs Monitoring Custom Memory Structures

The automatic leak detection feature captures memory usage data for all Collections objects in a JVM. However, custom memory structures might or might not contain Collections objects. For example, you can have a custom cache or a third party cache like Ehcache about which you can collect memory usage statistics.

The following illustration provides the work flow for configuring, monitoring, and troubleshooting custom memory structures. You have to configure custom memory structures manually.



Supported JVMs

The supported JVMs for the AppDynamics custom memory monitoring feature are:

- Sun JVM 1.51
- BEA JRockit JVM 1.5 (requires a JVM restart)
- IBM JVM 1.5 (content inspection functionality only)
- Sun JVM 1.6
- BEA JRockit JVM 1.6 (content inspection functionality only)
- IBM JVM 1.6 (content inspection functionality only)

To identify custom memory structures

Enable On-demand Access Tracking in automatic leak detection to capture information on what classes are accessing which Collections objects. Use this information to identify custom memory structures.

AppDynamics captures the top 1000 classes, by instance count.

To Add a Custom Memory Structure

1. From the left navigation pane select **Configure -> Instrumentation**.
2. Click the **Memory Monitoring** tab.
3. In the Tier panel select the tier for which you want to configure a custom memory structure.
4. In the Custom memory Structures section click **Add** to add a new memory structure. The Add Memory Structure window opens.
5. In the Create Memory Structure window

- Specify the configuration name.
 - Check **Enabled**.
6. Specify the discovery method.
The discovery method provides three options to monitor the Custom Memory Structure. The discovery method determines how the agent gets a reference to the Custom Memory Structure. AppDynamics needs this reference to monitor the size of the structure. Select any of the three options for the discovery method:
- Discover using Static Field.
 - Discover using Constructor.
 - Discover using Method.

In many cases, especially with caches, the object for which a reference is needed is created early in the life cycle of the application.

Example for using static field	Example for using Constructor	Example for using method
<pre>public class CacheManager { private static Map userCache<String> User>; }</pre>	<pre>public class CustomerCache { public CustomerCache(); }</pre>	<pre>public Class CacheManager{ public List<Order>; getOrderCache(); { } }</pre>

Notes: Monitors deep size of this Map.

Notes: Monitors deep size of CustomerCache object(s).

Notes: Monitors deep size of this list.

Restart the JVM after the discovery methods are configured to get the references for the object.

5. (Optional) Define accessors.
Click **Define Accessors** to define the methods used to access the custom memory structure. This information is used to capture the code paths accessing the custom memory structure.

6. (Optional) Define the naming convention.
Click **Define Naming Convention**. These configurations differentiate between custom memory structures.

There are situations where more than one custom Caches are used, but only few of them need monitoring. In such a case, use the **Getter Chain** option to distinguish amongst such caches. For all other cases, use either value of the field on the object or a specific string as the object name.

6. Click **Save** to save the configuration.

Identifying Potential Memory Leaks

Start monitoring memory usage patterns for custom memory structures. An object is automatically marked as a potentially leaking object when it shows a positive and steep growth slope. The Memory Leak Dashboard provides the following information:

- Collection Size:** It provides the number of elements in a Collection.
- Potentially Leaking:** Potentially leaking Collections are marked as red. It is recommended to start diagnostic sessions on potentially leaking objects.
- Status:** Indicates if a diagnostic session has been started on an object.
- Collection Size Trend:** A positive and steep growth slope indicates potential memory leak.



Identify long-lived Collections by comparing the JVM start time and Object Creation Time. After the potentially leaking Collections are identified, start the diagnostic session.

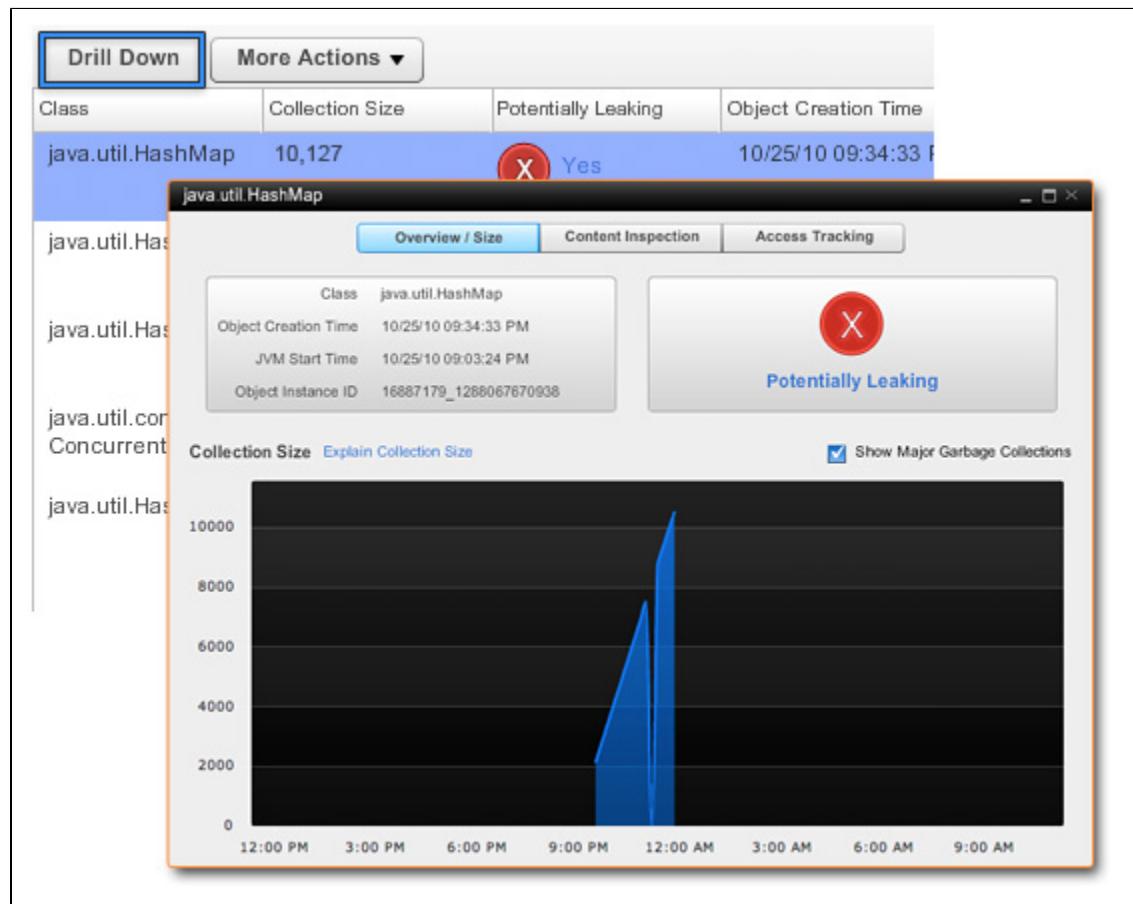
Diagnosing memory leaks

Select the class name to monitor and click **Drill Down** or right-click on the class name and select **Drill Down**.

Isolating a leaking collection

Use Content Inspection to identify to which part of the application the Collection belongs. It allows monitoring histograms of all the elements in a particular memory structure. Start a diagnostic session on the object and then follow these steps:

1. Select **Content Inspection**.
2. Click **Start Content Summary Capture Session**.
3. Enter the session duration. Allow at least 1-2 minutes for the data to generate.
4. Click **Refresh** to retrieve the session data.
5. Click on the particular snapshot to view the details about an individual session.



Access Tracking

Use Access Tracking to view the actual code paths and business transactions accessing the memory structure. Start a diagnostic session on the object and follow these steps:

1. Select the "Access Tracking".
2. Select "Start Access Tracking Session".
3. Enter the session duration. Allow at least 1-2 minutes for data generation.
4. Click the refresh button to retrieve the session data.
5. Click on the particular snapshot, to view the details about an individual session.

Learn More

- [Troubleshoot Java Memory Leaks](#)

Configure Object Instance Tracking (Java)

- Prerequisites for Object Instance Tracking
 - To enable object instance tracking on a node
- Tracking Specific Classes
 - To track instances of custom classes
- [Learn More](#)

This topic helps you understand how to configure object instance tracking. For more information about why you may need to configure this, see [Troubleshoot Java Memory Thrash](#).

Prerequisites for Object Instance Tracking

- Object Instance Tracking can be used only for Sun JVM v1.6.x and later.
- If you are running with the JDK then tools.jar will already be in the CLASSPATH, but if you are running with the JRE, you must add tools.jar to the CLASSPATH and restart the JVM for this feature to start working.

Adding tools.jar file:

For AppDynamics to read the tools.jar, add this file to jre/lib/ext directory and to the classpath as shown below (along with other -jar options):

```
java -classpath <complete-path-to-tools.jar>:%classpath% -jar myApp.jar  
OR  
java -classpath <complete-path-to-tools.jar>:$classpath -jar myApp.jar
```

To enable object instance tracking on a node

1. In the left navigation pane, click **Servers -> App Servers -> <tier> -> <node>**. The Node Dashboard opens.
2. Click the Memory tab.
3. Click the Object Instance Tracking subtab.
4. Click the ON button.

Tracking Specific Classes

Check against the required set of classes to enable instance tracking for each set. For improved performance, only the top 20 classes are tracked.

Use Configure Custom Classes to track instances of specific classes.

Classes configured here will only be tracked if their instance count is among the top 1000 instance counts in the JVM.

To track instances of custom classes

1. In the left navigation pane, click **Servers -> App Servers -> <tier> -> <node>**. The Node Dashboard opens.
2. Click **Configure Custom Classes To Track** on the rightmost corner of the screen.
3. Click **Add**.
4. Click **Enabled**.

Learn More

- [Troubleshoot Java Memory Thrash](#)

Configure Memory Monitoring (Java)

- [Prerequisites for Object Instance Tracking](#)
 - To enable object instance tracking
- [Tracking Specific Classes](#)
 - To Specify Custom Classes to Track
- [Learn More](#)

This topic helps you understand how to configure memory monitoring, also known as object instance tracking. For more information about why you may need to configure this, see [Troubleshoot Java Memory Thrash](#).

Prerequisites for Object Instance Tracking

Object Instance Tracking can be used only for Sun JVM v1.6.x and later.

If you are running with the JDK, tools.jar will already be in the CLASSPATH, but if you are running with the JRE, you must add tools.jar to the CLASSPATH and restart the JVM for this feature to start working.

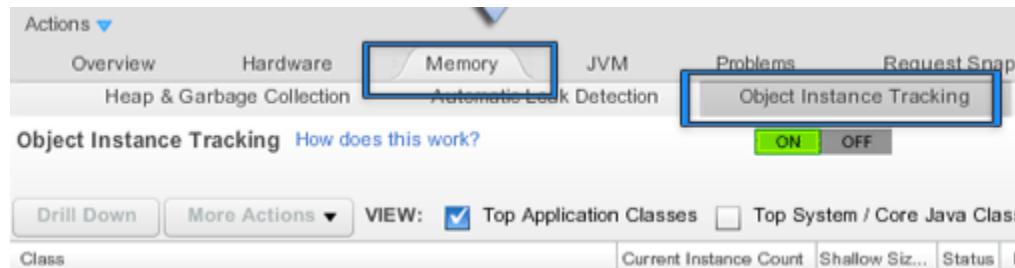
Adding tools.jar file:

For AppDynamics to read the tools.jar, add this file to jre/lib/ext directory and to the classpath as shown below (along with other -jar options):

```
java -classpath <complete-path-to-tools.jar>:%classpath% -jar myApp.jar
OR
java -classpath <complete-path-to-tools.jar>:$classpath -jar myApp.jar
```

To enable object instance tracking

1. Navigate to the node dashboard of the node for which you want to enable object instance tracing..
2. Click the **Memory** tab.
3. Select **Object Instance Tracking**.
4. Turn the object instance tracking mode ON.



Tracking Specific Classes

Check against the required set of classes to enable instance tracking for each set. For improved performance, only the top 20 classes are tracked.

VIEW: Top Application Classes Top System / Core Java Classes Custom Classes All

Use Configure Custom Classes to track instances of specific classes.
Classes configured here will only be tracked if their instance count is among the top 1000 instance counts in the JVM.

To Specify Custom Classes to Track

1. Click **Configuration -> Instrumentation -> Memory Monitoring**.
2. Select the tier in which to configure object tracking.
3. In the Object Instance Tracking. Define Custom Classes to Track section, click **Add**.
4. In the Create New Instance Tracker window, check **Enabled**.
5. Enter the fully qualified class name of the class to track.
6. Click **Save**.

You can edit or delete the object tracing configuration after it has been created.

Learn More

- Troubleshoot Java Memory Thrash

Configure Multi-Threaded Transactions (Java)

- Default Configuration
- Custom Configuration
 - Managing Thread Correlation Using a Node Property
- Enabling and Disabling Asynchronous Monitoring
 - To Disable Asynchronous Monitoring

- To Enable Asynchronous Monitoring
- Learn More

AppDynamics collects and reports key performance metrics for individual threads in multi-threaded Java applications. See [Trace Multi-Threaded Transactions \(Java\)](#) for details on where these metrics are reported.

Default Configuration

Classes for multi-threaded correlation are configured in the <excludes> child elements of the <fork-config> element in the <App_Server_Agent_Installation_Directory>/conf/app-agent-config.xml file.

The default configuration excludes the java, org, weblogic and websphere classes:

```
<fork-config>
  <!-- exclude java and org -->
  <excludes filter-type="STARTSWITH"
    filter-value="java/, javax/, com.sun/, sun/, org/" />
  <!-- exclude weblogic and websphere -->
  <excludes filter-type="STARTSWITH"
    filter-value="com.bea/, com.weblogic/, weblogic/, com.ibm/" />
  . . .

```

Custom Configuration

You can edit the app-agent-config.xml file to exclude additional classes from thread correlation. All classes not excluded are by default included.

You can also explicitly include sub-packages and subclasses of excluded packages and classes. Use the <includes> or <include> child element to specify the included packages and classes.

Use the <excludes> and <includes> elements to specify a comma-separated list of classes or packages. Use the <exclude> and <include> elements to specify a single class or package

Managing Thread Correlation Using a Node Property

You can also configure which classes or packages to include or exclude using a node property. See [thread-correlation-classes](#) and [thread-correlation-classes-exclude](#).

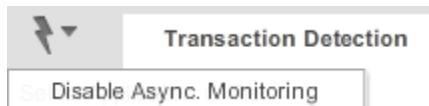
Enabling and Disabling Asynchronous Monitoring

 You should disable monitoring of multi-threaded transactions on all agents if all of your agents and your controller are not at AppDynamics version 3.6 or higher. You can enable the feature after all of your agents have been upgraded.

You must restart the agent after you enable or disable this feature.

To Disable Asynchronous Monitoring

1. In the left navigation pane click **Configure->Instrumentation->Transaction Detection**.
2. From the Actions menu in the upper left corner click **Disable Async Monitoring**.



To Enable Asynchronous Monitoring

1. In the left navigation pane click **Configure->Instrumentation->Transaction Detection**.
2. From the Actions menu in the upper left corner click **Enable Async Monitoring**.



Learn More

- [Configure Business Transaction Detection](#)
- [App Agent Node Properties](#)

Configure Background Tasks (Java)

- Pre-Configured Frameworks for Java Background Tasks
- Enabling Automatic Discovery for Background Tasks
 - To enable discovery for a background task using a common framework
- Configuring Background Batch or Shell Files using a Main Method
 - To instrument the main method of a background task
- [Learn More](#)

In a Java environment background tasks are detected using POJO entry points. It is the same basic procedure as defining entry points for business transactions, except that you check the Background Task check box. For instructions see [Configure Background Tasks](#).

Pre-Configured Frameworks for Java Background Tasks

When enabled, AppDynamics provides discovery for the following Java background-processing task frameworks:

- Quartz
- Cron4J
- JCronTab
- JavaTimer

Configure Background Tasks

Enabling Automatic Discovery for Background Tasks

Automatic discovery of background tasks is disabled by default. When you know that there are background tasks in your application environment and you want to monitor them, first enable automatic discovery so that AppDynamics will detect the task.

AppDynamics provides preconfigured support for some common frameworks. If your application is not using one of the default frameworks you can create a custom match rule.

To enable discovery for a background task using a common framework

1. In the left navigation pane, click **Configure -> Instrumentation**.
2. On the Transaction Detection tab, select the tier for which you want to enable monitoring.
3. Click **Use Custom Configuration for this Tier**.
4. Scroll down to the Custom Match Rules pane.
5. Do one of the following
 - If you are using a pre-configured framework, select the row of the framework and click the pencil icon, or double-click on the row to open the Business Transaction Match Rule window. By default the values are populated with rule name and the class and method names for the particular framework. Verify that those are the correct names for your environment.
OR
 - If you are using a custom framework, select the match criteria and enter the Class Name and Method Name.

The Background Task check box should be already checked.

6. Check **Enabled**.

7. Click **Save**.

The custom match rule for the background task will take effect and the background task will display in the Business Transaction List.

Once you enable discovery, every background task is identified based on following attributes:

- Implementation class name
- Parameter to the execution method name

Configuring Background Batch or Shell Files using a Main Method

Sometimes background tasks are defined in batch or shell files in which the main method triggers the background processing. In this situation, the response time of the batch process is the duration of the execution of the main method.

 **IMPORTANT:** Instrument the main method only when the duration of the batch process is equal to the duration of the main method. Otherwise choose another method that accurately represents the unit of work for the background process.

To instrument the main method of a background task

1. In the left navigation pane, click **Configure -> Instrumentation**.
2. In the **Transaction Detection** section select the tier for which you want to instrument the main method.
3. In the **Custom Rules** section click **Add** (the "+" icon).
4. From the **Entry Point** type drop down list, click **POJO**.
5. Enter a name for the custom rule.
6. Check **Background Task**.

7. Check **Enabled**.

8. Enter "main" as the match value for **Method Name**.

New Business Transaction Match Rule - POJO

Name	OvernightBatchRun
Enabled	<input checked="" type="checkbox"/>
Background Task	<input checked="" type="checkbox"/>

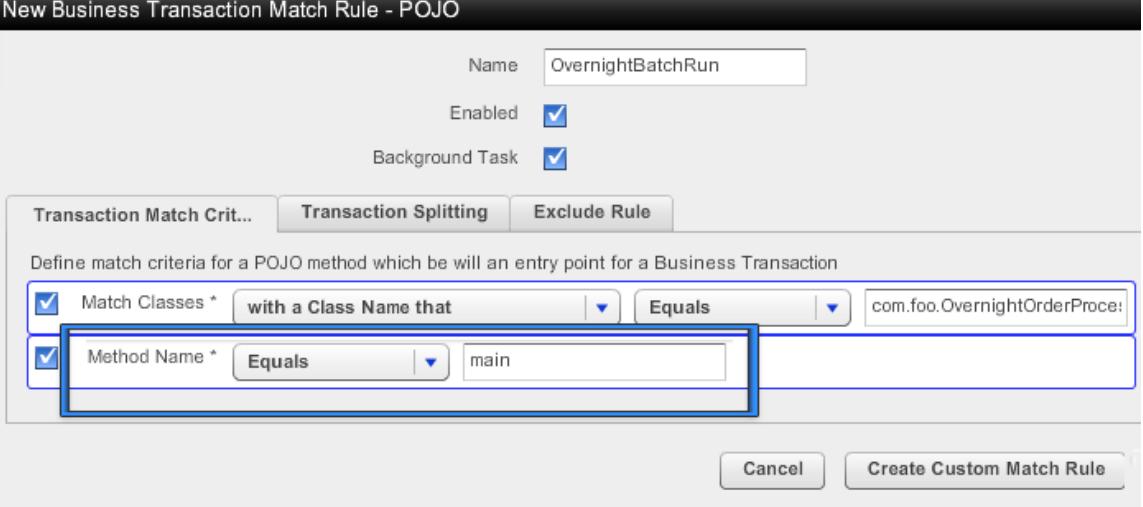
Transaction Match Crit... Transaction Splitting Exclude Rule

Define match criteria for a POJO method which will be an entry point for a Business Transaction

Match Classes * with a Class Name that Equals com.foo.OvernightOrderProce:

Method Name * Equals main

Cancel Create Custom Match Rule



9. Save the changes.

10. To ensure that the name of the script file is automatically picked up as a background task, configure your Java Agent for that node. See [Configure App Agent for Java for Batch Processes](#).

Learn More

- [Configure Background Tasks](#)
- [Configure App Agent for Java for Batch Processes](#)
- [POJO Entry Points](#)

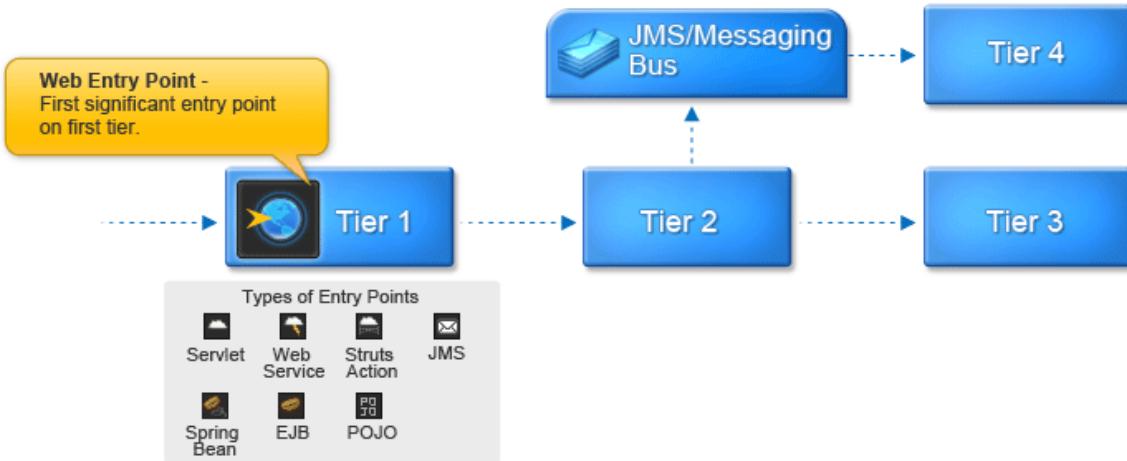
Web Application Entry Points

- [Tiers and Web Application Entry Points](#)
- [Other Web Application Frameworks Based on Servlets or Servlets Filter](#)
- [Learn More](#)

This section discusses web application entry points for different types of business transactions.

Tiers and Web Application Entry Points

A tier can have multiple entry points.



For example, for Java frameworks a combination of pure Servlets or JSPs, Struts, Web services, Servlet filters, etc. may all co-exist on the same JVM.

The middle-tier components like EJBs and Spring beans are usually not considered Entry Points because they are normally accessed using either the front-end layers such as Servlets or from classes that invoke background processes.

Other Web Application Frameworks Based on Servlets or Servlets Filter

AppDynamics provides out-of-the-box support for most of the common web frameworks that are based on Servlets or Servlet Filters. When using any of the frameworks listed below, refer to the [Servlet discovery rules](#) to configure transaction discovery.

- Spring MVC
- Wicket
- Java Server Faces (JSF)
- JRuby
- Grails
- Groovy
- Tapestry
- ColdFusion

Learn More

[Collapse all](#) [Expand all](#) [Collapse all](#)

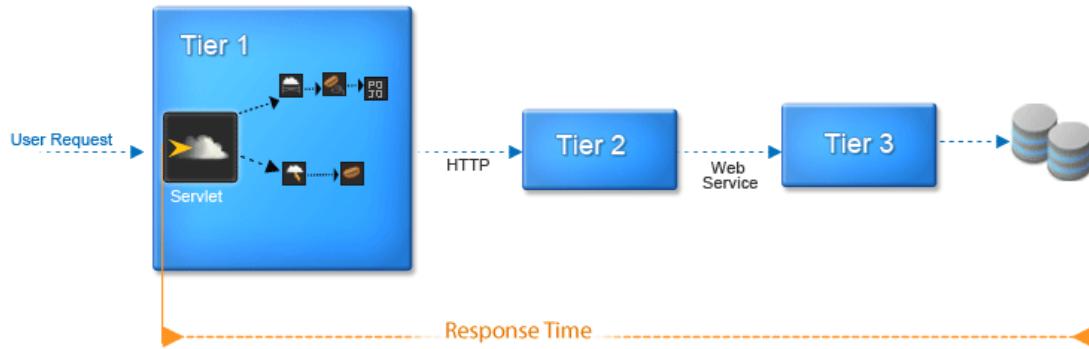
Servlet Entry Points

- **Servlet-Based Business Transactions**
 - Default Naming for Servlet-Based Transactions
 - Naming Using Other URI Patterns
 - Using Headers, Cookies, and Other Parts of HTTP Requests for identifying Transactions
 - Naming Servlet Based Transactions when Different Web Contexts Require Different Naming Strategies
 - Custom Match Rules for Servlet Based Transactions
 - Example 1: URL is <http://acmeonline.com/store/checkout>
 - Example 2: URL is <http://acmeonline.com/secure/internal/updateinventory>
 - Example 3: URL is <http://acmeonline.com/orders/process?type=creditcard>
 - Using Custom Expressions in Custom Match Rules
 - Request Attributes in the Custom Expression
- Configuring Transaction Identification for REST-Style URLs
- Configuring Transaction Identification Based on Information in the Body of the Servlet Request

This describes how to configure transaction entry points for Servlet-based methods that may or may not be used as part of an application framework.

Servlet-Based Business Transactions

AppDynamics allows you to configure a transaction entry point on the invocation of the service method of a Servlet. The response time for the Servlet transaction is measured when the Servlet entry point is invoked.



Default Naming for Servlet-Based Transactions

By default, AppDynamics identifies all Servlet-based transactions using the first two segments of the URI.

For example, if the URI for a checkout operation in an online store is

<http://acmeonline.com/store/checkout>

AppDynamics automatically names this transaction "store/checkout".

If the URI for a transfer funds operation in an online bank is

<http://acmebank.com/account/transferfunds/northernncalifornia>

AppDynamics automatically names this transaction "/account/transferfunds".

Servlet Transaction Naming Configuration

What part of the URI should be used in the Transaction Name?

Use the full URI
 Use a part of the URI (for example, if you have dynamic URIs)
Use the first ▾ 2 segments of the URI in Transaction Names [What does this do?](#)

Name Transactions dynamically using part of the request

Use URI segment(s) in Transaction names
Segment Numbers Enter a comma separated list of parameter numbers (e.g. 1,3,4)

Use a parameter value in Transaction names
Parameter Name

Use a header value in Transaction names
Header Name

Use a cookie value in Transaction names
Cookie Name

Use a session attribute value in Transaction names
Session Attribute Key

Use the request method (GET/POST/PUT) in Transaction names

Use the request host Transaction in names

Use the request originating address in Transaction names

Apply a custom expression on HttpServletRequest and use the result in Transaction Names [Explain This](#)

Naming Using Other URI Patterns

The AppDynamics default naming convention might not be optimal for all applications. You can configure AppDynamics to use different segments of the URI or other values (such as a header value, a cookie the request method, etc.).

For example the following URL represents checkout operation in ACME Online:

<http://acmeonline.com/web/store/checkout>

AppDynamics' default naming scheme identifies these transactions by the first two segments of the URI: "/web/store".

This naming convention does not indicate the functionality of the operation (checkout, add to cart, etc.) A better approach would identify such URLs using the last two segments: "store/checkout".

Servlet Transaction Naming Configuration

What part of the URI should be used in the Transaction Name?

Use the full URI
 Use a part of the URI (for example, if you have dynamic URIs)

Use the last ▾ 2 segments of the URI in Transaction Names

In certain situations, the URI might not contain enough information to be able to name the transaction. This is very common in dispatcher-style Servlet patterns where the Servlet dispatches requests based on a query parameter.

For example, for the following URL

<http://acmeonline.com/dispatcher?action=checkout>

it is ideal to name the transaction based on the value of the action parameter.

To configure this, specify the transaction detection based on the parameter value for the "action" parameter.

Servlet Transaction Naming Configuration

What part of the URI should be used in the Transaction Name?

Use the full URI
 Use a part of the URI (for example, if you have dynamic URIs)

Use the first ▾ 2 segments of the URI in Transaction Names

Name Transactions dynamically using part of the request

Use URI segment(s) in Transaction names
Segment Numbers Enter a comma separated list of segment numbers

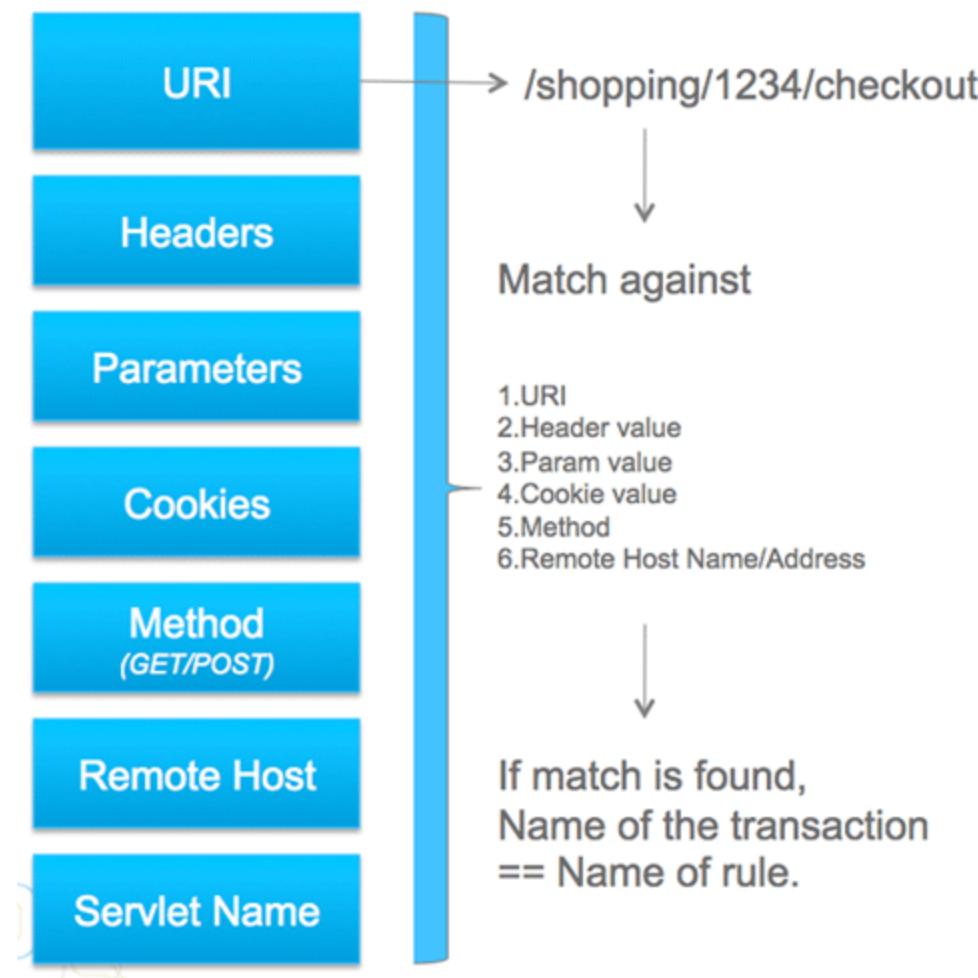
Use a parameter value in Transaction names

Parameter Name action

Discovered transactions will be automatically named as "Checkout".

Using Headers, Cookies, and Other Parts of HTTP Requests for identifying Transactions

You can also configure the naming for your Servlet based transactions using headers, cookies, and other parts of HTTP requests.



To identify all your Servlet based transactions using particular parts of the HTTP request., use the Name Transactions dynamically using part of the request option:

Servlet Transaction Naming Configuration

What part of the URI should be used in the Transaction Name?

Use the full URI
 Use a part of the URI (for example, if you have dynamic segments in your URI, choose this option)
 Use the first 2 segments of the URI

Enable this option to name the transactions using part of the request.

Name Transactions dynamically using part of the request

Use URI segment(s) in Transaction names
 Segment Numbers Enter a comma separated list of parameter numbers (e.g. 1,3,4)

Use a parameter value in Transaction names
 Parameter Name

Use a header value in Transaction names
 Header Name

Use a cookie value in Transaction names
 Cookie Name

Use a session attribute value in Transaction names
 Session Attribute Key

Use the request method (GET/POST/PUT) in Transaction names

Use the request host Transaction in names

Use the request originating address in Transaction names

Apply a custom expression on HttpServletRequest and use the result in Transaction Names

Cancel **Save**

Alternatively, you can also configure detection for only specific transactions using a custom match rule.

Naming Servlet Based Transactions when Different Web Contexts Require Different Naming Strategies

In certain situations, it is ideal to modify the existing naming (which is based on the URI patterns) and name your transactions based on different web contexts. You can also configure the transaction naming to directly map the WAR files deployed on the same tier.

For example, ACME Online's entry point tier has multiple contexts deployed on a single JVM, as listed in the table below. These web contexts represent two different parts of the same business application and therefore require following different naming strategies.

Web context	Ideal naming strategy
http://acmeonline.com/store/checkout	Should use first two segments to name these requests as: /store/checkout
http://acmeonline.com/secure/internal/updateinventory	Should use last two segments to name these requests as: /internal/updateinventory

<http://acmeonline.com/orders/process?type=creditcard>

Should use the combination of parameter value for "type" and the last two segments to name such requests as: **/orders/process.creditcard**
Such a naming strategy will ensure that the credit card orders are differentiated from other orders.

To learn more about how to configure transaction identification for these situations see [Custom match rules for Servlet based requests](#).

Custom Match Rules for Servlet Based Transactions

To handle situations as discussed in the previous section, use custom match rules for each of the web contexts or URIs. Custom match rules let you create mutually exclusive transaction names for specific contexts.

See [Custom Match Rules](#) for general information about accessing custom match rule configuration. To configure a rule for Servlets, select **Servlet** the drop-down list.

Here are some sample rules for different URLs.

Example 1: URL is <http://acmeonline.com/store/checkout>

New Business Transaction Match Rule - Servlet

Name: ACME

Enabled:

Priority: 0

This custom rule will name all the qualifying requests as "ACME.store.checkout" transaction.

1 Transaction Match Criteria Split Transactions Us

Method: URI: Starts With: /store

2 3

4 Split Transactions using request data

5 Use the first 2 segments in Transaction names

Example 2: URL is <http://acmeonline.com/secure/internal/updateinventory>

New Business Transaction Match Rule - Servlet

Name: ACME
Enabled:
Priority: 0

1 Transaction Match Criteria

Method:

2 URI: Starts With: /secure

3 Split Transactions Using Request Data

4 Split Transactions using request data

Use the first segments in Transaction names

5 Use the last segments in Transaction names

This custom rule will name all the qualifying requests as "ACME.internal.updateinventory" transaction.

Example 3: URL is <http://acmeonline.com/orders/process?type=creditcard>

New Business Transaction Match Rule - Servlet

Name: ACME
Enabled:
Priority: 0

1 Transaction Match Criteria

Method:

2 URI: Starts With: /orders

3 HTTP Parameter
(Both GET query parameters and POST parameters can be used)

Check for parameter value:

Parameter Name: type
Value: Equals: creditcard

4 Split Transactions Using Request Data

5 Split Transactions using request data

Use the first segments in Transaction names

6 Use the last segments in Transaction names

This custom rule will name all the qualifying requests as "ACME.orders.process.creditcard" transaction.

Using Custom Expressions in Custom Match Rules

You can create custom expressions to name transactions based on the evaluation of a custom expression on the HttpServletRequestObject.

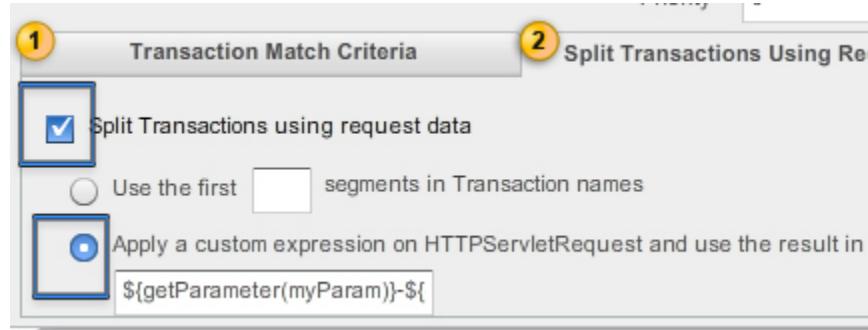
Suppose you want to monitor the HttpServletRequest request variable to identify the names and values of all the request parameters passed with the request.

The following example shows a custom rule configuration based on the expression:

```
 ${getParameter(myParam)}-${getUserPrincipal().getName()}
```

which evaluates to

```
 request.getParameter( "myParam" )+"-"+request.getUserPrincipal()
```



You can create a custom expression on the HttpServletRequest to identify all Servlet based requests (modify global discovery at the transaction naming level) or for a specific set of requests (custom rule).

Request Attributes in the Custom Expression

A custom expression can have a combination of any of the following getter chains on the request attributes:

Getters on Request Attributes	Transaction Identification
getAuthType()	Use this option to monitor secure (or insecure) communications.
getContextPath()	Use this option to identify the user requests based on the portion of the URI.
getHeader()	Identify the requests based on request headers.
getMethod()	Identify user requests invoked by a particular method.
getPathInfo()	Identify user requests based on the extra path information associated with the URL (sent by the client when the request was made).
getQueryString()	Identify the requests based on the query string contained in the request URL after the path.
getRemoteUser()	Identify the user requests based on the login of the user making this request.
getRequestedSessionId()	Identify user requests based on the session id specified by the client.
getUserPrincipal()	Identify user requests based on the current authenticated user.

For example, the following custom expression can be used to name the business transactions using the combination of header and a particular parameter:

```
 ${getHeader(header1)}-${getParameter(myParam)} .
```

The identified transaction will be named based on the result of following expression in your application code:

```
 request.getHeader( "header1" )+ " - "+ request.getParameter( "myParam" )
```

Configuring Transaction Identification for REST-Style URLs

REST applications typically have dynamic URLs where the application semantics are a part of the URL.

For example, suppose customer id or the order id can be a part of the URL. The following URL represents the checkout transaction invoked by a customer with id 1234: <http://acmeonline.com/store/cust1234/checkout>. The default discovery mechanism names this transaction "/store/cust1234". Ideally, all the customers performing a checkout operation should also be part of the checkout business transaction. Therefore, a better approach would be to name this transaction "/store/checkout".

To configure the transaction to be named "/store/checkout", use the first segment of the URL and split that URL using the third segment, thus avoiding the second (dynamic) segment.

Configuration for global discovery

The following screenshot illustrates this configuration for global discovery.

The screenshot shows the "Servlet Transaction Naming Configuration" dialog. It asks "What part of the URI should be used in the Transaction Name?". Two options are available: "Use the full URI" (radio button) and "Use a part of the URI (for example, if you have dynamic URIs)" (radio button, selected). Below this is a dropdown menu set to "Use the first" with a value of "1" and the text "segments of the URI in Transaction Names". A checked checkbox "Name Transactions dynamically using part of the request" has a tooltip explaining it modifies global discovery for all Servlet based transactions and names the transaction "Store.checkout". Another checkbox "Use URI segment(s) in Transaction names" is checked, with a dropdown menu set to "3" and the text "Segment Numbers 1,3 Enter a comma separated list of parameter numbers".

Configuration for custom match rule:

The following screenshot illustrates this configuration using a custom match rule.

The screenshot shows the "New Business Transaction Match Rule - Servlet" dialog. It includes fields for "Name" (ACME), "Enabled" (checked), and "Priority" (0). Under "Transaction Match Criteria", there are checkboxes for "Method" (GET) and "URI" (selected, Equals, value: /store). An arrow points from the "URI" section to the "Split Transactions Using Request Data" tab. This tab contains a checked checkbox "Split Transactions using request data" and three radio button options: "Use the first" (radio button), "Use the last" (radio button), and "Use URI segment(s)" (radio button, selected). The "Segment Numbers" dropdown is set to "1,3" with the placeholder "Enter a comma separated list of parameter numbers".

Another example illustrates the rules for the following URL: <http://acmeonline.com/user/foo@bar.com/profile/profile2345/edit>. The ideal detection strategy should avoid multiple segments (in this case, we need segments 1,3, and 5).

Configuration for global discovery:

The following screenshot illustrates the configuration for global discovery.

Servlet Transaction Naming Configuration

What part of the URI should be used in the Transaction Name?

Use the full URI
 Use a part of the URI (for example, if you have dynamic URIs)

Use the first 1 segments of the URI in Transaction names

Name Transactions dynamically using part of the request
 Use URI segment(s) in Transaction names

Segment Numbers Enter a comma separated list of parameter numbers

This configuration modifies the default global discovery for all Servlet based transactions. The requests will be named as "User.profile.edit" transaction.

Configuration for custom match rule:

The following screenshot illustrates the configuration for creating a custom match rule.

New Business Transaction Match Rule - Servlet

Name: <input type="text" value="ACME"/>	Transaction Match Criteria	Split Transactions Using Request Data
Enabled: <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Split Transactions using request data	
Priority: <input type="text" value="0"/>	<input type="radio"/> Use the first <input type="text" value="1"/> segments in Transaction names	
	<input type="radio"/> Use the last <input type="text" value="1"/> segments in Transaction names	
Method: <input type="button" value="GET"/>	<input checked="" type="radio"/> Use URI segment(s) in Transaction names	
URI: Equals <input type="button" value="▼"/> /user	Segment Numbers <input type="text" value="1,3,5"/> Enter a comma separated list of parameter numbers	

Configuring Transaction Identification Based on Information in the Body of the Servlet Request

In certain situations, you might have an application that receives an XML/JSON payload as part of the POST request.

If the category of processing is a part of the XML/JSON, neither the URI nor the parameters/headers have enough information to name the transaction. Only the XML contents that can provide correct naming configuration.

See [Identify Transactions Based on DOM Parsing Incoming XML Payload](#) and [Identify Transactions for Java XML Binding Frameworks](#).

Servlet Transaction Detection Scenarios

Identify Transactions Based on DOM Parsing Incoming XML Payload

- Business Transactions and XML Payload
 - Identifying the Business Transaction Using the XPath Expression
 - To configure a custom match rule
 - Learn More

This topic describes how to identify transactions when an XML is posted to a Servlet.

Business Transactions and XML Payload

The XML contains the naming information for the Business Transaction. The Servlet uses a DOM parser to parse the posted XML into a DOM object.



Posted XML is parsed into a DOM object

Identifying the Business Transaction Using the XPath Expression

For example, the following XML posts an order for three items. The order uses credit card processing.

```

<acme>
  <order>
    <type>creditcard</type>
    <item>Item1</item>
    <item>Item2</item>
    <item>Item3</item>
  </order>
<acme>

```

The URL is:

```
http://acmeonline.com/store
```

The doPost method of the Servlet is:

```

public void doPost(HttpServletRequest req, HttpServletResponse resp)
{
    DocumentBuilderFactory docFactory =
    DocumentBuilderFactory.newInstance();
    DocumentBuilder docBuilder = docFactory.newDocumentBuilder();
    Document doc = docBuilder.parse(req.getInputStream());

    Element element = doc.getDocumentElement();

    //read the type of order
    //read all the items
    processOrder(orderType,items)
    ....
}

```

The XPath expression "//order/type" on this XML payload evaluates to "creditcard".

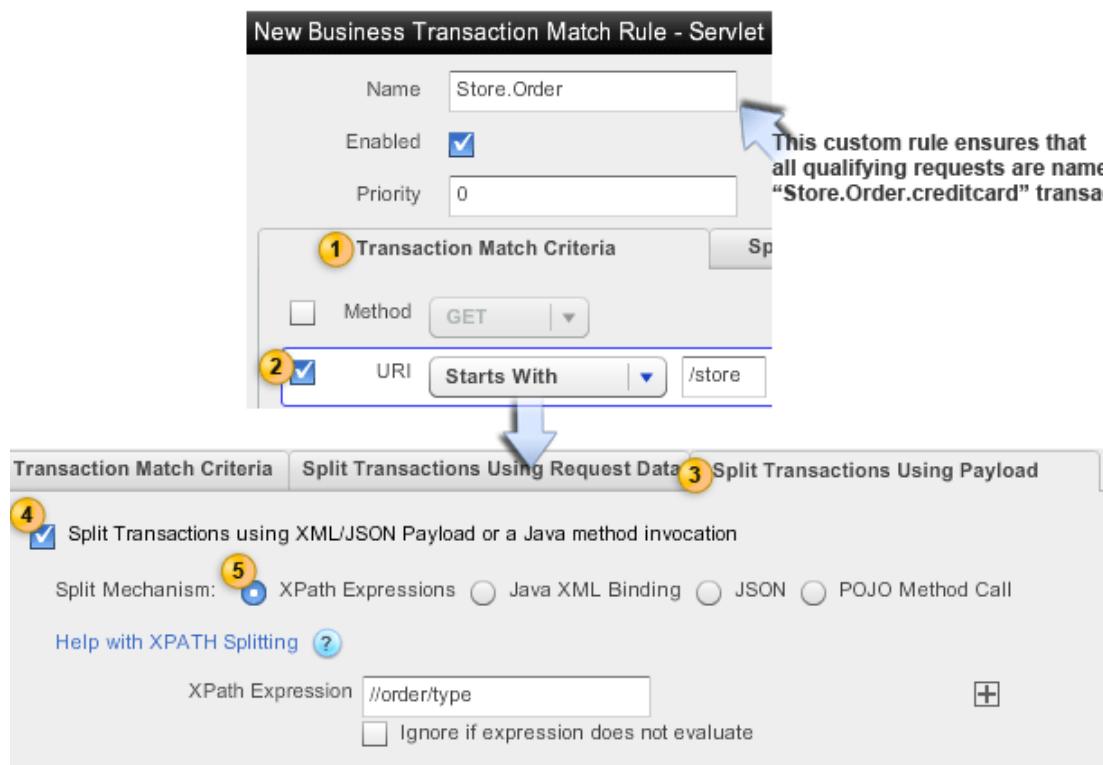
This value correctly identifies the type of the order and therefore should be used to name the "Order" transaction.

To identify the Business Transactions in this manner, first configure a custom match rule that automatically intercepts the method that parses the XML and gets the DOM object.

You use the XPath expression in the custom rule so that it names the transaction, for example "Store.order.creditcard". Though the name is not obtained until the XML is parsed, AppDynamics measures the duration of the business transaction to include the execution of the doPost() method.

To configure a custom match rule

1. Navigate to the custom rule section for Servlets.
2. In the **Transaction Match Criteria** tab, specify the URI.
3. In the **Split Transactions Using Payloads** tab, enable **Split transactions using XML/JSON Payload or a Java method invocation**.
4. Set the split mechanism to **XPath Expressions**.
5. Enter the XPath expression that you want to set as the Entry Point. The result of the XPath expression will be appended to the name of the Business Transaction.



You can use one or more XPath expressions to chain the names generated for the Business Transaction.

If the expression does not evaluate to a value, the transaction will not be identified.

Learn More

- [Servlet Entry Points](#)

Identify Transactions Based on POJO Method Invoked By a Servlet

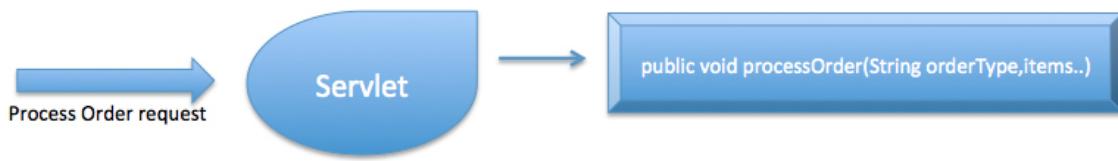
- [Using a Java Method to Name a Transaction](#)
 - To configure the custom match rule
- [Learn More](#)

Using a Java Method to Name a Transaction

You can use a Java method to name the transaction where:

- You might not have a clear URI pattern or
- You are using XML/JSON frameworks that are currently not supported by AppDynamics.

The following illustration shows how the Servlet that invokes the POJO method holds the transaction name.



For example, consider the `processOrder()` method. The order is parsed using any type of method and eventually when the `processOrder()` method is invoked, a correct approach to name transaction is to capture the first parameter to the `processOrder()` method.

The following URL is used by these requests: <http://acmeonline.com/store>. Following code snippet shows the `doPost()` method of the Servlet:

```

public void doPost(HttpServletRequest req, HttpServletResponse resp)
{
    //process the data from the sevlet request and get the orderType and the items
    processOrder(orderType,item)
    .....
}
public void processOrder(String orderType, String item)
{
    //process order
}
  
```

The `processOrder()` method has the information which can correctly derive the type of the order and also the name of the transaction. To identify these requests as a single transaction, configure a custom match rule.

To configure the custom match rule

1. Go to the custom rule section for Servlet Entry Points
2. In the **Transaction Match Criteria** tab, specify the URI.
3. In the **Split Transactions Using Payload** tab, enable Split transactions using XML/JSON Payload or a Java method invocation.
4. Select POJO Method Call as the split mechanism .
5. Enter the class and the method name.
6. If the method is overloaded, also specify the details for the arguments.
7. Optionally, you can name your transactions by defining multiple methods in a getter chain in the Method Call Chain field.

The following screenshot displays the configuration of a custom match rule which will name all the qualifying requests into a "Store.order.creditcard" transaction:

New Business Transaction Match Rule - Servlet

Name: Store.Order
Enabled:
Priority: 0

1 Transaction Match Criteria

Method: GET URI Equals /store

2 Split Transactions Using Request Data

3 Split Transactions Using Payload

4 Split Transactions using XML/JSON Payload or a Java method invocation

Split Mechanism: XPath Expressions Java XML Binding JSON POJO Method Call

Help with POJO Splitting [?](#)

Class Name: com.acme.Order.ProcessOrder
Method Name: processOrder()
Return Type:
Number of Arguments: 2
Argument Index: 0
Method Call Chain: +
For example: getPerson().getAddress().getStreet()

This custom rule ensures that the processOrder method is automatically intercepted.

Although the name is not obtained till the processOrder() method is called, the time for the transaction will include all of the doGet() method.

In addition to the parameter, you can also specify either the return type or a recursive getter chain on the object to name the transaction. For example, if the method parameter points to a complex object like PurchaseOrder, you can use something like getPurchaseDetails().getType() to correctly name the transaction.

Learn More

- [Servlet Entry Points](#)

Identify Transactions for Java XML Binding Frameworks

- To Configure the Custom Match Rule
- Supported Java XML data binding frameworks
- [Learn More](#)

The following illustration shows the situation in which an XML is posted to a Servlet. The Servlet uses an XML-to-Java binding framework, such as XMLBeans or Castor, to unmarshal and read the posted XML payload.



The XML payload contains the naming information for the transactions.

In the following example, an XML payload posts an order for three items. It uses a credit card to process the order.

The URL is: <http://acmeonline.com/store>

```

<acme>
  <order>
    <type>creditcard</type>
    <item>Item1</item>
    <item>Item2</item>
    <item>Item3</item>
  </order>
</acme>

```

The following code snippet shows the `doPost()` method of the Servlet:

```

public void doPost(HttpServletRequest req, HttpServletResponse resp)
{
    PurchaseOrderDocument poDoc = PurchaseOrderDocument.Factory.parse(po);

    PurchaseOrder po = poDoc.getPurchaseOrder();
    \\
    String orderType = po.getOrderType();

    //read all the items
    processOrder(orderType,items)

    ...
}

```

After the posted XML is unmarshalled to the `PurchaseOrder` data object, the `getOrderType()` method should be used to identify the type of the order.

To Configure the Custom Match Rule

1. Navigate to the custom rule section for **Servlet Entry Points**.
2. In the **Transaction Match Criteria** tab, specify the URI.
3. In the **Split Transactions Using Payload** tab, check Split transactions using XML/JSON Payload or a Java method invocation.
4. Select Java XML Binding as the split mechanism.
5. Enter the class name and the method name.

The screenshot below shows a custom match rule which identifies the business transaction for this example as "Store.order.creditcard":

New Business Transaction Match Rule - Servlet

This custom rule ensures that all qualifying requests are named as "Store.Order.creditcard" transaction.

Name	Store.Order
Enabled	<input checked="" type="checkbox"/>
Priority	0

1 Transaction Match Criteria

Method: GET

2 URI Equals /store

3 Split Transactions Using Request Data

4 Split Transactions using XML/JSON Payload or a Java method invocation

Split Mechanism: XPath Expressions Java XML Binding JSON POJO Method Call

[Help with Java XML Binding Splitting](#)

Unmarshaled Class Name: PurchaseOrderDocument

Method Call Chain: getPurchaseOrder() [+](#)

For example: `getPerson().getAddress().getStreet()`

This custom rule ensures that the method in XMLBeans (which unmarshals XML to Java objects) is automatically intercepted. It also ensures that the `getOrderType()` method is applied on the Java data object only if it is the `PurchaseOrder` data object.

If the name of the transaction is not on a first level getter on the unmarshalled object, you can also use a recursive getter chain such as `getOrderType().getOrder()` to get the name.

Although the transaction name is not obtained until the XML is unmarshalled, the response time for the transaction is calculated from the `doGet()` method.

Supported Java XML data binding frameworks

The following Java XML data binding frameworks are supported:

- Castor
- JAXB
- JibX
- XMLBeans
- XStream

Learn More

- [Servlet Entry Points](#)

Identify Transactions Based on JSON Payload

- [Example Configuration for a JSON Payload](#)
- [Learn More](#)

Example Configuration for a JSON Payload

The following illustration shows a JSON payload posted to a Servlet and the Servlet unmarshalls the payload.



The JSON contains the naming information for the transactions.

For example, the following JSON payload posts an "order" for an item "car" and uses creditcard for processing the order. The URL is:

<http://acmeonline.com/store>

```

order
: {
type:creditcard,
id:123,
name:Car,
price:23
}}

```

The following code snippet shows the doPost method of the Servlet:

```

public void doPost(HttpServletRequest req, HttpServletResponse resp)
{
    //create JSONObject from servlet input stream
    String orderType = jsonObject.get("type")\\
    //read the item for the order\\
    processOrder(orderType,item)\\
    ....\\
}

```

After the posted JSON payload is unmarshalled to the JSON object, the "type" key is required to identify the type of the order. In this case, this key uniquely identifies the business transaction.

To configure this rule:

1. Go to the custom rule section for [Servlet Entry Points](#).
2. Under "Transaction Match Criteria" specify the URI.
3. Under "Split Transactions Using Payloads", enable "Split transactions using XML/JSON Payload or a Java method invocation".
4. Select the split mechanism as "JSON".
5. Enter the JSON object key.

The following screenshot displays the configuration for a custom match rule that will name all the qualifying requests into a single transaction called "Store.Order.creditcard".

This custom rule ensures that all qualifying requests are named as "Store.Order.creditcard" transaction.

Name: Store.Order
Enabled:
Priority: 0

1 Transaction Match Criteria

Method: GET

2 URI Equals /store

3 Split Transactions Using Request Data

4 Split Transactions using XML/JSON Payload or a Java method invocation

Split Mechanism: XPath Expressions Java XML Binding **5** JSON POJO Method Call

NOTE: The 'enable-json-bci-rules' agent property must be set to true for each node in this Tier for this match rule to work.

Help with JSON Splitting [?](#)

JSON Object Key: type

6. Set the property "enable-json-bci-rules" to "true" for each node to enable this custom rule. See [App Agent Node Properties](#).

This configuration ensures that the JSONObject and the get("\$JSON_Object_Key") method on this object are intercepted automatically to get the name of the transaction. Although the transaction name is not obtained until the JSON object is unmarshalled, the response time for the transaction will be calculated from the doGet method.

Learn More

- [Servlet Entry Points](#)
- [App Agent Node Properties](#)

Struts Entry Points

- [Struts-based Transactions](#)
- [Struts Request Names](#)
- [Custom Match Rules for Struts Transactions](#)
- [Exclude Rules for Struts Actions or Methods](#)

Struts-based Transactions

When your application uses Struts to service user requests, AppDynamics intercepts individual Struts Action invocations and names the user requests based on the Struts action names. A Struts entry point is a Struts Action that is being invoked.

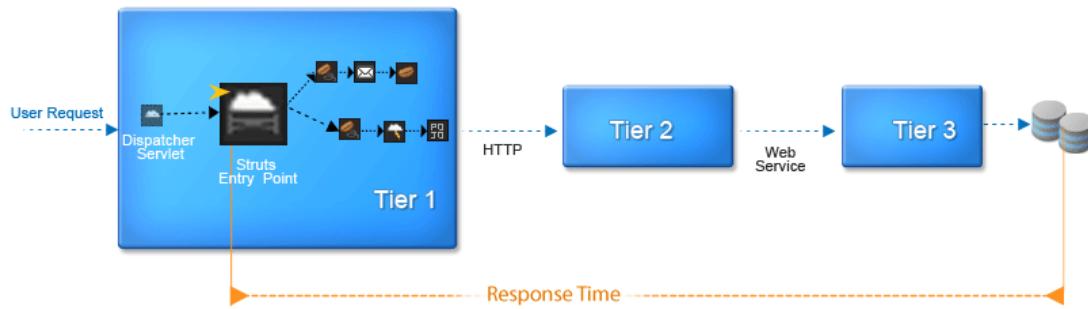
AppDynamics supports following versions of Struts:

- [Struts 1.x](#)
- [Struts 2.x](#)

Struts Action invocations are typically preceded by a dispatcher Servlet, but identification is deferred to the Struts Action. This ensures that the user requests are identified based on the Struts Action and not from the generic URL for Dispatcher Servlet.

The response time for the Struts-based transaction is measured when the Struts entry point is invoked.

The following figure shows the identification process for Struts Action entry points.



Struts Request Names

When a Struts Action is invoked, by default AppDynamics identifies the request using the name of Struts Action and the name of the method. All automatically discovered Struts-based transactions are thus named using the convention <Action Name>.<Method Name>.

For example, if an action called ViewCart is invoked with the SendItems(), the transaction is named "ViewCart.SendItems".

For Struts 1.x the method name is always "execute".

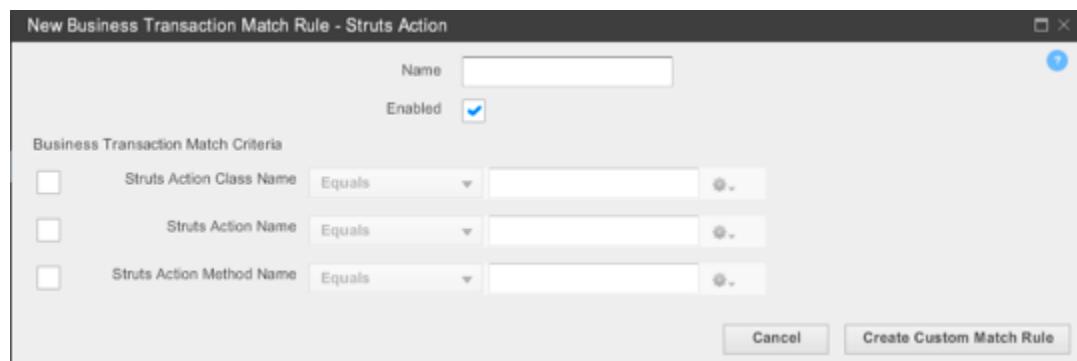
You can rename or exclude auto-discovered transactions. See [Business Transaction List Operations](#).

Custom Match Rules for Struts Transactions

For finer control over the naming of Struts-based transactions, use custom match rules.

A custom match rule lets you specify customized names for your Struts-based requests. You can also group multiple Struts invocations into a single business transaction using custom match rules. See [Custom Match Rules](#) for information about accessing the configuration screens.

The matching criteria for creating the rule are: Struts Action class names, Struts Action names, and Struts Action method names.



Exclude Rules for Struts Actions or Methods

To prevent specific Struts Actions and methods from being monitored add an exclude rule. See [Exclude Rules](#). The criteria for Struts exclude rules are the same as those for custom match rules.

Web Service Entry Points

- Web Services-based Transactions
 - Default Naming
 - Grouping Web Service Actions or Operation Names into a Business Transaction
 - Exclude Rules

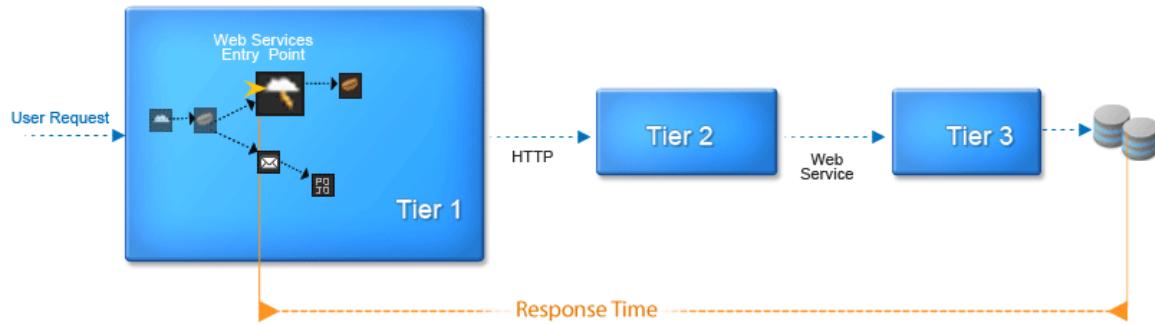
This topic discusses Web Service Entry Points.

Web Services-based Transactions

When your application uses Web Services to service user requests, AppDynamics intercepts the Web Service invocations and names requests based on the Web Service action names and operation name. A Web Service entry point is a Web Service end point that is being invoked.

This is relevant only when the Web Service invocation is part of the entry point tier and not in a downstream tier.

Web Service invocations are usually preceded by a dispatcher Servlet, but identification is deferred to the Web Service endpoints. This configuration ensures that the requests are identified based on the Web Service and not based on the generic URL for the dispatcher Servlet.



Default Naming

When the Web Service end point is invoked, the request is named after the Web Service name and the operation name.

For example, if a service called CartService is invoked with the Checkout operation, the is named "CartService.Checkout".

You can rename or exclude these automatically discovered transactions. See [Business Transaction List Operations](#).

Grouping Web Service Actions or Operation Names into a Business Transaction

If you want finer control over naming of Web Service requests, you can create custom match rules for Web Services. See [Custom Match Rules](#) for information about accessing the configuration screens.

The matching criteria for Web Service rules are Web Service Name and Operation Name

The following example names all operations for the Web Service named "CartService":

New Business Transaction Match Rule - Web Service

Name	CartService	?	
Enabled	<input checked="" type="checkbox"/>		
Business Transaction Match Criteria			
<input checked="" type="checkbox"/>	Web Service Name	Equals	CartService
<input checked="" type="checkbox"/>	Operation Name	Is Not Empty	
<input type="button"/> Cancel <input type="button"/> Create Custom Match Rule			

Exclude Rules

To exclude specific Web Services or operation names from detection, add an exclude rule. See [Exclude Rules](#). The criteria for Web Service exclude rules are the same as those for custom match rules.

POJO Entry Points

- Default Identification of POJO Transactions
- Recommended Practices for Defining a POJO Entry Point
- Custom Match Rules for POJO Transactions
 - POJO Transaction as a Background Task
 - Names for Custom POJO-Based Transactions
- Splitting POJO-Based Transactions
 - To configure transaction splitting
 - Using Method parameter for dynamically naming the transactions
- Exclude Rules for POJO Transactions

Not all business processing can be implemented using Web entry points for popular frameworks. Your application may perform batch processing in all types of containers. You may be using a framework that AppDynamics does not automatically detect. Or maybe you are using pure Java.

In these situations, to enable detection of your business transaction, configure a custom match rule for a POJO (Plain Old Java Object) entry point. The rule should be defined on the class/method that is the most appropriate entry point. Someone who is familiar with your application code should help make this determination. See [Recommended Practices for Defining a POJO Entry Point](#).

AppDynamics measures performance data for POJO transactions as for any other transactions. The response time for the transaction is measured from the POJO entry point, and the remote calls are tracked the same way as remote calls for a Servlet's Service method.

Default Identification of POJO Transactions

By default, the AppDynamics discovery for POJO-based requests captures either a single or a very small group of Servlet URLs. AppDynamics identifies the POJO-based transactions automatically using the class name and method name convention: <Class Name>.<Method Name>. These transactions are displayed on the Business Transactions List.

In certain cases the POJO transactions are not identified separately either because the POJO does not implement a predefined interface or because it does not have a predefined annotation. Auto-discovery for these transactions is not possible.

In some circumstances, AppDynamics' default discovery rules for Servlets take precedence over the POJO-based transaction. For example, the ACME Online application has a Servlet-handled checkout operation with URI pattern: /cart. The Servlet handling the operation reads the HTTP POST parameters to determine if this is a checkout operation. With this type of user request, the Servlet dispatcher parses the request data to determine the requested operation and then dispatches the request to the POJO for handling.

Recommended Practices for Defining a POJO Entry Point

The POJO entry point is the Java method that starts the transaction.

The most important consideration in defining a POJO entry point is to choose a method that begins and ends every time the specific business transaction is invoked.

For example, consider the method execution sequence:

```
com.foo.threadpool.WorkerThread.run()
  calls com.foo.threadpool.WorkerThread.runInternal()
    calls com.foo.Job.run()
```

The first two calls to run() method are the blocking methods that accept a job and invoke it.

The Job.run() method is the actual unit of work, because Job is executed every time the business transaction is invoked and finishes when the business transaction finishes.

Methods like these are the best candidates for POJO entry points.

Custom Match Rules for POJO Transactions

If you are not getting the required visibility with auto-discovered transactions, create a custom match rule for a POJO transaction. See

Custom Match Rules.

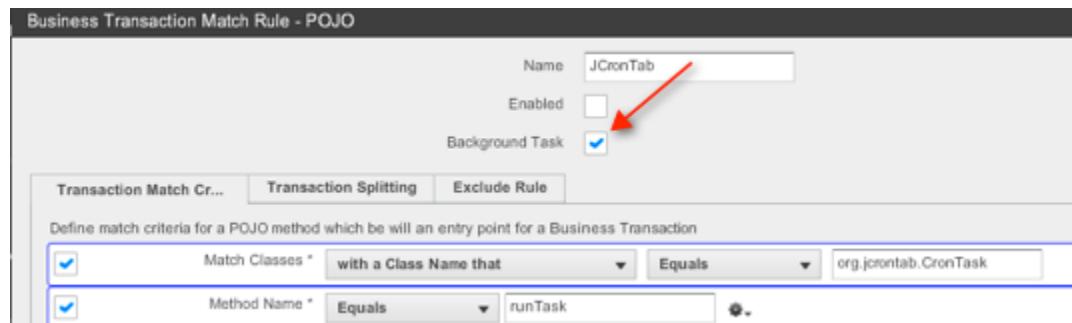
Creating a POJO transaction at a minimum involves setting the entry point.

You may optionally refine the naming of a POJO-based transaction by transaction splitting.

If you are running on IBM JVM v1.5 or v1.6, you must restart the JVM after defining the custom rules.

POJO Transaction as a Background Task

You can specify that a POJO request run as a background task by checking the Background Task check box in the configuration.



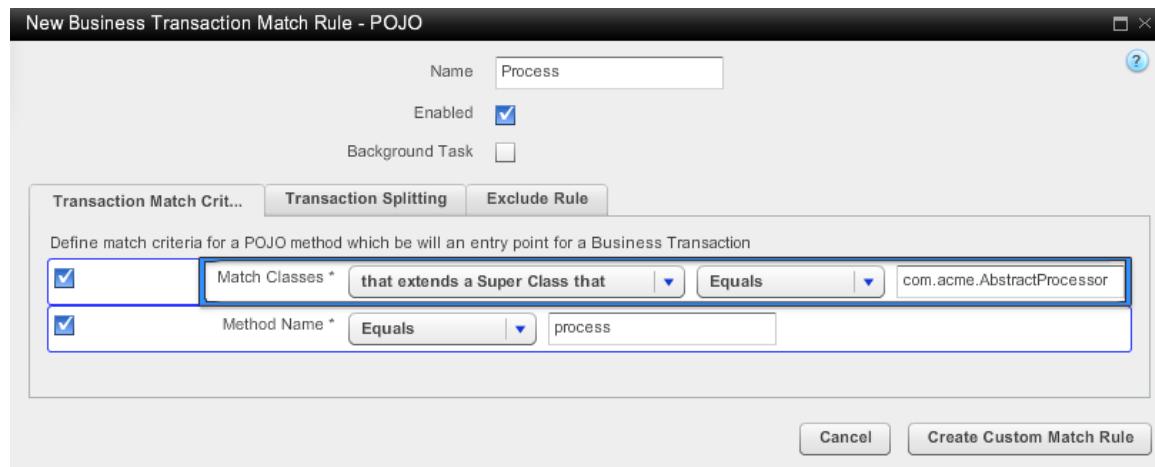
When a request runs as a background task, AppDynamics reports only Business Transaction metrics for the request. It does not aggregate response time and calls metrics at the tier and application levels for background tasks. This ensures that background tasks do not distort the baselines for the business application. Also, you can set a separate set of thresholds for background tasks. See [Background Task Monitoring](#)

Names for Custom POJO-Based Transactions

By default, when you create a custom match rule for a POJO-based transaction, the name of the transaction is based on the name of the custom rule.

For example, consider entry point based on the com.acme.AbstractProcessor superclass, which defines a process() method, which is extended by its child classes: SalesProcessor, InventoryProcessor, BacklogProcessor.

You can define the custom rule on the method defined in the superclass:



Similarly, you can define a custom rule matching on an interface named com.acme.IProcessor, which defines a process() method that is implemented by the SalesProcessor, InventoryProcessor, BacklogProcessor classes.

New Business Transaction Match Rule - POJO

Name	<input type="text"/>	?
Enabled	<input checked="" type="checkbox"/>	
Background Task	<input type="checkbox"/>	
<input type="button" value="Transaction Match Criteria"/> <input type="button" value="Transaction Splitting"/> <input type="button" value="Exclude Rule"/>		
Define match criteria for a POJO method which will be an entry point for a Business Transaction		
<input checked="" type="checkbox"/> Match Classes * <input type="button" value="that implements an Interface which"/> <input type="button" value="Equals"/> <input type="text" value="com.acme.IProcessor"/>		
<input checked="" type="checkbox"/> Method Name * <input type="button" value="Equals"/> <input type="text" value="process"/>		
<input type="button" value="Cancel"/> <input type="button" value="Create Custom Match Rule"/>		

You can also configure a custom rule based on the Annotations.

For example, if all processor classes are annotated with "@com.acme.Processor", a custom rule should be defined using annotation.

New Business Transaction Match Rule - POJO

Name	<input type="text" value="Process"/>	
Enabled	<input checked="" type="checkbox"/>	
Background Task	<input type="checkbox"/>	
<input type="button" value="Transaction Match Criteria"/> <input type="button" value="Transaction Splitting"/> <input type="button" value="Exclude Rule"/>		
Define match criteria for a POJO method which will be an entry point for a Business Transaction		
<input checked="" type="checkbox"/> Match Classes * <input type="button" value="that has an Annotation which"/> <input type="button" value="Equals"/> <input type="text" value="com.acme.Processor"/>		
<input checked="" type="checkbox"/> Method Name * <input type="button" value="Equals"/> <input type="text" value="process"/>		
<input type="button" value="Cancel"/> <input type="button" value="Create Custom Match Rule"/>		

By default, in these cases the business transaction started when any process() is invoked is named Process, based on the name of the custom rule. To refine the transaction name to reflect the specific method called (Process.SalesProcessor, Process.InventoryProcessor, Process.BacklogProcessor) use transaction splitting.

Splitting POJO-Based Transactions

By default, when you create a custom rule for POJO transactions, all the qualifying requests are identified by the name of the custom rule.

However, in many situations it is preferable to split POJO-based transactions, especially for nodes that execute scheduled jobs.

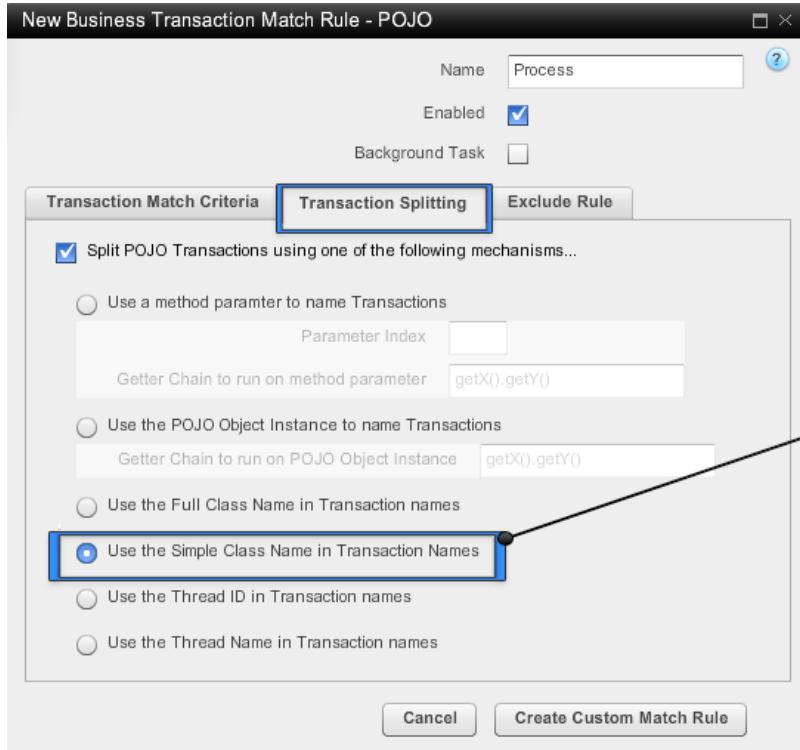
For example, if multiple classes have the same method and are instrumented using the same rule, when the method is invoked the class name of the instance being invoked can be used to classify the request.

If you split the transaction based on the simple class name, instead of one business transaction named Process, the transaction that is started when the process() method is invoked is named based on the rule name combined with the class name: either Process.SalesProcessor, Process.InventoryProcessor, or Process.BacklogProcessor.

In some cases you want to split the transaction based on the value of a parameter in the entry point method. For example, you could configure the split on the following process() method:

```
public void process(String jobType, String otherParameters...)
```

where the jobType parameter could be Sales, Inventory or Backlog.



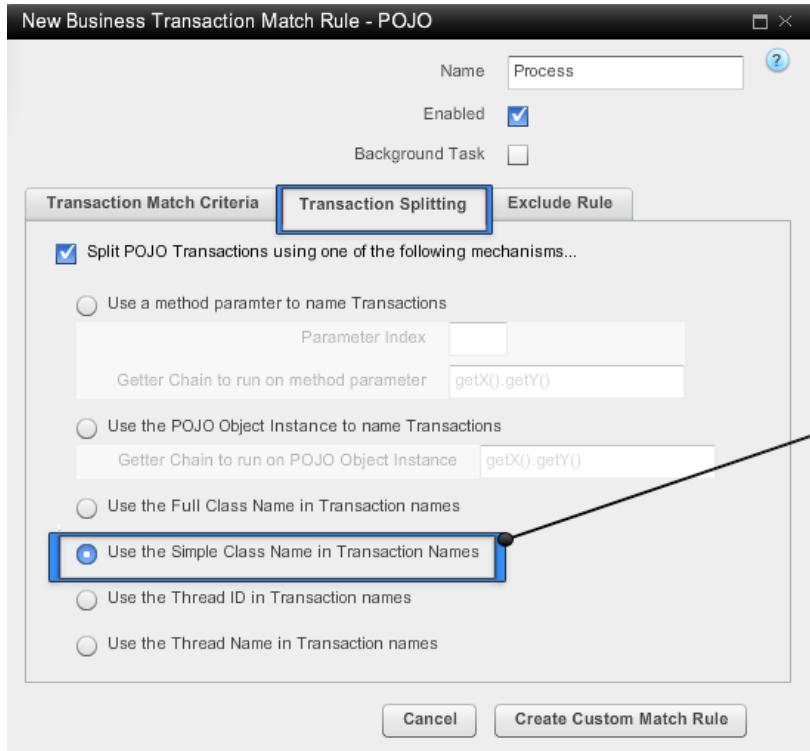
You can name POJO transactions dynamically using the following mechanisms:

- method parameter
- POJO object instance
- fully qualified class name
- simple class name
- thread ID
- thread name

In all cases, the name of the rule is prepended to the dynamically-generated name to form the business transaction name.

To configure transaction splitting

1. In the Transaction Splitting tab of the Business Transaction Match Rule - POJO window, check the Split POJO Transactions box to enable transaction splitting.
2. Select the mechanism to use to split the transaction.
If you are specifying a method parameter, enter the zero-based parameter index of the parameter
If the parameter is a complex type, specify the getter chain to use used to derive the transaction name.
If you are specifying a POJO object instance, specify the getter chain. See [Getter Chains in .NET Configurations](#).
3. Click **Save**.



Using Method parameter for dynamically naming the transactions

Suppose in the ACME Online example, instead of the super-class or interface, the type of the processing is passed in as a parameter.

For example:

```
public void process(String jobType, String otherParameters...)
```

In this case, it would be appropriate to name the transaction based on the value of Job type. This Job type is passed as the parameter. To specify a custom rule for method parameter:

1. Specify the details for the custom rule in the **Transaction Match Criteria** tab.
2. Click the **Transaction Splitting** tab.
3. Check the Split POJO transactions using following mechanisms checkbox.
4. Select the option for method parameter.
5. Specify the details for the parameters.

You can use a getter chain if the parameter is of complex type in order to derive a string value, which can then be used for the transaction name. See [Getter Chains in .NET Configurations](#).

Exclude Rules for POJO Transactions

To prevent configured transaction splitting from being applied in certain situations, create an exclude rule defined on the output of the transaction splitting.

Transaction Match Criteria Transaction Splitting Exclude Rule

The following condition will be applied to the output of the Transaction Splitting configuration. All traffic matching this Exclude Rule will be registered with a single Business Transaction with the same name as this Match Rule (splitting will not happen). This is a way to define a custom rule to split Transactions in all but certain specific conditions.

Exclude Criteria

<input checked="" type="checkbox"/> Transaction Splitting Output	Equals	<input type="text"/>
Equals		
Starts With		
Ends With		
Contains		
Matches Reg Ex		

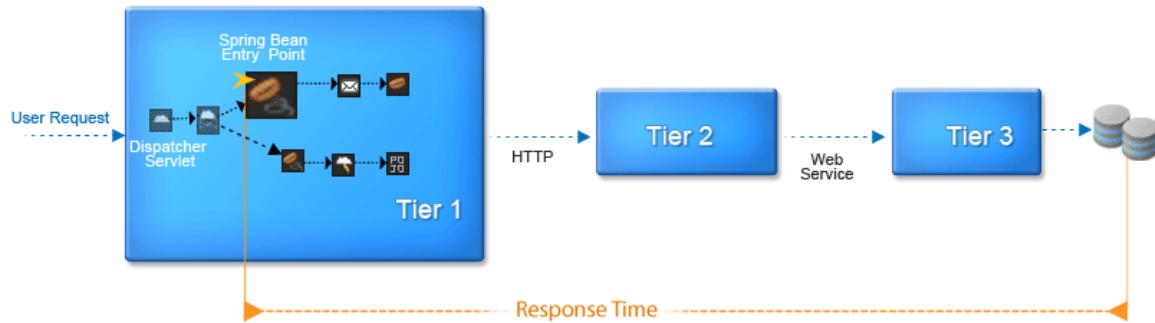
Spring Bean Entry Points

- Spring Bean-based Transactions
- Default Naming for Spring Bean Requests
 - To Enable Auto-discovery for Spring Bean entry points
- Custom Match Rules for Spring Bean Requests
- Exclude Rules Spring Bean Transactions

This topic describes how to configure transaction entry points for Spring Bean requests.

Spring Bean-based Transactions

AppDynamics allows you to configure a transaction entry point for a particular method for a particular bean in your environment. The response time is measured from when the Spring Bean entry point is invoked.



Default Naming for Spring Bean Requests

When the automatic discovery for a Spring Bean based request is turned on, AppDynamics automatically identifies all the Spring Beans based transactions and names these transactions using the following format:

BeanName . MethodName

By default, the transaction discovery for Spring Bean-based requests is turned off.

To Enable Auto-discovery for Spring Bean entry points

1. Access the transaction detection configurations screen and select the tier to configure. See [To Access Business Transaction Detection Configuration](#)
2. In the Spring Bean entry in the Entry Points section check the Automatic Transaction Detection check box.

Entry Points

Type	Transaction Monitoring	Automatic Transaction Detection
Servlet	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Servlet requests <input type="checkbox"/> Enable Servlet Filter Detection
Struts Action	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Struts Action invocations Transactions will be named: ActionName.MethodName
Web Service	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Web Service requests Transactions will be named: ServiceName.OperationName
POJO	<input checked="" type="checkbox"/> Enabled	Any Java method can be the entry point for a Business Transaction. The class to which the method belongs to can be picked using different parameters like its name, its super class name, the interfaces it implements, or the annotations it has.
Spring Bean	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically for all Spring Bean invocations Transactions will be named: BeanName.MethodName
EJB	<input checked="" type="checkbox"/> Enabled	<input type="checkbox"/> Discover Transactions automatically for all EJB invocations Transactions will be named: EJBNName.MethodName

Custom Match Rules for Spring Bean Requests

If you are not getting the required visibility with the auto-discovered transactions, you can create a custom match rule for a Spring Bean based transaction. See [Custom Match Rules](#).

The following example creates a custom match rule for the placeOrder method in the orderManager bean.

New Business Transaction Match Rule - Spring Bean

Name	ACME	?	
Enabled	<input checked="" type="checkbox"/>		
Business Transaction Match Criteria			
<input checked="" type="checkbox"/>	Bean ID	Contains	orderManager
<input checked="" type="checkbox"/>	Method	Equals	placeOrder
<input type="checkbox"/>	Class Name	Equals	
<input type="checkbox"/>	Extends	Equals	
<input type="checkbox"/>	Implements	Equals	

[Cancel](#) [Create Custom Match Rule](#)



This custom rule will name all the qualifying requests as "ACME.orderManager.placeOrder".

Exclude Rules Spring Bean Transactions

To exclude specific Spring Bean transactions from detection add an exclude rule. See [Exclude Rules](#). The criteria for Spring Bean exclude rules are the same as those for custom match rules.

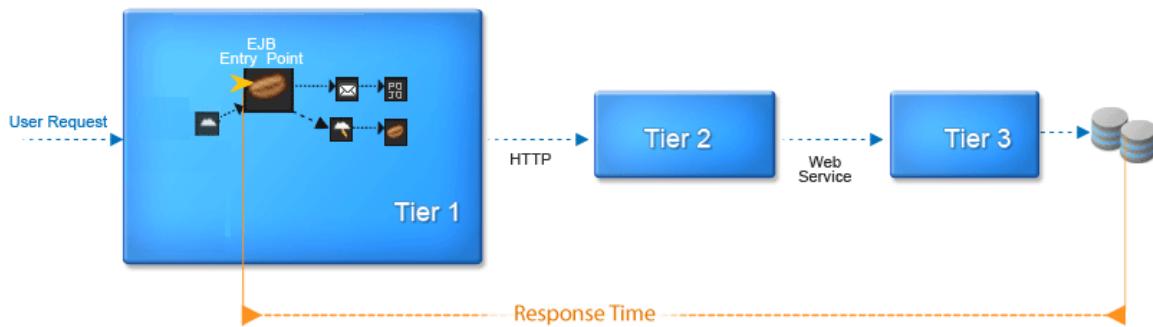
EJB Entry Points

- EJB-Based Business Transactions
- Default Naming for EJB Entry Points
 - To Enable the Auto-discovery for EJB Transactions
- Custom Match Rules for EJB based Transactions
- Exclude Rules for EJB Transactions

This topic describes how to configure transaction entry points for the EJB based requests.

EJB-Based Business Transactions

AppDynamics allows you to configure an EJB-based transaction entry point on either the bean name or method name. The response time for the EJB transaction is measured when the EJB entry point is invoked.



Default Naming for EJB Entry Points

AppDynamics automatically names all the EJB transactions <EJBName>.<MethodName>. By default, automatic transaction discovery for EJB transactions is turned off. To get visibility into these transactions, enable the auto-discovery for EJB based transactions explicitly.

Keep in mind the following before you enable auto-discovery for EJB based transactions:

- If the EJBs use Spring Beans on the front-end, the transaction is discovered at the Spring layer and the response time is measured from the Spring Bean entry point. This is because AppDynamics supports distributed transaction correlation.
- AppDynamics groups all the participating EJB-based transactions (with remote calls) in the same business transaction. However, if your EJBs are invoked from a remote client where the App Server Agent is not deployed, these EJBs are discovered as new business transactions.

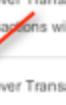
To Enable the Auto-discovery for EJB Transactions

1. Access the transaction detection configurations screen and select the tier to configure. See [To Access Business Transaction Detection Configuration](#).
2. In the EJB entry in the Entry Points section check the Automatic Transaction Detection check box.

Java - Transaction Detection .NET - Transaction Detection

▼ Entry Points

Type	Transaction Monitoring	Description
Servlet	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically <input type="checkbox"/> Enable Servlet Filter Detection
Struts Action	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically Transactions will be named: ActionName
Web Service	<input checked="" type="checkbox"/> Enabled	<input checked="" type="checkbox"/> Discover Transactions automatically Transactions will be named: ServiceName
POJO	<input checked="" type="checkbox"/> Enabled	Any Java method can be the entry point for a transaction. The class to which the method belongs will be picked using different parameters like name, the interfaces it implements, or the annotations it has.
Spring Bean	<input checked="" type="checkbox"/> Enabled	<input type="checkbox"/> Discover Transactions automatically Transactions will be named: BeanName
EJB	<input checked="" type="checkbox"/> Enabled	<input type="checkbox"/> Discover Transactions automatically Transactions will be named: EJBNName



Custom Match Rules for EJB based Transactions

If you are not getting the required visibility with auto-discovered transactions, create a custom match rule for a EJB based transaction. See [Custom Match Rules](#).

The following example creates a custom match rule for the receiveOrder method in the TrackOrder bean. The transactions are named "ACME_EJB.TrackOrder.receiveOrder".

New Business Transaction Match Rule - EJB

Name	ACME_EJB	<input type="button" value="?"/>		
Enabled	<input checked="" type="checkbox"/>			
Business Transaction Match Criteria				
<input checked="" type="checkbox"/>	EJB Name	Equals	TrackOrder	<input type="button" value="..."/>
<input checked="" type="checkbox"/>	Method	Equals	receiveOrder	<input type="button" value="..."/>
<input type="checkbox"/>	EJB Type	Message Driven	<input type="button" value="..."/>	
<input type="checkbox"/>	Class Name	Equals	<input type="button" value="..."/>	
<input type="checkbox"/>	Extends	Equals	<input type="button" value="..."/>	
<input type="checkbox"/>	Implements	Equals	<input type="button" value="..."/>	
<input type="button" value="<"/> <input type="button" value=">"/>				
<input type="button" value="Cancel"/> <input type="button" value="Create Custom Match Rule"/>				

In addition to the bean and method names, other match criteria that could be used to define the transaction are the EJB type, class name, superclass name and interface name.

Exclude Rules for EJB Transactions

To exclude specific EJB transactions from detection add an exclude rule. See [Exclude Rules](#). The criteria for EJB exclude rules are the same as those for custom match rules.

POCO Entry Points

- Default Discovery of POCO Transactions
- Defining a POCO Entry Point
 - Example POCO Entry Point
 - To specify a POCO entry point
 - POCO Transaction as a Background Task
- Further Refining POCO-Based Transactions using "Splitting"
 - To configure transaction splitting
 - To configure exclude rules

Some business processing is not implemented using common or popular frameworks. Your application may perform processing in all types of containers. It may be using a framework that AppDynamics does not automatically detect. You can specify entry points using custom match rules for POCOs (Plain Old C++ Objects). Once defined, AppDynamics measures performance data for POCO transactions as for any other transactions.

Define the custom match rule on the class/method that is the most appropriate entry point for the transaction. Someone who is familiar with your application code should help make this determination.

Default Discovery of POCO Transactions

By default, the AppDynamics discovery mechanism for POCO-based requests captures ... and displays them in the Business Transactions List. AppDynamics identifies the POCO-based transactions using the class name and method name convention:

```
<Class Name>. <Method Name>.
```

When AppDynamics does not automatically discover a POCO transactions, it may be because....?... the POCO does not implement a predefined interface or because it does not have a predefined annotation...?

AppDynamics' default discovery rules for.... takes precedence over the POCO-based transaction discovery...?

Defining a POCO Entry Point

The POCO entry point is the method that starts the transaction. Choose a method that begins and ends every time the specific business transaction is invoked. You use custom match rules to enable AppDynamics to discover the transaction.

You can refine the naming of a POJO-based transaction by transaction splitting.

Example POCO Entry Point

For example, consider the following method execution sequence: REDO CODE IF APPLICABLE

```
REDO CODE IF APPLICABLE

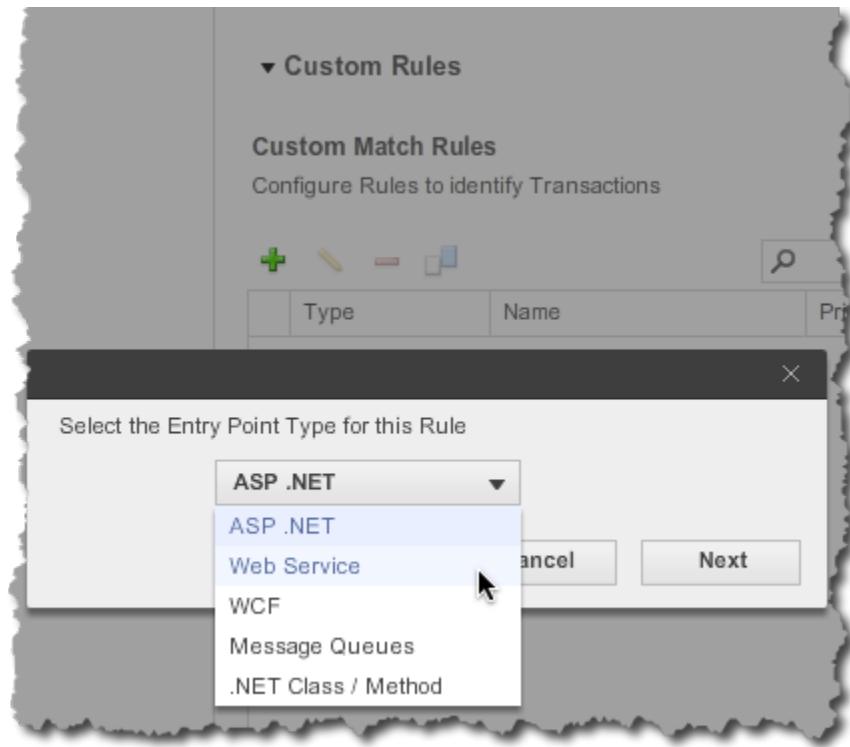
com.foo.threadpool.WorkerThread.run()
    calls com.foo.threadpool.WorkerThread.runInternal()
        calls com.foo.Job.run()
```

The first two calls to run() method are the blocking methods that accept a job and invoke it.

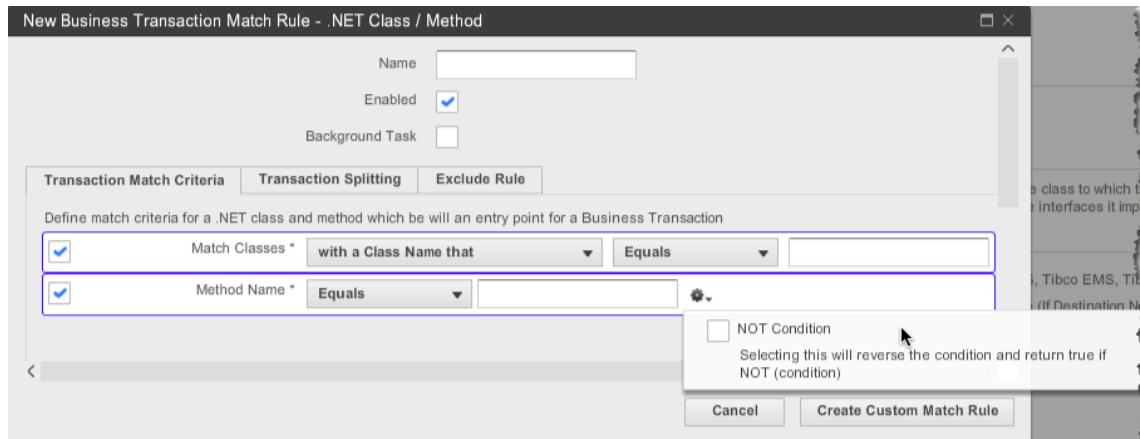
The Job.run() method is the actual unit of work, because Job is executed every time the business transaction is invoked and finishes when the business transaction finishes.

To specify a POCO entry point

1. Click **Configure -> Instrumentation -> Transaction Detection**.
2. Click the **.NET - Transaction Detection** tab.
2. From the application and tier list at the left, select either:
 - an application, to configure at the level of the entire business application.
 - a tier, to configure at the tier level. At the tier level click **Use Custom Configuration for this Tier**. AppDynamics copies the application configuration to the tier level so that you can modify it for the tier.
3. In the Custom Rules panel under Custom Match Rules, click **Add** (the plus sign).



4. From the dropdown menu, select ***.NET Class/Method]**.



5. Name the match rule.

- By default the rule is Enabled; you can disable it later if needed
- By default it is not a background task; see [POCO Transaction as a Background Task](#).

6. In the Transaction Match Criteria, use the dropdowns to specify the class name:

- The type of class
- The match condition
- A string

7. If also needed, use the dropdowns to specify the method name:

- The match condition
- A string

X. If configuring an application level, click **Configure all Tiers to use this Configuration**.

POCO Transaction as a Background Task

You can indicate that a POCO transaction runs as a background task by checking the Background Task check box in the Match Rule window.

When a request runs as a background task, AppDynamics reports only Business Transaction metrics for the request. It does not aggregate response time and calls metrics at the tier and application levels for background tasks. This ensures that background tasks do not distort the baselines for the business application. Also, you can set a separate set of thresholds for background tasks.

Further Refining POCO-Based Transactions using "Splitting"

When you create a custom rule for POCO transactions all of the qualifying transactions follow the custom rule. In some situations you may need to further refine the discovery rules. For example:

- When you need to identify a transaction based on the value of a parameter in the entry point method.
- When multiple classes have the same method and are instrumented using the same rule, when the method is invoked the class name of the instance being invoked can be used to classify the request.

This process is called "transaction splitting".

In addition, if there are some transactions that result from the custom match and splitting rules that you want to ignore, you can specify custom exclude rules.

To configure transaction splitting

New Business Transaction Match Rule - .NET Class / Method

Enabled

Background Task

Transaction Match Criteria **Transaction Splitting** **Exclude Rule**

Split .NET class / Method Transactions using one of the following mechanisms...

Use a method parameter to name Transactions
Parameter Index
Property or Field chain to run on method parameter getX().getY()

Use the .NET class / Method Object Instance to name Transactions
Getter Chain to run on .NET class / Method Object Instance getX().getY()

Use the Full Class Name in Transaction names

Use the Simple Class Name in Transaction Names

Use the Thread ID in Transaction names

Use the Thread Name in Transaction names

< Cancel

To configure exclude rules

New Business Transaction Match Rule - .NET Class / Method

Name	<input type="text"/>
Enabled	<input checked="" type="checkbox"/>
Background Task	<input type="checkbox"/>

Transaction Match Criteria Transaction Splitting Exclude Rule

The following condition will be applied to the output of the Transaction Splitting configuration. All traffic matching this Exclude Rule will be registered with a single Business Transaction with the same name as this Match Rule (splitting will not happen). This is a way to define a custom rule to split Transactions in all but certain specific conditions.

Exclude Criteria

Transaction Splitting Output Equals

Equals
Starts With Ends With
Contains
Matches Reg Ex

Cancel **Create**

Getter Chains in Java Configurations

- Separators in Getter Chains
- Getter Chain Examples
- Curly Braces Enclosing Getter Chains
- Learn More

This topic provides some guidance and examples of the correct syntax for getter chains in AppDynamics configurations.

Sepators in Getter Chains

The following special characters are used as separators:

- comma (,) for separating parameters
- forward slash (/) for separating a type declaration from a value in a parameter
- Dot (.) for separating the methods and properties in the getter chain

If a slash or a comma character is used in a string parameter, use the backslash (\) escape character.

If a literal dot is used in a string parameter, use the backslash escape character before the dot.

Getter Chain Examples

- Getter chain with integer parameters in the substring method using the forward slash as the type separator:

```
getAddress(appdynamics, sf).substring(int/0, int/10)
```

- Getter chain with various non-string parameter types:

```
getAddress(appdynamics, sf).myMethod(float/0.2, boolean/true, boolean/false, int/5)
```

- Getter chain with forward slash escaped; escape character needed here for the string parameter:

```
getUrl().split(/\)/ # node slash is escaped by a backward slash
```

- Getter chain with an array element:

```
getUrl().split(/\)/.[4]
```

- Getter chain with multiple array elements separated by commas:

```
getUrl().split(/\)/.[1,3]
```

- Getter chain using backslash to escape the dot in the string parameter;
the call is getParam (a.b.c).

```
getAddress.getParam(a\.\b\.\c\.)
```

- In the following getter chain, the first dot requires an escape character because it is in a string method parameter (inside the parentheses). The second dot does not require an escape character because it is not in a method parameter (it is outside the parentheses).

```
getName(suze\.smith)getClass().getSimpleName()
```

- The following getter chain is from a transaction splitting rule on URIs that use a semicolon as a delimiter; for example:
`/my-webapp/xyz;jsessionid=BE7F31CC0235C796BF8C6DF3766A1D00?act=Add&uid=c42ab7ad-48a7-4353-bb11-0dfcab`

The getter chain splits on the API name, so the resulting split transactions are "API.abc", API."xyz" and so on.
The call gets the URL using getRequestURI() and then splits it using the escaped forward slash. From the resulting array it takes the third entry (as the split treats the first slash as a separator) and inserts what before the slash (in this case, nothing) into the first entry. Then it splits this result using the semicolon, getting the first entry of the resulting array, which in this case contains the API name.

```
getRequestURI().split(/\)/.[2].split(;)![0]
```

Curly Braces Enclosing Getter Chains

In most cases curly braces are not used to enclose getter chains in AppDynamics configurations. An exception is the use of a getter chain in a custom expression on the HttpServletRequest object.

Custom expressions on the HTTP request are configurable in the Java Servlet Transaction Naming Configuration window and in the Split Transactions Using Request Data tab of the servlet custom match and exclude rules. In these cases, curly braces are required to delineate the boundaries of the getter chains.

Business Transaction Match Rule - Servlet

Name: API	Enabled: <input type="checkbox"/>	Priority: 10
<input checked="" type="radio"/> Transaction Match Criteria <input type="radio"/> Split Transactions Using Request Data <input type="radio"/> Split Transactions Using Payload		
<input type="radio"/> Use the first <input type="text"/> segments in Transaction names <input type="radio"/> Use the last <input type="text"/> segments in Transaction names <input type="radio"/> Use URI segment(s) in Transaction names Segment Numbers <input type="text"/> <i>Enter a comma separated list of parameter numbers (e.g. 1,3,4)</i> <input type="radio"/> Use a parameter value in Transaction names Parameter Name <input type="text"/> <input type="radio"/> Use a header value in Transaction names Header Name <input type="text"/> <input type="radio"/> Use a cookie value in Transaction names Cookie Name <input type="text"/> <input type="radio"/> Use a session attribute value in Transaction names Session Attribute Key <input type="text"/> <input type="radio"/> Use the request method (GET/POST/PUT) in Transaction names <input type="radio"/> Use the request host in Transaction names <input type="radio"/> Use the request originating address in Transaction names <input checked="" type="radio"/> Apply a custom expression on HttpServletRequest and use the result in Transaction Names Explain This <code> \${getRequestURI().split('/')[2].split('.')[0]} </code>		
<input type="button" value="Cancel"/> <input type="button" value="Save"/>		

Curly braces required here.

Getter chains in custom expressions on the HTTP request in diagnostic data collector should also be enclosed in curly braces:

Transaction Detection Backend Detection Error Detection **Diagnostic Data Collectors** Call Graph Settings >>

Create HTTP Request Gatherer

Specify the names of the parameter/cookie values to be collected. The value will be displayed in the Transaction Snapshot against the display name chosen here.

Method Invocation

Any method invocation or part of a method invocation of a method might tell you something about the method invocation data.

Name: user-agent Apply to new Business Transactions

HTTP Parameters

Display Name	HTTP Parameter Name

HTTP Request Attributes

URL
 Session ID
 User Principal
 Get User Principal by `httpServletRequest.getUserPrincipal().toString()`.
 Get User Principal by evaluating a custom expression on the `HttpServletRequest`:
 `${getHeader(user-agent)}`
Enter a custom expression to be applied on the `HttpServletRequest` object.

Add **Curly braces required here.**

Learn More

- Configure Business Transaction Detection
- Configure Data Collectors

Troubleshoot Java Application Problems

Detect Code Deadlocks (Java)

- Code Deadlocks and their Causes
 - Detecting deadlocks using the AppDynamics UI
 - To Examine a Code Deadlock

- Detecting deadlocks using the REST API
- Learn More

This topic describes how to use AppDynamics to detect code deadlocks.

Code Deadlocks and their Causes



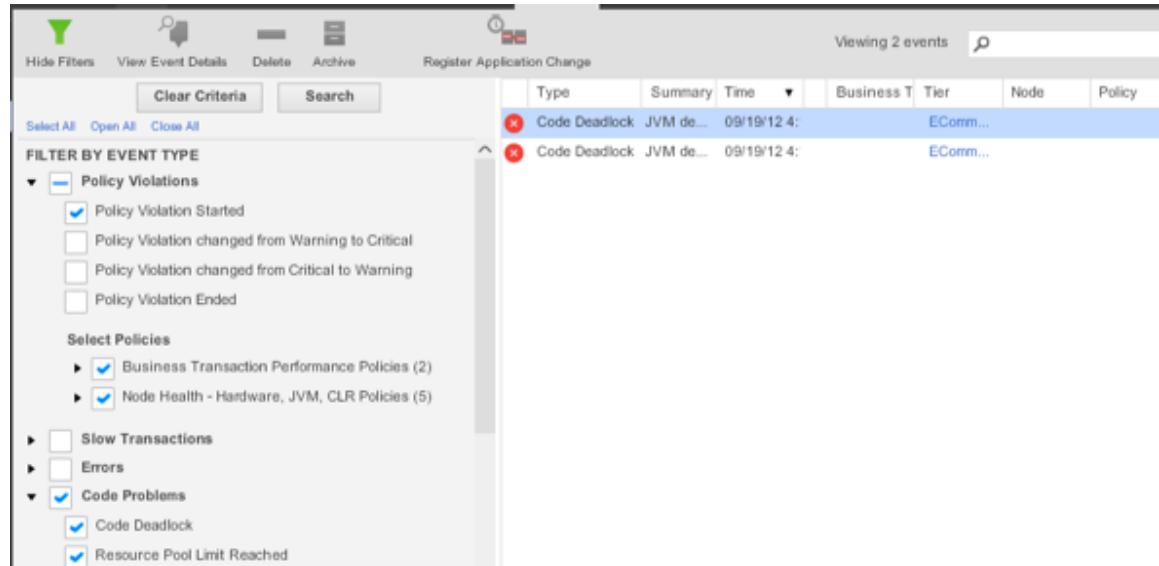
AppMan says: **Read a real-life story about how AppDynamics helped identify code deadlocks and reduce the risk to revenue!**

In a multi-threading development environment, it is common to use more than a single lock. However sometimes deadlocks will occur. Here are some possible causes:

- The order of the locks is not optimal
- The context in which they are being called (for example, from within a callback) is not correct
- Two threads may wait for each other to signal an event

Detecting deadlocks using the AppDynamics UI

If Code Deadlock is checked in the Event configuration for an application, AppDynamics reports code deadlocks in the Events list. See [Filter and Analyze Events](#). The following list shows two deadlocks in the ECommerce tier.



The screenshot shows the AppDynamics Events list interface. At the top, there are buttons for Hide Filters, View Event Details, Delete, Archive, and Register Application Change. A search bar indicates "Viewing 2 events". Below the header, there are filters for "Clear Criteria" and "Search". The main area displays a table of events with columns: Type, Summary, Time, Business T, Tier, Node, and Policy. Two rows are listed, both labeled "Code Deadlock" with the summary "JVM de..." and time "09/19/12 4:...". The "Business T" column shows "EComm..." and the "Tier" column shows "EComm...". On the left side, there is a sidebar titled "FILTER BY EVENT TYPE" with sections for "Policy Violations" (selected), "Select Policies" (Business Transaction Performance Policies and Node Health - Hardware, JVM, CLR Policies are checked), and "Code Problems" (Slow Transactions, Errors, and Code Problems are listed, with Code Deadlock checked). There are also sections for "Slow Transactions" and "Errors".

To Examine a Code Deadlock

1. Double-click the deadlock event in the events list.
The Code Deadlock **Summary** tab displays.

The screenshot shows a summary of a deadlock event. The summary includes:

- Severity:** Critical
- Type:** Code Deadlock
- Time:** 09/19/12 4:16:49 PM
- Summary:** JVM deadlock detected
- Tier:** ECommerce Server
- Node:** Node_8000

2. To see details about the deadlock click the **Details** tab and scroll down.

The screenshot shows the detailed stack traces for three threads (t1, t2, t3) involved in the deadlock:

- t1:** Name[t1]Thread ID[968]
Deadlocked on Lock[java.lang.String@52be1266] held by thread [t2] Thread ID[969]
Thread stack [
com.appdynamics.DeadlockingThread.g(DeadlockingThread.java:34)
com.appdynamics.DeadlockingThread.f(DeadlockingThread.java:28)
com.appdynamics.DeadlockingThread.run(DeadlockingThread.java:20)
]
- t2:** Name[t2]Thread ID[969]
Deadlocked on Lock[java.lang.String@4140ac33] held by thread [t3] Thread ID[970]
Thread stack [
com.appdynamics.DeadlockingThread.g(DeadlockingThread.java:34)
com.appdynamics.DeadlockingThread.f(DeadlockingThread.java:28)
com.appdynamics.DeadlockingThread.run(DeadlockingThread.java:20)
]
- t3:** Name[t3]Thread ID[970]
Deadlocked on Lock[java.lang.String@4d17a75f] held by thread [t1] Thread ID[968]
Thread stack [
com.appdynamics.DeadlockingThread.g(DeadlockingThread.java:34)
com.appdynamics.DeadlockingThread.f(DeadlockingThread.java:28)
com.appdynamics.DeadlockingThread.run(DeadlockingThread.java:20)
]

Detecting deadlocks using the REST API

You can detect a DEADLOCK event-type using the AppDynamics REST API. For details see the example [Retrieve event data](#).

Learn More

- Use the AppDynamics REST API

Troubleshoot Java Memory Issues

Troubleshoot Java Memory Leaks

- Memory Leaks in a Java Environment
- AppDynamics Automatic Java Leak Detection
 - Prerequisites
 - Monitoring Memory for Potential JVM Leaks
- Enabling Memory Leak Detection
- Conditions for Troubleshooting Java Memory Leaks
- Workflow to troubleshoot memory leaks
- Troubleshooting Memory Leaks
 - Select the Collection Object that you want to monitor
 - Use Content Inspection
 - Use Access Tracking
- Learn More

Memory Leaks in a Java Environment

While the JVM's garbage collection greatly reduces the opportunities for memory leaks to be introduced into a codebase, it does not eliminate them completely. For example, consider a web page whose code adds the current user object to a static set. In this case, the

size of the set grows over time and could eventually use up significant amounts of memory. In general, leaks occur when an application code puts objects in a static Collection and does not remove them even when they are no longer needed.

In high workload production environments if the Collection is frequently updated, it may cause the applications to crash due to insufficient memory. It could also result in system performance degradation as the operating system starts paging memory to disk.

AppDynamics Automatic Java Leak Detection

AppDynamics automatically tracks every Java Collection (for example, HashMap and ArrayList) that meets a set of criteria defined below. The Collection size is tracked and a linear regression model identifies whether the Collection is potentially leaking. You can then identify the root cause of the leak by tracking frequent accesses of the Collection over a period of time.

You can also monitor memory leaks for custom memory structures. Typically custom memory structures are used as caching solutions. In a distributed environment, caching can easily become a prime source of memory leaks. It is therefore important to manage and track memory statistics for these memory structures. To do this, you must first configure custom memory structures. For details, see [Configure Custom Memory Structures \(Java\)](#).

Prerequisites

For supported JVMs see [Memory Monitoring Compatibility in Java Environments](#).

Monitoring Memory for Potential JVM Leaks

Use the Node Dashboard to identify the memory leak. A possible memory leak is indicated by a growing trend in the heap as well as the old/tenured generation memory pool.

An object is automatically marked as a potentially leaking object when it shows a positive and steep growth slope. The Memory Leak Dashboard shows:

- **Collection Size:** The number of elements in a Collection.
- **Potentially Leaking:** Potentially leaking Collections are marked as red. AppDynamics recommends that you start diagnostic sessions on potentially leaking objects.
- **Status:** Indicates if a diagnostic session has been started on an object.
- **Collection Size Trend:** A positive and steep growth slope indicates potential memory leak.



Tip: To identify long-lived Collections compare the **JVM start time** and **Object Creation Time**.

If you are not able to see any captured Collections, ensure that you have correct configuration for detecting potential memory leaks.

Enabling Memory Leak Detection

Before enabling memory leak detection, identify the potential JVMs that may have a leak.

Leak Detection mode tracks all frequently used Collections; therefore, using this mode results in a higher overhead. Turn on leak detection mode only when a memory leak problem is identified. Turn this mode off after you have identified and resolved the leak.

To achieve optimum performance, start diagnosis on an individual Collection at a time.

Ensure that you identify the potentially leaking JVMs before enabling memory leak mode.

Leak Detection mode tracks all frequently used collections; therefore, using this mode results in a higher overhead. Turn on leak detection mode only when a memory leak problem is identified. Turn this mode off after the leak has been identified and resolved.

To achieve optimum performance, start diagnosis on one individual Collection at a time.

Conditions for Troubleshooting Java Memory Leaks

For a Collection object to be identified and monitored, it must meet the following conditions:

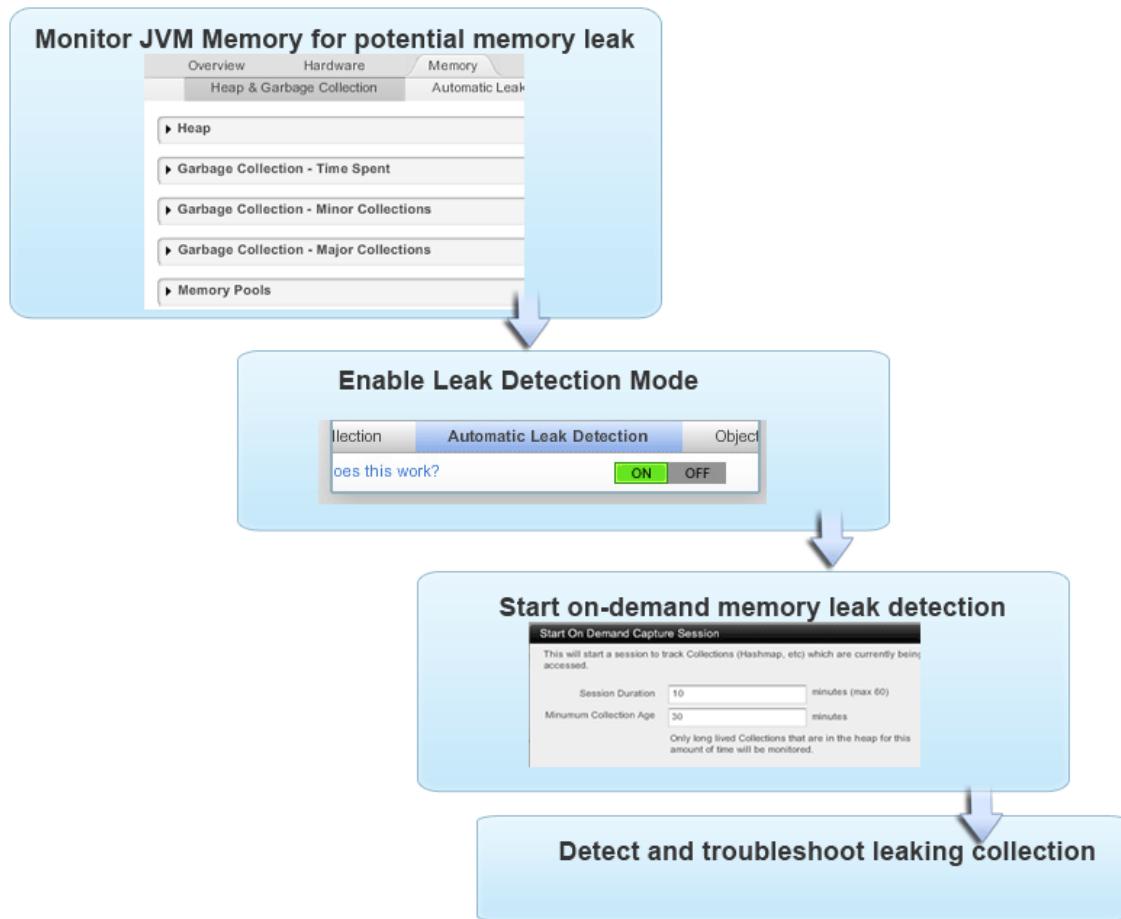
- The Collection has been alive for at least N minutes. Default is 30 minutes, configurable with the minimum-age-for-evaluation-in-minutes node property.
- The Collection has at least N elements. Default is 1000 elements, configurable with the minimum-number-of-elements-in-collection-to-deep-size node property.
- The Collection Deep Size is at least N MB. Default is 5 MB, configurable with the minimum-size-for-evaluation-in-node property.

The Deep Size is calculated by traversing recursive object graphs of all the objects in the Collection.

See [App Agent Node Properties](#) and [App Agent Node Properties Reference](#).

Workflow to troubleshoot memory leaks

Use the following workflow to troubleshoot memory leaks on JVMs that have been identified with a potential memory leak problem:



Troubleshooting Memory Leaks

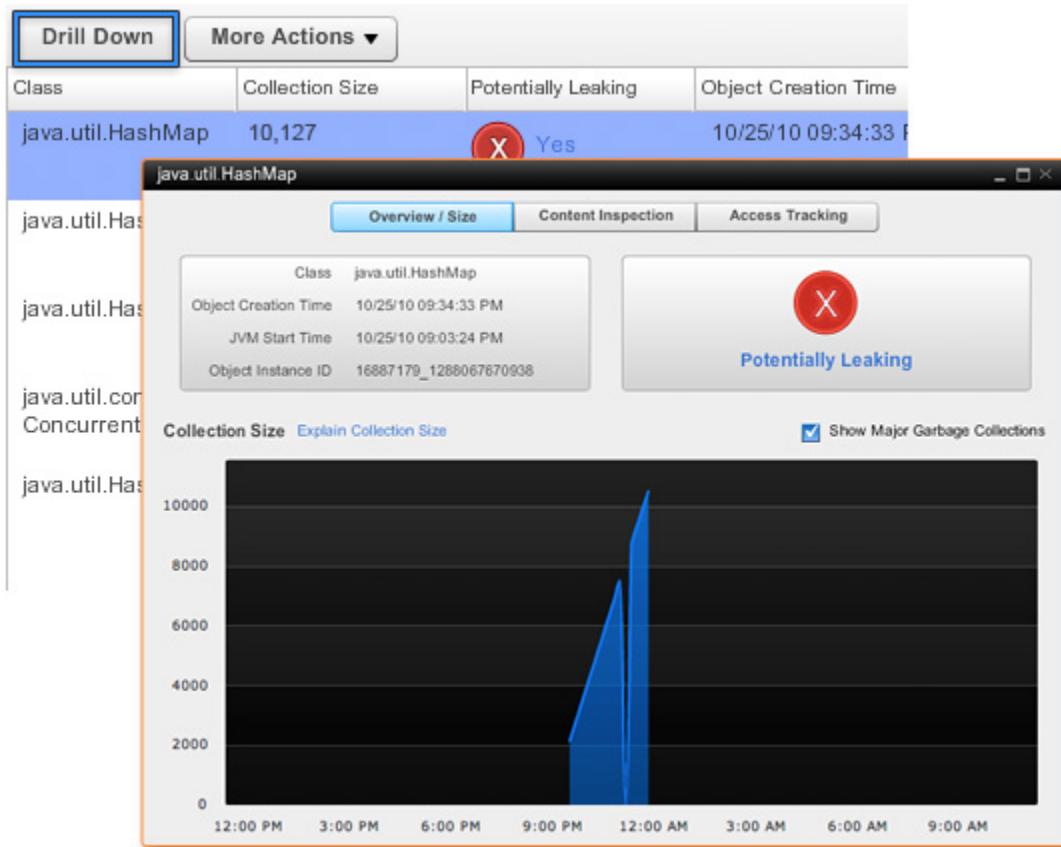
Troubleshooting a memory leak involves performing the following three actions:

- Select the Collection Object that you want to monitor
- Use Content Inspection
- Use Access Tracking

Select the Collection Object that you want to monitor

- Select the class name that you want to monitor.
 - Click on the "Drill Down" button provided on the top of the memory leak dashboard.
- Alternatively right click on the class name and select the "Drill Down" option.

⚠️ IMPORTANT: Ensure that all the instructions from the "Before you get started" section have been followed. To achieve optimum performance, start the troubleshooting session on a single Collection Object at a time.



Use Content Inspection

Use Content Inspection to identify which part of the application the Collection belongs to so that you can start troubleshooting. It allows monitoring histograms of all the elements in a particular Collection. Start diagnostic session on the object and then follow the steps listed below:

1. Select the **Content Inspection** tab.
2. Click **Start Content Summary Capture Session** to start the content inspection session.
3. Enter the session duration. Allow at least 1-2 minutes for data generation.
4. Click the refresh button to retrieve the session data.
5. Click on the snapshot to view details about an individual session.

1 Select the "Content Inspection"

START

Overview / Size Content Inspection Access Tracking

Time

10/12/10 03:16:25 PM
10/12/10 03:17:24 PM
10/12/10 03:22:24 PM

Content Summary Time: 10/12/10 03:22:24

Classname	Coun
java.lang.String	2650
com.appdynamics.qa.model.Address	3300
com.appdynamics.qa.model.Order	1650
java.lang.Object[]	1
java.util.HashMap\$Entry	1466
	2
java.util.HashMap	2
java.util.ArrayList	1

Refresh
Query for the latest available Content Summaries

Start Content Summary Capture Session

SESSION IN PROGRESS
Start a session to capture the summary of the contents of this Collection

Dump Contents to Disk
Dump the full contents of this Collection to the AppDynamics App Server

Start Content Summary Capture Session
This will start a session to capture a summary of the contents of this Collection. The Heap by this object.

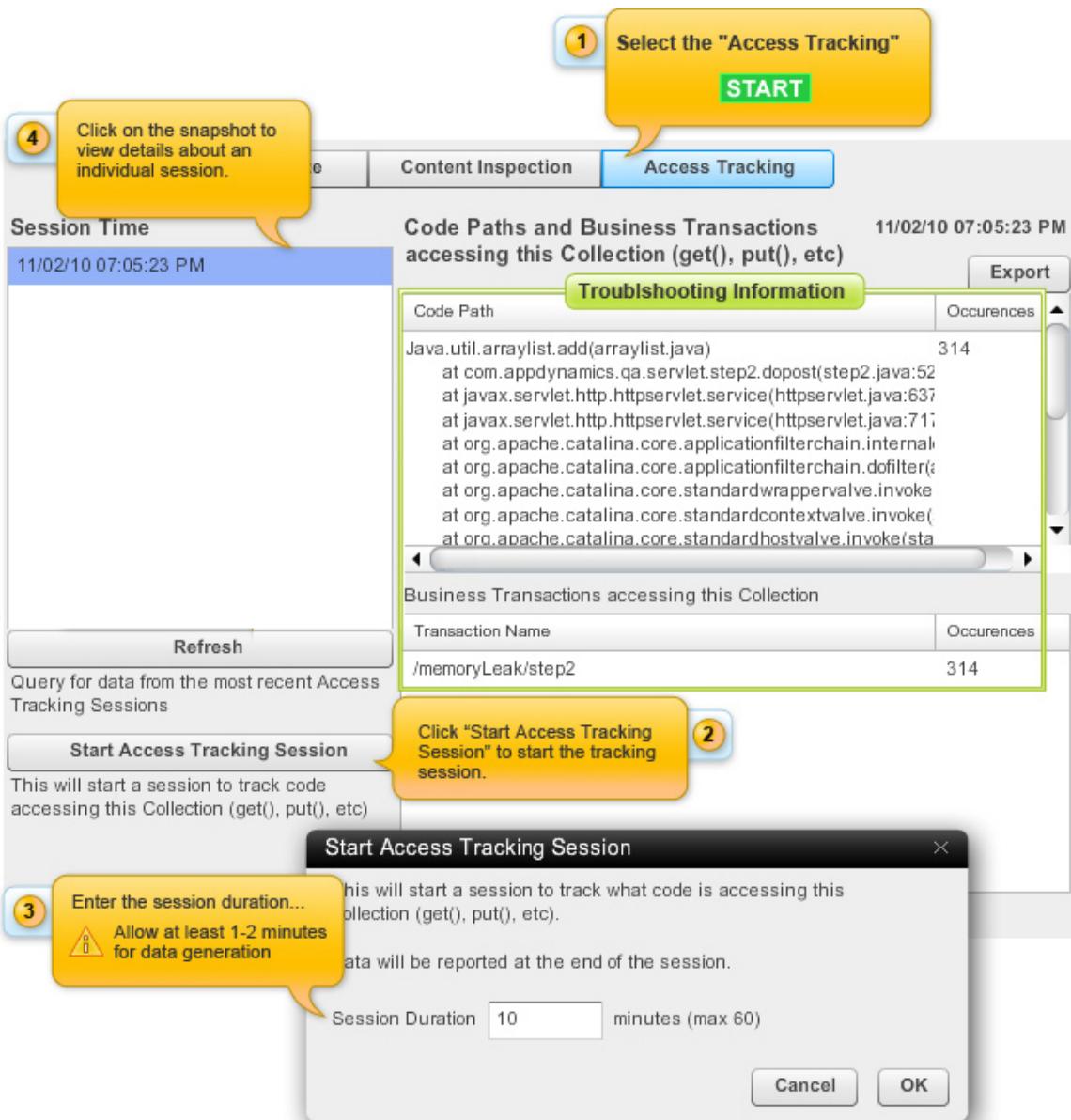
Enter the session duration...
Allow at least 1-2 minutes for data generation

Session Duration minutes (max)

Use Access Tracking

Use Access Tracking to view the actual code paths and business transactions accessing the Collections object. Start diagnostic session on the object and then follow the steps listed below:

1. Select the **Access Tracking** tab
 2. Click **Start Access Tracking Session** to start the tracking session.
 3. Enter the session duration. Allow at least 1-2 minutes for data generation.
 4. Click the refresh button to retrieve session data.
 5. Click on the snapshot to view details about an individual session.
- Note:** You can also export the troubleshooting information into excel files using the "Export" button on the right side.



Learn More

- App Agent Node Properties

Troubleshoot Java Memory Thrash

- Memory Thrash and Object Instance Tracking
- Workflow for Detecting and Troubleshooting Memory Thrash
- Isolating and Analyzing Memory Thrash
 - To analyze memory thrash problems
 - To verify memory thrash
- Using Allocation Tracking
 - To use allocation tracking:

Memory Thrash and Object Instance Tracking

Memory thrash is caused when a large number of temporary objects are created in very short intervals. Although these objects are temporary and are eventually cleaned up, the garbage collection mechanism struggles to keep up with the rate of object creation. Use object instance tracking to isolate the root cause of the memory thrash. To configure and enable object instance tracking, see [Configure](#)

Object Instance Tracking (Java).

AppDynamics automatically tracks the following classes:

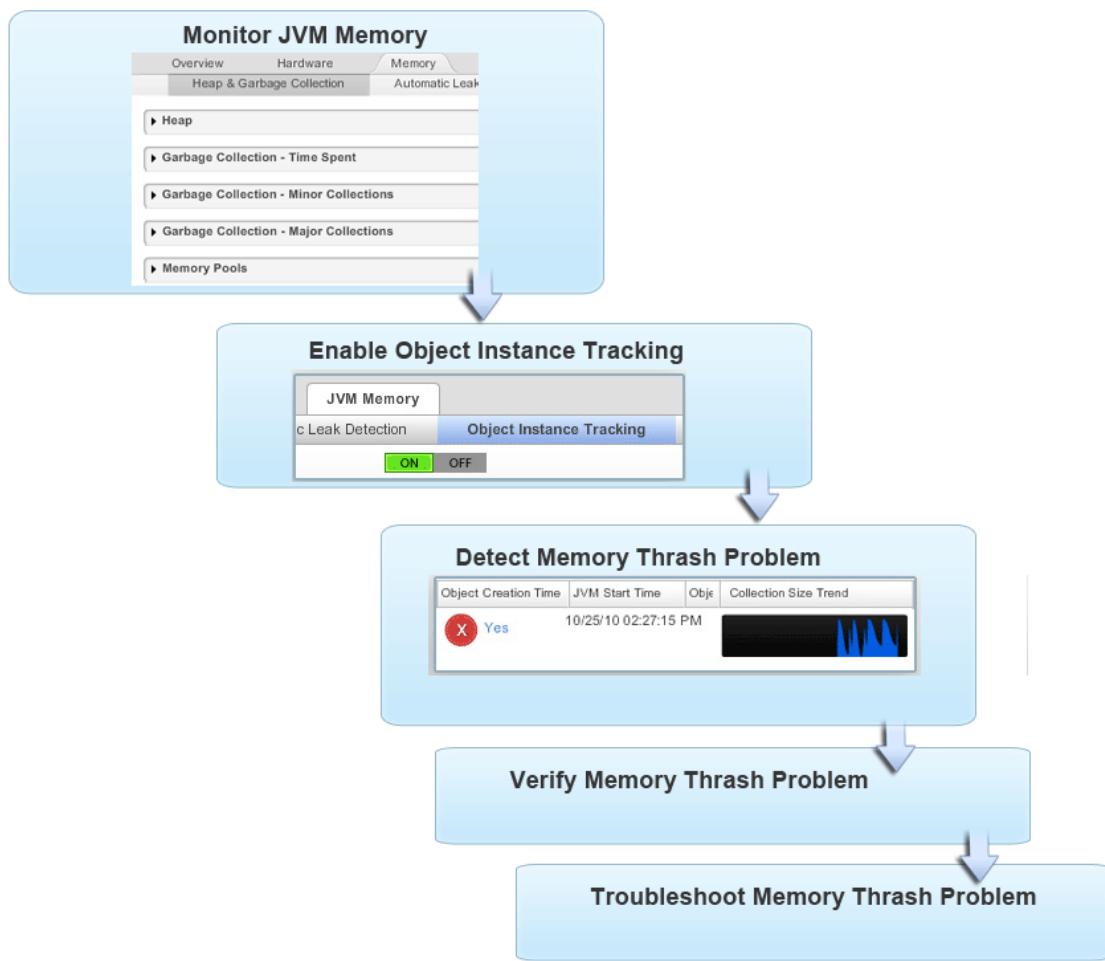
- Application Classes
- System Classes

Object instance tracking maps a histogram of every object in the JVM. The instance dashboard not only provides the number of instances for a particular class but also provides the shallow memory size (the memory footprint of the object and the primitives it contains) used by all the instances.

Workflow for Detecting and Troubleshooting Memory Thrash

The following diagram outlines the workflow for monitoring and troubleshooting memory thrash problems in a production environment.

Use the **JVM** tab in node dashboards to monitor leaks. See [Troubleshoot Java Memory Leaks](#).

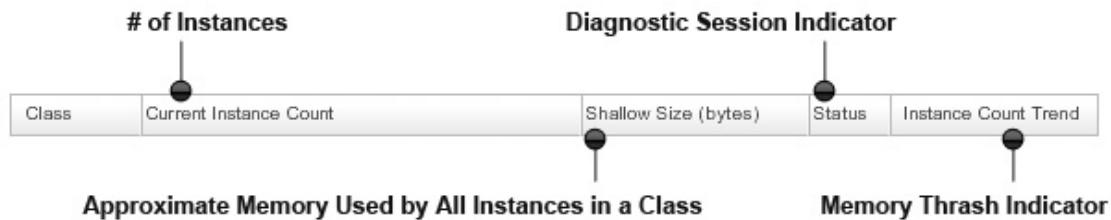


Isolating and Analyzing Memory Thrash

The Instance dashboard indicates possible cases of memory thrash. Prime indicators of memory thrash problems are:

- **Current Instance Count:** A high number indicates possible allocation of large number of temporary objects.
- **Shallow Size:** A large number for shallow size signals potential memory thrash.
- **Instance Count Trend:** A saw wave is an instant indication of memory thrash.

Once a memory thrash problem is identified in a particular Collection, start the diagnostic session.



To analyze memory thrash problems

1. Select the class name to monitor and click **Drill Down** at the top of the Object Instance Dashboard.

- Or right click on the class name and select the "Drill Down" option.

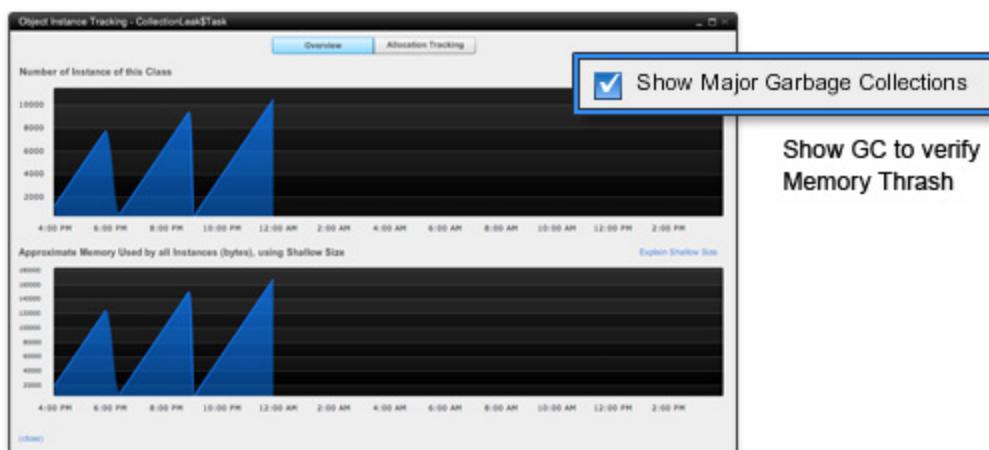
Note: To achieve optimum performance, trigger drill down action on a single instance /class name at a time.

After the drill down action is triggered, data collection for object instances is performed every minute.

To verify memory thrash

1. Enable "Show Major Garbage Collections".

2. If the instance count doesn't vary with the GC cycle, it is an indication of potential leak (see [Troubleshoot Java Memory Leaks](#)).



Using Allocation Tracking

Allocation Tracking tracks all the code paths and those business transactions that are allocating instances of a particular class.

Allocation tracking detects those code path/business transactions that are creating and throwing away instances.

To use allocation tracking:

1. Trigger Drill Down action.
2. Select "Allocation Tracking" from the pop-up window.
3. Choose "Start Allocation Tracking Session" to start tracking code paths and business transactions.
4. Enter the session duration and allow at least 1-2 minutes for data generation.
5. Click the refresh button to retrieve the session data.
6. Click on the particular snapshot to view the details about an individual session.

1 Select the "Allocation Tracking" START

5 Click on the snapshot to view details about an individual session.

4 Click the refresh button to retrieve the session data.

2 Click "Start Allocation Tracking Session" to start tracking code paths and business transactions

3 Enter the session duration... Allow at least 1-2 minutes for data generation

The screenshot shows the AppDynamics Java Application Monitoring interface. At the top, there's a navigation bar with 'Overview' and 'Allocation Tracking' tabs. Below it, a section titled 'Code Paths and Business Transactions creating instances of this Class' lists various Java method names and their occurrence counts. A 'Troubleshooting Information' table shows transaction names and their occurrences. A 'Start Allocation Tracking Session' dialog box is open in the foreground, prompting for a session duration (set to 10 minutes). A yellow callout points to the 'Start' button in the dialog with step 3. Another yellow callout points to the 'Start Allocation Tracking Session' button in the main interface with step 2. A third yellow callout points to the 'Refresh' button in the sidebar with step 4. A fourth yellow callout points to the 'Allocation Tracking' tab with step 1. A fifth yellow callout points to the 'Allocation Tracking' link in the sidebar with step 5.

Monitor Java Applications

Monitor JVMs

- Infrastructure Monitoring in a Java Environment
- JVM Key Performance Indicators
 - Memory Usage and Garbage Collection
 - To view heap usage, garbage collection, and memory pools
 - Heap Usage
 - Garbage Collection
 - Memory Pools
 - Classes, Threads, and Process CPU Usage
 - To view class, thread, and CPU usage
 - Alerting for JVM Health

- Monitoring JVM Configuration Changes
- Detecting Memory Leaks
 - Automatic Leak Detection
 - To enable automatic leak detection
- Detecting Memory Thrash
 - Object Instance Tracking
 - To monitor Java object instances
- Monitoring Long-lived Collections
 - To view or configure custom memory structures
- Learn More

AppMan Advice



JVM/container configuration can often be a root cause for slow performance because not enough resources are available to the application.

Infrastructure Monitoring in a Java Environment

A Java application environment has multiple functional subsystems. These are usually instrumented using JMX (Java Management Extensions) or IBM Performance Monitoring Infrastructure (PMI). AppDynamics automatically discovers JMX and PMI attributes.

JMX uses objects called MBeans (Managed Beans) to expose data and resources from your application. In a typical application environment, there are three main layers that use JMX:

- JVMs provide built-in JMX instrumentation, or platform-level MBeans that supply important metrics about the JVM.
- Application servers provide server or container-level MBeans that reveal metrics about the server.
- Applications often define custom MBeans that monitor application-level activity.

MBeans are typically grouped into domains to indicate where resources belong. Usually in a JVM there are multiple domains. For example, for an application running on Apache Tomcat there are “Catalina” and “Java.lang” domains. “Catalina” represents resources and MBeans relating to the Tomcat container, and “Java.lang” represents the same for the JVM Hotspot runtime. The application may have its own custom domains.

For more information about JMX see the [JMX overview and tutorial](#). To learn about PMI see [Writing PMI Applications Using the JMX Interface](#).

JVM Key Performance Indicators

There are often thousands of attributes, however, you may not need to know about all of them. By default, AppDynamics monitors the attributes that most clearly represent key performance indicators and provide useful information about short and long term trends. The preconfigured JVM metrics include:

- Total classes loaded and how many are currently loaded
- Thread usage
- Percent CPU process usage
- On a per-node basis:
 - Heap usage
 - Garbage collection
 - Memory pools and caching
 - Java object instances

You can configure additional monitoring for:

- Automatic leak detection
- Custom memory structures

Memory Usage and Garbage Collection

Monitoring garbage collection and memory usage can help you identify memory leaks or memory thrash that can have a negative impact application performance.

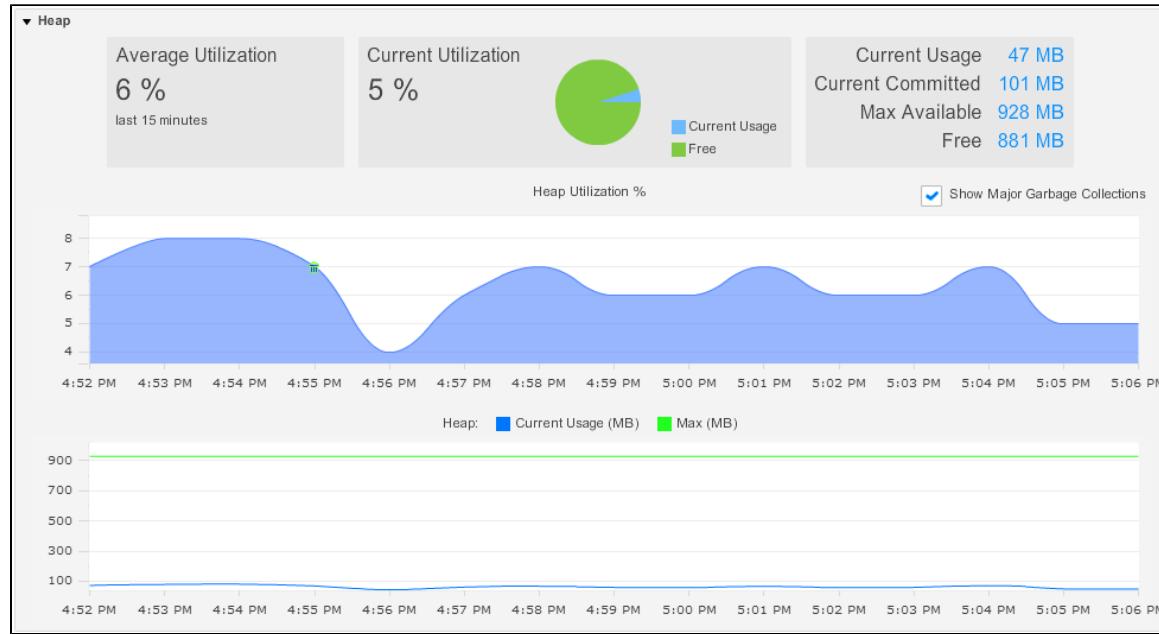
To view heap usage, garbage collection, and memory pools

1. In the left navigation pane, click **Servers -> App Servers -> <tier> -> <node>**. The Node Dashboard opens.
2. In the Node Dashboard, click the **Memory** tab.
3. In the Memory panel, click the **Heap & Garbage Collection** tab. The panels show data about the current usage.

See [Node Dashboard](#) for a complete description of the this dashboard.

Heap Usage

The Heap panel shows data about the current usage.

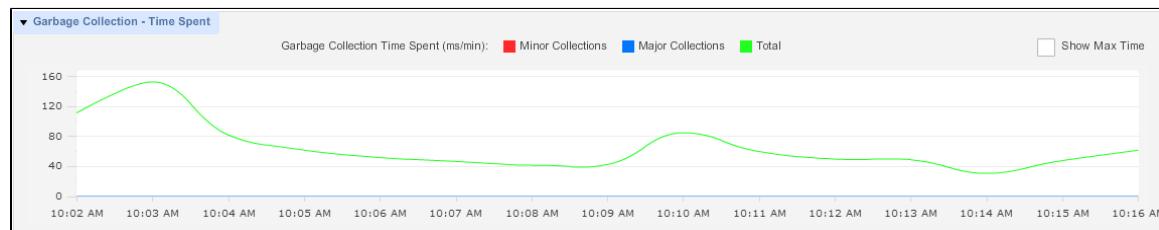


Garbage Collection

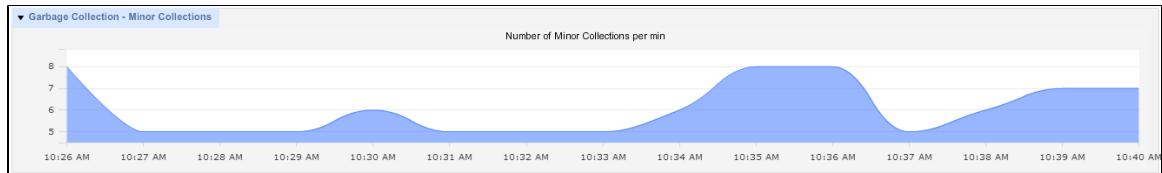
Java garbage collection refers to how the JVM monitors the objects in memory to find any objects which are no longer being referenced by the running application. Unused objects are deleted from memory to make room for new objects. For details see the Java documentation for [Tuning Garbage Collection](#).

Garbage collection is a well-known mechanism provided by Java Virtual Machine to reclaim heap space from objects that are eligible for Garbage collection. The process of scanning and deleting objects can cause pauses in the application. Because this can be an issue for applications with large amounts of data, multiple threads, and high transaction rates, AppDynamics captures performance data about the duration of the pauses for garbage collection.

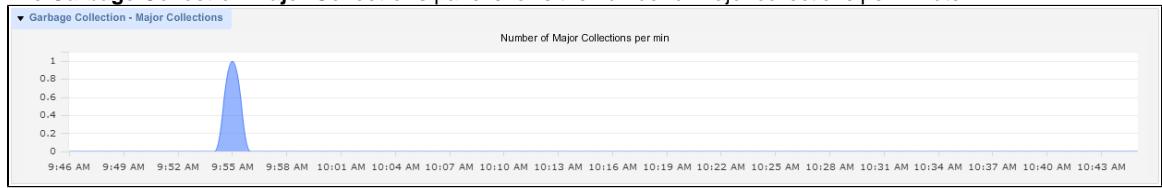
Below the **Heap** panel, the **Garbage Collection Time Spent** panel shows how much time, in milliseconds, it takes to complete both minor and major collections.



The **Garbage Collection Minor Collections** panel shows the number of minor collections per minute. The effectiveness of minor collections indicates better performance for your application.

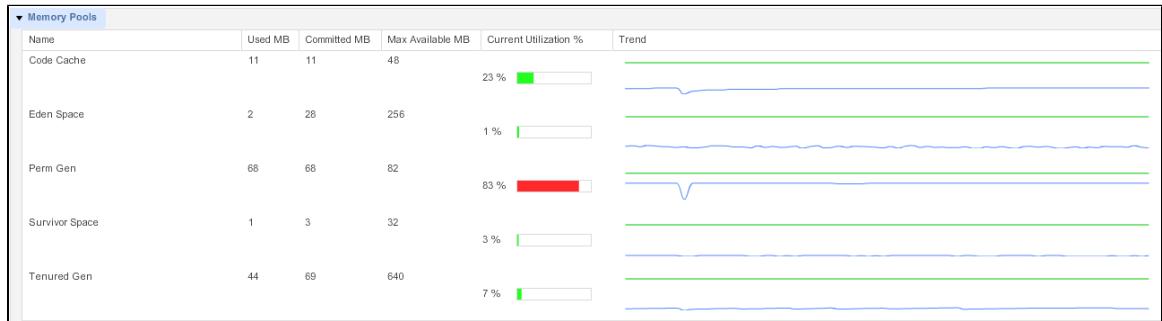


The **Garbage Collection Major Collections** panel shows the number of major collections per minute.



Memory Pools

The **Memory Pools** panel shows usage and trends about the Java memory pools.

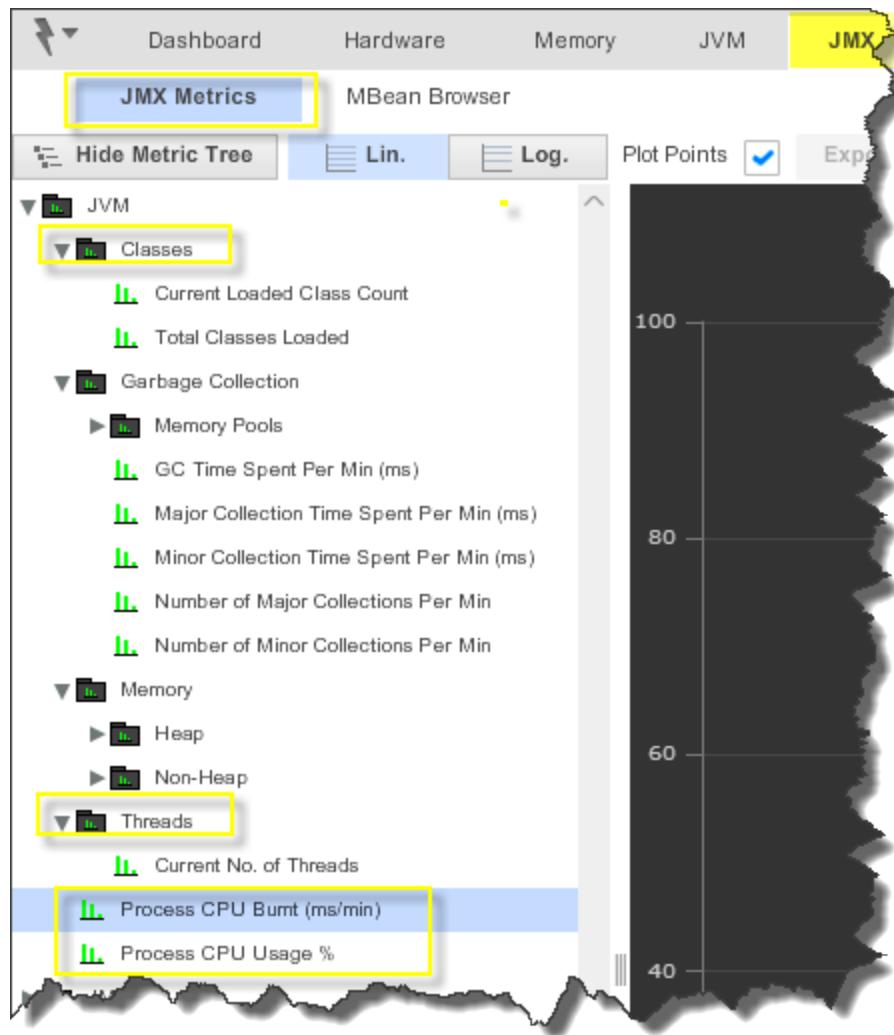


Classes, Threads, and Process CPU Usage

Information on JVM classes, threads and process CPU usage is available on the **JMX** tab of the Node Dashboard.

To view class, thread, and CPU usage

1. In the left navigation pane, click **Servers -> App Servers -> <tier> -> <node>**. The Node Dashboard opens.
2. In the Node Dashboard, click the **JMX** tab.
3. In the **JMX Metrics** panel, click **JVM**. The panels show data about the current usage.



Alerting for JVM Health

You can set up health rules based on JVM/JMX metrics. Once you have a health rule, you can create specific policies based on health rule violations. One type of response to a health rule violation is an alert. See [Alert and Respond](#) for a discussion of how health rules, alerts, and policies can be used.

You can also create additional persistent JMX metrics from MBean attributes. See [Configure JMX Metrics from MBeans](#).

Monitoring JVM Configuration Changes

The **JVM** tab of the Node Dashboard displays the JVM version, startup options, system options and environment properties for the node.

The screenshot shows the AppDynamics interface for monitoring an 'E-Commerce' application. The top navigation bar includes links for Dashboard, Hardware, Memory, **JVM**, JMX, Events, Slow Response Times, and Errors. The 'JVM' tab is currently active. Below the tabs, there are two main sections: 'Properties' and 'JVM Startup Options'. The 'Properties' section displays the following details:

- Version: Server Agent v3.6.2.0 GA #2013-02-08_20-00-53 rcb8a74384e7e15d1695cc4c3a08a620fa647958e 49
- JVM Version: Java HotSpot(TM) 64-Bit Server VM 1.6.0_33 Sun Microsystems Inc.
- Process ID: 22117

The 'JVM Startup Options' section contains a list of command-line arguments:

```
-Dcatalina.base=TIER1TOMCAT
-Dcatalina.home=TIER1TOMCAT
-Djava.endorsed.dirs=TIER1TOMCAT/common/endorsed
-Djava.io.tmpdir=TIER1TOMCAT/temp
-Djava.util.logging.config.file=TIER1TOMCAT/conf/logging.properties
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-javaagent:/mnt/appdynamics/applications/Cart/appagent/tier1/javaagent.jar
```

Below these sections is another 'JVM System Options' section with an 'Export Grid' button. It lists the following system properties:

Name	Value
catalina.base	TIER1TOMCAT
catalina.home	TIER1TOMCAT
file.encoding	UTF-8
file.encoding.pkg	sun.io
file.separator	/
java.awt.graphicsenv	sun.awt.X11GraphicsEnvironment

Changes to the application configuration generate events that can be viewed in the Events list.

For more information, see [Monitor Application Change Events](#).

Detecting Memory Leaks

By monitoring JVM heap utilization and memory pool usage you can identify potential memory leaks. Consistently increasing heap valleys may indicate either an improper heap configuration or a memory leak. You might identify potential memory leaks by analyzing the usage pattern of either the survivor space or the old generation. To troubleshoot memory leaks see [Troubleshoot Java Memory Leaks](#).

Automatic Leak Detection

AppDynamics supports automatic leak detection for some JVMs as listed at the [Compatibility Matrix for Memory Monitoring](#). By default this functionality is not enabled, because using this mode results in higher overhead on the JVM. AppDynamics recommends that you enable leak detection mode only when you suspect a memory leak problem and that you turn it off once the leak is identified and remedied.

Memory leaks occur when an unused object's references are never freed. These are the most common occurrences in Collections classes, such as `HashMap`. This is caused when an application code puts objects in Collections but does not remove them even when they are not being actively used. In production environments with high workloads, a frequently accessed Collection with a memory leak can cause the application to crash.

AppDynamics automatically tracks every Java Collection (HashMap, ArrayList, and so on) that has been alive in the heap for more than 30 minutes. The Collection size is tracked and a linear regression model identifies if the Collection is potentially leaking. You can then identify the root cause of the leak by tracking frequent accesses of the Collection over a period of time.

View this data on the Node Dashboard on the **Automatic Leak Detection** panel of the **Memory** tab. See [Node Dashboard](#) for a complete description of this dashboard and its tabs.

To enable automatic leak detection

To enable automatic leak detection follow the instructions at [Troubleshoot Java Memory Leaks](#).

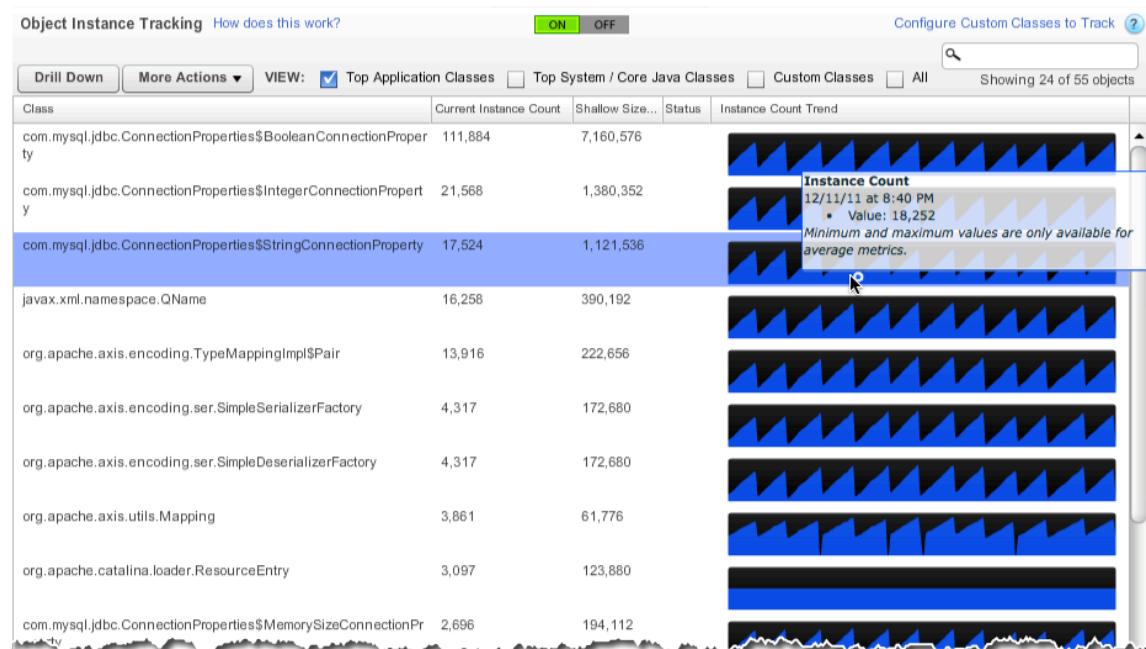
Detecting Memory Thrash

Memory thrash is caused when a large number of temporary objects are created in very short intervals. Although these objects are temporary and are eventually cleaned up, the garbage collection mechanism may struggle to keep up with the rate of object creation. This may cause application performance problems. Monitoring the time spent in garbage collection can provide insight into performance issues, including memory thrash. For example, an increase in the number of spikes for Major Collections either slows down a JVM or indicates potential memory thrash. To troubleshoot memory thrash, see [Troubleshoot Java Memory Thrash](#).

Object Instance Tracking

The **Object Instance Tracking** panel helps you isolate the root cause of possible memory thrash. By default, AppDynamics tracks the object instances for the top 20 core Java classes and the top 20 application classes. For the list of the supported JVMs see the [Compatibility Matrix for Memory Monitoring](#).

The **Object Instance Tracking** panel provides the number of instances for a particular class and graphs the count trend of those object in the JVM. It provides the shallow memory size (the memory footprint of the object and the primitives it contains) used by all the instances.



To monitor Java object instances

1. In the Node Dashboard, click the **Memory** tab.
3. In the Memory panel, click the **Object Instance Tracking** tab.

For details see [Configure Object Instance Tracking \(Java\)](#).

Monitoring Long-lived Collections

AppDynamics automatically tracks long lived Java Collections (HashMap, ArrayList, and so on) with Automatic Leak Detection. You can also configure tracking of specific classes using the Custom Memory Structures capability. You can use this capability to monitor a custom cache or other structure that is not a Java Collection. Custom memory structures are used as caching solutions. For example, you may have a custom cache or a third party cache such as Ehcache. In a distributed environment, caching can easily become a prime source of memory leaks. In addition, custom memory structures may or may not contain Collections objects that would be tracked using automatic leak detection. It is therefore important to manage and track these memory structures.

AppDynamics provides visibility into:

- Cache access for slow, very slow, and stalled business transactions
- Usage statistics (rolled up to Business Transaction level)
- Keys being accessed
- Deep size of internal cache structures

To view or configure custom memory structures

1. In the Node Dashboard, click the **Memory** tab.
3. In the **Memory** panel, click **Custom Memory Structures**.

For details see [Configure Custom Memory Structures \(Java\)](#).

Learn More

- Configure Policies
- Supported Environments and Versions
- Infrastructure Metrics
- Monitor Events

Monitor Java App Servers

- Infrastructure Monitoring in a Java Environment
- App Server Key Performance Indicators
- Alerting for App Server Health
- Learn More

AppMan Advice



JVM/container configuration can often be a root cause for slow performance because not enough resources are available to the application.

Infrastructure Monitoring in a Java Environment

A Java application environment has multiple functional subsystems. These are usually instrumented using JMX (Java Management Extensions) or IBM Performance Monitoring Infrastructure (PMI). AppDynamics automatically discovers JMX and PMI attributes.

JMX uses objects called MBeans (Managed Beans) to expose data and resources from your application. In a typical application environment, there are three main layers that use JMX:

- JVMs provide built-in JMX instrumentation, or platform-level MBeans that supply important metrics about the JVM.
- Application servers provide server or container-level MBeans that reveal metrics about the server.
- Applications often define custom MBeans that monitor application-level activity.

MBeans are typically grouped into domains to indicate where resources belong. Usually in a JVM there are multiple domains. For example, for an application running on Apache Tomcat there are "Catalina" and "Java.lang" domains. "Catalina" represents resources and MBeans relating to the Tomcat container, and "Java.lang" represents the same for the JVM Hotspot runtime. The application may have its own custom domains.

For more information about JMX see the JMX overview and tutorial. To learn about PMI see [Writing PMI Applications Using the JMX Interface](#).

App Server Key Performance Indicators

AppDynamics creates long-term metrics of the key MBean attributes that represent the health of the Java container. Depending on your application configuration, metrics may include:

- Session information such as the number of active and expired sessions, maximum active sessions, processing time, average and maximum alive times, and a session counter.
- Web container runtime metrics that represent the thread pool that services user requests. The metrics include pending requests and number of current threads servicing requests. These metrics are related to Business Transaction metrics such as response time.
- Messaging metrics related to JMS destinations, including the number of current consumers and the number of current messages.
- JDBC connection pool metrics including current pool size and maximum pool size.

To see the JMX metrics discovered in a node, see the JMX tab on the Node Dashboard.

To learn how to customize additional MBean attributes for long-term monitoring, see [Configure JMX Metrics from MBeans](#).

Alerting for App Server Health

AppDynamics discovers metrics for most Java platforms and applications. Some environments however are not instrumented by default, yet they have MBeans. For those situations you can enable monitoring using the MBean Browser. For details see [Monitor JMX MBeans](#).

In addition to the preconfigured metrics, you may be interested in additional JVM or Java container metrics. You can add custom metrics using JMX MBean attributes in the Metric Browser. To customize which MBean attributes are monitored, see [Configure JMX Metrics from MBeans](#).

Once you add a custom metric you can create a custom health rule for it and receive alerts if conditions indicate problems. For details see [Alert and Respond](#).

AppDynamics also provides the Application Server Agent API (Agent API) for access to metrics that are not supported by default or by MBeans. You can use the Agent API to:

- Inject custom events and report on them
- Create and report on new metrics
- Correlate distributed transactions when using protocols that AppDynamics does not support

Learn More

- [Configure JMX Metrics from MBeans](#)
- [Monitor JMX MBeans](#)
- [Configure Health Rules](#)
- [Supported Environments and Versions](#)

Monitor JMX MBeans

- [JMX MBeans and Monitoring Application Infrastructure](#)
 - [Prerequisites for JMX Monitoring](#)
 - [Preconfigured JMX Metrics](#)
 - To view the configuration of the preconfigured JMX metrics
- [Using AppDynamics for JMX Monitoring](#)
 - [To view JMX metrics in the Metrics Browser](#)
 - [Trending MBeans Using Live Graphs](#)
 - To monitor the real-time trend of an MBean
 - [Configuring New JMX Metrics](#)
 - [Reusing JMX Metric Configurations](#)
- [Learn More](#)

This topic discusses how to provide visibility into the JMX metrics for your JVM and application server.

JMX MBeans and Monitoring Application Infrastructure

As discussed at [Monitor JVMs](#) and [Monitor Java App Servers](#), AppDynamics uses JMX (Java Management Extensions) to monitor Java applications.

JMX uses objects called MBeans (Managed Beans) to expose data and resources from your application. You can use one or more MBean attributes to create persistent JMX metrics in AppDynamics. In addition, you can import and export JMX metric configurations from one version or instance of AppDynamics to another.

Prerequisites for JMX Monitoring

AppDynamics can capture MBean data, when these conditions are met:

- The monitored system must be running on Java 1.5 or later.
- Each monitored Java process must enable JMX. See [the JMX documentation](#).

Additional MBean data may be available when a monitored business application exposes Managed Beans (MBeans) using standard JMX. See [the MBean documentation](#).

Preconfigured JMX Metrics

AppDynamics provides preconfigured JMX metrics for several common app server environments:

- Apache ActiveMQ
- Cassandra
- GlassFish
- HornetQ
- JBoss
- Apache Solr
- Apache Tomcat
- Oracle WebLogic Server
- WebSphere PMI

For app server environments that are not instrumented out-of-the-box, you can configure a new JMX metrics configuration. You can also add new metric rules to the existing set of configurations. For example, Glassfish JDBC connection pools can be manually configured using MBean attributes and custom JMX metrics.

To view the configuration of the preconfigured JMX metrics

1. In the left navigation pane, click **Configure -> Instrumentation** and select the **JMX** tab.
The list of JMX Metric Configurations appears.

The screenshot shows the 'Instrumentation' section of the AppDynamics configuration interface. The 'JMX Metric Configurations' tab is selected. On the left, there's a sidebar with icons for Transaction Detection, Backend Detection, and End User Experience. Below the sidebar, there are two tables: 'JMX Metric Configurations' and 'JMX Metric Rules'. The 'JMX Metric Configurations' table lists various app servers with their status (Enabled). The 'JMX Metric Rules' table is currently empty.

Name	Enabled
ActiveMQ	✓
Cassandra	✓
Glassfish	✓
HornetQ	✓
JBoss	✓
Platform	✓
Solr	✓
Tomcat	✓
WebLogic	✓
WebSpherePMI	✓

Name	Enabled

2. Click a metric configuration to view the preconfigured JMX metrics for that app server.

For example, selecting Cassandra shows the preconfigured JMX Metric Rules for Apache Cassandra. Double-click a metric rule to see configuration details such as the MBeans matching criteria and the MBean attributes being used to define the metric.

The screenshot shows the 'ACME Book Store Application' dashboard with the 'Configure' tab selected. Under 'Instrumentation', the 'JMX Metric Configurations' section is active, displaying a list of configured servers: ActiveMQ, Cassandra, Glassfish, HornetQ, JBoss, Platform, Solr, Tomcat, and WebLogic. The 'Cassandra' row is selected. To the right, under 'Cassandra', the 'JMX Metric Rules' section is shown, listing specific rules for Cassandra components: Cassandra_Caches, Cassandra_CompactionManager, Cassandra_FailureDetector, Cassandra_GossipStage, Cassandra_ReadStage, and Cassandra_StorageProxy. All rules are enabled, indicated by green checkmarks.

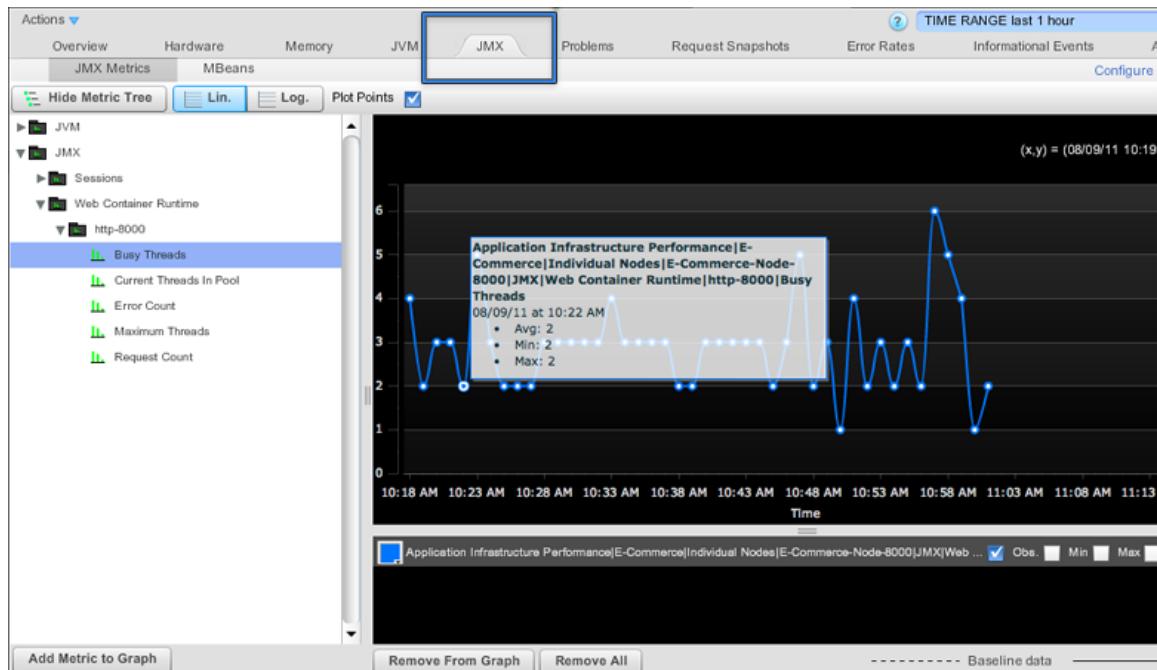
You can view, delete, and edit the existing JMX metric rules. You can add new JMX metric rules. See [37BKUP:Configure JMX Metrics from MBeans].

Using AppDynamics for JMX Monitoring

You can view MBean-based metrics using the Node Dashboard and the Metric Browser. In addition, the MBean Browser enables you to view all the MBeans defined in the system.

To view JMX metrics in the Metrics Browser

1. In the left navigation pane, click **Servers->App Servers-><Tier>><Node>**. The Node Dashboard opens.
2. Click the **JMX** tab. The JMX Metric Browser opens and displays the MBeans in a Metric Tree.
3. Browse the default JMX metrics.
4. To monitor a particular metric, double-click or drag and drop the metric onto the graph panel.



5. You can perform all the operations that are provided by the Metric Browser such as:

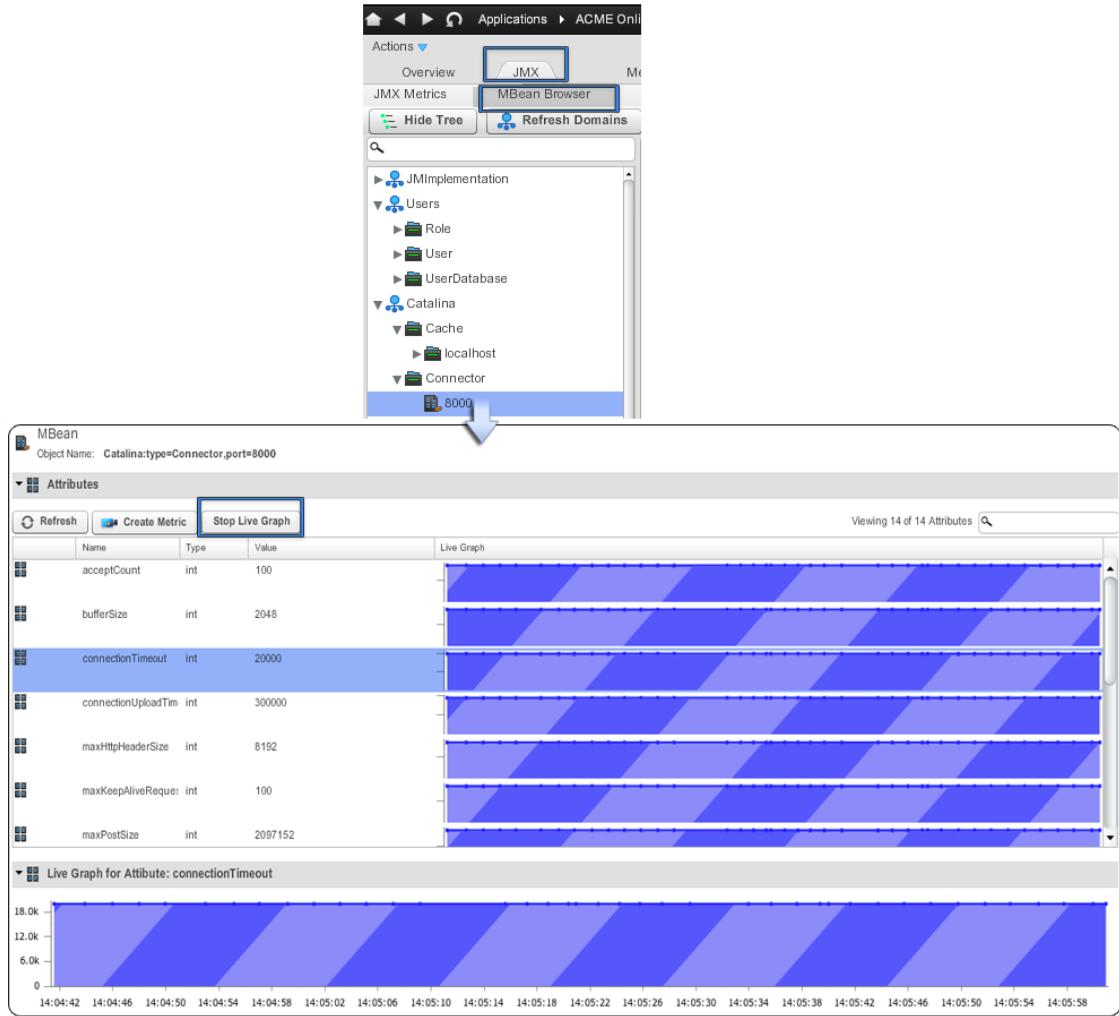
- Drill-down
- Analyze the transaction snapshot for a selected time duration
- Set the selected time range as a global time range

Trending MBeans Using Live Graphs

You can monitor the trend of a particular MBean attribute over time using the **Live Graph**.

To monitor the real-time trend of an MBean

1. In the left navigation pane, click **Servers->App Servers->><Tier>><Node>**. The Node Dashboard opens.
2. Click the **JMX** tab.
3. Click the **MBean Browser** sub-tab.
4. Select the domain for which you want to monitor MBeans. For a description of domains see [Monitor JVMs](#).
5. Select the MBean that is of interest to you.
6. Click **Start Live Graph**. You can see the runtime values.
7. Select an attribute and click **Live Graph for Attribute** to see a larger view of a particular graph.



Configuring New JMX Metrics

In addition to the preconfigured metrics, you can define a new persistent metric using a JMX Metric Rule that maps a set of attributes from one or more MBeans.

You can create a JMX metric from any MBean attribute or set of attributes. Once you create a persistent JMX metric, you can:

- View it in the Metric Browser
- Add it to a Custom Dashboard
- Create a health rule for it so that you can receive alerts

The JMX Metrics Configuration panel is the central configuration interface for all of the JMX metrics that AppDynamics reports. You can use the MBean Browser to view MBeans exposed in your environment. From there, you can access the JMX Metrics Configuration panel by selecting an MBean attribute and clicking **Create Metric**.

For details, see [Configure JMX Metrics from MBeans](#).

Reusing JMX Metric Configurations

Once you create a custom JMX metric, you can keep the configuration for upgrade or other purposes. The JMX metric information is stored in an XML file that you can export and then import to another AppDynamics system. For instructions see [Import or Export JMX Metric Configurations](#).

Learn More

- Configure JMX Metrics from MBeans
- Monitor JVMs
- Import or Export JMX Metric Configurations
- Configure JMX Without Transaction Monitoring

Trace Multi-Threaded Transactions (Java)

- Thread Visibility
 - Asynchronous Calls in Dashboards
 - Threads and Thread Tasks in the Metric Browser
- Threads in Call Graphs
 - To Drill Down into Downstream Calls on a Thread
- Thread Metrics in Health Rules
- Learn More

Multithreaded programming techniques are common in applications that require asynchronous processing. Threads are first class entities that you can monitor.

Although each thread has its own call stack, multiple threads can access shared data. This creates two potential problems: visibility and access.

- A visibility problem occurs if thread A reads shared data which is later changed by thread B, and thread A is not aware of the change.
- An access problem occurs if several threads are trying to access and change the same shared data at the same time.

Visibility and access problems can lead to:

- Liveness failure: Application performance becomes sluggish or stops processing also known as a deadlock.
- Safety failure: Race condition that results in difficult to discover programming errors.

Thread contention can occur when multiple threads attempt to access a synchronized method or block at the same time. If a thread remains in the synchronized method or blocks for a long time, the other threads must wait for access to shared resources. This situation has an adverse effect on application performance. Call graphs for multi-threaded transactions enable you to trace thread creation in a business transaction and provide an aggregated view of the overall processing for transactions that spawn threads for concurrent processing.

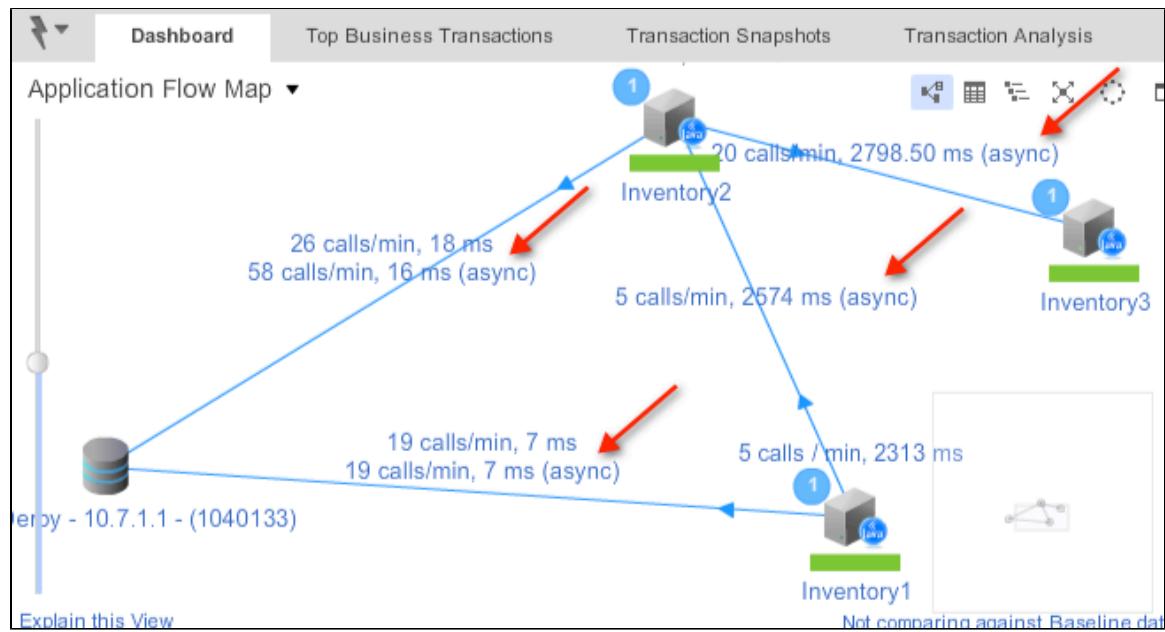
Thread Visibility

Application often spawn threads to perform concurrent tasks. You can monitor each thread as a separate entity, including exit calls and policies linked to a specific thread. All Runnables, Callables and Threads are instrumented by default except those that are explicitly excluded. In some environments, this could lead to too many classes being instrumented. In this case you can create custom rules to exclude them. If you do not want to monitor any threads, you can completely disable Asynchronous monitoring, which requires an agent restart. See [Configure Multi-Threaded Transactions \(Java\)](#).

AppDynamics provides thread visibility in dashboards, the metric browser and call graphs.

Asynchronous Calls in Dashboards

AppDynamics detects asynchronous calls in an application and labels them as "async" in the dashboards that display the asynchronous activity.



You can set the flow map lines to render as dotted lines for easier visibility. Click **Application Flow Map -> Edit Current Flow Map** and check **Use dotted line**.

Edit Flow Map

Overview Tiers Databases and Remote Services

Name: Default Flow Map
You cannot edit the name of a default flow map.

Scope: Acme Online Book Store (App)
 Shared
Shared Flow Maps can be used across multiple applications.

Colors (revert to default)
Use these options to apply custom colors to the flow map.

Text color	Check to make async transactions more visible in the flow map
Link color	
Background Type	
Background color	

Asynchronous Activity
 Use dotted line
Async. activity between items on the flow map will be rendered as dotted lines.

The tree view of a multi-threaded transaction dashboard shows the errors and time spent in asynchronous calls. You may need to expand the class to see all the threads.

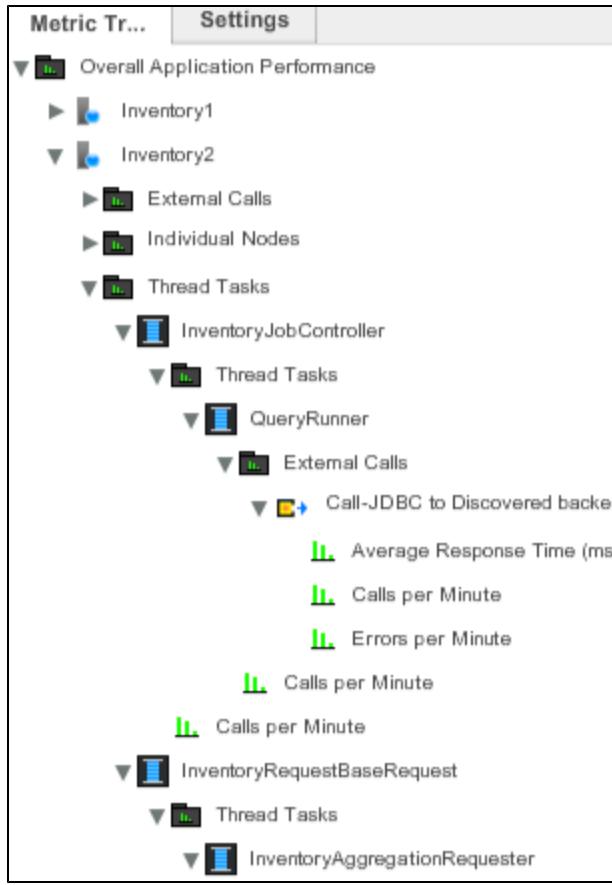
Transaction Flow - Tree View							
		Time Spent (ms)	Calls	Calls / min	Errors	Errors / min	N.
▼	Inventory2	2525.0 ms	100.0 %	72	5	0	0
►	JDBC call to Apache Der	14.0 ms	2.2 %	290	19	0	0
▼	InventorySupplierLookup	1593 ms	async	72	5	0	0
►	HTTP call to Inventor	1593 ms	async	72	5	0	0
▼	InventoryJobController	108 ms	async	72	5	0	0
►	QueryRunner	84 ms	async	72	5	0	0
►	LookupServlet\$1	506 ms	async	72	5	0	0
▼	InventoryRequestBaseR	2006 ms	async	72	5	0	0
►	InventoryAggregation	2008 ms	async	72	5	0	0

Threads and Thread Tasks in the Metric Browser

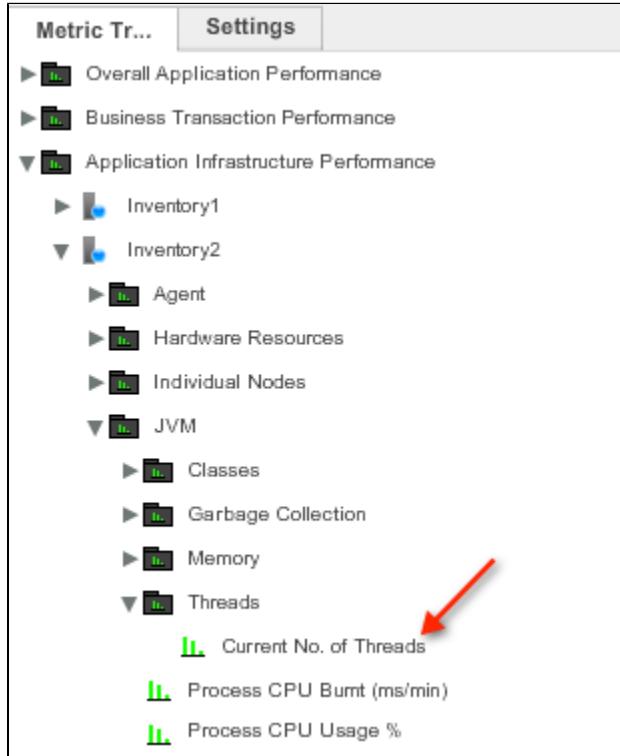
In a multi-threaded transaction, AppDynamics reports key performance metrics for the individual threads in Thread Tasks branch of the tier in the Metric Browser. The Thread Tasks branch is created only for multi-threaded transactions.

Metric Tr...	Settings
►	Business Transaction Groups
▼	Business Transactions
►	Inventory1
▼	Inventory2
▼	/inventory_check2/
►	External Calls
►	Individual Nodes
►	Thread Tasks
■	Average Block Time (ms)
■	Average CPU Used (ms)
■	Average Request Size
■	Average Response Time (ms)
■	Average Wait Time (ms)
■	Calls per Minute
■	End User Average Response Time (ms)
■	End User Network Average Response Time
■	End User Page Render Average Response
■	Errors per Minute
■	Normal Average Response Time (ms)

Thread Tasks are also reported in tiers under Overall Application Performance, where you can get metrics on specific calls made by each thread in a node or in a tier.



Under Application Infrastructure in the JVM section for a node or tier, you can get the number of threads spawned in the Current No. of Threads metric. This is a generic metric not necessarily limited to the thread tasks that AppDynamics tracks.



Threads in Call Graphs

When you drill down in a transaction snapshot with multiple calls, AppDynamics displays the list of calls that you can drill down into.

Select a Call to Drill Down into				
Multiple calls were made to this Tier as part of this Transaction.				
Drill Down into Call		Show: All Calls	Originating from: Show All	
1	10530 ms	Call from (end user)	4 Async. Activity calls (32 ms. max, 8.3 ms. avg.), and 4 JDBC calls (68 ms. max, 17.0 ms. avg.)	10/17/12 1:14:11.247 AM
✓	501 ms	Async Activity (LookupServlet\$1)	No exit calls made.	10/17/12 1:14:21.253 AM
✓	2049 ms	Async Activity (InventoryRequestBaseReq)	1 Async. Activity call (0 ms.)	10/17/12 1:14:21.538 AM
1	10422 ms	Async Activity (InventorySupplierLookup)	1 HTTP call (10420 ms.)	10/17/12 1:14:21.713 AM
✓	60 ms	Async Activity (InventoryJobController)	1 Async. Activity call (0 ms.)	10/17/12 1:14:21.716 AM
✓	47 ms	Async Activity (QueryRunner)	4 JDBC calls (10 ms. max, 2.5 ms. avg.)	10/17/12 1:14:21.727 AM

Select a call from the list and double-click or click **Drill Down into Call** to access the call graph for the thread.

To Drill Down into Downstream Calls on a Thread

If the call graph indicates Async Activity in the Exit Call/Threads column, you can drill down further into the downstream call on the thread:



1. Click **Async Activity** in the Exit Calls/Threads Column for the call that you want to drill down from.
2. In the Exit Calls and Async Activities window, click **Drill Down into Downstream Call**.

Exit Calls and Async Activities at LookupServlet\$1.<init>

Type	Details	Count	Time (ms)	% Time	From Tier	To Tier	Downstream Call Time (ms)
Async. Activity	Asynchronous activ	1	1	0.1	Inventory2	Inventory2	501 ms

 1 ms

Details

Asynchronous activity identified





A call graph for the downstream call opens.

Thread Metrics in Health Rules

You can create a health rule of the custom health rule type based on performance metrics for a thread task.

When you click the metric icon in the Health Rule Wizard, the embedded metric browser includes the Thread Tasks if the entity for which you are configuring the health rule spawns multiple threads.

See [Configure Health Rules](#).

Learn More

- [Configure Multi-Threaded Transactions \(Java only\)](#)
- [Metric Browser](#)
- [Call Graphs](#)
- [Health Rules](#)
- [Configure Health Rules](#)

AppDynamics for .NET

This information covers using AppDynamics for .NET applications and environments. For general information see [AppDynamics Essentials](#) and [AppDynamics Features](#).

Tutorials

Monitor .NET Applications

Troubleshoot .NET Application Problems

Configure AppDynamics for .NET

Administer App Agents for .NET

The following features are currently not available for .NET environments:

- Manual BCI EUM instrumentation also called Assisted Injection Using Injection Rules.
- Remediation Actions, such as running local scripts on nodes.
- Diagnostic thread dump actions are not supported.
- Reassigning nodes is not supported.
- Deleting tiers is not supported.
- Adding custom machine agent metrics (also called custom monitors) is not supported.
- Detecting code deadlocks is not supported.
- Aggressive snapshot collection is not supported.
- Memory monitoring (also known as Object Instance Tracking) is not supported.
- The embedded Machine Agent (installed by default) for the App Agent for .NET does not support extensions. If you want to use Machine Agent extensions such as custom plugins or the HTTP Metric Listener, or do orchestration to compute clouds, install the standalone Machine Agent. See [Configure the .NET Machine Agent](#).
- Some agent node properties are not supported in the .NET environment. Review [App Agent Node Properties Reference](#) for detailed information.

Install the App Agent for .NET

- Installation Prerequisites
 - COM+
 - Configuration Files
- Installing the App Agent for .NET
 - To install the App Agent for .NET
- Sample Silent Install Script
- Learn More

This topic describes how to install the App Agent for .NET.

For more information about upgrading see [Upgrade the App Agent for .NET](#).

All the ASP.NET functionality inside IIS runs under the scope of a worker process. You install an App Agent for .NET on each system that hosts the managed .NET applications. At start up, an individual instance of the agent is created for each application running in the CLR.

Installation Prerequisites

COM+

COM+ must be enabled for the agent to function. To enable COM+ see [Troubleshoot App Agent for .NET Installation and Configuration](#).

Configuration Files

The App Agent for .NET uses the myapp.exe.config and web.config files, XML files that define the configuration of your web application. You can define the agent configuration at different levels. See [App Agent for .NET Configuration Properties](#).

AppDynamics provides a configuration wizard that updates the XML files for IIS environments. See [Configure the App Agent for .NET](#).

For non-IIS configurations see [Enable the App Agent for .NET for Non-IIS Applications](#).

Installing the App Agent for .NET

To install the App Agent for .NET

1. Download the App Agent for .NET.

Download the App Agent for .NET from the [AppDynamics Download Center](#).

2. Launch the installer wizard.

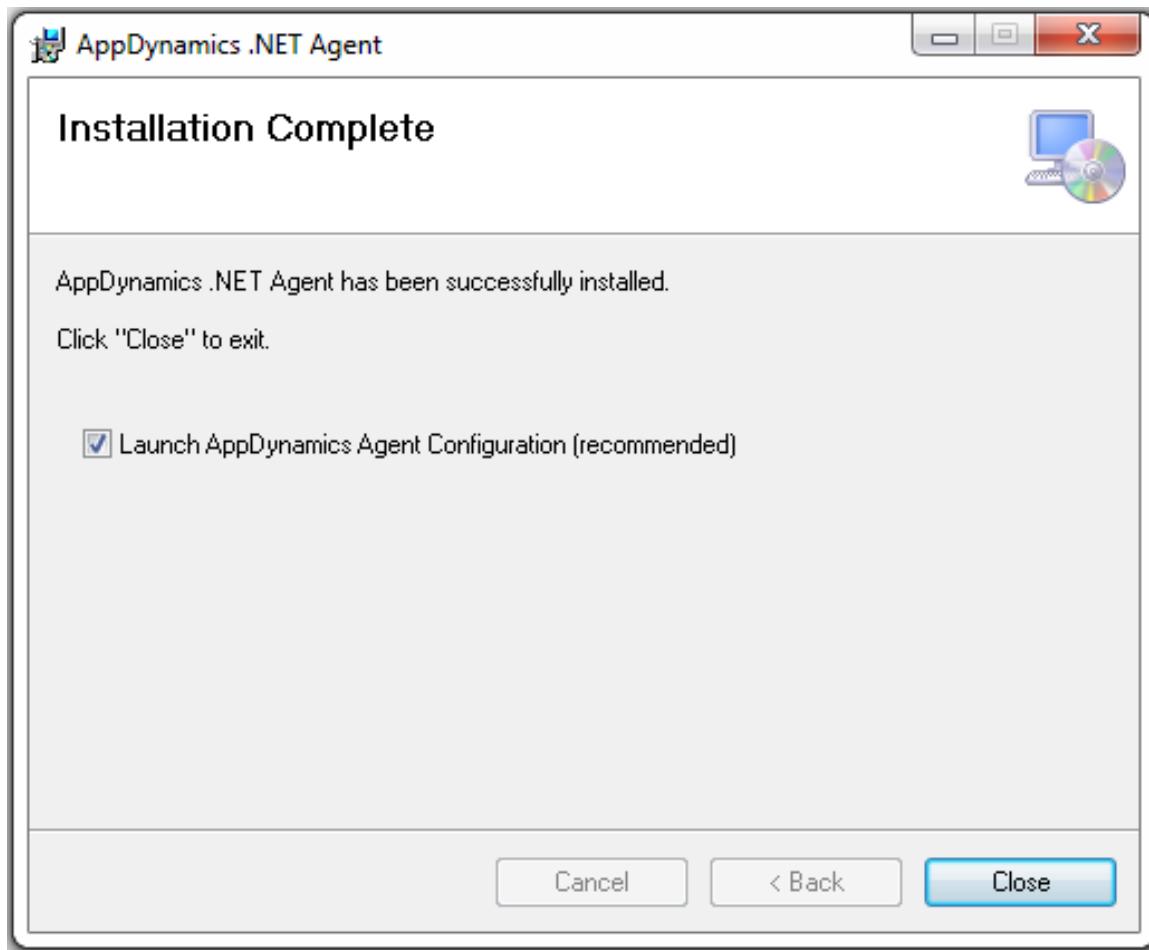
Your account must be administrator or an account with administrator privileges otherwise the installer will stop.

If you are not using the administrator account but the account has administrator privileges, right-click on **Start -> Command Prompt**, and select **Run as Administrator**, and enter the administrator credentials. Then navigate to the directory where the installer is located and run the installer from the command prompt.

3. Select the destination directory for the App Agent for .NET.

By default, the agent is installed at C:\Program Files\AppDynamics\AppDynamics.NET Agent 64-bit. You can also select other location for the installation.

4. Wait for the installation to complete.



- If this is a new installation and you are using IIS, click **Close** to use the App Agent for .NET Configuration Utility. See [Configure the App Agent for .NET](#).
- If this is a new installation and you are not using IIS uncheck the box and then click **Close**. See [Enable the App Agent for .NET for Non-IIS Applications](#).
- If this is an upgrade or if the configuration already exists, uncheck the box and then click **Close**.

For troubleshooting information see [Troubleshoot App Agent for .NET Installation and Configuration](#).

Sample Silent Install Script

The following is a sample script to install the agent. It assumes that you are first uninstalling an older agent, so if this is a fresh install, remove the uninstall command.

```
installscript.bat "olddotNetAgentSetup64.msi" "uninstall.log" "dotNetAgentSetup64.msi"
"C:\Program Files\AppDynamics\AppDynamics .NET Agent" "install.log"
"SavedSetupConfiguration.xml"
```

Learn More

- Multi-Agent Deployment for .NET
- Logical Model
- App Agent for .NET Configuration Properties
- Troubleshoot App Agent for .NET Installation and Configuration
- Disable Instrumentation for an IIS Application Pool
- Enable the App Agent for .NET for Non-IIS Applications
- Upgrade the App Agent for .NET

Upgrade the App Agent for .NET

This topic describes how to upgrade to a new version of the App Agent for .NET.

To upgrade the App Agent for .NET

1. Backup the <Agent_Installation_Directory>\Machine Agent\Configuration\application.config file for the Machine Agent configurations.
2. Backup the <Agent_Installation_Directory>\AppDynamicsAgentLog.config file.
3. Stop IIS, Windows services, and application executables.
4. Open services.msc and stop the AppDynamics.Agent.coordinator service.
5. In the Control Panel, select Add/Remove Program and remove the "AppDynamics .NET Agent".
6. Launch and run the installer for the App Agent for .NET.
 Uncheck the Installer option to run the Configuration Utility. Do not reconfigure at this point!
7. Replace the new application.config file with the backup copy.
8. Replace the AppDynamicsAgentLog.config with the backup copy.
9. (Optional) Run the Configuration Utility if you want to change anything. See [Configure the App Agent for .NET](#).
10. Start IIS, Windows services, and application executables.

Learn More

- [Install the App Agent for .NET](#)
- [Configure the App Agent for .NET](#)
- [Uninstall the App Agent for .NET](#)
- [Agent - Controller Compatibility Matrix](#)
- [Release Notes for AppDynamics Pro](#)

Configure the App Agent for .NET

- [Configuration Considerations](#)
- [Configuring an App Agent for .NET](#)
 - To access the .NET Agent Configuration Utility
 - To set up the logs and account permissions

- To provide Controller configuration information
- To map business applications, tiers, and nodes to your application environment
- To let AppDynamics automatically name the tiers
- To manually name the tiers
- Using a Configuration File from the Command Line
 - To create a configuration file
 - To run the configuration utility from the command line
- Learn More

The App Agent for .NET requires information about your IIS and non-IIS applications on .NET. You configure the agent using the App Agent for .NET Configuration Utility.

Configure the App Agent for .NET according to what kind of application you want to monitor:

- For IIS applications, use the Configuration Utility with either automatic or manual tier naming. See the instructions in this topic.
- For non-IIS applications and services, set an environment variable and use the Configuration Utility with automatic tier naming. See [Enable the App Agent for .NET for Non-IIS Applications](#).
- For non-IIS applications and services, set an environment variable, use the Configuration Utility with manual tier naming, and manually edit the application configuration file. See [Enable the App Agent for .NET for Non-IIS Applications](#).

Use the App Agent for .NET Configuration Utility to configure the agent just after installation, or to make changes to existing agent configurations.

Configuration Considerations

AppDynamics recommends that you install the Controller, or have access credentials to a SaaS Controller, before installing an agent.

The utility configures one agent at a time.

AppDynamics implements the profile API of the .NET CLR. Since only one profiler can be used at a time on a Windows machine, you may need to uninstall any pre-existing profiler, such as Ant, VS 2010 Performance Tools, or others. The utility will alert you if it finds a pre-existing profiler.

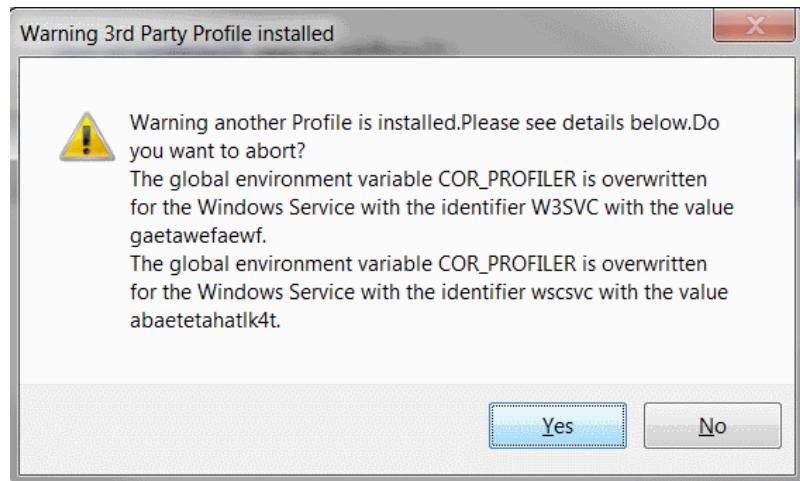
 To apply configurations, the .NET Agent Configuration Utility must restart IIS.

Prior to configuration, run the .NET Agent Installer. See [Install the App Agent for .NET](#).

Configuring an App Agent for .NET

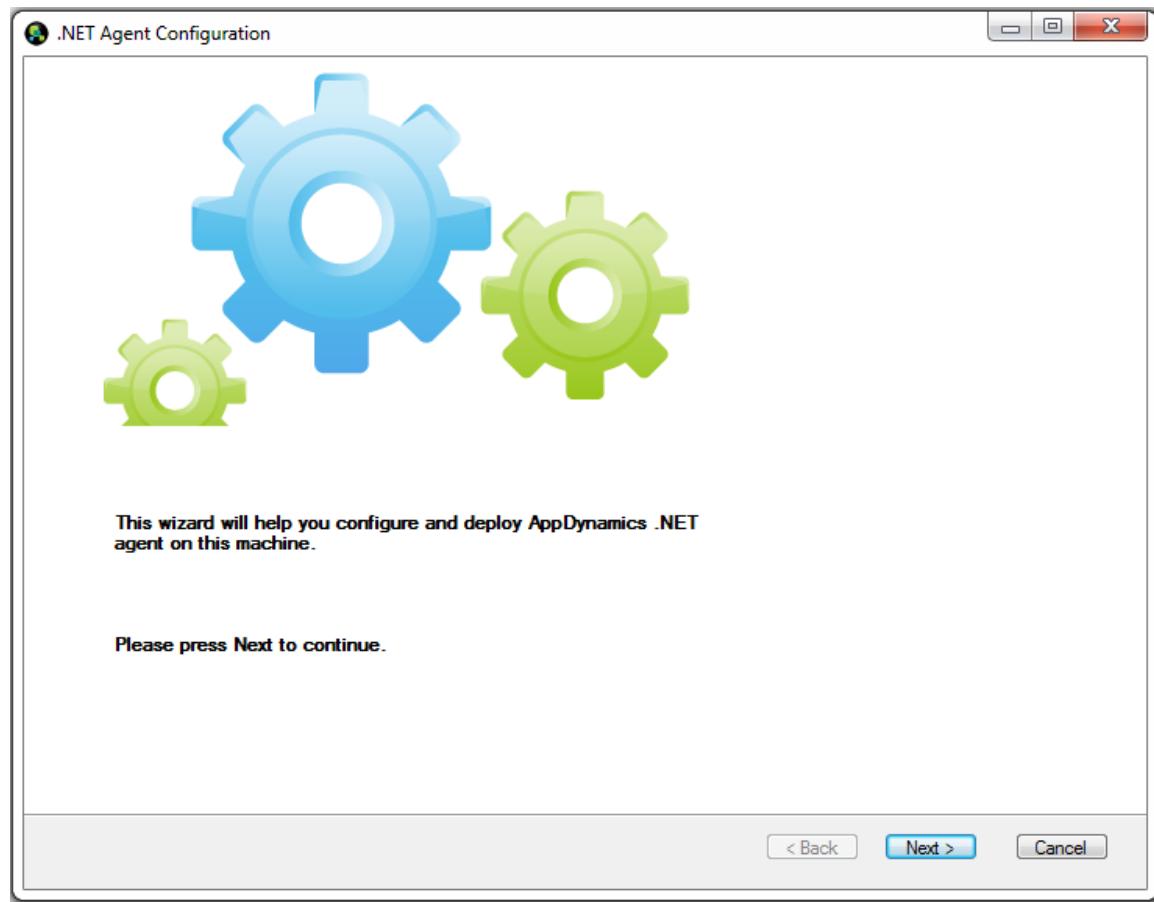
To access the .NET Agent Configuration Utility

1. In the Windows menu, click **AppDynamics -> .NET Agent Configuration**.
2. If you get a message "Warning: 3rd Party Profiler installed" that means that another profiler is already installed. There can be only one profiler per Windows machine, and since AppDynamics uses a profiler you must uninstall any other profilers.



Click **OK** to exit and uninstall any pre-existing profiler. Check the registry to make sure that the uninstall process cleaned up the registry entries. Use the warning message to identify any undeleted profiler environment variables.

3. When there are no profiler conflicts, the Welcome screen appears.



Click **Next**. The Log directory permissions window appears.

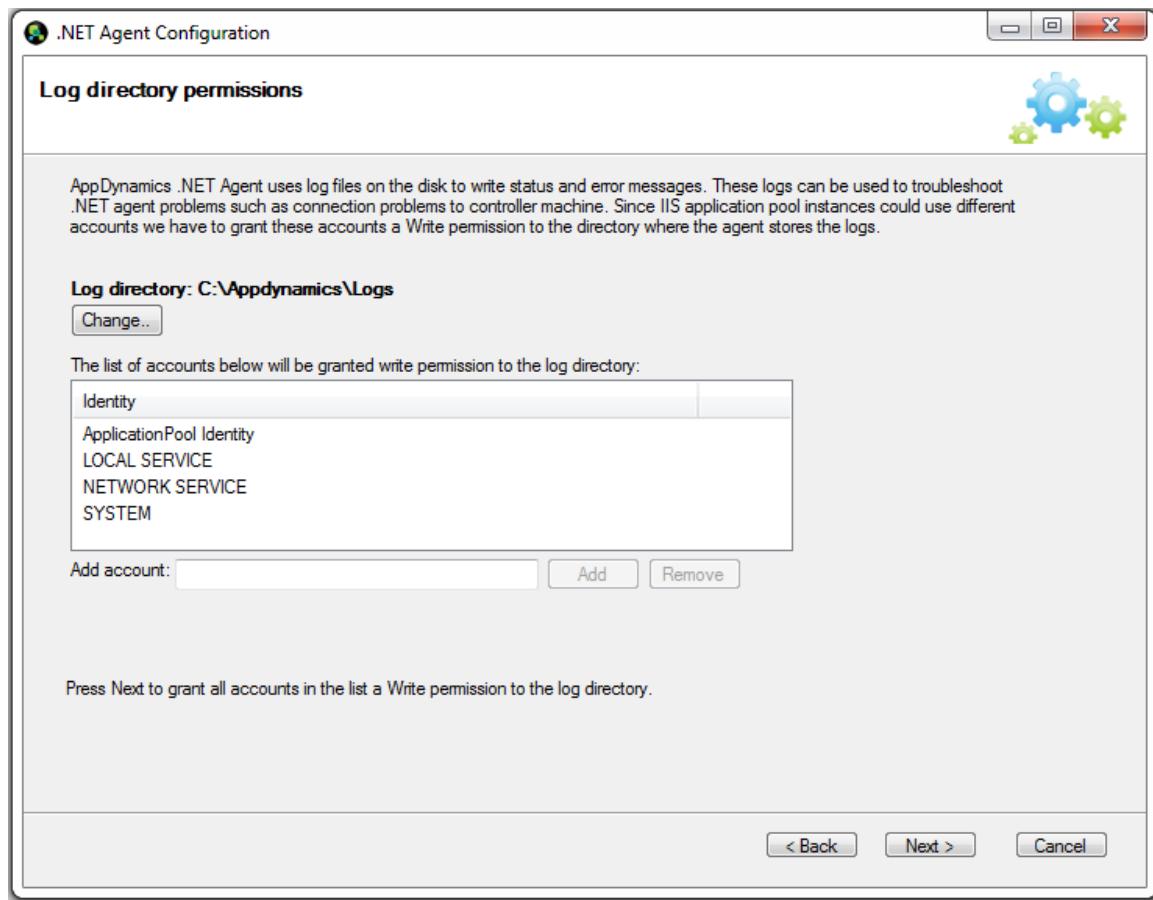
To set up the logs and account permissions

The first screen helps you set up the location of the agent logs and provide the correct account access to the logs.

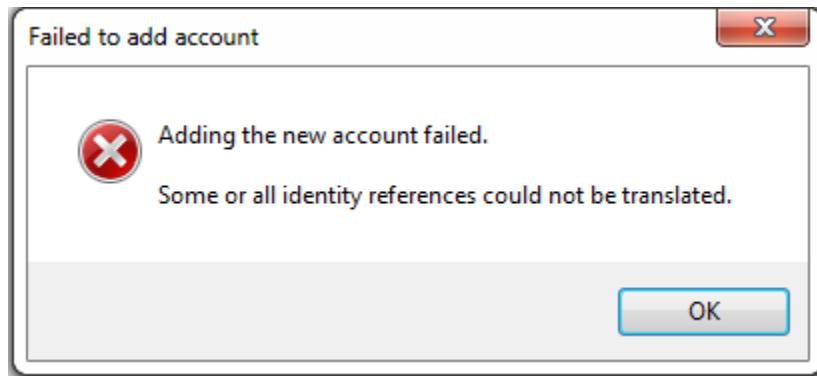
1. If you want to change the default location of the log directory, click **Change** and select a new location.

2. If needed, add accounts for log directory permissions.

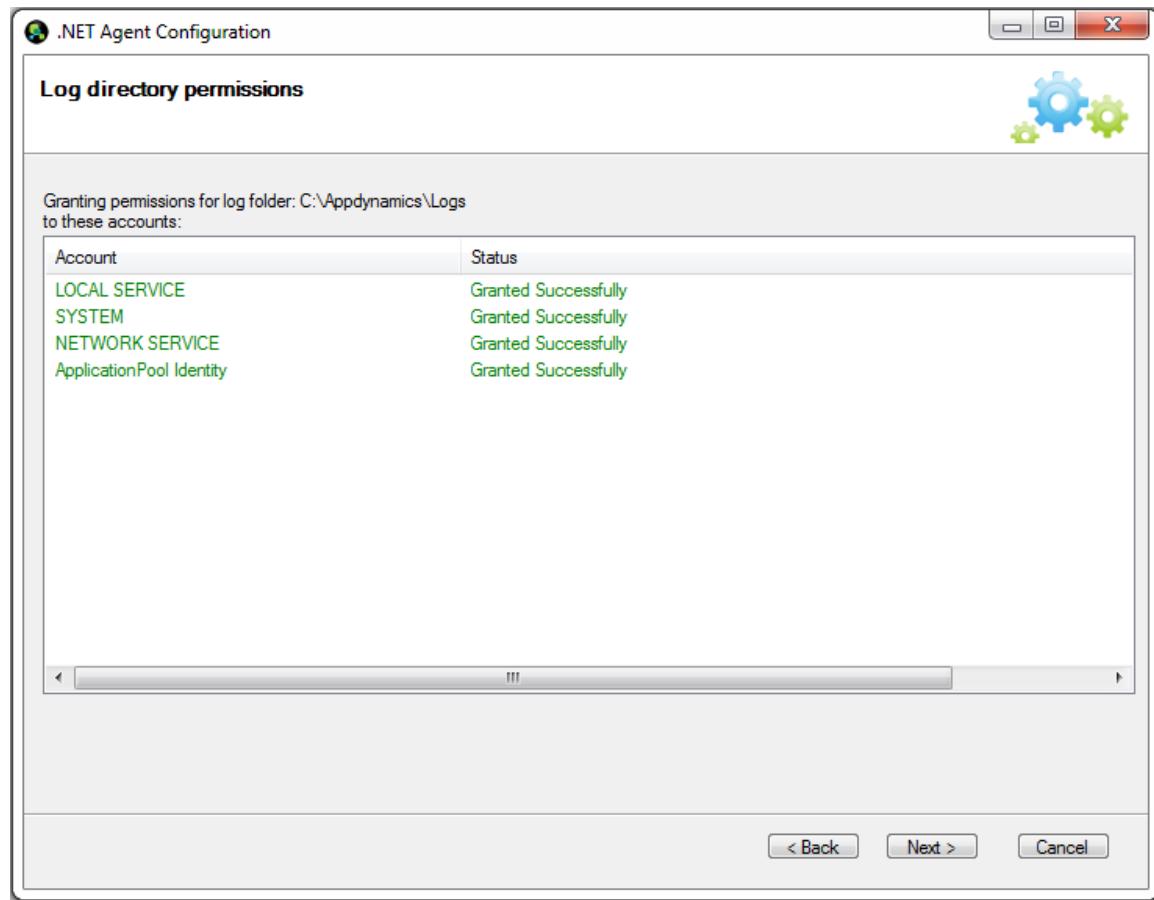
Commonly-used accounts are provided. If your application uses another account, enter the Windows account name that is used by the application and application pools to be monitored. The account name must be valid on the operating system and have permission to write to the log files directory.



Click **Add**. If you get a warning message, make sure that the account is valid on the system.



3. Click **Next** and the wizard confirms the list of accounts.



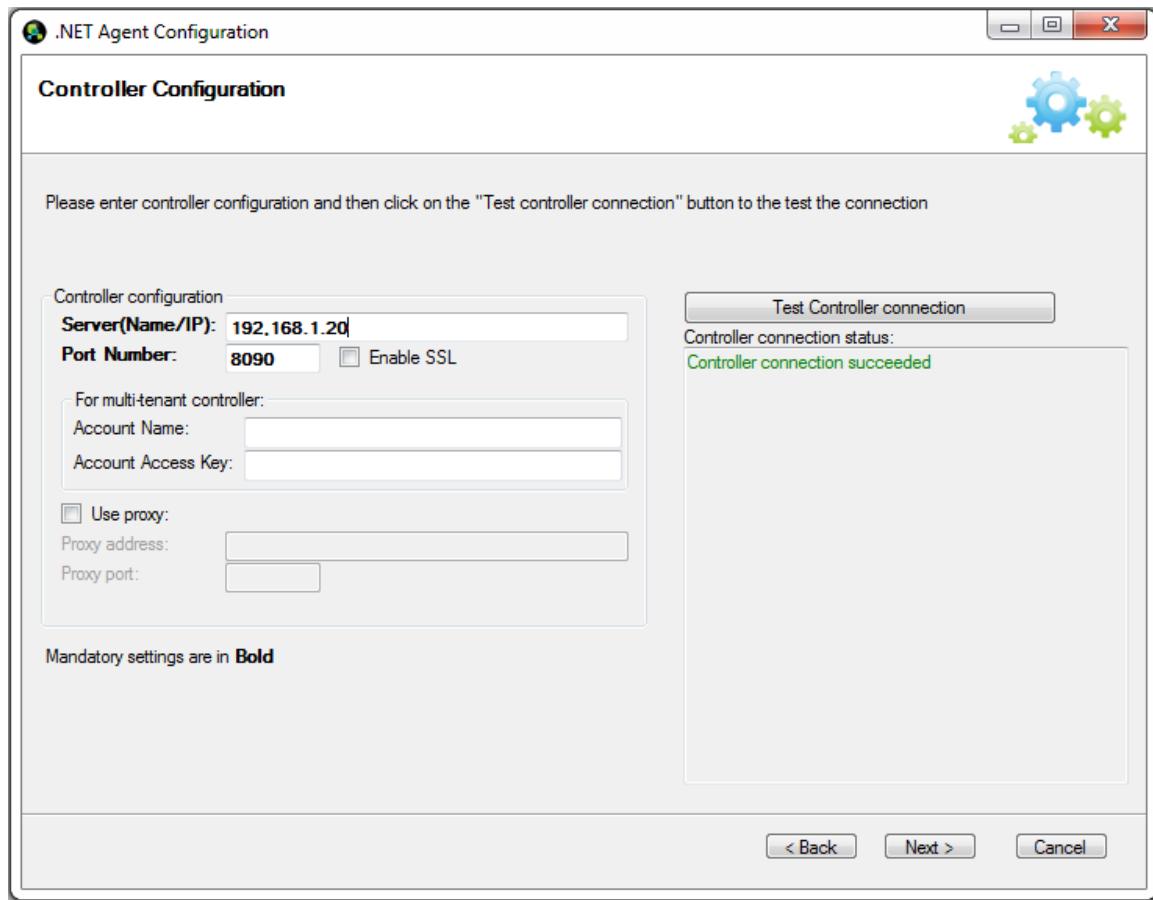
Click **Next**. The Controller Configuration window appears.

To provide Controller configuration information

Enter the Controller access information and credentials.

- For a SaaS Controller, enter the server name or IP, port number, account name, and access key as provided to you by AppDynamics.
- For an on-premise Controller, if you haven't already installed it, cancel this installation and see [Install the Controller](#). Otherwise enter the server name and port number of an existing Controller.
- For a secure connection, click **Enable SSL**.
 Note: The Controller must use a trusted certificate.
- If needed, fill in the proxy information. AppDynamics does not support proxies that require authentication out-of-the-box.

5. Click **Test Controller Connection** to verify the connection.



Click **Next**. The Tier Naming Decision window opens.

To map business applications, tiers, and nodes to your application environment

1. Read about how AppDynamics uses business applications, tiers, and nodes to organize application performance monitoring. In summary:

- A business application is a set of modules and distributed services that together provide business functionality.
- A node is the basic unit of processing that AppDynamics monitors.
- A tier represents a module in an application environment, such as an eCommerce website or Inventory application.

See Logical Model.

2. Decide how to identify and name the tiers. Either AppDynamics will automatically configure tier names, or you can manually configure them.

Use these guidelines for deciding whether to use automatic or manual naming:

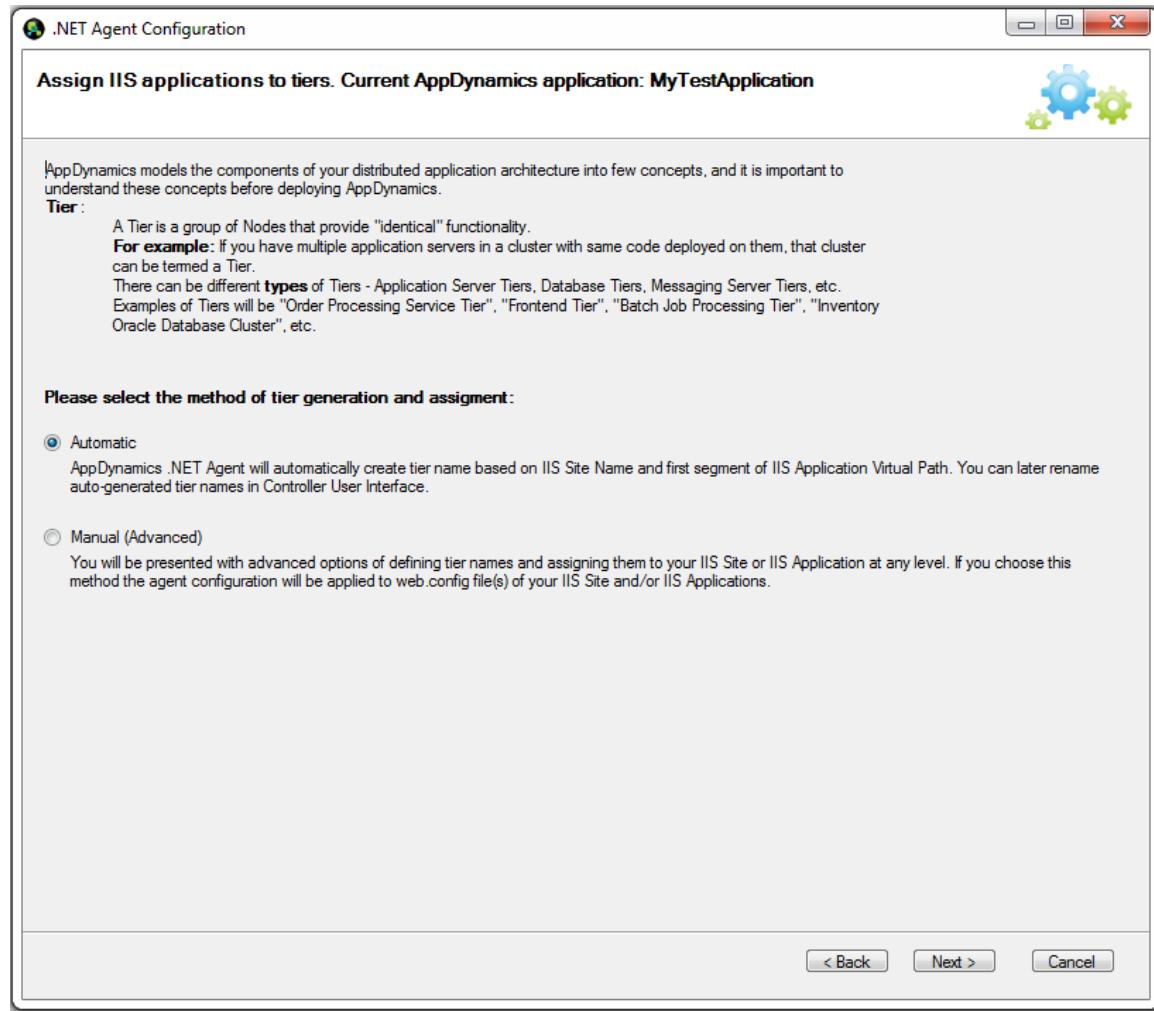
- When all IIS applications on a machine will be instrumented by AppDynamics, choose **automatic**.
- AppDynamics names tiers using this pattern:

IIS_site_name-IIS_application_name

- When only some IIS applications on a machine will be instrumented by AppDynamics, choose **manual**.
- In manual naming, you can name nodes as you like and AppDynamics will update the web.config file.

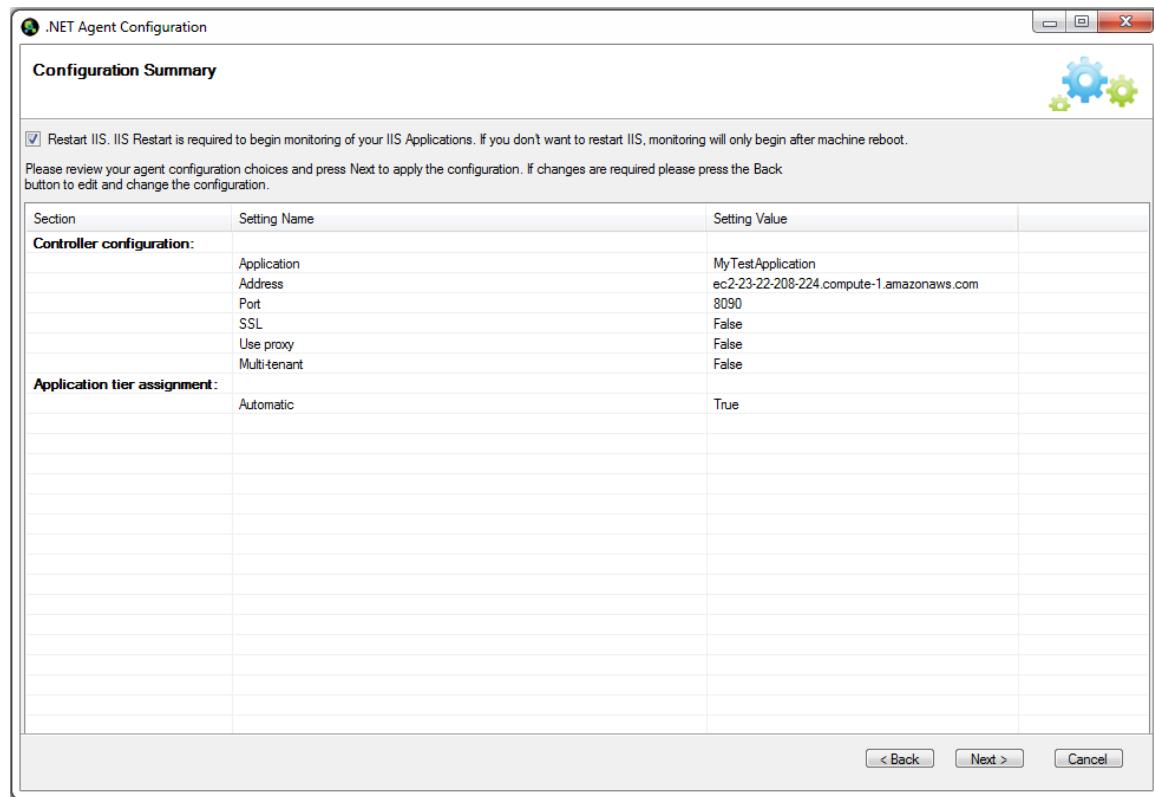
To let AppDynamics automatically name the tiers

1. In the Tier Naming Decision window click **Automatic**.

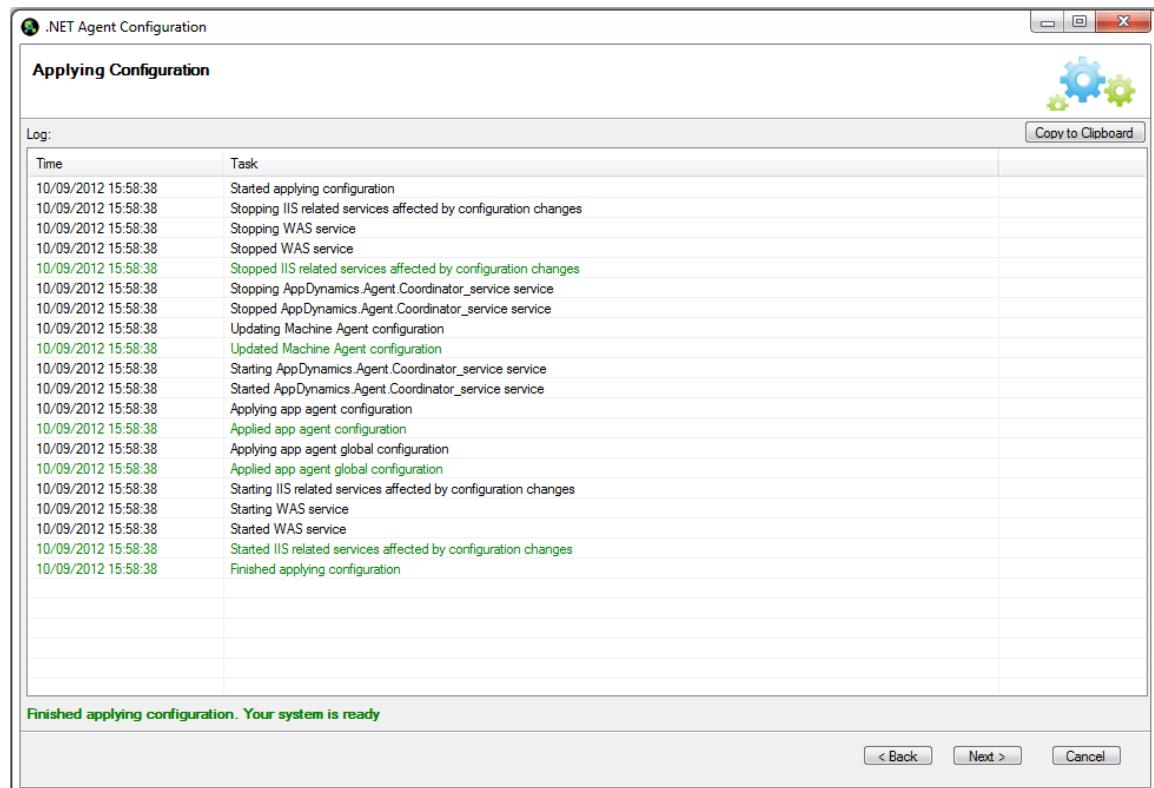


The Configuration Utility summarizes the configuration settings.

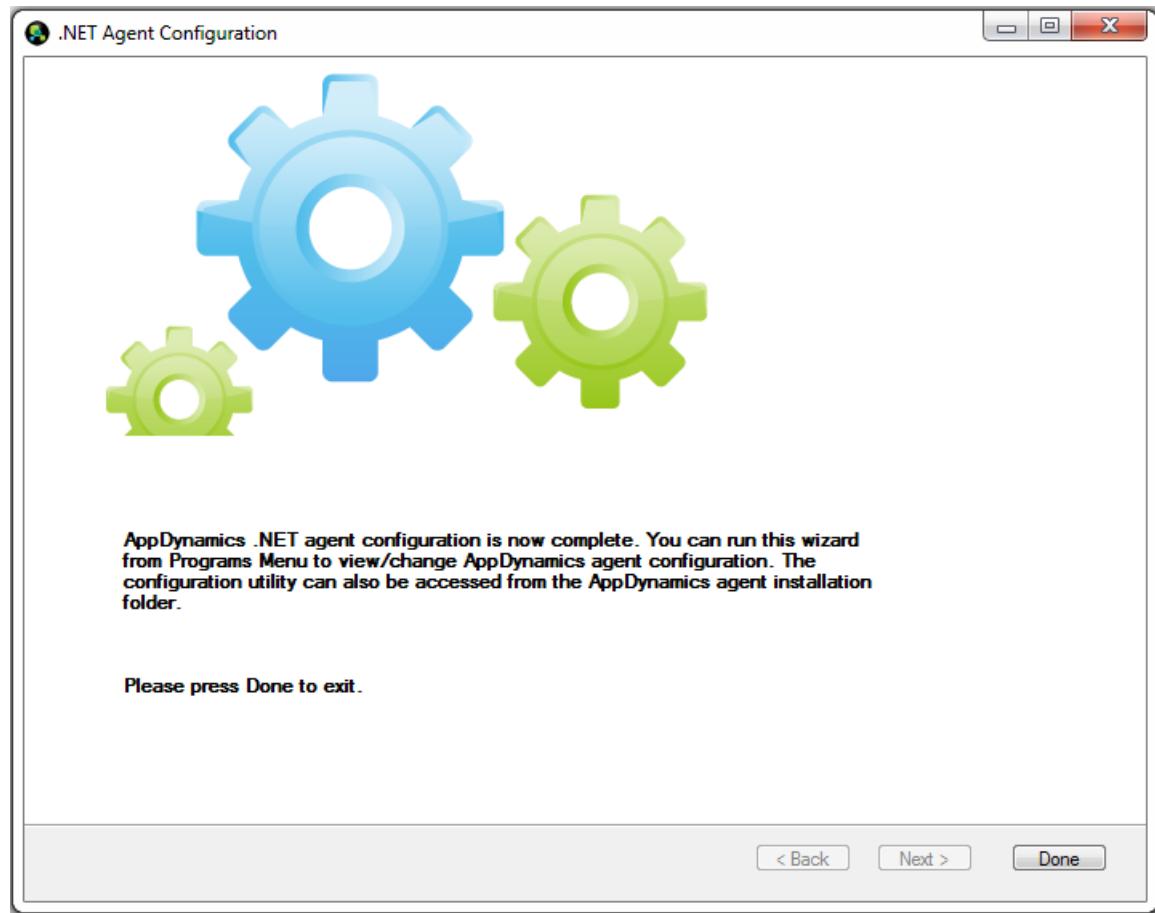
! By default when you click **Next** the Configuration Utility will restart IIS. If you do not want to apply the configuration right away, uncheck the box. The Configuration Utility will save the information and apply it the next time you restart IIS.



2. If you proceed and click **Next**, the Configuration Utility logs its activities, including stopping and restarting IIS, and reports any problems. Review the summary for any issues in red font. Green font indicates the more interesting logged events. The summary shows any Warnings (W) or Errors (E). If you have errors, contact [AppDynamics Support](#).



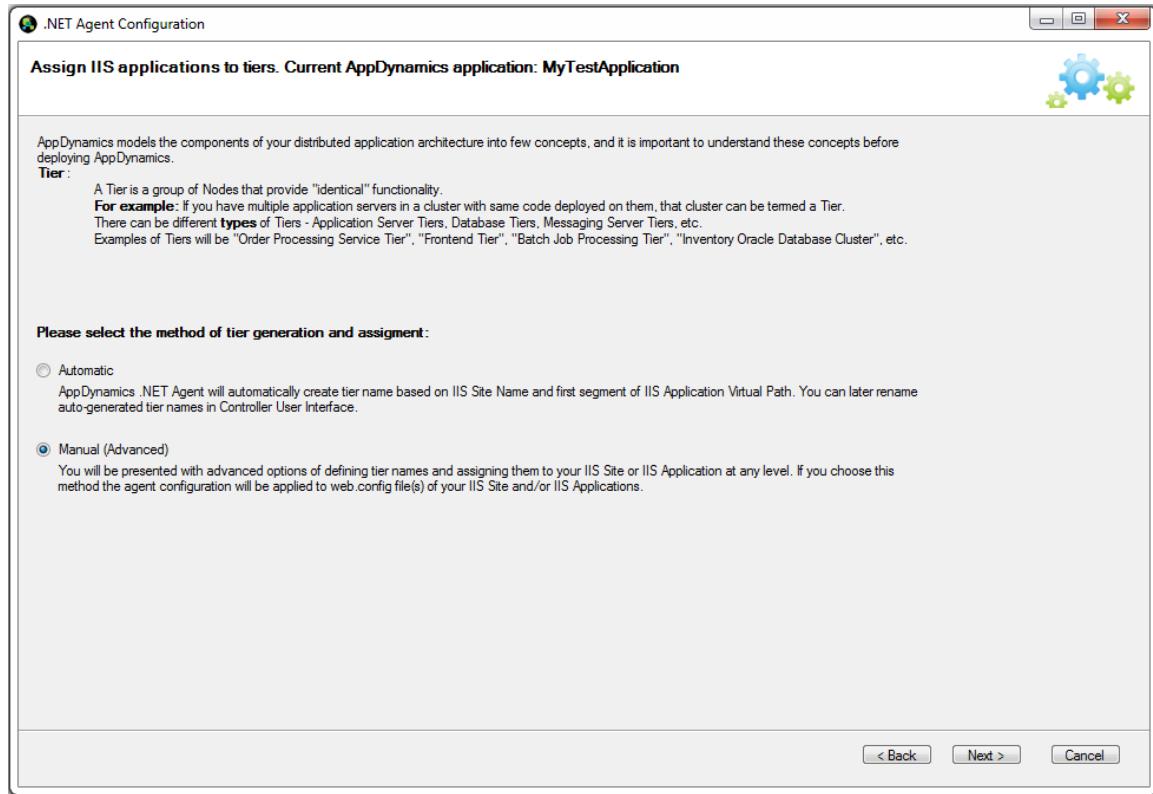
3. When there are no errors, click **Next**.



4. Click **Done** to close the Configuration Utility.

To manually name the tiers

1. In the Tier Naming Decision window click **Manual**.

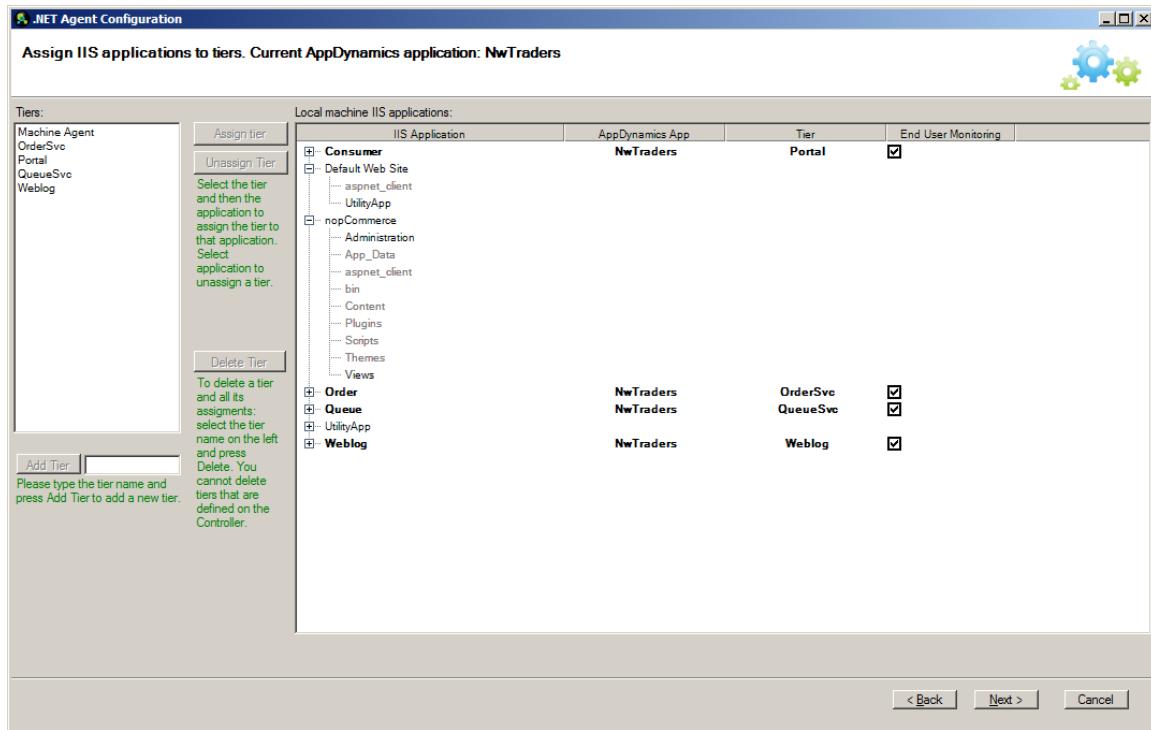


AppDynamics retrieves existing business application information from the Controller and displays it in the left column, and connection status in the right column.

If no business applications are already defined the utility displays an empty list.

2. Click **New Application** to define a new business application. Be careful about spellings and capitalization and note down the exact name.

Note: Do not use ampersands in the business application name; they are not supported at this time.



3. Click **Next**.

4. Assign IIS Applications to AppDynamics tiers.

Select a tier on the right and click a business application on the left. The assigned tier will be highlighted in boldface.

i Note: For large IIS installations, use the Max IIS tree depth pulldown to display all the projects. A large tree depth may take some time to view.

To add tiers that are not already configured, enter a name and click **Add Tier**.

5. When you are done click **Next**. AppDynamics displays a configuration summary.

6. Review the configuration. If you need to make changes click **Back**.

! By default when you click **Next** the Configuration Utility will restart IIS. If you do not want to apply the configuration right away, click *Cancel*. The Configuration Utility will save the information and apply it the next time you restart IIS.

.NET Agent Configuration

Configuration Summary



Note: Agent configuration requires the installation to stop the Internet Information Services (IIS). Once the IIS services are shutdown all Web applications and services running under IIS will no longer be available to users.

Please review your agent configuration choices and press Next to apply the configuration. If changes are required please press the Back button to edit and change the configuration.

Section	Setting Name	Setting Value
Controller configuration:	Application	MyTestApplication
	Address	ec2-23-22-208-224.compute-1.amazonaws.com
	Port	8090
	SSL	False
	Use proxy	False
	Multi-tenant	False
Application tier assignment:	/ADServiceHost.Client	Cleaning configuration
	/BugBash.DynamicCode	Cleaning configuration
	/BugBash.MongoDB	Cleaning configuration
	/CustomerSuccess.SiteA	SiteA, EUM disabled
	/CustomerSuccess.SiteB	SiteB, EUM disabled
	/EUMTests.RegularWeb	Cleaning configuration
	/MorningStar.MatchEntrySite1	Cleaning configuration
	/MorningStar.MatchEntrySite2	Cleaning configuration
	/MorningstarWeb	Cleaning configuration
	/MSOtherWebsite	Cleaning configuration
	/MVC3EUMTest.Web	Cleaning configuration
	/POC.NavmanWireless.NWWeb	Cleaning configuration
	/POC.OpenTable.NetworkIO	Cleaning configuration
	/POC.Proflowers.Web	Cleaning configuration
	/POC.Proflowers.Webservice	Cleaning configuration

[< Back](#) [Next >](#) [Cancel](#)

7. If you proceed and click **Next**, the Configuration Utility logs its activities, including stopping and restarting IIS, and reports any problems. Review the summary for any issues in red font. Green font indicates the more interesting logged events. The summary shows any Warnings (W) or Errors (E). If you have errors, contact [AppDynamics Support](#).

.NET Agent Configuration

Applying Configuration

Log:

Time	Task
10/09/2012 15:52:51	Started applying configuration
10/09/2012 15:52:51	Stopping IIS related services affected by configuration changes
10/09/2012 15:52:51	Stopping WAS service
10/09/2012 15:52:51	Stopped WAS service
10/09/2012 15:52:51	Stopped IIS related services affected by configuration changes
10/09/2012 15:52:51	Stopping AppDynamics.Agent.Coordinator_service service
10/09/2012 15:52:51	Stopped AppDynamics.Agent.Coordinator_service service
10/09/2012 15:52:51	Updating Machine Agent configuration
10/09/2012 15:52:51	Updated Machine Agent configuration
10/09/2012 15:52:51	Starting AppDynamics.Agent.Coordinator_service service
10/09/2012 15:52:51	Started AppDynamics.Agent.Coordinator_service service
10/09/2012 15:52:51	Applying app agent configuration
10/09/2012 15:52:51	Applying app agent configuration to application at: c:\users\petto\documents\visual studio 2010\Projects\CustomerSuccess\CustomerSuccess.SiteA\web.config
10/09/2012 15:52:51	Applied app agent configuration to application at: c:\users\petto\documents\visual studio 2010\Projects\CustomerSuccess\CustomerSuccess.SiteA\web.config
10/09/2012 15:52:51	Applying app agent configuration to application at: c:\users\petto\documents\visual studio 2010\Projects\CustomerSuccess\CustomerSuccess.SiteB\web.config
10/09/2012 15:52:51	Applied app agent configuration to application at: c:\users\petto\documents\visual studio 2010\Projects\CustomerSuccess\CustomerSuccess.SiteB\web.config
10/09/2012 15:52:51	Applied app agent configuration
10/09/2012 15:52:51	Applying app agent global configuration
10/09/2012 15:52:51	Applied app agent global configuration
10/09/2012 15:52:51	Starting IIS related services affected by configuration changes
10/09/2012 15:52:51	Starting WAS service
10/09/2012 15:52:51	Started WAS service
10/09/2012 15:52:51	Started IIS related services affected by configuration changes
10/09/2012 15:52:51	Finished applying configuration

Copy to Clipboard

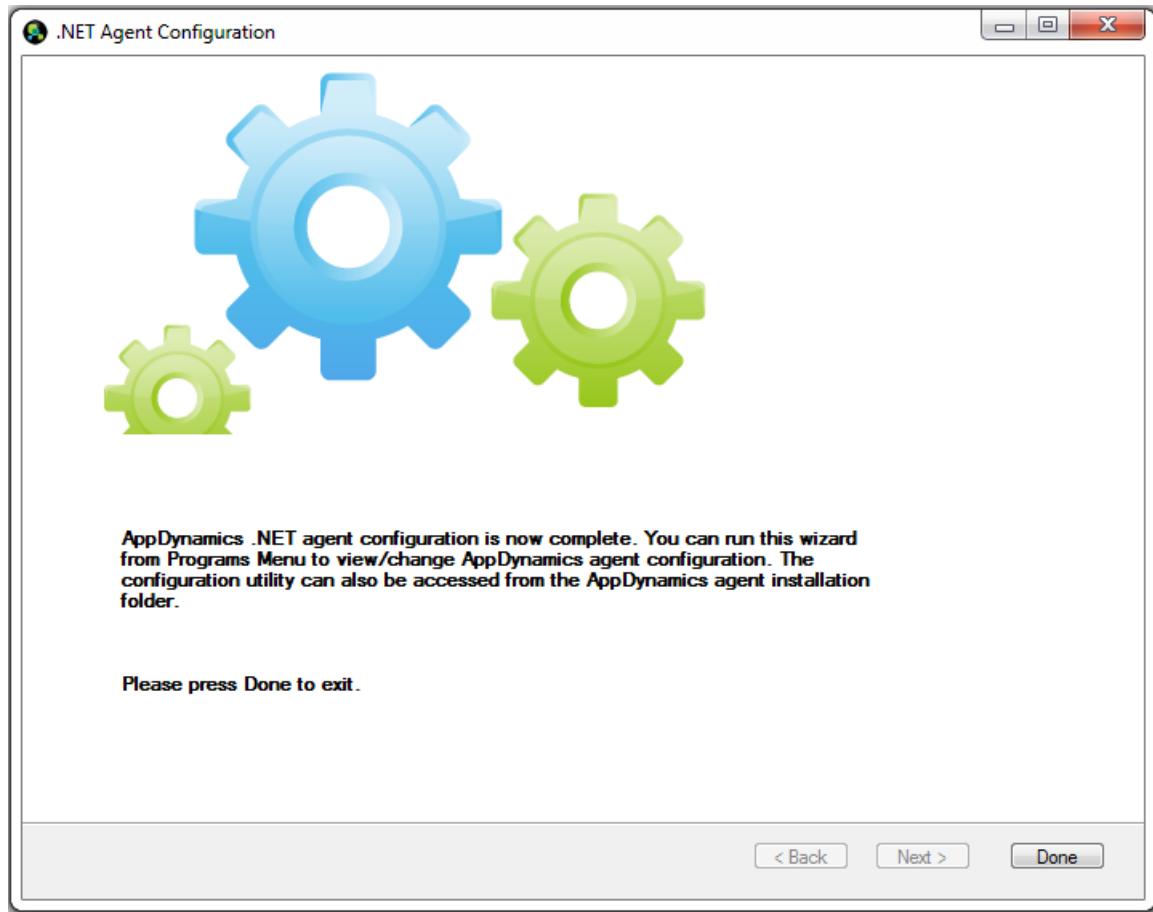
Finished applying configuration. Your system is ready

< Back Next > Cancel

8. Review the configuration log summary.

As it applies the configuration, AppDynamics generates a log of the configuration activities and displays a summary. Review the summary for any issues in red font. Green font indicates the more interesting logged events. The summary shows any Warnings (W) or Errors (E). If you have errors, contact AppDynamics Support.

9. Click **Next**. The wizard completes.



For troubleshooting information see [Troubleshoot App Agent for .NET Installation and Configuration](#).

Using a Configuration File from the Command Line

You can set up a .NET Agent configuration file and run it from the command line. This is useful when you have multiple agents to configure.

To create a configuration file

1. From a command line, start the configuration utility:

```
AppDynamics.Agent.Winston.exe -s <SetupConfigurationFilePath>
```

For example:

```
c:\Program Files\AppDynamics\AppDynamics .NET  
Agent\AppDynamics.Agent.Winston.exe -s  
"c:\temp\configurationSavedSetupConfiguration.xml"
```

The utility starts.

2. Configure the agent as described in the previous sections. The configuration is applied.
In addition, when the configuration completes, AppDynamics creates a setup file.

You use this setup file as an argument to the command line utility.

To run the configuration utility from the command line

1. Start the .NET Configuration Utility from the command line. Change the file path and setup file path as needed.

```
AppDynamics.Agent.Winston.exe -c <SetupConfigurationFilePath>
```

For example:

```
c:\Program Files\AppDynamics\AppDynamics .NET  
Agent\AppDynamics.Agent.Winston.exe -c  
"c:\temp\configurationSavedSetupConfiguration.xml"
```

The utility runs in command line mode; the user interface does not launch.

When it is done it exits the process with 0 for success or any other number for failure.

2. Review the Winston.txt log file in the default logs directory for details.

Learn More

- [Install the App Agent for .NET](#)
- [Naming Conventions for .NET Nodes](#)

Uninstall the App Agent for .NET

This topic describes how to do a complete uninstall of the App Agent for .NET.

 Do not follow these instructions if you are doing an upgrade. If you want to upgrade to a new version see [Upgrade the App Agent for .NET](#).

To completely uninstall the App Agent for .NET

1. Disable the agent by using the Configuration Utility or by manually editing the myapp.exe.config file. Then Reset IIS. Otherwise the Coordinator keeps restarting the Controller service and you cannot remove it.
2. Open services.msc and stop the AppDynamics.Agent.coordinator service.
3. Remove all AppDynamics configurations from the application config file (myapp.exe.config).
4. In the Control Panel, select Add/Remove Program and remove the "AppDynamics .NET Agent".

Learn More

- [Upgrade the App Agent for .NET](#)
- [Install the App Agent for .NET](#)
- [Agent - Controller Compatibility Matrix](#)
- [Release Notes for AppDynamics Pro](#)

Enable the App Agent for .NET for Non-IIS Applications

- [Configuring the App Agent for .NET for Non-IIS Applications](#)
 - Step 1 option A. Configure the AppDynamicsAgent_EnableInProcesses global environment variable
 - Step 1 option B. Create the environment variable when installing the script as a service
 - Step 2. Use the Configuration Utility

- Step 3. Update the application configuration file
- [Learn More](#)

The out-of-the-box App Agent for .NET is enabled only in IIS worker processes. This topic describes how to enable the App Agent for .NET for non-IIS processes by adding an environment variable to be used by the service.

Configuring the App Agent for .NET for Non-IIS Applications

1.A. Set the AppDynamicsAgent_EnableInProcesses global environment variable.

OR

1.B. Create an environment variable for services.

2. Use the Configuration Utility. See [Configure the App Agent for .NET](#).

3. If you used manual naming mode in the Configuration Utility, then update the application config file to add AppDynamics configuration elements.

Step 1 option A. Configure the AppDynamicsAgent_EnableInProcesses global environment variable

1. Set the AppDynamicsAgent_EnableInProcesses global environment variable to the name of your executable file:

```
AppDynamicsAgent_EnableInProcesses=myapp.exe|myotherapp.exe|myservice.exe ....
```

2. Restart the application.

3. Verify that this environment variable is used by the service by using the Processor Explorer application and look at the Environment properties. You can download the [Process Explorer](#) application from Microsoft.

Step 1 option B. Create the environment variable when installing the script as a service

If you are using the Windows Resource ToolKit to install scripts as a service (instsrv.exe and srvany.exe), follow these steps:

1. Use the following key - HKLM\SYSTEM\CurrentControlSet\Services\YourService\Parameters
2. Create a REG_MULTI_SZ called AppEnvironment
3. Add the variable:

```
AppDynamicsAgent_EnableInProcesses=myservice.exe
```

The script service will have its own environment variables.

Step 2. Use the Configuration Utility

See [Configure the App Agent for .NET](#).

If you use manual naming mode in the Configuration Utility, then go to Step 3.

Step 3. Update the application configuration file

If you used manual naming mode in the Configuration Utility in Step 2, update the myapp.exe.config file for the application to be monitored. If you used automatic naming mode you can skip Step 3.

You can copy the XML from the agent_install_directory/Samples/Configuration/web.config file.

1. In the <configSections> element in the myapp.exe.config file, add the following section:

```

<configSections>
    <section name="AppDynamicsAgent"
        type="AppDynamics.Agent.Config.Section, AppDynamics.Agent,
        Version=1.0.0.0, Culture=neutral, PublicKeyToken=3f604d9e4f8e4985"
        allowLocation="true" allowDefinition="Everywhere"/>
</configSections>

```

2. Add the following <AppDynamicsAgent> section:

```

<AppDynamicsAgent>
    <controller-info>
        <controller-host>host</controller-host>
        <controller-port>8090</controller-port>
        <controller-ssl-enabled>false</controller-ssl-enabled>

        <!-- If the controller is running in multi-tenant mode, specify the account name
            and access key for this agent to authenticate with the controller.
            If the controller is running in single-tenant mode, there is no need to
            specify these values. -->
        <account-name></account-name>
        <account-access-key></account-access-key>

        <!-- For Auto Agent Registration specify the application name, tier name,
            and optionally, node name. If the application and/or tier does not
            exist already it will be created. -->
        <application-name>Test .NET Application</application-name>
        <tier-name>Portal</tier-name>

        <!-- Recommended to leave it empty if your application tier node has more than one
            instance,
            as in case of IIS applications running in web garden (app pool) or in the case
            when nodes
            of the same tier run on different machines.
            If unsure - leave it empty, It will be auto-generated-->
        <node-name></node-name>
        <proxy disable="true">
            <host></host>
            <port></port>
        </proxy>
    </controller-info>
    <app-agent-configuration>
        <configuration-properties />
        <agent-services>
            <agent-service name="AgentOperationService" enabled="true">
                <configuration-properties />
                <configuration />
            </agent-service>
            <agent-service name='BCIEngine' enabled='true'>
                <configuration-properties>
                </configuration-properties>
                <configuration>
                </configuration>
            </agent-service>
            <agent-service name='CLRMetricsService' enabled='true'>
                <configuration-properties>
                </configuration-properties>
                <configuration>
                    <perf-counters>
                        <!-- In this section you can add CLR related performance counters
                            The list of possible .NET performance counters is documented here
                            Performance Counters in the .NET Framework
                            http://msdn.microsoft.com/en-us/library/w8f5kw2e%28v=VS.80%29.aspx -->

```

```
<!-- Example: <counter cat=".NET CLR Exceptions" name="# of Exceps Thrown"/> -->
</perf-counters>
</configuration>
</agent-service>
<agent-service name='SnapshotService' enabled='true'>
    <service-dependencies>BCIEngine</service-dependencies>
    <configuration-properties>
        </configuration-properties>
    </agent-service>
<agent-service name='TransactionMonitoringService' enabled='true'>
    <service-dependencies>BCIEngine, SnapshotService</service-dependencies>
    <configuration-properties>
        </configuration-properties>
    </agent-service>
</agent-services>
</app-agent-configuration>
```

```
</AppDynamicsAgent>
```

3. Configure how the App Agent for .NET connects to the Controller

In the myapp.exe.config file, modify the following items in the <controller-info> element:

	Required	Default
<controller-host>	Yes	None
<controller-port>	Yes	For On-premise Controller installations: By default, port 8090 is used for HTTP and 8181 is used for HTTPS communication. For SaaS Controller service: By default, port 80 is used for HTTP and 443 is used for HTTPS communication.

To configure the .NET Agent to use SSL see [App Agent for .NET Configuration Properties](#).

4. (for Multi-tenant mode or SaaS installations only) Configure the .NET Agent account information

This step is required only when the AppDynamics Controller is configured in multi-tenant mode (the Controller is configured with multiple user accounts) or when you [Use a SaaS Controller](#). Skip this step if you are using single-tenant mode, which is the default for the on-premise installation.

In the myapp.exe.config, specify the properties for Account Name and Account Key. This information is provided in the Welcome email from AppDynamics Support Team.

	Required	Default
<account-name>	Required only if your Controller is configured for multi-tenant mode or your Controller is hosted.	None.
<account-access-key>	Required only if your Controller is configured for multi-tenant mode or your Controller is hosted.	None.

5. Configure how the Agent Application and Tier are identified

In the myapp.exe.config file, specify to which Application and Tier this .NET Agent belongs. For more information see [Mapping Application Services to the AppDynamics Model](#).

	Required	Default
<application-name>	Yes	None.
<tier-name>	Yes	None.

6. Restart the application or service.

Learn More

- [Mapping Application Services to the AppDynamics Model](#)
- [Configure the App Agent for .NET](#)
- [Install the App Agent for .NET](#)

Troubleshoot App Agent for .NET Installation and Configuration

- Troubleshooting Agent-Controller Configuration
 - To verify that the App Agent for .NET is reporting to the Controller
- Checklist for Troubleshooting the App Agent for .NET Installation
- Troubleshooting .NET Application Server Agent Issues

- Troubleshooting Agent Installation
 - Verify Administrative privileges
 - To verify that COM+ Services are enabled
 - To get a log when the .NET Agent installer fails
 - The .NET Agent installation fails when there are other APM products
 - To remove associated "Environment" subkey for W2SVC and WAS services in the registry:
- Troubleshooting Configuration Errors
 - Verify the Machine Agent configuration
 - Verify the configuration in the web.config file
- Troubleshooting Log Issues
 - To verify that the .NET Agent directory has the correct permissions
 - Allowed groups for different IIS versions
- Learn More

This topic covers how to solve installation and configuration problems for the App Agent for .NET.

Troubleshooting Agent-Controller Configuration

To verify that the App Agent for .NET is reporting to the Controller

Use the AppDynamics UI to verify that the agent is able to connect to the Controller.

1. In a browser open:

```
http://<controller-host>:<controller-port>/controller
```

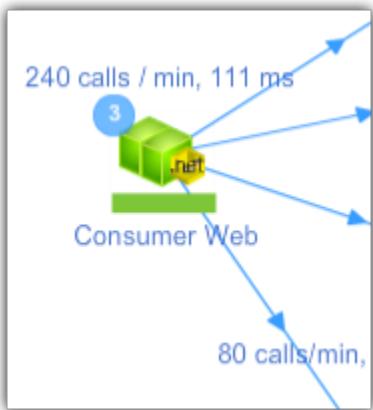
2. Log in to the AppDynamics UI.

3. Select the application to open the Application Dashboard.

4. In the left navigation panel click **Servers > App Servers** and open the Health tab.

The Health tab lists the tiers, their nodes, and App Agent Status. When an agent successfully reports to the Controller, you see an "up" arrow symbol. For details see [Verify App Agent - Controller Communication](#).

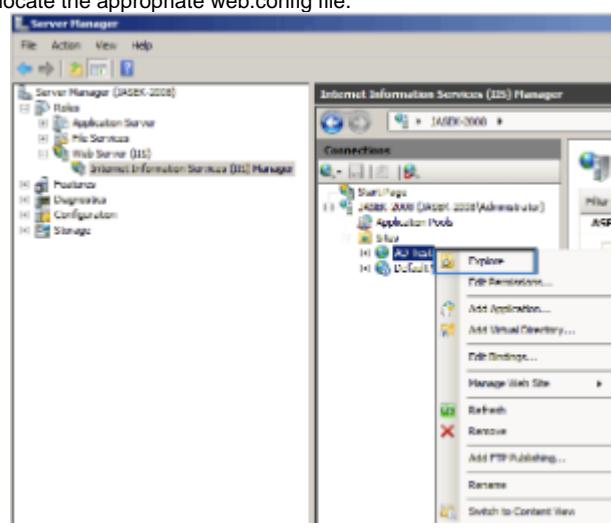
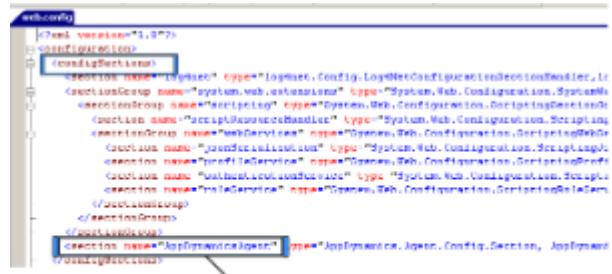
- When deploying multiple agents for the same tier, determine whether you get the correct number of nodes reporting into the same tier.
- After sending a request to your web application, data should appear on the AppDynamics UI. The agents should be displayed in the Application Flow Map of the Application Dashboard.



If no data appears after a few minutes:

- Verify that the Agent is writing its log files to \Appdynamics\Logs\AgentLog.txt.
 - If the log file exists, open it and review it for errors.
 - If no log file has been written, run the Windows Event Viewer and see the application messages.
 - If there are no AppDynamics event messages, look for messages from the .NET Runtime.

Checklist for Troubleshooting the App Agent for .NET Installation

	Item	Notes
	Run the installer as Administrator.	Verify Administrative privileges
	Verify that COM+ is enabled.	To verify that COM\+\+ Services are enabled
	Verify permissions for Agent directory.	To verify that the .NET App Server Agent directory has the correct permissions based on the site's application pool identity.
	Verify that the Agent is compatible with the Controller.	Agent - Controller Compatibility Matrix
	Ensure correct settings in the application's web.config file.	<p>Update the web.config file to include the App Agent for .NET Configuration Properties.</p> <p>Tip: Right click on "Explore" while viewing the IIS website will locate the appropriate web.config file.</p>  <p>Click to enlarge.</p>
	Ensure that the <configSections> section in your web.config file is the first child of "<configuration>" section.	 <p>This section must be explicitly added to the existing web.config file. Alternatively, you can use the sample web.config file.</p> <p>Click to enlarge.</p>

Troubleshooting .NET Application Server Agent Issues

Troubleshooting Agent Installation

If the Agent installation is failing, check the following configurations in your environment:

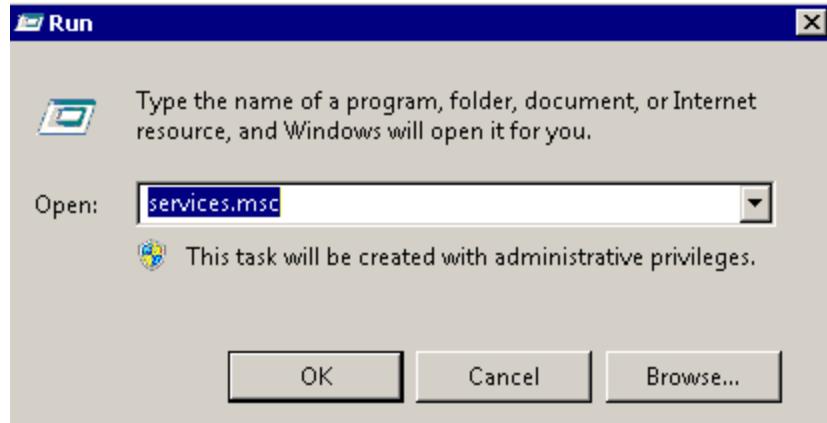
Verify Administrative privileges

- Ensure that you have the administrative privileges when you launch the installer.

To verify that COM+ Services are enabled

If you receive an error about the Appdynamics.Agent.Coordinator_service, ensure that the COM+ applications in your system are enabled.

1. From the Windows Start Menu, click **Run**.
2. Enter "services.msc" and click on the "OK" button. Windows opens the Services Console.



3. Enable the COM+ services (if not enabled).

Name	Description	Status
AppD_AutomationService	This service hosts a remoting server and provides methods for automation of performance metrics.	Started
Application Experience	Processes application compatibility cache requests for applications as they are launched.	Started
Application Host Helper Service	Provides administrative services for IIS, for example configuration history and Application Pool.	Started
Application Identity	Determines and verifies the identity of an application. Disabling this service will prevent interactive logons.	Started
Application Information	Facilitates the running of interactive applications with additional administrative privileges.	Started
Application Layer Gateway Service	Provides support for 3rd party protocol plug-ins for Internet Connection Sharing.	Started
Application Management	Processes installation, removal, and enumeration requests for software deployed through the Windows Update infrastructure.	Started
ASP.NET State Service	Provides support for out-of-process session states for ASP.NET. If this service is disabled, session state will be stored in memory.	Started
Background Intelligent Transfer Service	Transfers files in the background using idle network bandwidth. If the service is disabled, file transfers will be limited to foreground transfers.	Started
Base Filtering Engine	The Base Filtering Engine (BFE) is a service that manages firewall and Internet Protocol (IP) security (IPSec) policies.	Started
Certificate Propagation	Copies user certificates and root certificates from smart cards into the current user's certificate store.	Started
CNG Key Isolation	The CNG key isolation service is hosted in the LSA process. The service provides key isolation for cryptographic operations.	Started
COM+ Event System	Supports System Event Notification Service (SENS), which provides automatic distribution of event notifications.	Started
COM+ System Application	Manages the configuration and tracking of Component Object Model (COM)-based components.	Started
Computer Browser	Maintains an updated list of computers on the network and supplies this list to computers that are joining the network.	Started
Credential Manager	Provides secure storage and retrieval of credentials to users, applications and security systems.	Started
Cryptographic Services	Provides four management services: Catalog Database Service, which confirms the existence of a catalog.	Started
CYGWIN sshd	The CYGWIN sshd service launches COM and DCOM servers in response to object requests.	Started
DCOM Server Process Launcher	The DCOMLAUNCH service launches COM and DCOM servers in response to object requests.	Started

To get a log when the .NET Agent installer fails

If installer fails, use the command line utility to launch the installer.

```
msiexec /i $Path_to_the_MSI_File /l*v verbose.log
```

A verbose log for the .NET Agent is created at the same location where you saved the installer file. Send this log to the [AppDynamics Support Team](#).

The .NET Agent installation fails when there are other APM products

The .NET Agent installation may fail if there are other Application Performance Management (APM) products installed in the same managed environment. Remove the associated "Environment" subkey for certain services for the installed APM products.

To remove associated "Environment" subkey for W2SVC and WAS services in the registry:

1. Run Regedit or regedt32.
2. In regedit.exe, locate the following registry keys:
 HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\W3SVC
 HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\WAS
3. Expand the keys.
4. Modify the **Environment** subkey to delete the following values:

```
COMPLUS_ProfAPI_ProfilerCompatibilitySetting=EnableV2Profiler
COR_ENABLE_PROFILING=1
COR_PROFILER= {a GUID}
```

5. Restart the services. For more details see [How to restart the W2SVC and WAS services?](#).

Troubleshooting Configuration Errors

Verify the Machine Agent configuration

Ensure that the application.config file for the Machine Agent is configured correctly. See [Configure the .NET Machine Agent](#).

Verify the configuration in the web.config file

- Ensure that you have correctly configured the web.config file for AppDynamics Agent.
- If you are using your existing web.config file, ensure that the <configSections> section in your web.config file is the first child of "<configuration>" section (see the screenshot given below):



- If you are using an existing "web.config" file and receive an error related to missing "<section>" element, ensure that you have correctly configured the .NET Agent in the web.config file. For details see [Install the App Agent for .NET](#).

Troubleshooting Log Issues

The .NET App Server Agent logs are located at C:\AppDynamics\logs. Logs will not be created if the .NET Agent directory does not have correct permissions.

⚠️ IMPORTANT: If the AppDynamics directory is not being created, contact AppDynamics Support Team.

To verify that the .NET Agent directory has the correct permissions

1. Click IIS -> Application pools.
AppDynamics displays the list of application pools for your machine.

The screenshot shows the 'Application Pools' section of the IIS Manager. The left sidebar shows 'Start Page', 'IP-0A4D46B0 (IP-0A4D46B0)\Admin', 'Application Pools' (which is selected), and 'Sites'. The main area has a title 'Application Pools' with a description: 'This page lets you view and manage the list of application pools on the server. Application pools are associated with worker processes, contain one or more applications, and provide isolation among different applications.' Below is a table with the following data:

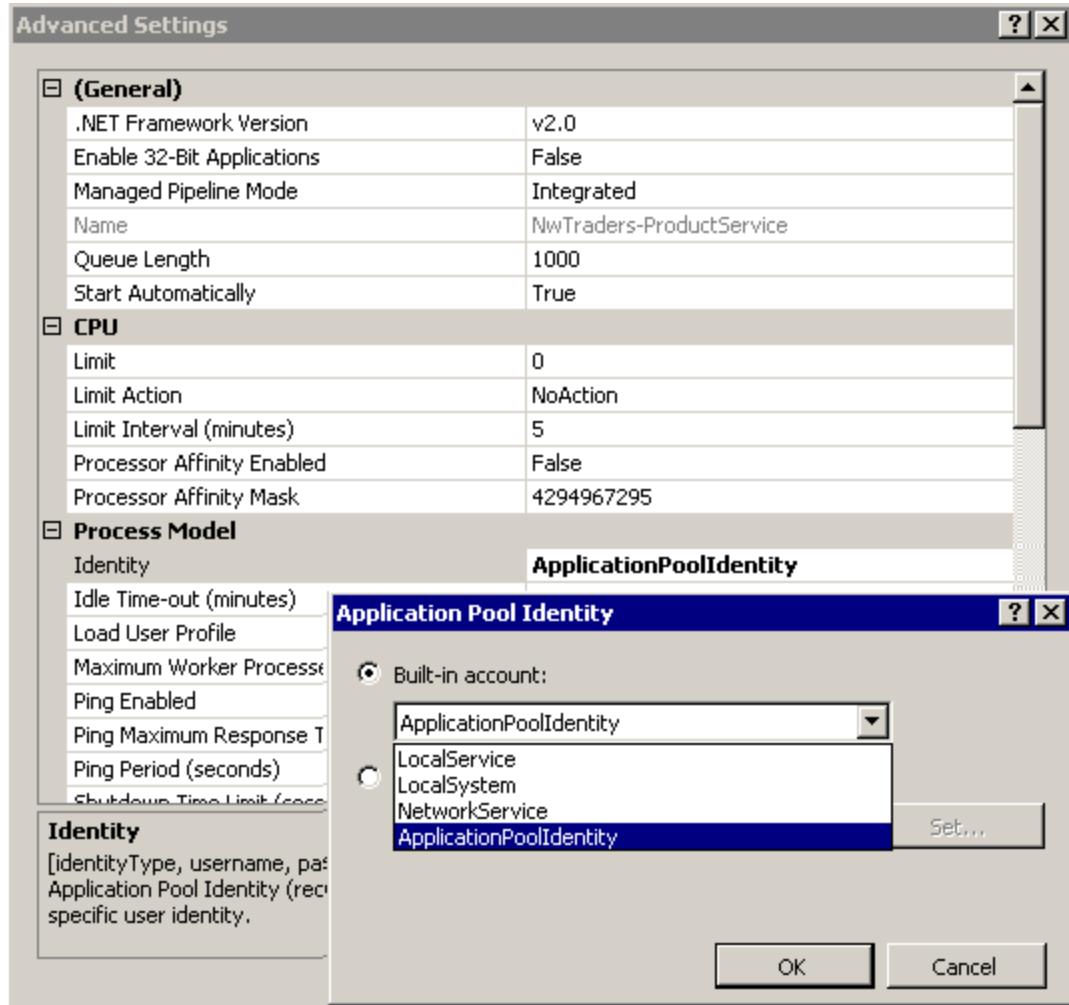
Name	Status	.NET Frame...	Managed I...
Classic .NET App...	Started	v2.0	Classic
DefaultAppPool	Started	v2.0	Integrated
NwTraders-Admin	Started	v2.0	Integrated
NwTraders-Cons...	Started	v2.0	Integrated

2. Right-click on a particular application pool.

3. Click Advanced Settings.

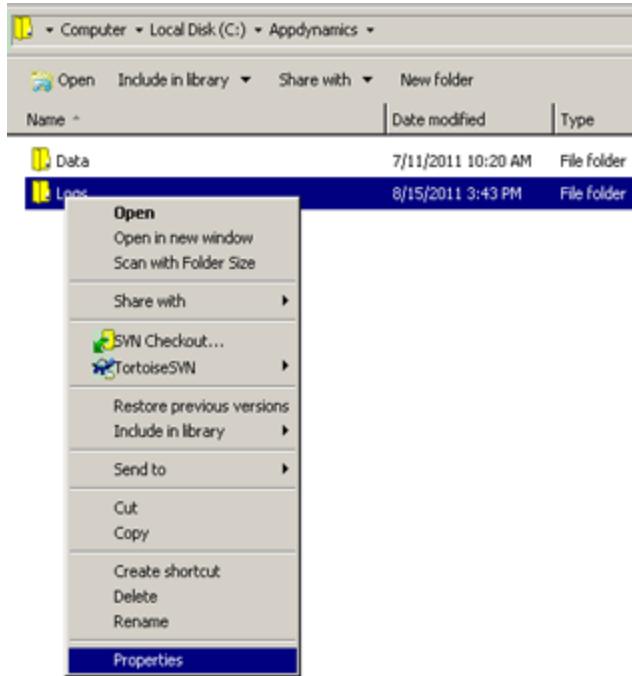
The screenshot shows a context menu for an application pool. The menu items are: 'Add Application Pool...', 'Set Application Pool Defaults...', 'Start', 'Stop', 'Recycle...', 'Basic Settings...', 'Recycling...', 'Advanced Settings...', 'Rename', 'Remove', 'View Applications', 'Help', and 'Online Help'. The 'Advanced Settings...' option is highlighted with a blue selection bar.

AppDynamics displays the Application Pool Identity for that application.

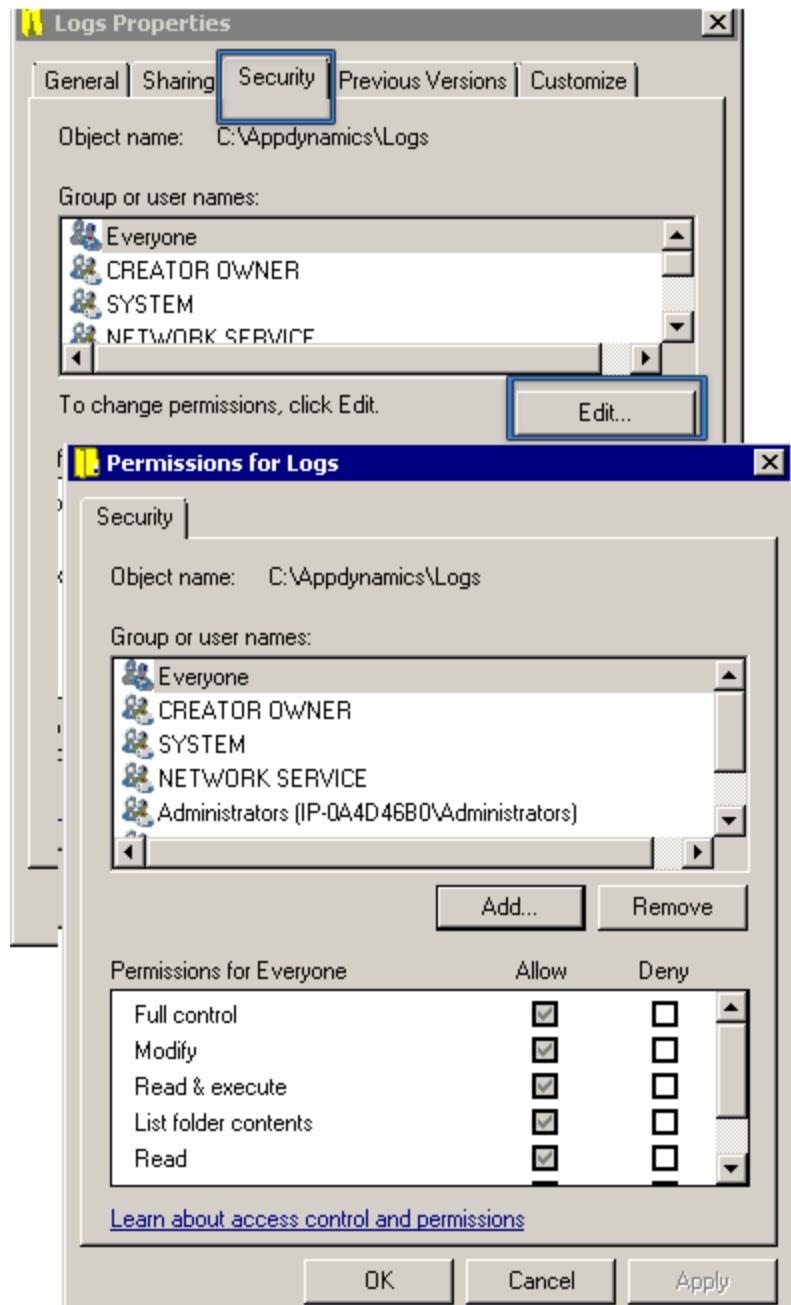


4. Ensure that your Agent Directory also has the same permissions as your site's application pools.

- Navigate to AppDynamics .NET App Server Agent directory location.
- Right-click on the "logs" directory for the App Server Agent and select **Properties**.

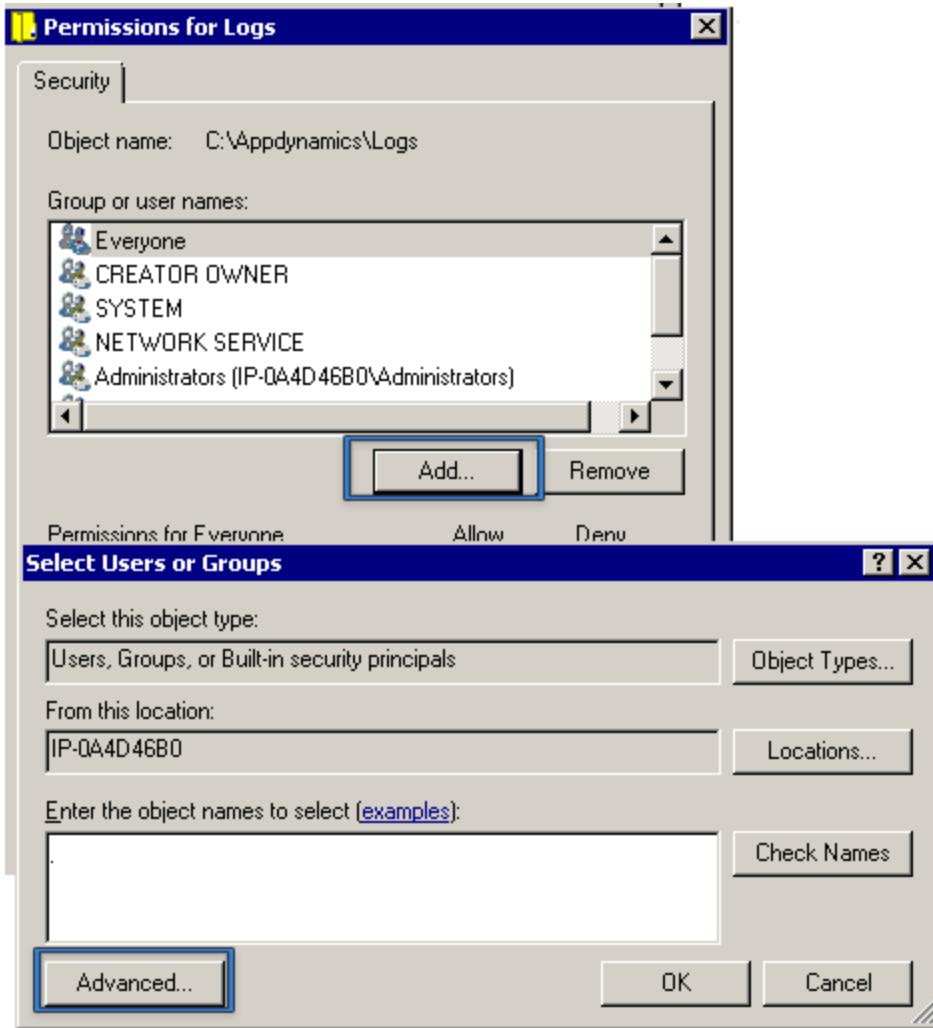


- Click the **Security** tab and verify that the same Application Pool Identity is specified for the .NET Agent directory.



If the Agent's logs directory does not have the requisite permissions:

1. In the Security tab, click **Edit**.
2. Click **Add** to add new permissions to the Agent directory.
3. Click **Advanced**.



4. Click **Find Now** to find all the users, groups, or built-in security principals on your machine.
5. Select the required group (see the information given below for "**Allowed groups**") from the list and click **OK**.
6. Provide the read and write permissions for the selected user/group/security principal to the Agent directory and click **OK**.
7. Click **Apply**.

Allowed groups for different IIS versions

For IIS v6.x, following settings are applicable for Application Pool Identities:

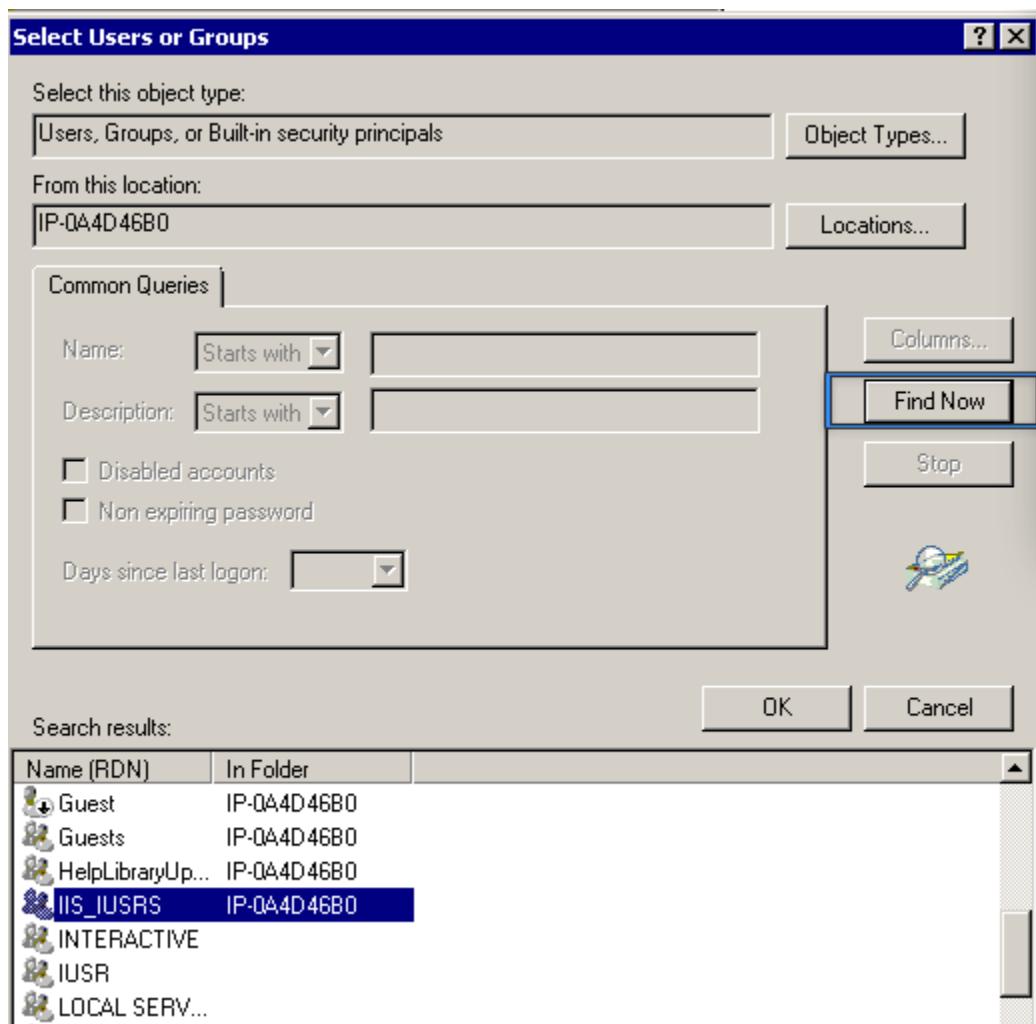
Application Pool Identity	Permission Level
LocalService	LOCAL SERVICE
LocalSystem	SYSTEM
NetworkService	NETWORK SERVICE
Custom Account	Provide the exact name of the account.

For IIS v7.0 and later, following settings are applicable for Application Pool Identities:

Application Pool Identity	Permission Level

LocalService	LOCAL SERVICE
LocalSystem	SYSTEM
NetworkService	NETWORK SERVICE
ApplicationPoolIdentity	Provide the group level permissions for IIS_IUSRS Group (see the screenshot given below).
Custom Account	Provide the exact name of the account.

For example, if your application has the identity "ApplicationPoolIdentity", you must provide the permissions for "IIS_IUSRS" group to your Agent's directory.



Learn More

- [Install the App Agent for .NET](#)

How to Configure the .NET Agent Manually

This topic was written by guest author Robert Petty of AppDynamics Customer Success. Thanks Robert!

Introduction

For instrumentation to occur you must define the configuration for the AppDynamics .NET Agent to properly be hooked inside the host (w3wp) process. In the case of IIS applications, this occurs via adding the appropriate agent configuration to the web.config. This configuration can reside higher in the hierarchical chain, but this how-to will show how to directly manipulate the IIS application at the same level in which it resides (the local web.config). This article assumes AppDynamics .NET Agent has already been successfully installed.

Web.config Explanation

Regardless of how the original web.config is created there are two areas that are required for agent instrumentation. The two sections required are the section definition (which is a child of the <configSections> element. If it is not present you must add this parent element. Here it is in its entirety:

```
<configSections>
  <section name="AppDynamicsAgent"
    type="AppDynamics.Agent.Config.Section, AppDynamics.Agent,
    Version=1.0.0.0, Culture=neutral, PublicKeyToken=3f604d9e4f8e4985"
    allowLocation="true"      allowDefinition="Everywhere"    />
</configSections>
```

The <section> element will alert the ASP.NET process, as it parses the web.config structure, from machine.config to local web.config, which section to use for the base .NET AppDynamics Agent configuration. This is defined in the other relevant element, which resides at the same level as <configSections>, called <AppDynamicsAgent>:

```
<AppDynamicsAgent>
  <controller-info>
    <controller-host>localhost</controller-host>
    <controller-port>8090</controller-port>
    <controller-ssl-enabled>false</controller-ssl-enabled>

    <!-- If the controller is running in multi-tenant mode, specify the account name
        and access key for this agent to authenticate with the controller.
        If the controller is running in single-tenant mode, there is no need to
        specify these values. -->
    <account-name></account-name>
    <account-access-key></account-access-key>

    <!-- For Auto Agent Registration specify the application name, tier name,
        and optionally, node name. If the application and/or tier does not
        exist already it will be created. -->
    <application-name>Test .NET Application</application-name>
    <tier-name>Portal</tier-name>

    <!-- Recommended to leave it empty if your application tier node has more than one instance,
        as in case of IIS applications running in web garden (app pool) or in the case when nodes
        of the same tier run on different machines.
        If unsure - leave it empty, It will be auto-generated-->
    <node-name></node-name>
    <proxy disable="true">
      <host></host>
      <port></port>
    </proxy>
  </controller-info>
  <app-agent-configuration>
    <configuration-properties>
      <property name='agent-overwrite' value='false'/>
    </configuration-properties>
    <agent-services>
      <agent-service name='BCIEngine' enabled='true'>
        <configuration-properties>
        </configuration-properties>
        <configuration>
        </configuration>
      </agent-service>
      <agent-service name='CLRMetricsService' enabled='true'>
        <configuration-properties>
          <property name='update-interval-in-seconds' value='60'/>
        </configuration-properties>
        <configuration>
          <perf-counters>
            <!-- In this section you can add CLR related performance counters
```

The list of possible .NET performance counters is documented here
 Performance Counters in the .NET Framework
<http://msdn.microsoft.com/en-us/library/w8f5kw2e%28v=VS.80%29.aspx> -->

```
<!- Example: <counter cat=".NET CLR Exceptions" name="# of Exceps Thrown">>
</perf-counters>
</configuration>
</agent-service>
<agent-service name='SnapshotService' enabled='true'>
<service-dependencies>BCIEngine</service-dependencies>
<configuration-properties>
</configuration-properties>
</agent-service>
<agent-service name='TransactionMonitoringService' enabled='true'>
<service-dependencies>BCIEngine,SnapshotService</service-dependencies>
<configuration-properties>
</configuration-properties>
</agent-service>
</agent-services>
</app-agent-configuration>
<interceptors>
<interceptor name="ADO.NET" disable="false"/>
<interceptor name="ASP.NET" disable="false"/>
<interceptor name="ASP.NET Web Services" disable="false"/>
<interceptor name="Directory Services Exit" disable="false"/>
<interceptor name="HttpWebRequest GetResponse exit" disable="false"/>
<interceptor name="Messaging exit receive" disable="false"/>
<interceptor name="Messaging exit send" disable="false"/>
<interceptor name="Remoting entry" disable="false"/>
<interceptor name="Remoting exit" disable="false"/>
<interceptor name="SOAP Exit" disable="false"/>
<interceptor name="WCF Entry" disable="false"/>
<interceptor name="WCF Exit" disable="false"/>
</interceptors>
<snapshots disable="false">
</snapshots>
</AppDynamicsAgent>
```

Application Pool Configuration

The IIS application runs under the context of the account defined to run the application pool (impersonation being an exception inside the application itself). For the .NET Application Agent to write to the logs folder, it must run under an account that has access to the logs folder (generally C:\AppDynamics\Logs unless changed previously). AppDynamics best practices generally give the logs folder by default the following accounts permission when running the .NET Agent Configuration:

- ApplicationPool Identity
- Local Service
- Network Service
- System

Not all applications run under the ApplicationPoolIdentity so it is advised when manually configuring the .NET AppDynamics Agent to run the application pool under one of these accounts and ensure the same accounts have access to the logs folder.

AppDynamics Configuration

The <AppDynamicsAgent> element is where you configure the controller communication for the application agent. The values here depend on whether the controller is on-premise, SaaS, single or multi-tenant, and if SSL is in use.
 In the web.config file, modify the following items in the <controller-info> element:

Item	Required	Default
<controller-host>	Yes	None
<controller-port>	Yes	For On-premise Controller installations: By default, port 8090 is used for HTTP and 8181 is used for HTTPS communication. For SaaS Controller service: By default, port 80 is used for HTTP and 443 is used for HTTPS communication.

To configure the .NET Agent to use SSL see [App Agent for .NET Configuration Properties](#).

(For Multi-tenant mode or SaaS installations only) Configure the .NET Agent account information

This step is required only when the AppDynamics Controller is configured in multi-tenant mode (the Controller is configured with multiple user accounts) or when you [Use a SaaS Controller](#). Skip this step if you are using single-tenant mode, which is the default for the on-premise installation.

In the web.config file, specify the properties for Account Name and Account Key. This information is provided in the Welcome email from AppDynamics Support Team.

Property	Required	Default
<account-name>	Required only if your Controller is configured for multi-tenant mode or your Controller is hosted.	None
<account-access-key>	Required only if your Controller is configured for multi-tenant mode or your Controller is hosted.	None

Configure How The Agent Application And Tier Are Identified

In the web.config file, specify to which Application and Tier this .NET Agent belongs. For more information see [Mapping Application Services to the AppDynamics Model](#).

Property	Required	Default
<application-name>	Yes	None
<tier-name>	Yes	None

AppDynamics automatically identifies the node name as:

<tier name><host name><app pool name><worker process index: 0,1,2...><clr index: 0,1,...><app domain index: 0,1,...>

Note: Node names tend to have low values for the index and attempts are made to reuse the same node name after a recycle event (such as application pool recycle).

Launching the Application Agent

If the machine agent has been configured and the relevant web.configs have been configured, it is now safe to launch the application. The steps are as follows:

1. Stop either the relevant application pools for the instrumented web application or stop IIS entirely.
2. Stop the AppDynamics.Agent.Coordinator_service (described as AppDynamics.Agent.Coordinator).
3. Restart the application pool or IIS.
4. Place load on the application to start instrumentation and reporting to the defined AppDynamics controller application.

Troubleshooting

To verify that instrumentation has occurred, visit the logs folder (generally c:\AppDynamics\Logs unless changed previously). This log folder contains the following if instrumentation has occurred:

- A folder called Data
- ByteCode.txt
- Possibly additional log files (AgentLog, BusinessTransactionLog, RESTHeartbeat, etc.)

If these are not seen, open AgentLog and look for lines containing w3wp and dllhost. If only dllhost is present and not w3wp, this means instrumentation has not occurred and generally will be due to the following issues:

- Incorrect permissions on the log folder (the application pool must have permission to write to the logs folder).
- A previous profiler has been installed. Even if removed, often software leaves behind residual hooks which can affect installs of other applications.

Learn More

- [Agent - Controller Compatibility Matrix](#)

- Upgrade the App Agent for .NET
- Install the App Agent for .NET
- App Agent for .NET Configuration Properties

Source

1. <http://msdn.microsoft.com/en-us/library/ms178685.aspx>
2. Configure the .NET Machine Agent

Multi-Agent Deployment for .NET

- Deployment Procedure
 - To Deploy .NET App Agents
- Sample Deployment Solutions
- Sample Silent Install Script
- Sample Multi-Agent Rollout

This topic describes the high-level procedures for deploying multiple AppDynamics app agents on .NET platforms.

In .NET the machine agent is embedded in the app agent so there is no separate machine agent installation procedure.

Deployment Procedure

To Deploy .NET App Agents

1. Download the latest .NET agent msi file from <http://download.appdynamics.com/>.
2. For each server role in your managed environment, create a setup file.
You can use a configuration utility to create this file. See [Using a Configuration File from the Command Line](#) for information about starting the configuration utility from a script.

See [Configure the App Agent for .NET](#) for details about configuring the server using the utility.

3. For each application server that you are instrumenting, execute the configuration file that was output by the configuration utility.

Sample Deployment Solutions

The following is a sample setup file named SavedSetupConfiguration.xml that was generated by the configuration utility.

```

<?xml version="1.0" encoding="utf-8"?>
<winston>
  <logFileDirectory directory="C:\Appdynamics\Logs" />
  <appDynamicsAgent>
    <controller controller-host="winbuild-server2" controller-port="8090"
    controller-ssl-enabled="False" account-name="" account-access-key="">
      <proxy disable="True">
        <host>
        </host>
        <port>
        </port>
      </proxy>
      <logFileFolderAccessPermissions>
        <account name="NT AUTHORITY\LOCAL SERVICE" displayName="LOCAL SERVICE" />
        <account name="NT AUTHORITY\SYSTEM" displayName="SYSTEM" />
        <account name="NT AUTHORITY\NETWORK SERVICE" displayName="NETWORK SERVICE" />
        <account name="IIS_IUSRS" displayName="ApplicationPool Identity" />
      </logFileFolderAccessPermissions>
    </controller>
    <controllerApplications>
      <controllerApplication name="New App">
        <tiers>
          <tier name="WWW">
            <applications>
              <application name="Default Web Site\Mvc3TestApplication" iisApplication="True"
endUserMonitoring="False" />
              <application name="Default Web Site\WebTestApplication" iisApplication="True"
endUserMonitoring="False" />
            </applications>
          </tier>
          <tier name="Services">
            <applications>
              <application name="Default Web Site\WasHostedWcfService" iisApplication="True"
endUserMonitoring="False" />
              <application name="Default Web Site\WASService" iisApplication="True"
endUserMonitoring="False" />
              <application name="Default Web Site\MyService" iisApplication="True"
endUserMonitoring="False" />
            </applications>
          </tier>
        </tiers>
      </controllerApplication>
    </controllerApplications>
  </appDynamicsAgent>
</winston>

```

Sample Silent Install Script

The following is a sample script to install the agent. It assumes that you are first uninstalling an older agent, so if this is a fresh install, remove the uninstall command.

```

installscript.bat "olddotNetAgentSetup64.msi" "uninstall.log" "dotNetAgentSetup64.msi"
"C:\Program Files\AppDynamics\AppDynamics .NET Agent" "install.log"
"SavedSetupConfiguration.xml"

```

Sample Multi-Agent Rollout

You can also download a sample solution that one of our customers created to perform multi-agent rollouts. Use this for sample for ideas on how to automate AppDynamics agent deployment for your environment.

Sample attached.

Name	Size	Creator	Creation Date	Comment
ZIP Archive DotNetExampleScripts.zip	1 kB	Lynn Davidson	Feb 27, 2013 16:42	

Naming Conventions for .NET Nodes

- [Naming Syntax](#)
 - Tier Name
 - Machine Name
 - App Pool Name or Exe Name
 - Process Index
 - CLR Count
 - App Domain Index
- [Maximum Number of Nodes Generated](#)

This topic describes the conventions that the App Agent for .NET Configuration Utility uses to name the nodes.

Note that different .NET versions of the same application run on independent processes, as each has its own version of the CLR.

Naming Syntax

The naming pattern of a node is as follows:

```
<tier-name>-<machine-name>-<app-pool-name | exe-name><process-index>-<clr-count>-<app-domain-index>
```

In the following example:

```
Order Server-IP-0A481DD8-NWTraders-P-0-1-4
```

- Order Server is the tier name
- IP-0A481DD8 is the machine name
- NWTraders-P is the app pool name
- 0 is the process index
- 1 is the CLR count
- 4 is the app domain index

Tier Name

This is the name of the tier to which the node belongs.

Machine Name

This is the name of the machine on which the CLR is running.

App Pool Name or Exe Name

This is the name of the process.

Process Index

The process index represents the zero-based index of the process. For a self-hosted process the index would be 0. For a web garden-IIS-hosted application with five worker processes, the index could be 0, 1, 2, 3 or 4.

CLR Count

The CLR count represents the CLR in process, which is normally 1. In some cases it could be greater than 1 if you have multiple applications pointing to the same application pool.

App Domain Index

This is the CLR's position in the zero-based app domain index.

Maximum Number of Nodes Generated

Below is the algorithm for calculating how many nodes will be generated. It refers to all nodes that are alive, as historical node counts can change via retention and deletion time frames as well as manual deletion, etc.

```
IISApp1_AppPool_MaxWorkerProcesses  
+  
IISApp2_AppPool_MaxWorkerProcesses  
+...  
IISAppN_AppPool_MaxWorkerProcesses  
+...  
Self-Hosted Process (Windows Service, Console Application, etc)  
=  
Number of Nodes
```

Administer App Agents for .NET

- AppDynamics Applications and IIS Applications
- Features Currently Unavailable
- Log Files
- Additional App Agent for .NET Administration Topics

AppDynamics Applications and IIS Applications

Usually there is not a 1-to-1 correspondence between an AppDynamics business application and an IIS application. A typical AppDynamics configuration assigns one IIS application per tier, and multiple tiers to a single AppDynamics business application. For example, a front end is a single tier, various services are represented as another tier or tiers, and they all belong to a single business application. See [Name Business Applications, Tiers, and Nodes](#).

Features Currently Unavailable

Some features that are available for the App Agent for Java are not yet supported for the App Agent for .NET.

- Manual BCI EUM instrumentation also called [Assisted Injection Using Injection Rules](#).
- [Remediation Actions](#), such as running local scripts on nodes.
- Diagnostic thread dump actions are not supported.
- Reassigning nodes is not supported.
- Deleting tiers is not supported.
- Adding custom machine agent metrics (also called custom monitors) is not supported.
- Detecting code deadlocks is not supported.
- Aggressive snapshot collection is not supported.
- Memory monitoring (also known as Object Instance Tracking) is not supported.
- The embedded Machine Agent (installed by default) for the App Agent for .NET does not support extensions. If you want to use Machine Agent extensions such as custom plugins or the HTTP Metric Listener, or do orchestration to compute clouds, install the standalone Machine Agent. See [Configure the .NET Machine Agent](#).
- Some agent node properties are not supported in the .NET environment. Review [App Agent Node Properties Reference](#) for detailed information.

Log Files

The configuration file that controls log files for the App Agent for .NET is located at:

```
C:\Program Files\AppDynamics\appdynamics .NET Agent\appdynamicsagentlog.config
```

The configuration file uses NLog rules. See <http://nlog-project.org/>.

Additional App Agent for .NET Administration Topics

App Agent for .NET Configuration Properties

- Properties to Configure the .NET App Server Agent
 - Configuration Scopes
- Required Agent Properties
 - Agent-Controller Communication Properties
 - Agent Identification Properties
 - Advanced Properties
 - SSL for the Controller
 - Agent Runtime Directory
 - Multi-tenant Mode
 - To use an existing Web.config file
- Learn More

This topic describes the .NET App Server Agent configuration properties.

Properties to Configure the .NET App Server Agent

You can configure App Server Agent properties using either of the following options:

- The sample Web.config file provided at <agent_install_dir>\Samples\Configuration
- An existing Web.config file

Configuration Scopes

There are different scopes to configuration settings:

- a global scope
- the scope of the application
- the root Web.config file

For details see [ASP.NET Configuration File Hierarchy](#).

Configuration Level	File Name	Notes
Root Web	Web.config	The Web.config file for the server is stored in the same directory as the Machine.config file and contains default values for most of the system.web configuration sections.
Web site	Web.config	The Web.config file for a specific Web site contains settings that apply to the Web site and inherit downward through all ASP.NET applications and subdirectories of the site.
ASP.NET application root directory	Web.config	The Web.config file for a specific ASP.NET application is located in the root directory of the application and contains settings that apply to the Web application and inherit downward through all of the subdirectories.
ASP.NET application sub directory	Web.config	The Web.config file for an application subdirectory contains settings that apply to this subdirectory and inherit downward through all of the subdirectories.
Client application directory	ApplicationName.config	The ApplicationName.config file contains settings for a Windows client application (not a Web application). The <ApplicationName> includes the .exe extension (app.exe.config).

Required Agent Properties

Mandatory properties include:

- Agent-Controller Communication Properties configure how the Agent connects with the Controller.
- Agent Identification Properties configure how the Agent identifies itself.

Additionally, there are Advanced Properties, used for special cases.

Agent-Controller Communication Properties

Modify the following properties in the Web.config file:

Description	Property	Required	Default
Controller Host <i>For an on-premise Controller installation: The value as configured for "Application Server Host Name". For the SaaS Controller service, refer to the Welcome email.</i>	<controller-host>	Yes	None
Controller Port <i>For an on-premise Controller installation: Provide the value for "HTTP Listener Port". For SaaS Controller service, refer to the Welcome email.</i>	<controller-port>	Yes	For on-premise: Port 8090 is the default for HTTP and port 8181 is the default for HTTPS. For SaaS Controller Service: Port 80 is used for HTTP and port 443 is used for HTTPS.

For example:

```

<configurations>
  <AppDynamicsAgent>
    <controller-info>
      <controller-host>localhost</controller-host>
      <controller-port>8090</controller-port>
      ...
    </controller-info>
  </AppDynamicsAgent>
</configurations>

```

Agent Identification Properties

Provide information about the business application and tier in the Web.config file:

Description	Property	Required	Default
Application Name <i>Name of the business application</i>	<application-name>	Yes	None
Tier Name <i>Name of the tier/cluster to which this Agent belongs</i>	<tier-name>	Yes	None

For example:

```

<configurations>
  <AppDynamicsAgent>
    <controller-info>
      .....
      <application-name>ACMEOnline</application-name>
      <tier-name>InventoryTier</tier-name>

      .....
    </controller-info>
  </AppDynamicsAgent>
</configurations>

```

Advanced Properties

Advanced properties configure the Agent for special cases.

SSL for the Controller

Use this property to enable or disable SSL for Agent-Controller communication.

Property	Default Value
<controller-ssl-enabled>	False (Allowed values: True, False)

Agent Runtime Directory

Use this property to specify the runtime directory for all runtime files (logs, transaction configuration) for those nodes that use this Agent. If this property is specified, all Agent logs are written to <agent-runtime-dir>/logs/node-name and transaction configuration is written to <agent-runtime-dir>/conf/node-name.

Property	Default Value
<agent-runtime-dir>	<agent_install_dir>/nodes

Multi-tenant Mode

Use these properties to specify the account name and account key in a multi-tenant setup.

i If you are using the SaaS Controller Service, these properties are mandatory. The <account-name> is provided to you in the Welcome email sent by the AppDynamics Support Team.

Property	Default Value
<account-name>	None
<account-access-key>	None

To use an existing Web.config file

You can update an existing web.config file present in your system.

1. Go to the <configSections> element in your web.config file and add following section:

```
<configSections>
    <section name="AppDynamicsAgent"
        type="AppDynamics.Agent.Config.Section, AppDynamics.Agent,
        Version=1.0.0.0, Culture=neutral, PublicKeyToken=3f604d9e4f8e4985"
        allowLocation="true"          allowDefinition="Everywhere"      />
</configSections>
```

2. Add the following <AppDynamicsAgent> section:

```
<AppDynamicsAgent>
    <controller-info>
        <controller-host>host</controller-host>
        <controller-port>8090</controller-port>
        <controller-ssl-enabled>false</controller-ssl-enabled>

        <!-- If the controller is running in multi-tenant mode, specify the account name
            and access key for this agent to authenticate with the controller.
            If the controller is running in single-tenant mode, there is no need to
            specify these values. -->
        <account-name></account-name>
        <account-access-key></account-access-key>

        <!-- For Auto Agent Registration specify the application name, tier name,
            and optionally, node name. If the application and/or tier does not
            exist already it will be created. -->
        <application-name>Test .NET Application</application-name>
        <tier-name>Portal</tier-name>
    </controller-info>
</AppDynamicsAgent>
```

```

        <!-- Recommended to leave it empty if your application tier node has more than one
instance,
           as in case of IIS applications running in web garden (app pool) or in the case
when nodes
               of the same tier run on different machines.
               If unsure - leave it empty, It will be auto-generated-->
<node-name></node-name>
<proxy disable="true">
    <host></host>
    <port></port>
</proxy>
</controller-info>
<app-agent-configuration>
    <configuration-properties>
        <property name='agent-overwrite' value='false' />
    </configuration-properties>
    <agent-services>
        <agent-service name='BCIEngine' enabled='true'>
            <configuration-properties>
            </configuration-properties>
            <configuration>
            </configuration>
        </agent-service>
        <agent-service name='CLRMetricsService' enabled='true'>
            <configuration-properties>
                <property name='update-interval-in-seconds' value='60' />
            </configuration-properties>
            <configuration>
                <perf-counters>
                    <!-- In this section you can add CLR related performance counters
The list of possible .NET performance counters is documented here
Performance Counters in the .NET Framework
http://msdn.microsoft.com/en-us/library/w8f5kw2e%28v=VS.80%29.aspx -->
                    <!-- Example: <counter cat=".NET CLR Exceptions" name="# of Exceps Thrown" /> -->
                </perf-counters>
            </configuration>
        </agent-service>
        <agent-service name='SnapshotService' enabled='true'>
            <service-dependencies>BCIEngine</service-dependencies>
            <configuration-properties>
            </configuration-properties>
        </agent-service>
        <agent-service name='TransactionMonitoringService' enabled='true'>
            <service-dependencies>BCIEngine,SnapshotService</service-dependencies>
            <configuration-properties>
            </configuration-properties>
        </agent-service>
    </agent-services>
</app-agent-configuration>
<interceptors>
    <interceptor name="ADO.NET" disable="false"/>
    <interceptor name="ASP.NET" disable="false"/>
    <interceptor name="ASP.NET Web Services" disable="false"/>
    <interceptor name="Directory Services Exit" disable="false"/>
    <interceptor name="HttpWebRequest GetResponse exit" disable="false"/>
    <interceptor name="Messaging exit receive" disable="false"/>
    <interceptor name="Messaging exit send" disable="false"/>
    <interceptor name="Remoting entry" disable="false"/>
    <interceptor name="Remoting exit" disable="false"/>
    <interceptor name="SOAP Exit" disable="false"/>
    <interceptor name="WCF Entry" disable="false"/>
    <interceptor name="WCF Exit" disable="false"/>
</interceptors>
<snapshots disable="false">

```

</snapshots>

```
</AppDynamicsAgent>
```

Learn More

- [Install the App Agent for .NET](#)
- [Troubleshoot App Agent for .NET Installation and Configuration](#)

Disable Instrumentation for an IIS Application Pool

- [When to Consider Disabling Instrumentation](#)
 - To disable an IIS application pool or pools

When to Consider Disabling Instrumentation

By default when you install the App Agent for .NET on a machine, every IIS application on the machine will be instrumented. You may not need all application pools to be monitored.

AppDynamics uses the system variable AppDynamicsAgent_DisableAppPools to disable instrumentation for specific application pools.

To disable an IIS application pool or pools

1. Add this system environment variable:

```
AppDynamicsAgent_DisableAppPools=<DefaultAppPool>|<MyAppPool1>|<MyAppPool2>
```

Use the '|' (pipe character) is used as a separator for multiple application pools.

2. Restart IIS to pick up this system variable.

App Agent for .NET Logs

To Request Agent Log Files

1. In the **Agents** tab of the node dashboard, scroll down to the Agent Logs panel.

2. Click **Request Agent Log Files**.

The Request Agent Log Files window opens with the "All logs in the logs directory" option preselected.

Request Agent Log Files

What would you like to collect from the agent on this node?

All logs in the logs directory

Output from a specific logger, at a set level, for a fixed duration

Logger Name	ex. com.singularity
Logger Level	All
Duration (minutes)	

Thread dump samples

Number to Collect	1	Maximum 50
Collection Interval (ms)	500	Minimum 500 ms

(optional) Enter a name for this request:

Cancel **Request Agent Log Files**

4. (Optional) Enter a name for your request. The name is used to identify your request. It is not used to name the generated zip file.

5. Click Request Agent Log Files.

The AppDynamics controller processes your request. The status field changes as follows:

- PENDING - request submitted
- IN_PROGRESS - request being processed
- SUCCESSFUL - request is ready

When the status reads as SUCCESSFUL, you can click the icon at the end of the line to download the zip file to your local machine for review as shown in the following screen capture.

Agent Logs				
Request Agent Log Files				
Name	Description	Date Log Request	Status	
test	Zipped agent logs directory	10/03/12 2:36:25 PM	SUCCESSFUL	

Click to download the zip file.

Note: If you accidentally request the logs for a node that is not valid, for example IIS is turned off for that node, the request eventually changes to IN_PROGRESS, but does not progress to SUCCESSFUL. You can remove such a request manually by selecting it, right-clicking and selecting Delete Data from the contextual menu.

Monitor .NET Applications

- Monitor CLRs
- Monitor IIS
- Windows Performance Counters

The .NET embedded machine agent reports hardware metrics such as:

- CPU activity
- Memory usage

- Disk reads and writes
- Network traffic

When IIS is installed on the machine, the .NET machine agent also reports IIS, ASP.NET and ASP.NET Application metrics. See [Monitor IIS](#).

The App Agent for .NET reports CLR metrics, which are also based on MS Performance Counters. See [Monitor CLRs](#).

If you install the Standalone Machine Agent, which is a Java application, in a .NET environment, then you can add custom monitors and extensions such as the HTTP listener. See [Standalone Machine Agent](#) for details.

Learn More

- [Monitor Databases](#)
- [Monitor Remote Services](#)

Monitor CLRs

- [CLR Metrics](#)
 - To access the Node Dashboard
 - To access the Metric Browser
- [Alerting for CLR Health](#)
- [Learn More](#)

AppDynamics uses Microsoft Windows Performance Counters to gather .NET metrics. These preconfigured CLR metrics can be viewed from the Node Dashboard and in the Metric Browser.

Refer to the Microsoft documentation for more information:

- General overview: [Performance Counters](#)
- ASP.NET Counters: [Performance Counters for ASP.NET](#)

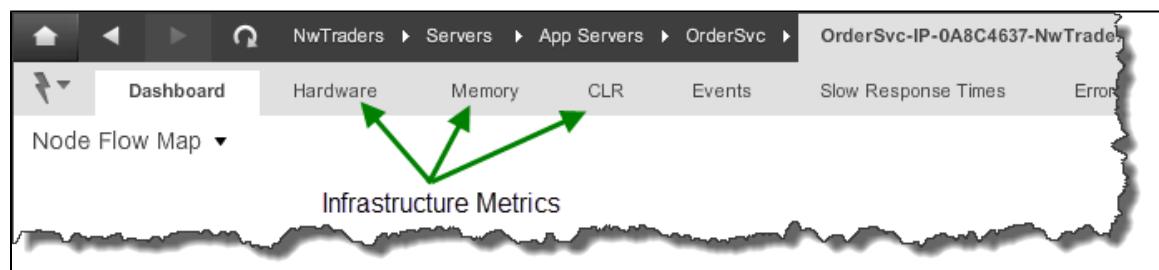
CLR Metrics

CLR metrics give insight into how the .NET runtime is performing. The AppDynamics preconfigured CLR metrics include:

- .NET CLR memory usage
- Total classes loaded and how many are currently loaded
- Garbage collection time spent, and detailed metrics about GC memory pools and caching
- Locks and thread usage
- Memory heap and non-heap usage, including the large object heap
- Percent CPU process usage

To access the Node Dashboard

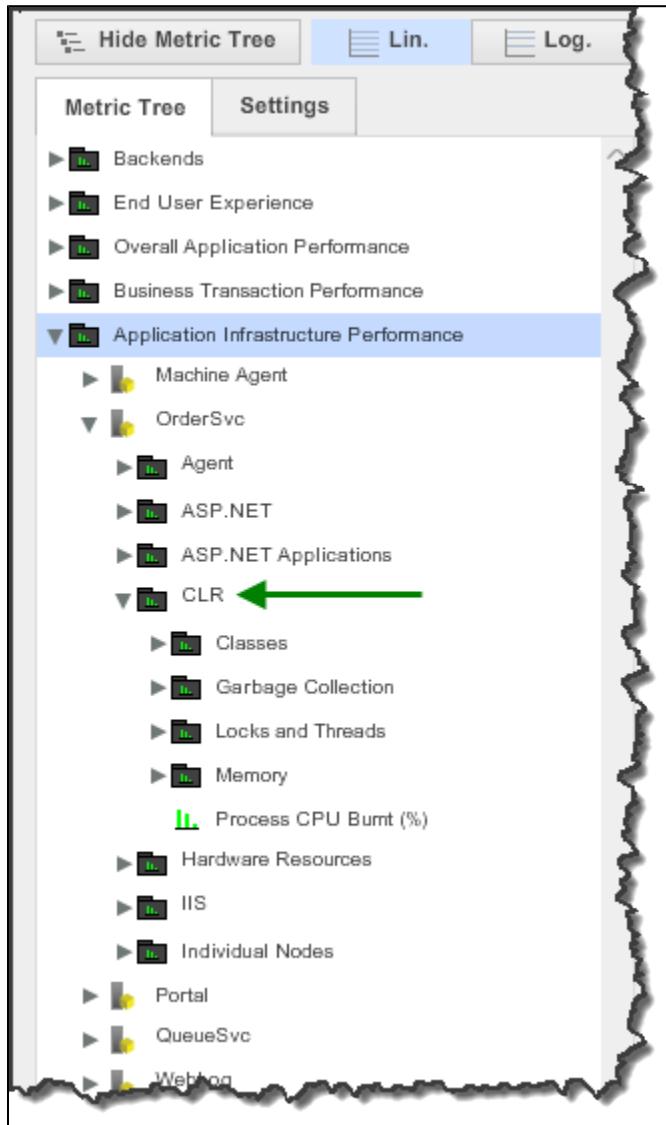
1. Select the business application.
2. In the left navigation pane, click **Servers > App Servers > <Tier> -> <Node>**.
AppDynamics displays the Node Dashboard for the selected node.
3. Click the tab for the metrics you want to view.



See [Node Dashboard](#) for more details.

To access the Metric Browser

1. In the left navigation pane of the Controller, click **Analyze -> *Metric Browser**.
2. Expand the **Application Infrastructure Performance** branch of the **Metric Tree**.
3. Expand the tier that you want to view.
4. Expand the tree for each category of metrics.
5. To view the CLR metrics in the Metric Browser, expand **Application Infrastructure Performance -> <Node> -> CLR**.



See [Metric Browser](#) for more details.

Alerting for CLR Health

You can set up health rules based on the infrastructure metrics. Once you have a health rule, you can create specific policies based on health rule violations. One type of response to a health rule violation is an alert.

In addition to the default metrics, you may be interested in additional metrics. You can specify additional performance counters to be reported by the .NET Machine Agent. See [Windows Performance Counters](#) for details.

Once you add a custom metric you can create a health rule for it and receive alerts when conditions indicate problems.

See [Alert and Respond](#) for details on health rules and policies.

i Note: Automatically running a script for remediation of problems and taking thread dumps are not supported in the .NET environment.

Learn More

- Infrastructure Monitoring
- Performance Counters for ASP.NET (external website)
- Install the App Agent for .NET
- Windows Performance Counters

Monitor IIS

- Default Metrics Available in the .NET Environment
 - ASP.NET Metrics
 - ASP.NET Application Metrics
 - IIS Metrics
- Monitoring IIS Application Pools
 - To view the IIS application pools:

AppDynamics uses Microsoft Windows Performance Counters to gather .NET metrics. These preconfigured IIS related metrics can be viewed in the Metric Browser.

For more information on Windows Performance Counters, refer to the Microsoft documentation [Performance Counters for ASP.NET](#).

Default Metrics Available in the .NET Environment

 Internet Information Services (IIS) must be installed on the machine for you to view the metrics for the following:

- IIS
- ASP.NET
- ASP.NET Application

ASP.NET Metrics

To view the ASP.NET metrics in the Metric Browser, expand **Application Infrastructure Performance -> <Node> -> ASP.NET**.

AppDynamics reports the following ASP.NET metrics:

- Application Restarts
- Applications Running
- Requests Disconnected
- Requests Queued
- Requests Rejected
- Request Wait Time
- Worker Process Restarts

ASP.NET Application Metrics

To view the ASP.NET Application metrics in the Metric Browser, expand **Application Infrastructure Performance -> <Node> -> ASP.NET Applications**.

AppDynamics reports the following ASP.NET Application metrics:

- Anonymous Requests
- Anonymous Requests/sec
- Cache Total Entries
- Cache Total Hit Ratio
- Cache Total Turnover Rate
- Cache API Entries
- Cache API Hit Ratio
- Cache API Turnover Rate
- Errors Unhandled During Execution/sec
- Errors Total/sec
- Errors During Preprocessing
- Errors During Compilation
- Errors During Execution
- Errors Unhandled During Execution
- Errors Unhandled During Execution/sec
- Errors Total
- Errors Total/sec

- Output Cache Entries
- Output Cache Hit Ratio
- Output Cache Turnover Rate
- Pipeline Instance Count
- Requests Executing
- Requests Failed
- Requests In Application Queue
- Requests Not Found
- Requests Not Authorized
- Requests Succeeded
- Requests Timed Out
- Requests Total
- Requests/sec
- Session State Server Connections Total
- Session SQL Server Connections Total
- Sessions Active
- Sessions Abandoned
- Sessions Timed Out
- Sessions Total
- Transactions Aborted
- Transactions Committed
- Transactions Pending
- Transactions Total
- Transactions/sec

IIS Metrics

From the Metric Browser:

- To view the IIS metrics for a tier, expand **Application Infrastructure Performance -> <Tier> -> IIS**.
- To View the IIS metrics for a node, expand **Application Infrastructure Performance -> <Tier> -> Individual Nodes -> <Node> -> IIS**.

Each metric is reported for the entire tier, each individual application pool, and each individual node as follows:

- **Application Infrastructure Performance -> <Tier> -> IIS** = combined for all IIS processes in all Application Pools for this tier
- **Application Infrastructure Performance -> <Tier> -> Application Pools -> <application pool name>** = combined for all processes in this specific Application Pool
- **Application Infrastructure Performance -> <Tier> -> Individual Nodes -> <Node>** = metrics for the specific node.

Monitoring IIS Application Pools

You can monitor the health of IIS application pools for the instrumented .NET nodes in a tier. You can view the information by application pool, machine, and process IDs in varying hierarchies.

These groupings enable you to visualize key performance indicators for your infrastructure:

- Health of the node in the specified time-range for a particular group.
- App Server Agent's status in the specified time-range.
- Last CLR restart.
- A link to the parent tier.

To view the IIS application pools:

1. In the left navigation pane, select the tier that you want to monitor: **Servers -> AppServers -> <Tier>**.
2. Click the **IIS App Pools** tab.
The IIS App Pools list displays.
3. (Optional) From the **Group Nodes By** dropdown menu, select how you want to view the application pools and machines for this tier.

4. To view a node's dashboard, double-click the node in the list. From there you can select the various tabs for details about performance of the node. See [Node Dashboard](#).

i If a machine or application pool name is not available for a .NET node, an "Unknown App Pool" / "Unknown Machine" grouping is created.

Windows Performance Counters

Performance Counters and the .NET Machine Agent

By default, the .NET Machine Agent uses Microsoft Performance Counters to gather and report .NET metrics. For details on the preconfigured .NET metrics see [Monitor CLR](#)s and [Monitor IIS](#).

You can specify additional performance counters to be reported by the .NET Machine Agent.

To configure additional performance counters for .Net

1. Shut down the Coordinator process using the Component Services (COM+) console.
2. Open the application.config file located at <agent_install_dir>\Machine Agent\Configuration.
3. Go to the <perf-counters> element located in the "<AppDynamicsAgentCoordinator>" section.
4. Add a performance counter by creating a new element <counter> and specifying the following items for it:
 - cat: Category of the Performance Counter
 - Name: Name of the metrics that will be displayed on the Metric Browser
 - instance: Instance of the performance counter.

i **Tip:** If a particular performance counter has many instances you can specify the following options:

- instance = "*" OR
- instance = "all" (This will report the sum of all instances)

For example, to add the performance counter for measuring CPU Idle time(%), add the following element in the <perf-counters> section:

```
<counter cat="Processor" name="% Idle Time" instance="_Total"/>
```

You can add any of the performance counters as specified here: [Performance Counters in .NET Framework](#).

5. Restart the Appdynamics.Agent.Coordinator_service process to ensure that the changes are applied to the application.config file.

Configure AppDynamics for .NET

- Configure Custom Exit Points (.NET)
- Getter Chains in .NET Configurations
- Enable Thread Correlation (.NET)
- Configure the .NET Machine Agent
- Enable Correlation for .NET Remoting

Configure Custom Exit Points (.NET)

- Default Backends Discovered by the Agent for .NET
- Configure Custom Exit Points for .NET Backends
- To create a custom exit point
 - To split an exit point
 - To group an exit point
- To define custom metrics for a custom exit point
- To define transaction snapshot data collected
- Learn More

AppDynamics provides default automatic discovery for commonly-used backends. If a backend used in your environment is not discovered, first compare the list of default backends to determine whether you need to modify the default configuration. If it is not on the list then configure a custom exit point according to these instructions.

Default Backends Discovered by the Agent for .NET

The default data access backends:

- ADO.NET

The default remote services for .NET agents are:

- WCF
- HTTP
- Web Services, including SOAP
- Directory Services, including LDAP*
- Queues
 - Apache ActiveMQ
 - IBM WebSphere MQ (also known as IBM XMS)
 - Microsoft Message Queuing (MSMQ)*
 - .NET Remoting*
 - Tibco Enterprise Message Service (EMS)
 - Tibco Rendezvous (RV)
 - MicrosoftServiceBus (Windows Azure Service Bus)
 - MicrosoftServiceBusQueue (Windows Azure Service Bus Queues)
 - AzureQueue (Windows Azure Queues)

* Notes: Correlation is supported unless otherwise indicated below.

- For LDAP, correlation is non-applicable
- For MSMQ, correlation for downstream calls is not supported.
- For .NET remoting, additional configuration is needed to get downstream correlation. See [Enable Correlation for .NET Remoting](#).

For instructions on modifying default backend discovery see [Configure Backend Detection](#).

Configure Custom Exit Points for .NET Backends

Custom exit points provide identification for backend types that are not automatically detected, such as file systems, mainframes etc. For example, you can define a custom exit call to monitor the file system read method. Custom exit points appear as unresolved

backends in the flow maps. Unresolved backends are shown on flow maps with this icon .

You define a custom exit point by specifying the class and method used to identify the backend. If the method is overloaded, you need to add the parameters to identify the method uniquely.

You can restrict the method invocations for which you want AppDynamics to collect metrics by specifying match conditions for the method. The match conditions can be based on a parameter or the invoked object.

You can also optionally split the exit point based on a method parameter, the return value, or the invoked object.

You can also configure custom metrics and transaction snapshot data to collect for the backend.

See [Configurations for Custom Exit Points](#) for suggested custom configurations for some common backends.

To create a custom exit point

1. In the left navigation panel, click **Configure -> Instrumentation**.

2. Click the **Backend Detection** tab.
3. Click the tab corresponding to the backend platform.
4. Select the application or tier for which you are configuring the custom exit point. Backend detection configuration is applied on a hierarchical inheritance model. See [Hierarchical Configuration Model](#).
5. Scroll down to Custom Exit Points.
6. Click **Add** (the + icon).
7. In the Create Custom Exit Point window, click the **Identification** tab if it is not selected.
8. Enter a name for the exit point. This is the name that identifies the backend.
9. Select the type of backend from the Type drop-down menu or check Use Custom if the type is not listed.
10. Configure the class and method name that identify the custom exit point.
If the method is overloaded, check the Overloaded check box and add the parameters.
11. If you want to restrict metric collection based on a set of criteria that are evaluated at runtime, click **Add Match Condition** and define the match condition(s).
For example, you may want to collect data only if the value of a specific method parameter contains a certain value.
12. Click **Save**.

The following screenshot shows a custom exit point of Type Cache. This exit point is defined on the `getAll()` method of the specified class. The exit point appears in flow maps as an unresolved backend named CoherenceGetAll.

Edit Custom Exit Point

Name:	CoherenceGetAll	Type:	Cache				
<input type="radio"/> Identification <input type="radio"/> Custom Metrics <input type="radio"/> Snapshot Data							
Define the class and method name which, when called, will be identified as a Custom Exit Point							
Class	that implements an Interface which	equals	com.tangosol.net.NamedCache				
Method Name	getAll	<input type="checkbox"/> Is this Method Overloaded?					
Method Parameters (optional)							
<input type="button" value="Add Parameter"/>							
Match Conditions (optional)							
<input type="button" value="Add Match Condition"/>							
Calls to the specified class and method name can be further split based by a combination of name and description.							
<table border="1"> <thead> <tr> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>Cachename</td> <td>Collect data from the invoked object and capture the result.</td> </tr> </tbody> </table>		Name	Description	Cachename	Collect data from the invoked object and capture the result.	<input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/>	
Name	Description						
Cachename	Collect data from the invoked object and capture the result.						
<input type="button" value="Cancel"/> <input type="button" value="Save"/>							

To split an exit point

1. Click **Add**.
2. Enter a display name for the split exit point.
3. Specify the source of the data (parameter, return value, or invoked object).
4. Specify the operation to invoke on the source of the data (`toString()` or getter chain for complex objects).
5. Click **Save**.

The following example shows a split configuration of the previously created CoherenceGetAll exit point based on the `getCacheName()` method of the invoked object.

Edit Custom Exit Point Identifier

Specify the parameter index or indicate if it the return value of the diagnostic data to be collected.
Simple getters without parameters can be used on the parameter or the return value to be displayed against the display name specified here.

Display Name

Create your own name for the data collected. This will be the display name for the data in Request Snapshots

Collect Data From Method Parameter @ Index:
 Return Value
 Invoked Object

Operation on Invoked Object Use `toString()`

Use Getter Chain

for example: `getAccount().getBalance()`

To group an exit point

You can group methods as a single exit point. The only requirement is that these methods point to the same key.

For example, ACME Online has an exit point for `NamedCache.getAll`. This exit point has a split configuration of `getCacheName()` on the invoked object as illustrated in the previous screenshot.

Suppose we also define another exit point for `NamedCache.entrySet`. This is another exit point, but it has the split configuration that has `getCacheName()` method of the invoked object.

Edit Custom Exit Point

Name:	CoherenceCacheAccess	Type:	Cache				
<input checked="" type="radio"/> Identification <input type="radio"/> Custom Metrics <input type="radio"/> Snapshot Data							
Define the class and method name which, when called, will be identified as a Custom Exit Point:							
Class	that implements an Interface which		equals com.tangosol.net.NamedCache				
Method Name	entrySet	<input type="checkbox"/> Is this Method Overloaded?					
Method Parameters (optional)							
Add Parameter							
Match Conditions (optional)							
Add Match Condition							
Calls to the specified class and method name can be further split based by a combination of method parameters and match conditions.							
<table border="1"> <thead> <tr> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>CacheName</td> <td>Collect data from the invoked object and capture the result of getCacheName().</td> </tr> </tbody> </table>				Name	Description	CacheName	Collect data from the invoked object and capture the result of getCacheName().
Name	Description						
CacheName	Collect data from the invoked object and capture the result of getCacheName().						
<input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/>							

If the getAll() and the entrySet() methods point to the same cache name, they will point to the same backend.

Matching name-value pairs identify the back-end. In this case, there is only one key, i.e. cache name, has to be matched. So, here both exit points have the same name for the cache and they resolve to the same backend.

To define custom metrics for a custom exit point

Custom metrics are collected in addition to the standard metrics.

The result of the data collected from the method invocation must be an integer value, which will either be averaged or added per minute, depending on your selection of data roll-up.

To configure custom business metrics that can be generated from the Java method invocation:

1. Click the **Custom Metrics** tab.
2. Click **Add**.
3. In the Add Custom Metric window, enter a name for the metric.
4. Select the Collect Data From radio button to specify the source of the metric data.
5. Select the Operation on Method Parameter to specify how the metric data is processed.
6. Select how the data should be rolled up (average or sum) from the Data Rollup drop-down menu.
7. Click **Create Custom Metric**.

To define transaction snapshot data collected

1. Click the **Snapshot Data** tab.
2. Click **Add**.
3. In the Add Snapshot Data window, enter a display name for the snapshot data.
4. Select the Collect Data From radio button to specify the source of the snapshot data.
5. Select the Operation on Method Parameter to specify how the snapshot data is processed.
5. Click **Save**.

Learn More

- Configurations for Custom Exit Points
- Monitor Remote Services
- Monitor Databases

Getter Chains in .NET Configurations

- Property Getter Chains in .NET Configurations
- Learn More

This topic provides some guidance and examples of the correct syntax for getter chains in AppDynamics configurations.

Property Getter Chains in .NET Configurations

Support in .NET configurations is limited to property getter chains. Only simple properties and fields can be chained.

Use the syntax:

<Property>.<Value>.<Member>

as in

```
Request.Url
```

or

```
Amount.Total
```

Getter chains of methods and array elements are not supported for .NET.

Learn More

- Configure Business Transaction Detection
- Configure Data Collectors

Enable Thread Correlation (.NET)

- To enable thread correlation for .NET applications
- Learn More

This topic describes how to enable thread correlation for .NET applications.

To enable thread correlation for .NET applications

AppDynamics supports instrumentation for Thread.Start and ThreadPool.QueueUserWorkItem on the Common Language Runtime (CLR) 2.x and CLR 4.x.

1. Edit the application.config file and uncomment the relevant code:

```

<instrumentation>
    <!-- Uncomment the below to enable thread correlation -->
    <!--instrumentor name="ThreadCorrelationThreadPoolCLR2Instrumentor"
disable="false"/-->
    <!--instrumentor name="ThreadCorrelationThreadPoolCLR4Instrumentor"
disable="false"/-->
    <!--instrumentor name="ThreadStartCLR2Instrumentor" disable="false"/-->
    <!--instrumentor name="ThreadStartCLR4Instrumentor" disable="false"/-->
</instrumentation>

```

2. Save the file.
3. Restart the Coordinator and relevant CLR processes.

Learn More

- Configure Backend Detection
- Monitor Remote Services

Configure the .NET Machine Agent

- The .NET Machine Agent
 - To configure the .NET Machine Agent
- The Standalone Machine Agent
 - To Download and Install the Standalone Machine Agent
 - To Configure the Standalone Machine Agent to Run Automatically on Startup
- Learn More

The .NET Machine Agent

There is a Machine Agent embedded in AppDynamic's App Agent for .NET, which is automatically installed with the app agent. You must configure the .NET Machine Agent before it can capture metrics in your managed environment.

The .NET Machine Agent does not support the extensions (plugins, metric listener, orchestration) that are available in the Standalone Machine Agent (which is a Java application).

You configure the .NET Machine Agent using the AppDynamics Agent Coordinator (the Coordinator) that also ships with the App Agent for .NET. The Coordinator determines the instrumentation and specifies the metrics captured by the Machine Agent.

To configure the .NET Machine Agent

1. Open the application.config file located at <agent_install_dir>\Machine Agent\Configuration.
2. Specify how the machine agent connects to the Controller by modifying the <controller-host> and <controller-port> properties in the application.config file.

Property Name		Required	Default
Controller Host <i>For on-premise Controller installation:</i> Provide value as configured for "Application Server Host Name". <i>For SaaS Controller service, refer to the Welcome mail.</i>	<controller-host>	Yes	None
Controller Port <i>For on-premise Controller installation:</i> Provide value for "HTTP Listener Port". <i>For SaaS Controller service, refer to the Welcome mail.</i>	<controller-port>	Yes	For On-premise installations: Port 8090 is the default for HTTP and port 8181 is the default for HTTPS For SaaS Controller Service: Port 80 for HTTP and port 443 for HTTPS

For example:

```
<AppDynamicsAgentCoordinator>
  <controller-info>
    <controller-host>localhost</controller-host>
    <controller-port>8090</controller-port>
    ...
  </controller-info>
</AppDynamicsAgentCoordinator>
```

3. Specify how the Machine Agent should identify the Business Application.

The machine agent requires the name of its business application and tier. For more information see [Logical Model](#).

Property Name		Required	Default
Application Name <i>Name of the business application.</i>	<application-name>	Yes	None
Tier Name <i>Name of the tier</i>	<tier-name>	Required only if no app agent is deployed on the machine.	None

For example:

```
<AppDynamicsAgentCoordinator>
  <controller-info>
  .....
  <application-name>ACMEOnline</application-name>
  <tier-name>InventoryTier</tier-name>

  .....
</controller-info>
</AppDynamicsAgentCoordinator>
```

The Standalone Machine Agent

If you want to monitor hardware metrics for an application that is running on Windows or use extensions such as the HTTP Metric Listener, or automate a local script as part of a policy, install and configure the [standalone Machine Agent](#). The standalone Machine Agent, which is a Java application, is different from the .NET Machine Agent. The Standalone Machine Agent provides the capability to use extensions (plugins, metric listener, orchestration) that are missing from the embedded .NET Machine Agent.

To Download and Install the Standalone Machine Agent

1. From the AppDynamics download site at <http://download.appdynamics.com/>, locate the AppDynamics Machine Agent.
2. Click **Download Now**.
3. Unzip the MachineAgent.zip file.

To Configure the Standalone Machine Agent to Run Automatically on Startup

1. Make sure that Java is installed on the machine.
2. Create a batch file containing the following line:

```
java -Xmx8m -jar <pathtomachineagent>/machineagent.jar
```

If your application loads a large number of extensions into memory, you may want need to increase the size of the memory allocation or drop it altogether:

```
java -jar <path to machineagent>/machineagent.jar
```

3. Click **Control panel -> Scheduled Tasks.**
4. Create a new task.
5. Select the batch file to execute. This is the file created in step 2.
6. Select **When my computer starts.**
7. Enter the administrator's credentials to run the Machine Agent as that user.
8. Click **Finish.**

To start the Machine Agent for the first time, right-click on the scheduled task, and click **Run**.

Learn More

- Install the Machine Agent
- Install the App Agent for .NET
- Monitor CLRs

Enable Correlation for .NET Remoting

Correlation for .NET Remoting is not enabled by default. This topic describes how to enable correlation for .NET Remoting.

To enable correlation for .NET Remoting

1. Edit the application.config file and locate the <instrumentation> element. The XML looks similar to the following:

```
<instrumentation>
    <!-- Uncomment the below to enable thread correlation -->
    <!--instrumentor name="ThreadCorrelationThreadPoolCLR2Instrumentor"
disable="false"/-->
    <!--instrumentor name="ThreadCorrelationThreadPoolCLR4Instrumentor"
disable="false"/-->
    <!--instrumentor name="ThreadStartCLR2Instrumentor" disable="false"/-->
    <!--instrumentor name="ThreadStartCLR4Instrumentor" disable="false"/-->
</instrumentation>
```

2. Add lines these to the <instrumentation> element in machine agent configuration file:

```
<instrumentor name="RemotingEntryInstrumentor" disable="false">\>
<instrumentor name="RemotingExitInstrumentor" disable="false">\>
```

2. Save the file.
3. Restart the Coordinator and relevant CLR processes.

 For .NET Remoting, AppDynamics supports HTTP and Net.TCP protocols, but does not support the net.pipe protocol.

Learn More

- Configure Backend Detection
- Monitor Remote Services

Tutorials for .NET

Monitoring Tutorials for .NET

Coming Soon!

Overview Tutorials for .NET

Coming Soon!

Troubleshooting Tutorials for .NET

Coming Soon!

Automate Scaling in Windows Azure

- Set Up the Windows Azure Configuration
 - To Set Up Your Windows Azure Configuration:
- Create Scale Actions
 - To Create a Scale Action
- Automate Scale Actions
 - To Associate an Action with a Policy
- Learn More

You can create actions to scale up or scale down the number of instances in a tier in response to your application's changing requirements. Then you can configure these actions to execute automatically when a policy violation occurs.

A maximum of 10 policies can have automatic scaling actions on them.

For information about policies, see [Policies](#).

To use automatic scaling for Windows Azure:

1. Set up your Windows Azure configuration.
2. Create scale up and scale down actions.
3. Automate when the scale actions are triggered.
4. Observe automatic scaling activity.

To access Automate Scaling in Windows Azure configuration, click **Alert & Respond -> Azure Auto-Scaling** in the left navigation pane.

Set Up the Windows Azure Configuration

Before you can create automatic scaling actions, you need to set up your Windows Azure configuration to enable the AppDynamics Controller to connect with your Azure account.

You need your Windows Azure account credentials and your Windows Azure Key Store and Trust Store certificates, as well as your AppDynamics password, to complete this step.

The first time you publish on Azure from Visual Studio, it automatically downloads a publish settings file in your personal certificate store, called <certificate_name>.publishsettings. From the Certificate Wizard, export the .publishsettings file once to a .pfx file and once

as a .cer file, using the same base name (<certificate_name>.pfx, <certificate_name>.cer).

For more information about Windows Azure certificates, see [Overview of Certificates in Windows Azure](#).

To Set Up Your Windows Azure Configuration:

1. Click **Automate -> Automatic Scaling in Azure** in the left navigation pane.
2. In the Azure Configuration panel click **Setup Azure Configuration**.
3. In the Azure Configuration screen, enter your Azure credentials: subscription ID, certificate alias (same as the base name as the .cer and .pfx files, <certificate_name>), and the key store password that corresponds to your key store certificate.
4. If you are not already logged into AppDynamics with your username and password, you will also need to enter your AppDynamics password.

Azure Configuration

Subscription ID

Certificate Alias

Key Store Password

AppDynamics Password

Please enter your AppDynamics password

Cancel Save

5. Click **Save**.

6. In the Azure Certificates section that appears after you save the configuration, click **Upload Key Store(.pfx) file** and **Upload Trust Store(.cer) file** to validate your Windows Azure credentials. (Remember that the .pfx file and the .cer file must have the same base name, which must be the same name as the certificate alias.)

Azure Certificates

Upload the Key Store and Trust Store Certificates from your Azure Account.

Key Store (.pfx)

Upload Key Store (*.pfx) file

Trust Store (.cer)

Upload Trust Store (*.cer) file

After Azure Configuration is set up, you can edit it at any time. You need to do this if your Windows Azure certificates change.

Create Scale Actions

When you need more capacity, create a scale up action to spawn additional VM instances.

When you have excess capacity, create a scale down action to terminate unneeded VM instances.

Scale actions are created at the tier level, so if you want to create similar actions for multiple tiers you need to create separate actions, one for each tier. A tier in AppDynamics corresponds to a role in Windows Azure.

To Create a Scale Action

1. Click **Automate -> Automatic Scaling in Azure** from the left navigation pane.

2. In the Define Scale Up/Down Actions panel click the + icon.

3. In the Create Action screen, enter a unique name and an optional description for the action. Choose a name that clearly defines the action but is not too long. This name identifies this action in the Define Scale Up/Down Actions panel below as well as in drop-down menus in the policy configuration screens that let you configure the policy that to trigger the action.

4. Enter the name of the Windows Azure hosted service for which you are creating this action.

5. From the Tier drop-down menu, select the tier in which the action will be performed. If you want to apply the action to multiple tiers, create a separate action for each tier.

6. Select Scale Up or Scale Down from the Scale Up or Down drop-down menu. Scaling up spawns new instances. Scaling down terminates instances.

7. From the Maximum Instances drop-down menu, set the maximum number of instances permitted in the tier. If a scale up action would cause the number of instances to exceed this level, the action is not performed. This value is constrained by your CPU core limit for the role in Windows Azure. For example, if your core limit per role is 100 and each instance in the role takes 4 cores, then the maximum number of instances in the core should be 25 or less.

Your application could end up in an undefined state if you try to spawn instances that exceed your core limit per role. You need to track the number instances being consumed so that you do not exceed your core limit.

8. From the Minimum Instances drop-down menu, set the minimum number of instances permitted in the tier. If a scale down action would cause the number of instances to fall below this level, the action is not performed.

9. From the Instances to Spawn/Instances to Terminate drop-down menu, select the number of instances to spawn (if scaling up) or to terminate (if scaling down).

When the action executes, this is the number of instances that it will spawn or terminate on a single invocation.

10. From the Instance Deployment Slot drop-down menu, select Production or Staging.

11. Click **Create** to create the action.

Create Action

Name	ScaleUp Order Service
Description	Spawn 1 instance in OrderService tier.
Hosted Service Name	NwTraders-Azure
Hosted Service Name in Azure. Normally this will be the same as the Application name.	
Tier	OrderService
Scale Up or Down	Scale Up
Maximum Instances	10
The maximum number of Instances (Nodes) for this Tier. Scaling will stop when this limit is reached.	
Minimum Instances	1
The minimum number of Instances (Nodes) for this Tier. Scaling will stop when this limit is reached.	
Instances to Spawn	1
Instance Deployment Slot	Production
Cancel Create	

You can edit an existing action by selecting it in this panel and clicking the edit icon, and you can remove it by clicking the remove icon. You cannot remove an action that a policy is configured to trigger. It is necessary first to disassociate the policy from the action by setting its autoscaling action to another action or to Do Nothing.

Automate Scale Actions

To automate execution of a scale action, associate the action with one or more policies. Then, when a violation of the associated policy occurs, the scale action is automatically triggered.

To Associate an Action with a Policy

Perform these steps for every policy for which you want to automate an action.

1. Do one of the following:

- Click **Automate -> Automatic Scaling in Azure** in the left navigation pane.
- In the Define Scale When Scale Up/Scale Down Actions will Automatically Run panel, click **Configure All Policies**.

OR

- Click **Configure -> Policies** in the left navigation pane.

The Policies panel opens.

2. Either select an existing policy to trigger the action and click the edit icon, or click the add (+) icon to create a new policy to trigger this action.

The policy wizard opens. For information about creating or editing a policy see [Configure Policies](#).

3. In the policy wizard, navigate to the Critical Condition page.

4. From the Azure Auto-Scaling Action to execute if this condition is violated drop-down menu, select the action to trigger.

A screenshot of a dropdown menu titled "Azure Auto-Scaling Action to execute if this condition is violated:". The menu contains four options: "Do nothing", "AzureAction", and "AzureScaleDown". The "AzureScaleDown" option is highlighted with a blue background, indicating it is selected.

If there is no appropriate action to trigger for this policy, click **Configure Azure Auto-Scaling**, which sends you back to the Create Action screen where you can create an action. See [Create Scale Actions](#).

5. If you want to trigger the action for the warning condition also, navigate to Warning Condition page of the wizard and the repeat step 4 there.

6. Click **Save** to update the policy.

A row for the policy and its associated action appears in the Define when Scale Up/ Down Actions will Automatically Run list in the Automatic Scaling for Azure screen.

The action will be automatically performed when its associated policy is violated.

You can observe messages generated by automatically executed scale actions in the Azure Scaling Activity panel at the bottom of the Automatic Scaling for Azure screen.

Learn More

- [Use Service Management API of Azure Java SDK](#)

AppDynamics Administration

This section provides system administration information for managing AppDynamics installations.

Release Notes for AppDynamics Pro

- New and Enhanced Features in 3.7
 - End User Monitoring
 - Custom Dashboards
 - New Alert and Respond Mechanism Including Runbook Automation
 - Upgrading from 3.6 Policies
 - Custom Correlation
 - Controller Updates
 - Sub-Millisecond Metrics
 - REST API Updates
 - New Health Rule Violation URI
 - IP Addresses Retrieved for Nodes
 - App Agent for Java Updates
 - Automatic Naming for Application, Tier, and Node
 - New Supported Environments for the App Agent for Java
 - App Agent for .NET Updates
 - New User Interface
 - New UI for Setup
 - Renamed Menu Item
 - New UI for Integrations
 - Integrate with AppDynamics for Databases
 - New Compute Clouds
- Agent-Controller Compatibility Matrix
- Controller Notes
 - Installation Notes for the Controller
 - Linux Considerations
 - Upgrade Notes for the Controller
- Agent Notes
 - App Agent for Java
 - App Agent for .NET

This topic covers release information for AppDynamics Pro versions 3.7.x.

If you are reading this in PDF format, many links will open in the product documentation wiki. You must have a registered user login to access the product documentation.

AppDynamics continuously improves its product documentation. The most current release notes are on the wiki at [Release Notes for AppDynamics Pro](#).

New and Enhanced Features in 3.7

See the AppDynamics [marketing](#) video overviews of the new features.

See the new product demo videos at [AppDynamics in Action Videos](#).

End User Monitoring

AppDynamics has redesigned the End User Monitoring (EUM) feature originally introduced with release 3.4.

- Separate account registration for EUM
- JavaScript injection into the page. Injection is no longer serviced by the app agent
- User experience tracked in terms of pages, not business transactions. A page is the user's view on a single browser window. There is a page list and page dashboards for each instrumented page with visibility into pages that are slow or have errors with browser snapshots.
- There is a new set of EUM metrics.

For details see:

- [EUM License](#)
- [Monitor End User Experience](#)
- [EUM Metrics](#)
- [Geo Dashboard](#)
- [Page List](#)
- [Page Dashboard](#)
- [Browser Snapshots](#)
- [Configure End User Experience](#)
- [Use the AppDynamics REST API](#)

Custom Dashboards

In release 3.7.0 custom dashboard functionality is completely revised, including:

- A contemporary look-and-feel
- Dashboards now render in HTML5 for viewing on many devices
- New widgets for image and iframe
- New editor supports advanced metrics including "Top 10" lists
- Full dashboards can be embedded in other dashboards using sharing, iframes, and URLs
- External data can be embedded in dashboards using iframes and URLs
- Support for metric wild-carding such as "all nodes whose names start with..."
- Supports anonymous sharing to provide access without requiring credentials
- Can copy dashboards on the same Controller
- Can include baseline data in charts

Pre-3.7 dashboards are still supported and can be edited using pre-3.7 functionality. However pre-3.7 dashboards cannot be imported to the new 3.7 editor.

Dashboard import/export is not supported.

The scatterplot widget is no longer provided.

See [Custom Dashboards](#) and [Create a Custom Dashboard](#).

New Alert and Respond Mechanism Including Runbook Automation

The policy engine has been simplified to separate the identification of problems and the creation of response actions to allow for greater flexibility and more finely-tuned proactive monitoring and problem remediation. The conditional part of the pre-3.7 policy creation is now captured in health rules, which are created as separate entities. A health rule violation is a type of event.

Remedial actions are also created as separate entities and have been expanded to include local scripts, thread dumps and diagnostic sessions. Email and SMS alerts are now a type of action called a notification action.

A policy now consists of a trigger, configured as one or more events which might or might not be health rule violation events, and one or more actions to execute when the triggering events occur.

See:

- [Policies](#)
- [Health Rules](#)
- [Actions](#)

Upgrading from 3.6 Policies

When a 3.6 application is upgraded to 3.7, the upgrade process performs the following conversions with regard to policies:

- Converts the critical and warning conditions in your 3.6 policies to 3.7 health rules.
- Converts the alerts in your 3.6 policies to 3.7 notification actions.
- Converts the actions in your 3.6 policies to 3.7 cloud auto-scaling actions.
- Converts your 3.6 policies to 3.7 policies using the newly-converted health rules and actions.

The names of the 3.7 policies are similar to those of the 3.6 policies, but there are two 3.7 policies for each 3.6 policy: one for the critical condition and one for the warning condition. For example, the 3.6 policy named "CPU utilization it too high" would generate two policies in 3.7: "CPU utilization it too high_critical" and "CPU utilization it too high_warning", each with its own actions if they were configured with actions and alerts in 3.6.

A "xxx_critical" policy is violated (and its actions triggered) when the "Health Rule Violation Started - Critical" or the "Health Rule Violation Upgraded - Warning to Critical" event occurs. A "xxx_warning" policy is violated (and its actions triggered) when the "Health Rule Violation Started - Warning" or the "Health Rule Violation Downgraded - Critical to Warning" event occurs.

The upgrade process deletes 3.6 alert digests and event notifications.

Before upgrading, save the information in your 3.6 event notifications and digests. Then recreate the event notifications and digests as 3.7 email digests.

See:

- [Actions](#)
- [Notification Actions](#)
- [Policies](#)
- [Configure Policies](#)
- [Alerting Wizard](#)
- [Events](#)

- Email Digests

Custom Correlation

For complex multi-threaded, distributed applications in which the AppDynamics default correlation mechanisms are not sufficient, it is possible to configure correlation across tiers, threads, and queues though configuration without requiring any code changes. Contact [AppDynamics Support](#) for details on how to configure custom correlation.

Controller Updates

- Now supported on Windows 8 and Windows Server 2012. See [Supported Environments and Versions#Controller Operating System Requirements](#).
- Controller authentication now supports LDAP nested groups, referrals, and paging. See [Configure Authentication Using LDAP](#).
- There is a new utility, modifyJvmOptions, to change the Controller's Glassfish server JVM options. When you upgrade to the latest version of the Controller the JVM settings created by this utility are preserved. See [Modify Glassfish JVM Options](#).
- When you install the Controller, there is an option to specify a parent data directory that is separate from the installation directory. The installer creates a /data subdirectory. During the upgrade process, you specify the parent directory that contains the /data directory.
- The Glassfish server used by the Controller is updated to version 3.1.2.2.

Sub-Millisecond Metrics

It is possible to retrieve Average Response Time metrics for some backends (databases and remote services) in units smaller than milliseconds. Call [AppDynamics Support](#) for instructions on how to enable this feature.

REST API Updates

New Health Rule Violation URI

As part of the new alert and respond mechanism a new URI has been added to retrieve health rule violations.

See [Retrieve all health rule violations in a particular business application](#).

The old URI (/controller/rest/applications/<application-name|application-id>/problems/policy-violations) is currently maintained for backward compatibility with pre-3.7 REST requests.

IP Addresses Retrieved for Nodes

The /controller/rest/applications/<application-name | application-id>/nodes and controller/rest/applications/<application-name | application-id>/nodes/<node-name> URIs have been updated to retrieve the IP addresses of the nodes.

App Agent for Java Updates

Automatic Naming for Application, Tier, and Node

The App Agent for Java javaagent command has a new uniqueID argument that AppDynamics uses to automatically name the node and its tier. For example, using this command argument AppDynamics will name the node "my-app-jvm1" and the tier "my-app-jvm1":

```
-javaagent:<agent_home>/javaagent.jar=uniqueID=<my-app-jvm1>
```

When uniqueID is used, AND when an application name is not provided either through the system property or in the controller-info.xml, AppDynamics creates a new business application called "MyApp".

The new naming mechanism is used by the new Self-Service process. See [Install Agents for 5 or fewer JVMs or CLRs \(Self-Service Installations\)](#).

New Supported Environments for the App Agent for Java

- IBM JVM 1.7.x
- OpenJDK 1.6.x
- TomEE 1.0

- WebLogic 11.x, 12.x
- WebSphere 8.x
- Coherence 3.7.1 caching

See [Supported Environments and Versions](#).

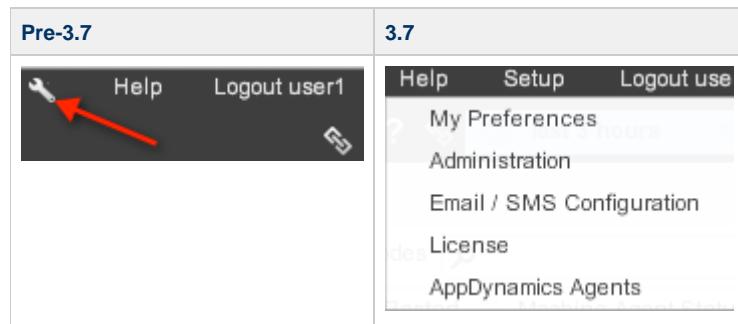
App Agent for .NET Updates

The App Agent for .NET is now supported on Windows Server 2012. See [Supported Environments and Versions](#).

New User Interface

New UI for Setup

The wrench icon in the upper right corner of the console is replaced with a Setup menu. The Security Configuration menu item has been renamed Administration.



Renamed Menu Item

The Automation menu in the left navigation pane, enabled through the Setup UI, is renamed Cloud Auto-Scaling.

New UI for Integrations

Setup -> Administration now leads to a new tab for integrations.

Not every possible integration is on this tab; as of 3.7.0 it includes Splunk and AppDynamics for Databases.

Integrate with AppDynamics for Databases

You can right-click in the databases list or on any database icon in a flow map to link to AppDynamics for Databases to get database performance metrics if you have integrated with AppDynamics for Databases. See [Integrate with AppDynamics for Databases](#).

New Compute Clouds

IBM Smart Cloud Enterprise, VMware vSphere, and Rackspace OpenStack Private Cloud connectors for compute clouds are now included in the installation. For instructions on registering compute clouds, see [Compute Clouds](#).

Agent-Controller Compatibility Matrix

The Controller supports older Agents. Newer Agents will not work with an older Controller version. For more information, see the [Agent - Controller Compatibility Matrix](#).

Controller Notes

Install the Controller if you are not using the AppDynamics SaaS Controller Service. If you are using the SaaS Controller, skip the Controller installation.

Obtain the Controller for your operating system from the [AppDynamics Download Center](#).

Installation Notes for the Controller

- Before you install or upgrade the Controller, validate the hardware requirements as listed at [Controller System Requirements](#). The sizing requirements have been updated and now include recommendations based on metrics per minute in addition to the number of nodes in the installation. Features such as monitoring asynchronous threads and End User Monitoring increase the number of metrics per minute flowing to the Controller.
- AppDynamics strongly recommends that you install the Controller on a dedicated machine for adequate stability and performance.
- The disk space requirements differ for each of the Controller performance profiles. Verify the disk space requirements for your performance profile at [Controller System Requirements](#).

Linux Considerations

- Before installing Controller on Linux environments, check the [limits for file descriptors](#).
- You must have a 64-bit Linux system if you plan to use a medium, large, or extra large performance profile.
- Ensure that you assign executable permissions to the installer binary before launching the installer.
- AppDynamics Controller requires that libaio is installed on your machine. For more information see [Install libaio on Linux](#).

Upgrade Notes for the Controller

- For all major upgrades, AppDynamics recommends that you upgrade both the Controller and Agents. This ensures that you get all the latest Agent features and fixes.
- If you are upgrading both the Controller and the Agents, first upgrade the Controller then upgrade the Agents.
- Make sure that you have saved a backup of the following files before starting the upgrade procedure:
 - <Controller_Installation-Directory>/db/db.cnf
 - <Controller_InstallationDirectory>/appserver/domains/domain1/config/domain.xml
 - <Agent_Installation_Directory>/conf/controller-info.xml
- If you want to save any snapshots or events, archive them before upgrading. Upgrade to 3.4 deletes all snapshots and events.
- If you have changed any global (Controller-wide) properties using the Controller Administration (admin.html) interface, note your changes. You will have to reset them after upgrade. Controller upgrade does not persist properties set in the Admin interface.

For more information see [Upgrade the Controller](#).

Agent Notes

For general Agent information see [Administer Agents](#).

App Agent for Java

Obtain the AppDynamics App Agent for Java download for your system from the [AppDynamics Download Center](#):

- For Sun and JRockit JVMs, download AppServerAgent-x.x.x.zip
- For IBM JVMs, download AppServerAgent-ibm-x.x.x.zip

For more information see:

- [Install the App Agent for Java](#)

- Java Server-Specific Installation Settings
- App Agent for Java Configuration Properties

App Agent for .NET

Obtain the AppDynamics App Agent for .NET download for your system from the [AppDynamics Download Center](#):

- For Windows 32-bit, download dotNetAgentSetup.msi
- For Windows 64-bit, download dotNetAgentSetup64.msi

For more information see:

- [Install the App Agent for .NET](#)
- [App Agent for .NET Configuration Properties](#)
- [Troubleshoot App Agent for .NET Installation and Configuration](#)
- [Configure the .NET Machine Agent](#)

For full support of the new EUM functionality, enabled EUM in the Configuration Utility. See [Configure the App Agent for .NET](#) and [Configure the .NET App Agent for EUM](#).

The following features are currently not available for .NET environments:

- Manual BCI EUM instrumentation also called [Assisted Injection Using Injection Rules](#).
- [Remediation Actions](#), such as running local scripts on nodes.
- Diagnostic thread dump actions are not supported.
- Reassigning nodes is not supported.
- Deleting tiers is not supported.
- Adding custom machine agent metrics (also called custom monitors) is not supported.
- Detecting code deadlocks is not supported.
- Aggressive snapshot collection is not supported.
- Memory monitoring (also known as Object Instance Tracking) is not supported.
- The embedded Machine Agent (installed by default) for the App Agent for .NET does not support extensions. If you want to use Machine Agent extensions such as custom plugins or the HTTP Metric Listener, or do orchestration to compute clouds, install the standalone Machine Agent. See [Configure the .NET Machine Agent](#).
- Some agent node properties are not supported in the .NET environment. Review [App Agent Node Properties Reference](#) for detailed information.

Upgrade Policies from 3.6 to 3.7

- [Overview](#)
- [Pre-Upgrade Requirements](#)
 - To copy digest and event notification configurations in 3.6
- Converts the 3.6 policies configuration, minus the alerts and actions, to health rules in 3.7
- Converts the alerts in the 3.6 policies to notification actions in 3.7
- Converts the actions in the 3.6 policies to cloud auto-scaling actions in 3.7
- Converts the 3.6 policies to 3.7 policies using the newly-converted health rules and actions
- [Post-Upgrade Requirements](#)
 - To re-create email digests in 3.7:
- [Learn More](#)

Overview

When a 3.6 application is upgraded to 3.7, the upgrade process performs the following conversions with regard to policies:

- Converts the 3.6 policies configuration, minus the alerts and actions, to health rules in 3.7
- Converts the alerts in the 3.6 policies to notification actions in 3.7
- Converts the actions in the 3.6 policies to cloud auto-scaling actions in 3.7
- Converts the 3.6 policies to 3.7 policies using the newly-converted health rules and actions

The upgrade process deletes 3.6 alert digests and event notifications, so you need to save and recreate them.

Pre-Upgrade Requirements

Before you upgrade to 3.7, manually save your 3.6 event notification and digest configurations.

To copy digest and event notification configurations in 3.6

1. In the left navigation pane click **Configure -> Alerts**.
2. In the alerts configuration screen scroll down to the Send Alerts as Digest section. Your digests are listed here.
3. For every digest that you will want to recreate in 3.7:
 - a. Select the digest.
 - b. Click the Edit icon.
 - c. Copy the digest configuration.

Name	Enabled	Event Types	Alerts
Email Digest	✓	5 Types: Policy Violation Started, Application	2 Alerts: Admins, Incident SMS
Splunk Event Notification	✓	11 Types: Application	1 Alerts: Notify Splunk of Events

Converts the 3.6 policies configuration, minus the alerts and actions, to health rules in 3.7

The upgrade process gives the new 3.7 health rules names that are similar to the names of the 3.6 policies.

The upgrade process does not convert the actions and alerts in the 3.6 policies to health rules, because health rules do not include alerts and actions.

Name
Notify Splunk of Policy Violation

Name

Click **Alert & Respond -> Health Rules** in the left navigation pane to see your health rules in 3.7.

Converts the alerts in the 3.6 policies to notification actions in 3.7

Alerts configured in the alert section of the 3.6 policy wizard

Name
Notify Splunk of Policy Violation

or in this section of the 3.6 alert configuration screen

Configure Alerts

Configure when Alert will be sent

Name
Notify Splunk of Policy Violation
Notify Splunk of Events
Admins
Incident SMS

become notification actions in 3.7.

Notification actions include notification by email, SMS or custom action.

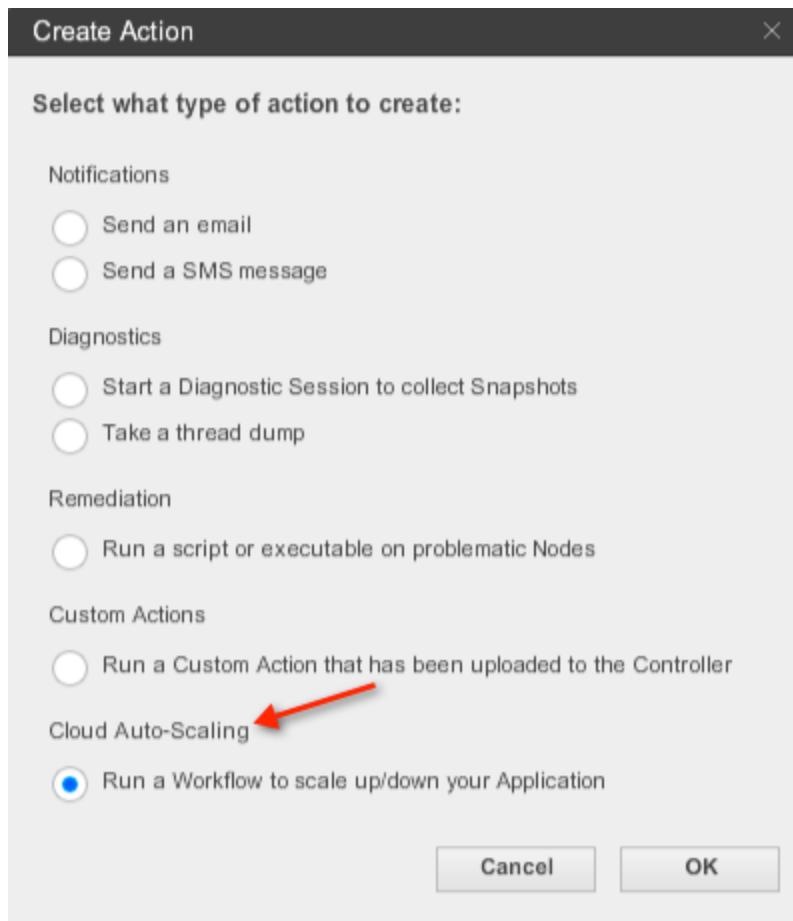
The screenshot shows a list of actions under the 'Actions' tab. The columns are 'Name', 'Type', and 'Policies or Email Digests that execute this Action'. The actions listed are:

Name	Type	Policies or Email Digests that execute this Action
Notify Splunk of Policy Violation	Custom Action	Policy Notify Splunk of Violations_Warning Condition, Policy Notify Splunk of Violations_Critical Condition
Notify Splunk of Event	Custom Action	Policy Notify Splunk of Events
tester1@appdynamics.com	Email	Email Digests
6505551234	SMS	

Click **Alert & Respond ->Actions** in the left navigation pane to see your actions in 3.7. If you have different types of actions and want to see only notification actions (formerly known as alerts), check the Notifications check box in the filter above the list.

Converts the actions in the 3.6 policies to cloud auto-scaling actions in 3.7

Actions configured in the action section of the 3.6 policy wizard become cloud-auto-scaling actions in 3.7:



Click **Alert & Respond ->Actions** in the left navigation pane to see the actions in 3.7. If you have different types of actions and want to see only cloud auto-scaling actions, check the Cloud Auto-Scaling check box in the filter above the list.

Converts the 3.6 policies to 3.7 policies using the newly-converted health rules and actions

The names of the 3.7 policies are similar to those of the 3.6 policies, but there are two 3.7 policies for each 3.6 policy: one for the critical condition and one for the warning condition. For example, the 3.6 policy named "CPU utilization is too high" would generate two policies in 3.7: "CPU utilization it too high_critical" and CPU utilization it too high_warning", each with its own actions if they were configured with actions and alerts in 3.6.

A "xxx_critical" policy is violated (and its actions triggered) when the "Health Rule Violation Started - Critical" or the "Health Rule Violation Upgraded - Warning to Critical" event occurs. A "xxx_warning" policy is violated (and its actions triggered) when the "Health Rule Violation Started - Warning" or the "Health Rule Violation Downgraded - Critical to Warning" event occurs.

You can fine-tune the triggers for these policies as well as modify their actions. See [Configure Policies](#).

Post-Upgrade Requirements

After you upgrade to 3.7, manually recreate your saved 3.6 event notifications and digest configurations as email digests.

To re-create email digests in 3.7:

1. In the left navigation pane click **Email Digests**.
2. For every digest that you want to recreate in 3.7, configure an email digest using the events and recipients that you saved from the 3.6 configuration. See [Configure Email Digests](#) for details on how to create an email digest.

Edit Email Digest - Email Digests

CONTENTS	Name <input type="text" value="Email Digests"/> Enabled <input checked="" type="checkbox"/>	
RECIPIENTS	This Email Digest will include ▾ these Events when they are occurring on ▾ any object	
HOW OFTEN	Health Rule Violation Events <input checked="" type="checkbox"/> Health Rule Violation Started - Warning <input checked="" type="checkbox"/> Health Rule Violation Started - Critical <input type="checkbox"/> Health Rule Violation Upgraded - Warning to Critical <input type="checkbox"/> Health Rule Violation Downgraded - Critical to Warning <input type="checkbox"/> Health Rule Violation Ended Other Events <input type="checkbox"/> Slow Transactions <input type="checkbox"/> Code Problems <input type="checkbox"/> Application Changes <input type="checkbox"/> AppDynamics Config Warnings	
	What Health Rules? <input checked="" type="radio"/> Any Health Rule <input type="radio"/> These Health Rules:	

Edit Email Digest - Email Digests

CONTENTS	Name <input type="text" value="Email Digests"/> Enabled <input checked="" type="checkbox"/>					
RECIPIENTS	Digest Notification Recipients					
HOW OFTEN	 <table border="1" style="width: 100%;"> <thead> <tr> <th>Name</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td>tester1@appdynamics.com</td> <td>Email</td> </tr> </tbody> </table>		Name	Type	tester1@appdynamics.com	Email
Name	Type					
tester1@appdynamics.com	Email					

Learn More

- Actions
- Notification Actions
- Policies
- Configure Policies
- Alerting Wizard
- Events
- Email Digests
- Configure Email Digests
- Custom Actions

Upgrade End User Monitoring

- EUM Upgrade Overview
 - To upgrade to the new and improved EUM:
- 3.6 EUM Metrics
- Learn More

End User Monitoring (EUM) was completely redesigned in AppDynamics 3.7. and the metrics collected by the new version are different from those in 3.6. For a description of the new metrics see [EUM Metrics](#).

This topic explains the procedure for upgrading EUM from 3.6 to 3.7.

EUM Upgrade Overview

When you upgrade your controller from 3.6.x to 3.7, the Controller upgrade script determines whether your agents have been using EUM based on existing metrics.

AppDynamics disables EUM for all users that are upgrading.

To upgrade to the new and improved EUM:

1. Contact your AppDynamics sales representative to get an EUM license key for 3.7.
2. After you receive the license key, upgrade all agents to 3.7.

3. Configure and enable EUM for 3.7. See [Configure End User Experience](#).

After the license key is provided, the agents upgraded, and EUM is enabled and configured, the new EUM metrics will begin appearing in the Metric Browser and EUM dashboards.

3.6 EUM Metrics

After upgrade, your 3.6 EUM metrics are displayed in the Metric Browser under a node named EUM 1.0. These metrics do not appear in any of the EUM dashboards.

All 3.6 EUM configurations are retained in the 3.7 EUM configuration except for Geo Server URL and Attribute Injection, both of which are set to their defaults which are blank for the Geo Server URL and disabled for Attribute Injection. For information about resetting them in 3.7 see [Configure the Geo Server Location](#) and [To enable attribute injection](#).

Learn More

- Monitor End User Experience (EUM)
- Configure End User Experience
- EUM Metrics

Agent - Controller Compatibility Matrix

- Java Agent - Controller Compatibility Matrix
- .NET Agent - Controller Compatibility Matrix
- Learn More

Java Agent - Controller Compatibility Matrix

The Controller supports older Agents. Prior to 3.6.2, newer agents will not work with an older Controller. 3.6.2 and newer agents will work with the Controller that has the same major and minor version (X.x). Caveats or exceptions are noted in the matrix.

Controller	3.5.2	3.5.3	3.5.4	3.5.5	3.5.6	3.5.7	3.6	3.6.1	3.6.2-6	3.7.0
Java Agent										
3.5.2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3.5.3		✓	✓	✓	✓	✓	✓	✓	✓	✓
3.5.4			✓	✓	✓	✓	✓	✓	✓	✓
3.5.5				✓	✓	✓	✓	✓	✓	✓
3.5.6					✓	✓	✓	✓	✓	✓
3.5.7						✓	✓	✓	✓	✓
3.6							✓	✓	✓	✓
3.6.1								✓	✓	✓
3.6.2-6									✓	✓
3.7.0										✓

.NET Agent - Controller Compatibility Matrix

The Controller supports older Agents. Prior to 3.6.2, newer agents will not work with an older Controller. 3.6.2 and newer agents will work with the Controller that has the same major and minor version (X.x). Caveats or exceptions are noted in the matrix.

3.5.3 is the version of the .NET Agent released following 3.4.2. There were no releases in-between 3.4.2 and 3.5.3.

The .NET Agent did not release a 3.5.6 version.

Controller	3.5.2	3.5.3	3.5.4	3.5.5	3.5.7	3.6.0	3.6.1	3.6.2-6	3.7.0
.NET Agent									
3.5.3	✓	✓	✓	✓	✓	✓	✓	✓	✓
3.5.4		✓	✓	✓	✓	✓	✓	✓	✓
3.5.5			✓	✓	✓	✓	✓	✓	✓
3.5.7				✓	✓	✓	✓	✓	✓
3.6					✓	✓	✓	✓	✓
3.6.1						✓	✓	✓	✓
3.6.2-6							✓	✓	✓
3.7.0								✓	

Learn More

- Supported Environments and Versions

Supported Environments and Versions

- Supported Platform Matrix for the AppDynamics Controller
 - Controller Operating System Requirements
 - Supported Web Browsers for the Controller UI
- Supported Platform Matrix for the App Agent for Java
 - Supported JVMs for the App Agent for Java
 - Supported Application Servers and Portals for the App Agent for Java
 - Supported Backends for the App Agent for Java
 - Supported JDBC Drivers for the App Agent for Java
- Supported Platform Matrix for the App Agent for .NET
 - Supported Versions of .NET for the App Agent for .NET
 - Supported Versions of Windows and IIS for the App Agent for .NET
 - Supported Standalone .NET Applications
 - Supported Versions of Microsoft Windows Server for the App Agent for .NET
 - Supported Frameworks and Protocols for the App Agent for .NET
 - Supported Data Storage for the App Agent for .NET
 - Supported Remote Services for the App Agent for .NET
 - Supported ADO.NET Clients for the App Agent for .NET
- Supported Platform Matrix for the Standalone Machine Agent
 - Supported Platforms for Default Hardware Monitor Plugin
- Feature-Specific Compatibility Information
 - Memory Monitoring Compatibility
 - Memory Monitoring Compatibility in Java Environments
 - Automatic Leak Detection in Java Environments
 - Object Instance Tracking in Java Environments
 - Custom Memory Structures in Java Environments
 - End User Monitoring (EUM) Compatibility
 - End User Monitoring Browser Compatibility
 - End User Monitoring (EUM) Compatibility in Java Environments
 - End User Monitoring (EUM) Compatibility in .NET Environments
 - Supported Application Servers for .NET EUM

Supported Platform Matrix for the AppDynamics Controller

Controller Operating System Requirements

The Controller is supported on the following Operating Systems:

Linux (32 and 64-bit)	Microsoft Windows (32 and 64-bit)
<ul style="list-style-type: none"> Redhat Enterprise Linux (RHEL) 6.1, 6.2 CentOS 6.1, 6.2 Fedora 14 Ubuntu 8, 12 Open SUSE 11.x SUSE Linux Enterprise Server Cloud: Amazon EC2, Rackspace, Azure 	<ul style="list-style-type: none"> Windows 2003 Server Windows 2008 Server Windows 2012 Server Windows 8

Supported Web Browsers for the Controller UI

The following browsers are supported:

- Mozilla FireFox v 3.x, 4.x, 5.0
- Internet Explorer 6.x, 7.x, 8.x, 9.0.
- Safari 4.x, 5.x
- Google Chrome 10.x, 11.x, 12.x

The Controller UI requires Flash Player 10 or greater; AppDynamics recommends version 11.

Supported Platform Matrix for the App Agent for Java

See the following sections for App Agent for Java compatibility matrix:

- Supported JVMs for the App Agent for Java
- Supported Application Servers and Portals for the App Agent for Java
- Supported Backends for the App Agent for Java
- Supported JDBC Drivers for the App Agent for Java

Supported JVMs for the App Agent for Java

The following JVMs are supported for application server agents:

- Java Hotspot (TM) 1.5.x, 1.6.x, 1.7.x
- BEA JRockit (R) 1.5.x, 1.6.x
- IBM JVM 1.5.x, 1.6.x, **1.7.x**
- OpenJDK **1.6.x**, 1.7.x

i Note: See [Compatibility Matrix for Memory Monitoring](#) for information on supported JVMs for the AppDynamics memory management feature.

Supported Application Servers and Portals for the App Agent for Java

The following application servers are supported:

Application Servers	Messaging	Frameworks	SOA	RMI

ColdFusion 8.x, 9.x GlassFish v2, v3 JBoss 4.x, 5.x, 6.x, 7.x Jetty 6.x, 7.x Oracle 10.3.0, 10.0.2 OC4J (<i>Oracle Application Server</i>) OSGi Infrastructure (<i>Felix, Equinox, Apache Sling</i>) Resin Solr Tomcat 5.x, 6.x, 7.x TomEE 1.0 WebLogic 9.x, 10.x, 11.x, 12.x Webmethods WebSphere 6.1+, 7.x, 8.x	Active MQ 5.x Fiorano MQ IBM MQ Series 6.x, 7.x IBM WAS Embedded JMS 6.1+, 7.x JBoss MQ 4.x, 5.x Oracle AQ-JMS Open MQ Progress Sonic MQ Rabbit MQ Rabbit MQ Spring client Tibco RV Weblogic Embedded JMS 9.x, 10.x	Applets AWT/Swing/RCP BlazeDS Cassandra with Thrift Framework EJB 2.x, 3.x Hibernate JMS Message Listeners 1.x Wicket JSF Oracle Coherence Spring Beans 2.x, 3.0 Struts 1.x Servlets 2.x Struts Action 1.x, 2.x Tapestry	Axis 1.x, 2.x CXF 2.1 Glassfish Metro JAX WS JBoss Native Stack (<i>For JBoss 4.x, 5.x</i>) OC4J (<i>Oracle Application Server</i>) Weblogic JAX-WS (<i>For Weblogic 9.x, 10.x</i>) Weblogic JAX-RPC (<i>For Weblogic 9.x, 10.x</i>) Webmethods Glue Websphere JAX-WS (<i>For Websphere 6.1.x, 7.x</i>) Websphere JAX-RPC (<i>For Websphere 6.1.x, 7.x</i>) XFire 1.x	JBoss RMI (<i>JBoss 4.x, 5.x</i>) OC4J (<i>Oracle Application Server</i>) Weblogic RMI (<i>Weblogic 10.x</i>) Websphere RMI (<i>Websphere 7.x</i>)
---	--	--	---	---

Supported Backends for the App Agent for Java

The following backends are supported:

JDBC Backends and Databases	Binary Remoting Protocol	Third Party Caches	Other Backends
Oracle 8i Oracle 9i Oracle 10g Oracle 11g IBM DB2 9.x Informix MS SQL Server 2005 MS SQL Server 2008 Postgres 8.x, 9.x MySQL 5.x PointBase Derby Sybase	Apache Thrift Spring HTTP Remoting (3.4.2)	Coherence 3.7.1 Ehcache JBoss Tree Cache / Tree Cache AOP MemCache	Cassandra MongoDB Javamail SAP JCO LDAP

Supported JDBC Drivers for the App Agent for Java

The following JDBC drivers are supported:

Database	JDBC Driver
Oracle 9i,10g,11g,	Oracle- Thin, Oracle OCI, Data Direct
Oracle 8i	Oracle- Thin, Oracle OCI
DB2 v9.x	DB2, Data Direct
Informix	IBM/Informix, Data Direct
MySQL 5.x, 6.0	Microsoft SQL Server, Connector/J
MS SQL Server 2005, 2008	Microsoft SQL Server, MS -Type4, jTDS - Type4
MS SQL Server 7.0, 2005	Microsoft SQL Server, Data Direct
Postgres 8.4	JDBC3, JDBC4
Postgres 8.2, 8.3	JDBC2, JDBC2 EE, JDBC3, JDBC4
Postgres 8.0, 8.1	JDBC2, JDBC2 EE, JDBC3

Supported Platform Matrix for the App Agent for .NET

See the following sections for the App Agent for .NET compatibility matrix:

- Supported Versions of .NET for the App Agent for .NET
- Supported Versions of Windows and IIS for the App Agent for .NET
- Supported Standalone .NET applications for the App Agent for .NET
- Supported Versions for Microsoft Windows Server for the App Agent for .NET
- Supported Frameworks and Protocols for the App Agent for .NET
- Supported Backends for the App Agent for .NET
- Supported ADO.Clients for the App Agent for .NET

Supported Versions of .NET for the App Agent for .NET

.Net Framework v 2.0, 3.0, 3.5, 4.0 and 4.5.

Supported Versions of Windows and IIS for the App Agent for .NET

Microsoft IIS v 6.0, 7.0, 7.5

Supported Standalone .NET Applications

- COM+ services
- Windows Service and Console Applications

Supported Versions of Microsoft Windows Server for the App Agent for .NET

- Microsoft Windows Server v2003 (32-bit and 64-bit)
- Microsoft Windows Server v2008 (32-bit and 64-bit)
- Microsoft Windows Server v2008 R2
- **Windows Server 2012**

Supported Frameworks and Protocols for the App Agent for .NET

- Apache ActiveMQ NMS framework and related MQs
- ASP.NET
- ASP.NET MVC 2 and 3
- HTTP
- SOAP
- WCF
- Web Services
- .NET Remoting
- IBM MQ
- Tibco EMS MQ

Supported Data Storage for the App Agent for .NET

- ADO.NET
- Windows Azure Blob Storage

Supported Remote Services for the App Agent for .NET

- WCF
- HTTP
- Web Services, including SOAP
- Directory Services, including LDAP*
- Queues
 - Apache ActiveMQ
 - IBM WebSphere MQ (also known as IBM XMS)
 - Microsoft Message Queuing (MSMQ)*
 - .NET Remoting*

- Tibco Enterprise Message Service (EMS)
- Tibco Rendezvous (RV)
- MicrosoftServiceBus (Windows Azure Service Bus)
- MicrosoftServiceBusQueue (Windows Azure Service Bus Queues)
- AzureQueue (Windows Azure Queues)

* Notes: Correlation is supported unless otherwise indicated below.

- For LDAP, correlation is non-applicable
- For MSMQ, correlation for downstream calls is not supported.
- For .NET remoting, additional configuration is needed to get downstream correlation. See [Enable Correlation for .NET Remoting](#).

Supported ADO.NET Clients for the App Agent for .NET

Database Name	Database Version	Client Type	Client Version
Oracle	10, 11	odp.net	2.112.2.0
Oracle	10, 11	ms provider	1.1
MySQL	5.2	Connector/Net and ADO.NET	5, 5.2
MySQL	5.5	Connector/Net	5.2
MySQL	5.5	Connector/ODBC	5.1.8
MS SQL Server	2005, 2008	ADO.NET	2.0

Supported Platform Matrix for the Standalone Machine Agent

The standalone Machine Agent provides platform-level metrics. It has a default built-in plugin for hardware monitoring. See [Install the Machine Agent](#).

Supported Platforms for Default Hardware Monitor Plugin

Operating System	Architecture	Versions
Linux	x86	2.2, 2.4, 2.6 kernels
Linux	amd64	2.6 kernel
Linux	ppc	2.6 kernel
Linux	ppc64	2.6 kernel
Linux	ia64	2.6 kernel
Linux	s390	2.6 kernel
Linux	s390x	2.6 kernel
Solaris	Sparc-32	2.6, 7, 8, 9, 10
Solaris	Sparc-64	2.6, 7, 8, 9, 10
Solaris	x86	8, 9, 10
Solaris	x64	8, 9, 10
AIX	ppc	4.3, 5.1, 5.2, 5.3, 6.1
AIX	ppc64	5.2, 5.3, 6.1
HP-UX	PA-RISC	11
HP-UX	ia64	11
FreeBSD	x86	4.x

FreeBSD	x86	5.x, 6.x
FreeBSD	x64	6.x
FreeBSD	x86,x64	7.x,8.x
OpenBSD	x86	4.x,5.x
NetBSD	x86	3.1
Mac OS X	PowerPC	10.3, 10.4
Mac OS X	x86	10.4, 10.5, 10.6
Mac OS X	x64	10.5, 10.6
Windows	x86	NT 4.0, 2000 Pro/Server, 2003 Server, XP, Vista, 2008 Server, 7
Windows	x64	2003 Server, Vista, 2008 Server, 7

The following Linux distributions have been certified:

Distribution	Versions
Red Hat	6.2, 7.3, 8.0, 9.0
RHEL	3, 4, 5, 6
CentOS	3, 4, 5
Fedora	2, 3, 4, 5, 6, 7, 8, 9, 10
SuSE	8, 9, 10, 11
Ubuntu	6.06, 8.04, 8.10, 9.04
Debian	2.6, 3.0, 3.1, 3.2, 4.0, 5.0
VMware ESX	2.x, 3.0
XenServer	3.1, 3.2, 4.0, 4.1, 5.0
Slackware	10, 11
Mandrake	10
Scientific Linux	5
Gentoo	

 **Note:** If you are using a 64-bit Operating System, use only a 64-bit Java Runtime Environment (JRE). For more details see [Supported Platform Matrix for Default Hardware Monitoring Plugin](#).

Feature-Specific Compatibility Information

- [Memory Monitoring](#)
- [End User Monitoring](#)

Memory Monitoring Compatibility

- [Memory Monitoring Compatibility in Java Environments](#)

Memory Monitoring Compatibility in Java Environments

Memory management solution for Java App Server Agents provide following capabilities:

- [Automatic Leak Detection in Java Environments](#)
- [Object Instance Tracking in Java Environments](#)

- Custom Memory Structures in Java Environments

Automatic Leak Detection in Java Environments

- JDK 1.5 ¹
- JDK 1.7
- BEA JRockit JVM 1.5 ¹
- Sun JVM 1.6
- BEA JRockit JVM 1.6
- IBM JVM 1.6 ¹

¹ AppDynamics Pro Edition 3.1.1 version onwards

 **Note:** The JVM needs a **restart** after enabling the 'Automatic Leak Detection' feature.

Object Instance Tracking in Java Environments

- Sun JVM 1.6

Custom Memory Structures in Java Environments

- Sun JVM 1.5
- BEA JRockit JVM 1.5
- IBM JVM 1.5 ¹
- Sun JVM 1.6
- BEA JRockit JVM 1.6
- IBM JVM 1.6*

¹. JVM restart is required after configuring the custom memory structure.

End User Monitoring (EUM) Compatibility

End User Monitoring Browser Compatibility

The following Web browsers are certified for the JavaScript Agent for EUM.

Browser	Windows	Linux	Mac	iOS(iPhone)	iOS(iPad)	Android phone
Chrome 23.x	x	x	x	x		x
Chrome 26.x			x			
Firefox 3.5	x	x	x			
Firefox 3.0	x	x	x			
Firefox 4.x	x	x	x			
Firefox 5.x	x	x	x			
Firefox 6.x	x	x	x			
Firefox 7.x	x	x	x			
Firefox 8.x	x	x	x			
Firefox 9.x	x	x	x			
Firefox 10.x	x	x	x			
Firefox 11.x	x	x	x			
Firefox 12.x	x	x	x			
Firefox 13.x	x	x	x			
Firefox 14.x	x	x	x			

Firefox 15.x	x	x	x				
Firefox 16.x	x	x	x				
Firefox 17.x	x	x	x				
Firefox 18.x	x	x	x				
Firefox 19.x	x	x	x				
Firefox 20.x	x		x				
Firefox 21.x	x						
IE 10	x						
IE 9.x	x						
IE 8.x	x						
IE 7.x	x						
IE 6.x	x						
Safari 5.x	x		x				
Safari 6.x			x	x	x		
Opera 11	x		x				
Opera 12	x	x					

End User Monitoring (EUM) Compatibility in Java Environments

The following frameworks are certified for the EUM instrumentation. All these frameworks support manual injection of the JavaScript Agent for EUM. Additional supported script injection strategies are listed in the Script Injection column. See [Inject the JavaScript Agent for EUM](#) for details about strategies for injecting the JavaScript Agent for EUM.

Web Application/ AJAX Frameworks	Version	Certified App Server	Script Injection
JSP	Servlet 2.3	Tomcat 7x , GlassFish v3, Weblogic (Assisted only)	Automatic / Assisted
JSF	MyFaces, ICEFaces, ADF	Tomcat 7x , Glassfish v3	Assisted
Tapestry	5.0		Assisted
Struts	2	Tomcat 7x, GlassFish v3	Automatic / Assisted
Spring MVC		Tomcat 7x	Automatic / Assisted
Grails		Tomcat 7x, Glassfish v3, Weblogic 12c	Assisted
Wicket		Tomcat 7	Automatic / Assisted
Web Objects			Assisted
Liferay			Assisted
ZK			Assisted
JQuery		Tomcat 7	Automatic / Assisted
mootools		Tomcat 7	Automatic / Assisted
DWR		Tomcat 7, Glassfish V3, Weblogic 12c	Automatic / Assisted
YUI		Tomcat 7	Automatic / Assisted

EXT JS		Tomcat 7	Automatic / Assisted
Dojo Web tool kits		Tomcat 7Glassfish V3, Weblogic 12c	Automatic / Assisted
GWT			Assisted
angular JS			Assisted
backbone			Assisted

End User Monitoring (EUM) Compatibility in .NET Environments

The following frameworks are certified for the EUM instrumentation. All these frameworks support manual injection of the JavaScript Agent for EUM. Additional supported script injection strategies are listed in the Script Injection column. See [Inject the JavaScript Agent for EUM](#) for details about strategies for injecting the JavaScript Agent for EUM.

Web Application/ AJAX Frameworks	Version	Script Injection
ASP.NET (Web Forms)	3	Automatic / Assisted Injection-Using Attribute Injection
ASP.NET	4	Automatic / Assisted Injection-Using Attribute Injection
MVC	3/ASPX	Assisted Injection-Using Attribute Injection
MVC	3/Razor	Assisted Injection-Using Attribute Injection
SharePoint	2007	Automatic
SharePoint	2010	Automatic

Supported Application Servers for .NET EUM

- IIS 6, 7 and 7.5

Install and Upgrade AppDynamics

- Standard Installations
 - On-Premise Controller
 - App Agent for Java
 - Machine Agent
 - App Agent for .NET
- Self-Service Installations

This section covers downloading and installing AppDynamics software.

Standard Installations

Before you install see:

- Architecture
- Logical Model
- Download AppDynamics Software

On-Premise Controller

- Install the Controller
- Upgrade the Controller
- Verify App Agent - Controller Communication

App Agent for Java

- Install the App Agent for Java
- Upgrade the App Agent for Java
- Uninstall the App Agent for Java

Machine Agent

- [Install the Machine Agent](#)

App Agent for .NET

- [Install the App Agent for .NET](#)
- [Upgrade the App Agent for .NET](#)
- [Uninstall the App Agent for .NET](#)

Self-Service Installations

For Self-Service Pro Trial installation see [Install Agents for 5 or fewer JVMs or CLRs \(Self-Service Installations\)](#).

Name Business Applications, Tiers, and Nodes

- [Naming Guidelines](#)
 - [Naming Components in a Java Environment](#)
 - [Automatic Naming for Application, Tier, and Node](#)
 - [Naming Components in a .NET Environment](#)
 - [Renaming Icon Labels in the UI](#)
- [Learn More](#)

This topic discusses naming AppDynamics business applications, tiers, and nodes. For an overview see [Mapping Application Services to the AppDynamics Model](#).

Naming Guidelines

Use names that are recognizable to the people in your group or company.

Naming Components in a Java Environment

Use these guidelines when configuring App Agents for Java:

- Configure items that are common for all the nodes in the controller-info.xml file.
- Configure information that is unique to a node in the JVM startup script.

See:

- [Install the App Agent for Java](#)
- [App Agent for Java Configuration Properties](#)

Automatic Naming for Application, Tier, and Node

Starting with release 3.7.0, the App Agent for Java javaagent command has a new uniqueID argument that AppDynamics uses to automatically name the node and its tier. For example, using this command argument AppDynamics will name the node "my-app-jvm1" and the tier "my-app-jvm1".

When uniqueID is used, AND when an application name is not provided either through the system property or in the controller-info.xml, AppDynamics creates a new business application called "MyApp".

The new naming mechanism is used by the new Self-Service process. See [Install Agents for 5 or fewer JVMs or CLRs \(Self-Service Installations\)](#).

Naming Components in a .NET Environment

The App Agent for .NET Configuration Utility helps you name the components. It automatically names the nodes. See [Configure the App Agent for .NET](#).

See also:

- [Install the App Agent for .NET](#)
- [App Agent for .NET Configuration Properties](#)

- Naming Conventions for .NET Nodes

Renaming Icon Labels in the UI

1. In the Application, Node, or Tier Dashboard, click **Actions -> Edit Properties**.
2. In the Properties dialog, type a new name.
3. Click **Save**.

Note: renaming in the UI is not persistent in the configuration. It changes the labels in the UI only.

Learn More

- Mapping Application Services to the AppDynamics Model
- Logical Model
- Naming Conventions for .NET Nodes

Install Agents for 5 or fewer JVMs or CLRs (Self-Service Installations)

 Note: the difference between Self-Service installations and standard installations is that some of the settings are pre-configured and supplied in the ZIP file you get from AppDynamics. Standard installations do not have pre-configured settings. For standard installations see [Install Agents for 5 or more JVMs or CLRs \(Standard Installations\)](#).

Install Agents for JVMs

- [Self-Service Install - App Agents for Java](#) 
includes standalone Machine Agent installation instructions

Install Agents for .NET CLRs

- [Self-Service Install - App Agents for .NET](#)

Install the Self-Service On-Premise Controller

- [Install the Self-Service Controller on Linux](#)
- [Install the Self-Service Controller on Windows](#)

Self-Service On-Premise Install - Controller and App Agents for Java

Previously signed up using the TRY FREE form and accepted the default option to use the SaaS Controller environment.
Steps:

1. Get email from AppDynamics
AppDynamics sends an email with the license and ZIP files
2. Follow the Controller installation instructions at [Install the Controller](#).
3. Extract the agent ZIP file to a destination directory as the same user or administrator of the JVM.
4. Add the agent properties as a javaagent argument to your JVMs

This step adds the agent to the startup script of your application servers. Use the server-specific instructions below to add this argument for different application server JVMs:

(??? App/Tier/Node?)

5. Confirm the five mandatory items for agent configuration in the Agent controller-info.xml file:

```
<controller-info>
  <controller-host>192.168.1.20</controller-host>
  <controller-port>8090</controller-port>
  <application-name>ACMEOnline</application-name>
  <tier-name>InventoryTier</tier-name>
  <node-name>Inventory1</node-name>
</controller-info>
```

6. Start the AppDynamics UI to verify that the Java Agent is able to connect to the Controller:

- Point your browser to: <http://<controller-host>:<controller-port>/controller>
- Provide the credentials to log into the AppDynamics UI.

Self-Service On-Premise Install - Controller and App Agents for .NET

essentially the same as

[Install the Controller](#)

and

[Install the App Agent for .NET](#)

Self-Service Install - App Agents for .NET

- [Prerequisites](#)
- [Instrumenting Your IIS with App Agents for .NET](#)
- [Learn More](#)

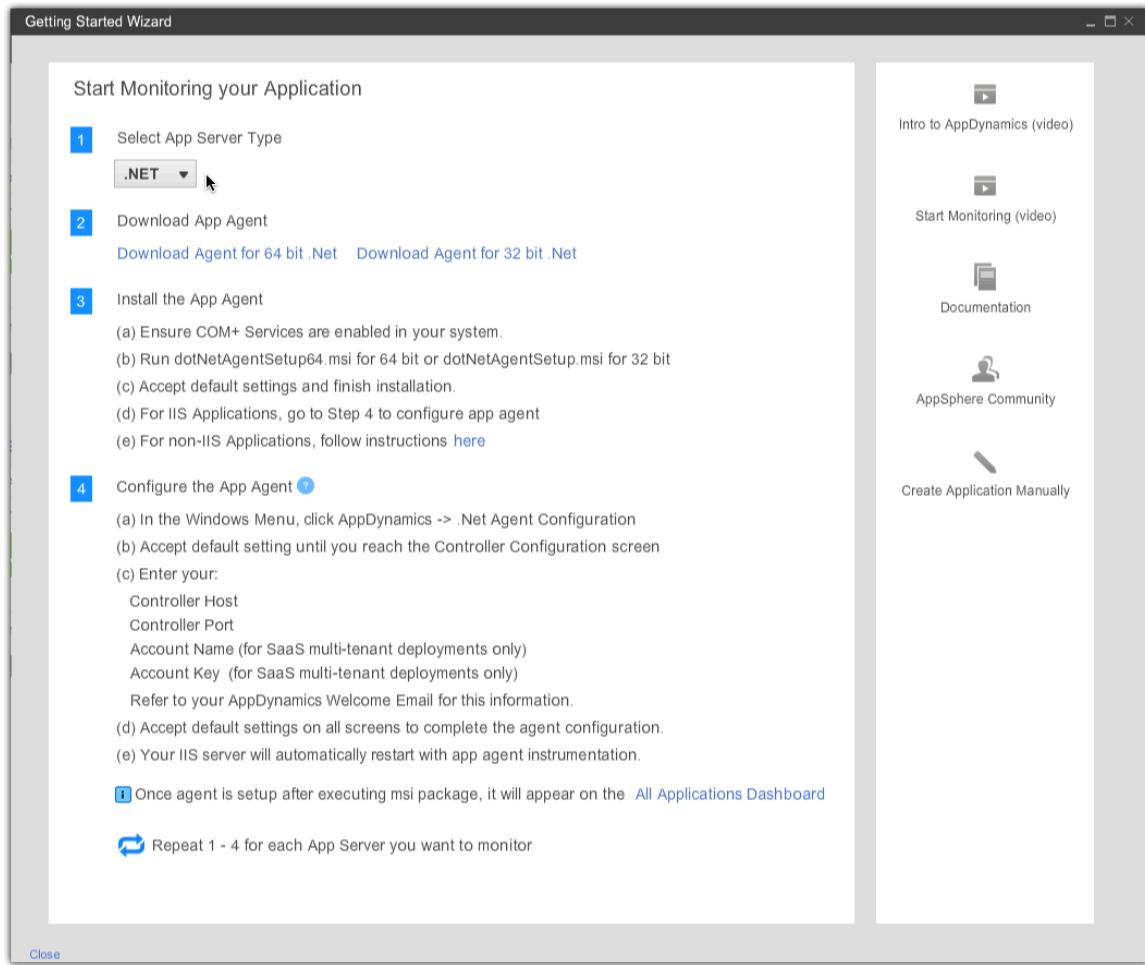
Prerequisites

These instructions assume that you have signed up for an AppDynamics Pro Trial and have access to your Controller.

Instrumenting Your IIS with App Agents for .NET

1. Log into your Controller using account details in your email titled "Welcome to your AppDynamics Pro SaaS Trial" or the account details you have entered during On-Premise installation.

2. Click **Add Application**. The Get Started Wizard opens.



3. Select .NET for your App Server Type and follow the Wizard steps 1 and 2 to download an App Agent for .NET.
4. Confirm that COM+ is enabled on the system. To enable COM+ see [Troubleshoot App Agent for .NET Installation and Configuration](#).
5. Run the setup msi file for your system and accept the default settings.
6. a. If you are instrumenting a non-IIS application see [Enable the App Agent for .NET for Non-IIS Applications](#).
or
b. If you are instrumenting IIS see [Configure the App Agent for .NET](#).
7. Repeat steps 4-6 for up to 5 installations.
8. Log into the Controller UI and start monitoring your application performance.

Learn More

- [Instructions for a standard installation](#)
- [AppDynamics Essentials](#)

Self-Service Install - App Agents for Java

- [Prerequisites](#)
- [Instrumenting Your JVMs with App Agents for Java](#)
- [Learn More](#)

Prerequisites

These instructions assume that you have signed up for an AppDynamics Pro Trial and have access to your Controller.

Instrumenting Your JVMs with App Agents for Java

1. Log into your Controller using account details in your email titled "Welcome to your AppDynamics Pro SaaS Trial" or the account details you have entered during On-Premise installation.

2. Click **Add Application**. The Get Started Wizard opens.

Start Monitoring your Application

1 Select App Server Type
Java ▾

2 Download App Agent
[Download Agent for Sun and JRockit JVM](#) [Download Agent for IBM JVM](#)

3 Install the App Agent

(a) Unzip App Agent

(b) Restart your Application with App Agent instrumentation
Add java agent parameter to app server startup script: ?
-javaagent:<AGENT_HOME>/javaagent.jar=uniqueID=<YOUR_JVM_NAME> Copy

Example: Java agent unzipped in /usr/local/ad and to name JVM as 'EcommerceNode'
-javaagent:/usr/local/ad/javaagent.jar=uniqueID=EcommerceNode

Once your application has restarted, it will appear on the [All Applications Dashboard](#)

4 (Optional) Monitor Hardware and Memory with the Machine Agent

(a) [Download Machine Agent](#)

(b) Unzip Machine Agent on the machine you want to monitor

(c) Start the Machine Agent
java -jar <MACHINE_AGENT_HOME>/machineagent.jar

Repeat 1-3 for each App Server/JVM you want to monitor and
4 for each Machine those App Servers/JVMs run on

3. Follow the Wizard steps 1 and 2 to download an App Agent for Java ZIP file for your operating system.

4. As an admin user in a directory on the machine where your Java application is running, extract the file. This location will be referred to as <AGENT HOME>.

5. Add the App Agent for Java javaagent argument to your JVM start script where <my-app-jvm1> is the name you use for the application running on that JVM.

```
java -javagent:<AGENT_HOME>/javaagent.jar=uniqueID=<my-app-jvm1>
```

Use the server-specific instructions below to add this argument for different application server JVMs:

6. Repeat steps 4 and 5 for up to 5 JVMs.

7. Optionally download the Machine Agent from the Get Started Wizard and install the files.

a. Extract the ZIP file to a directory on the same machine as the App Agent for Java.

Do not use spaces in the destination directory path.

For Windows environments, unblock the zip file before you extract it. To unblock the zip file, right-click on the zip file and select the "Properties" tab. On the properties tab, choose "unlock" to unblock the file.

b. Start the Machine Agent.

```
java -jar <MACHINE_AGENT_HOME>/machineagent.jar
```

8. Repeat step 7 for up to 5 machines.

9. In the Controller UI, verify the nodes in the default MyApp business application and start monitoring your application performance. See [Use AppDynamics for the First Time with Java](#).

(i) Note: When installing from the Get Started Wizard, you do not have to manually configure the AppDynamics business application, tier, and node names. The node and tier names are derived from <my-app-jvm1>, the name you provide for the javaagent command. The business application name is MyApp.

Learn More

-  [Creating a New Application](#)
- [AppDynamics Essentials](#)

Install Agents for 5 or more JVMs or CLRs (Standard Installations)

 Note: the difference between Self-Service installations and standard installations is that some of the settings are pre-configured and supplied in the ZIP file you get from AppDynamics. Standard installations do not have pre-configured settings.

For Java

For Java environments using a SaaS Controller see:

- [Install the App Agent for Java](#)
- [Install the Machine Agent](#)

For Java environments using an On-Premise Controller see:

- [Install the Controller](#)
- [Install the App Agent for Java](#)
- [Install the Machine Agent](#)

For .NET

For .NET environments using a SaaS Controller see:

- [Install the App Agent for .NET](#)

For .NET environments using an On-Premise Controller see:

- [Install the Controller](#)
- [Install the App Agent for .NET](#)

Automate Multi-Agent Deployment

There are two basic strategies for deploying large numbers of AppDynamics agents across a managed environment:

1. Deploy the agents independently of the application inside the application server. This method ensures that re-deployments of the application do not overwrite the agent deployment.

2. Integrate deployment of AppDynamics agents into the deployment of applications. This more sophisticated approach requires modifying the existing application deployment automation scripts.

Install the Controller

This section covers installing the AppDynamics Controller.

Controller System Requirements

- The AppDynamics Controller
- Controller System Requirements
 - Controller Operating System Requirements
 - Supported Web Browsers for the Controller UI
- Controller Hardware Requirements
 - Use Dedicated Hardware for the Controller
 - Controller Performance Profiles
 - Calculating Performance Profiles for a .NET Environment
 - Hardware Requirements per Performance Profile
 - Additional Disk I/O Information
 - Additional Sizing Considerations
 - Asynchronous Call Monitoring
 - End User Monitoring
 - Linux file descriptor limit
 - Running the Controller on Virtual Machines
 - To measure available Disk I/O
- Additional Requirements
 - Learn More

The AppDynamics Controller

The AppDynamics Controller is the central management server where all data is stored and analyzed. The Controller serves the Flash-based user interface (UI). All AppDynamics Agents connect to the Controller to report data.

The Controller uses an embedded Glassfish application server and an embedded MySQL database. It has a browser-based Administration interface for various system parameters.

The Controller installer ships with and automatically installs all of its required software. You do not need any additional software to install the AppDynamics Controller.

Controller System Requirements

Controller Operating System Requirements

The Controller is supported on the following Operating Systems:

Linux (32 and 64-bit)	Microsoft Windows (32 and 64-bit)
<ul style="list-style-type: none">• Redhat Enterprise Linux (RHEL) 6.1, 6.2• CentOS 6.1, 6.2• Fedora 14• Ubuntu 8, 12• Open SUSE 11.x• SUSE Linux Enterprise Server• Cloud: Amazon EC2, Rackspace, Azure	<ul style="list-style-type: none">• Windows 2003 Server• Windows 2008 Server• Windows 2012 Server• Windows 8

Supported Web Browsers for the Controller UI

The following browsers are supported:

- Mozilla FireFox v 3.x, 4.x, 5.0
- Internet Explorer 6.x, 7.x, 8.x, 9.0.
- Safari 4.x, 5.x
- Google Chrome 10.x, 11.x, 12.x

The Controller UI requires Flash Player 10 or greater; AppDynamics recommends version 11.

Controller Hardware Requirements

The Controller performs large-scale real-time data analysis and correlation supporting multiple agents at any given time. It constantly persists large amounts of data to the storage system. For the Controller to be responsive, it must have appropriate hardware. Use these recommendations in the performance profiles to determine the hardware needed for your Controller.

 These recommendations represent a good starting point only, your requirements may vary depending on the number of metrics per minute generated in your environment.

Use Dedicated Hardware for the Controller

AppDynamics strongly recommends that you install the Controller on a dedicated machine. The hardware requirements described in this document assume that no other major processes are running on the machine where the Controller is installed. A Controller with more than 250 nodes *must* run on a dedicated machine.

For information about using a Virtual Machine for the Controller, see [Running the Controller on Virtual Machines](#).

Controller Performance Profiles

Size the Controller hardware requirements using a performance profile based on the number of nodes in your environment and the number of metrics per minute being reported to the controller. For general information about applications and nodes, see [Logical Model](#).

The Controller performance profiles are based on the number of JVMs and/or CLRs (nodes). When the number of JVMs is static, it is easy to determine. However, WebSphere on z/OS, dynamic cloud environments, and .NET environments dynamically spawn new agents. For z/WebSphere and clouds you need to consider historic data to estimate the correct sizing. For .NET see [Calculating Performance Profiles for a .NET Environment](#).

Controller Performance Profile	Number of Nodes (1)	Number of Business Apps	Recommended for VMs
Demo	Up to 5	1	OK
Small	Up to 10	1	OK
Medium	Up to 50	5	OK
Large	Up to 250	5	Not supported*
Extra Large	250-500	20	Not supported*

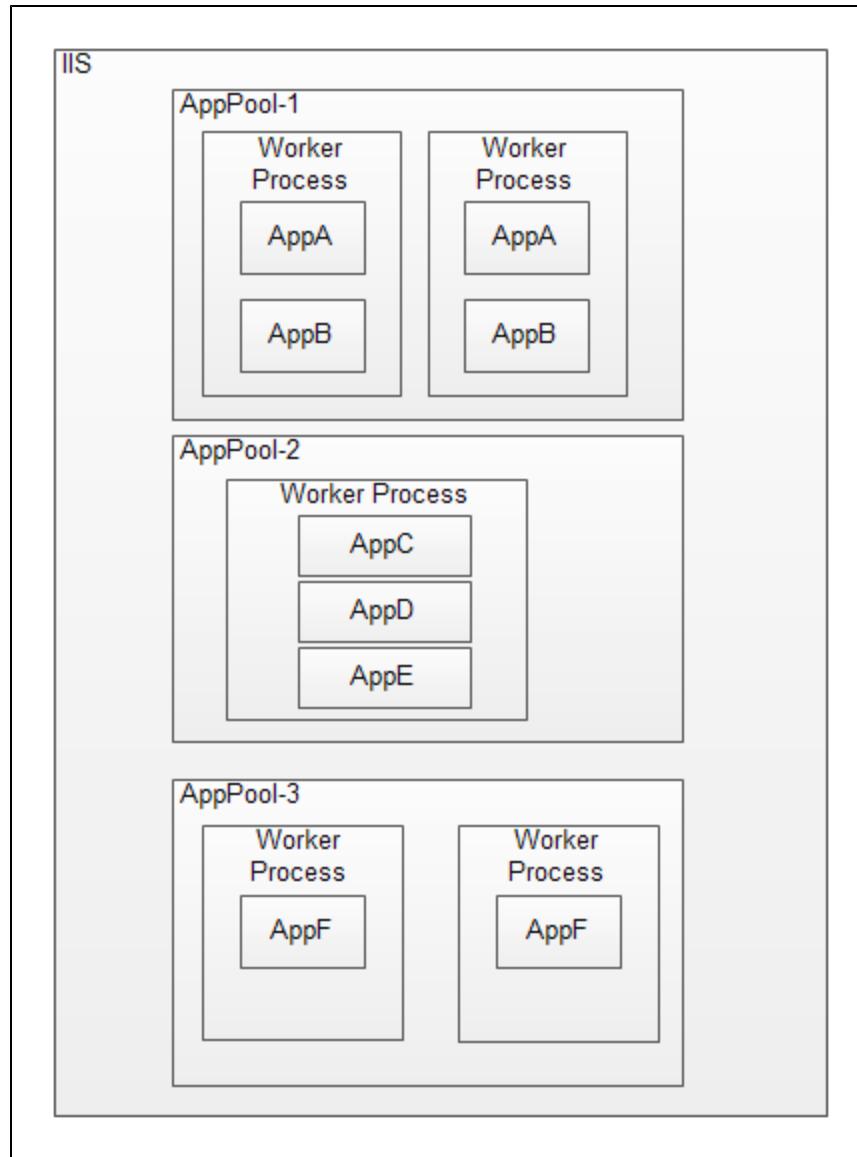
(1) In a Java environment there is usually one AppDynamics App Agent for Java per node. In contrast, one AppDynamics App Agent for .NET may support multiple nodes. See [Calculating Performance Profiles for a .NET Environment](#) to arrive at the number of nodes for a .NET environment.

 Note: AppDynamics neither recommends nor supports Large or Extra Large deployments in virtual environments.

Calculating Performance Profiles for a .NET Environment

The .NET Agent dynamically creates nodes depending on the monitored application's configuration in the IIS server. An IIS server can create multiple instances of each monitored IIS application. For every instance the .NET Agent creates a node. For example, if a IIS application has 5 instances the .NET Agent will create 5 nodes.

The maximum number of instances of a particular IIS application is determined by the number of worker processes configured for its application pool. Consider the following diagram:



The diagram shows three application pools: AppPool-1, AppPool-2, and AppPool-3 with the following characteristics:

- AppPool-1 and AppPool-3 can have a maximum of two worker processes (known as a web garden)
- AppPool-2 can have one worker process

If applications are assigned to application pools as follows:

AppPool-1: AppA and AppB

AppPool-2: AppC, AppD and AppE

AppPool-3: AppF

Use the following formula to estimate the number of nodes:

```

AppPool-1 * number of applications * max number of worker processes
+
AppPool-2 * number of applications * max number of worker processes
+
AppPool-3 * number of applications * max number of worker processes
= Number of Nodes

```

In this case the numbers would be:

AppPool-1 nodes = 4 (2 applications * 2 worker processes)

AppPool-2 nodes = 3 (3 applications * 1 worker process)

AppPool-3 nodes = 2 (1 application * 2 worker processes)

for a total of 9 nodes.

Hardware Requirements per Performance Profile

For larger deployments with hundreds of nodes, sizing the hardware for your controller is complicated. Depending on the number of agents and the complexity of your business transactions, the number of metrics can become very large. The following recommendations are a starting point.

Controller performance profile	Supported OS platforms	CPU # of cores	RAM	Disk storage	Disk I/O write/read/random read	Agent Count*	Metrics Count/minute
Demo	Linux (32 & 64-bit) Windows (32 & 64-bit)	2 CPU Cores 1.5 GHz minimum	2 GB	50 GB	50 MB/sec 50 MB/sec 1.5 MB/sec	up to 5	5 K max
Small	Linux (32 & 64-bit) Windows (32 & 64-bit)	4 CPU Cores 1.5 GHz minimum	4 GB	100 GB	50 MB/sec 50 MB/sec 1.5 MB/sec	6-10	25 K max
Medium	Linux (64-bit) Windows (64-bit)	8 CPU Cores 3.3 GHz minimum	16 GB	4,000 GB	60 MB/sec 60 MB/sec 1.6 MB/sec	11-50	250 K max
Large	Linux (64-bit) Windows (64-bit)	24 CPU Cores 2.5 GHz minimum	32 GB RAM	5,000 GB	100 MB/sec 100 MB/sec 3 MB/sec	51-250	512 K max
Extra Large **	Linux (64-bit)	24 CPU Cores 2.5 GHz minimum	64 GB RAM	10,000 GB	125 MB/sec 125 MB/sec 5 MB/sec	251-500	1 Million max

 The Demo profile is only for demonstration and evaluation environments and is not suited for production installations.

*Agent Count - also see [Controller Performance Profiles](#).

** When sizing in the Extra Large profile range, contact your AppDynamics Technical Account Manager or the [AppDynamics Support Team](#) for assistance.

Additional Disk I/O Information

The AppDynamics Controller always writes and reads using the following block sizes:

- Read block size: 16 K
- Write block size: 64 K

Additional Sizing Considerations

The number of metrics is affected by monitoring asynchronous calls and using End User Monitoring (EUM).

Extensive async monitoring and use of EUM is not supported on a Small profile controller due to size constraints. In addition, for a Medium profile controller running 40+ agents and with some async monitoring or EUM you might need to use a machine closer to the sizing of the Large profile recommendation.

Asynchronous Call Monitoring

Monitoring asynchronous calls increases the number of metrics per minute to a maximum number of 23000 per minute. This does not apply for .NET environments because async monitoring is not supported for .NET.

End User Monitoring

The use of End User Monitoring in your applications can increase the number of metrics per minute by up to 22000 metrics.

Linux file descriptor limit

For Linux environments, it is important to set the file descriptor limit to at least 65535. For more details see [Configure File Descriptor Limits on Linux](#).

Running the Controller on Virtual Machines

To run AppDynamics Controller on a virtual machine, the virtual machine must meet the hardware resource requirements equivalent to physical machines.

Large or Extra Large deployments in virtual environments are not supported.

Make sure that your disk I/O performance matches our recommended values. To ensure that your virtualized storage subsystem is providing the required I/O capacity, run the following disk I/O rate tests.

To measure available Disk I/O

These tests write 32GB of data to your disk and take approximately 15 minutes to complete. These tests produce numbers for write, rewrite, read, reread, random read, and random write.

1. Download the free IOzone tool.

- [Download IOzone for Linux](#)
- [Download IOzone for Windows](#)

2. Execute the following commands:

	For Linux	For Windows
Test for Write Speed	/opt/iozone/bin/iozone -s 32000m -r 64k -i 0 -i 1 -w -f <path-to-your-partition>/test1	<path-to-your-partition>/Benchmarks/lozone3.353/iozone -s 32000m -r 64k -i 0 -i 1 -w -f test1
Test for Random Read	/opt/iozone/bin/iozone -s 32000m -r 16k -i 2 -w -f <path-to-your-partition>/test1	<path-to-your-partition>/Benchmarks/lozone3.353/iozone -s 32000m -r 16k -i 2 -w -f test1

3. Verify available disk I/O.

The results from these tests should match the Disk I/O requirements specified in [Hardware Requirements Per Performance Profile](#).

Additional Requirements

In addition to the correct sizing of the hardware, special tuning of the operating system, the Glassfish application server, the MySQL as well as the Controller itself is required for very large installations, as detailed in [Tuning for Large Enterprise Environments](#).

Learn More

- Logical Model
- Configure File Descriptor Limits on Linux

Measuring Disk Performance Using a Script

- To edit the script
- To evaluate performance

You can use lozone to benchmark I/O performance. See <http://www.iozone.org/> for information about lozone.

Running iostat in parallel with lozone gives a better overall picture.

To start lozone and iostat in parallel, you can use our script. Click [ScriptToMeasureDiskIO](#) to download our script. Then edit it.

To edit the script

1. Change the -s 32g to reflect ½ of your total physical RAM. For example, if your machine has 32GB RAM use --s 16g.
2. Set the -F parameter to point to the same partition that your database is in.
3. The -t parameter defines the number of read/write threads. Change the -t parameter if you have more than 500 nodes. Set the -t parameter to match the number of files in the -F parameter. For example, if t=4, then 4 test files must be specified for -F. Use 8 for the write test (with 8 files for -F) and 4 for the read test (with 4 files for -F).

To evaluate performance

1. Check if the rewrite/reread from iozone meets the Controller requirements for your profile.
2. Check if the MIN value in the iostat.logs is higher than the minimum values of the Controller requirements.

See [Controller System Requirements](#) for the Controller requirements.

Contact AppDynamics Support if you have questions.

Install the Controller on Linux

- [The Controller Installer for Linux](#)
- [Pre-installation Checklist for Linux](#)
- Download and Launch the Linux Installer
 - Installer Modes
 - To download and launch the Controller on Linux
- [Installing the Controller on Linux](#)
 - To install the Controller on Linux in Console Mode
 - To install the Controller on Linux in GUI Mode
 - To install the Controller on Linux in Silent Mode
- [Verifying Controller Installation](#)
- [Post-installation Checklist for Linux](#)
 - Apply the License File
 - Configure swappiness if the Linux kernel is newer than 2.1.10
 - Configure Backups.
- [Learn More](#)

This topic describes how to install the AppDynamics Controller in a Linux environment.

For an overview of the Controller and its requirements, including supported operating systems, see [Controller System Requirements](#).

The Controller Installer for Linux

The AppDynamics Controller installer for Linux is an executable binary that includes a command-line wizard to guide you through the installation.

You do not require any additional software. All of the software components required for Controller installation are bundled with the installer binary.

Pre-installation Checklist for Linux

- For optimum stability and performance, AppDynamics strongly recommends that you install the Controller on a dedicated machine.

- The disk space requirements differ for each of the Controller Performance Profiles. Verify that there is enough disk space for the performance profile that describes your environment.
- Verify that you have the right CPU, RAM and Disk I/O capacity for the performance profile you plan to install. Run the Disk I/O measurement tests described at [To measure available Disk I/O](#).
- The Controller requires that libaio be installed on the same machine. For more information see [Install libaio on Linux](#).
- Make sure the file descriptor limit is set to at least 65535. For more information see [Configure File Descriptor Limits on Linux](#).
- Install the Controller on a 64-bit Linux system if you plan to use a Medium, Large, or Extra Large Performance Profile. For details see [Controller Performance Profiles](#).
- Verify that you have Read/Write/Execute permissions on the directory where you install the Controller.
- Verify that a ZIP/UNZIP utility is installed on the machine to which you will download the installer.
- Open the HTTP port that the Controller will use to communicate with agents and the AppDynamics UI.

Download and Launch the Linux Installer

The Controller is available at [AppDynamics Download Center](#).

Installer Modes

The Controller installer can run in three different modes:

- GUI
By default the installer tries first to start in GUI mode. If the system is not set up to display the GUI, it will start in console mode.
- Console
You can force the installer to start in console mode by passing the **-c** option to the installer.
- Silent
You can force the installer to start in silent mode by passing the **-q -varfile <path-to-response-file>** option. See [Install an On-Premise Controller Silently](#) for more information about silent mode and the response file.

To download and launch the Controller on Linux

1. Download the Linux Controller binary (controller_linux_<version>.sh) file from [AppDynamics Download Center](#).

2. Assign execute permissions to the downloaded installer binary.

3. Launch the installer by executing the controller_linux_<version>.sh that you downloaded.

If you want to install in console or silent mode, use the **-c** or **-q** options as described above under [Installer Modes](#). For example, to install in console mode:

```
./controller_32bit_linux_v3.7.0-3.7.0_121.sh -c
```

Installing the Controller on Linux

To install the Controller on Linux in Console Mode

1. When the installer starts, type **o**, then press the Enter key.

```
This will install AppDynamics Controller on your computer.
OK [o, Enter], Cancel [c]
o
Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.
```

2. Press the Enter key several times to get the entire license agreement. At the end, type **1** to accept the license agreement.

I accept the agreement
Yes [1], No [2]
1

3. Type **1** to grant or **2** to deny AppDynamics permission to collect usage data statistics from your controller, then press the Enter key. The purpose of collecting usage data is to improve the quality of AppDynamics products and services.

Do we have your consent to collect usage data statistics?
Yes [1], No [2]

4. Set the path of the Controller installation directory or accept the default, then press the Enter key.

Where should AppDynamics Controller be installed?
[/Applications/AppDynamics/Controller]

5. Type the host name or IP address that the AppDynamics agents and user interface will use to connect to the Controller, then press the Enter key.

What connection values should the Controller components use?
Basic Configuration:
This is the hostname or IP address that agents and the user interface use to connect to the Controller. If this is an upgrade and the application server host name is modified, existing agents may need to be reconfigured to connect to the new host name.
Application Server Host Name
[osxltldavi.local]

6. Type the primary HTTP port that the AppDynamics agents and user interface will use to connect to the Controller, or accept the default. Then press the Enter key.

This is the port that agents and the user interface use to connect to the Controller. If this is an upgrade and the application server primary port is modified, existing agents may need to be reconfigured to connect to the new port.
Application Server Primary Port
[8090]

7. Accept the default internal controller port settings or configure different ports if the defaults are already in use. Then press the Enter key.

These are internal Controller ports that should be modified only if other services are already bound to one or more of these ports.
Database Server Port
[3388]
Application Server Admin Port
[4848]
Application Server JMS Port
[7676]
Application Server IIOP Port
[3700]
Application Server SSL Port
[8181]

8. Type **1** to configure the Controller in single-tenancy mode or **2** in multi-tenancy mode, then press the Enter key. See [Controller Tenant Mode](#) for more information about tenancy modes.

Select the Controller tenancy mode configuration.
You can configure the AppDynamics Controller to run as a single or multi-tenant platform.

Select multi-tenancy mode if you want to set up access controls to managed applications at the account level. In this mode each application must be created within a specific account and only those agents and users assigned to that account will have visibility into that application. You will be required to configure additional account information in this mode for each user and agent connecting to the Controller.

Select single-tenancy mode if you do not need to use accounts to restrict access to managed applications. You will not need to configure account information for users and agents you want to connect to a Controller in single-tenancy mode.

You can always switch later on from single-tenancy to multi-tenancy mode, so if you are unsure about the mode, you should start with single-tenancy.

Single Tenancy Mode [1], Multi Tenancy Mode [2]

9. Configure Controller access and authentication. Which procedure you use depends on which tenancy mode you selected in the previous step.

If you selected single-tenancy mode:

Type the admin user name and password and confirm the password, then press the Enter key.

This user will be able to access the single-tenant Controller and create other users. The admin user is the user described as the AppDynamics Administrator.

Single Tenancy Mode [1], Multi Tenancy Mode [2]
1
What is the admin user info required to access the Controller UI?
User Name
[]
admin
Password

Reenter Password

Press the Enter Key.

If you selected multi-tenancy mode:

Configure the following settings:

- the root user password

This password is used to log into the AppDynamics administration console, where the user can create and manage AppDynamics accounts and configure various controller settings. See [Access the Administration Console](#) for more information about this console.

Single Tenancy Mode [1], Multi Tenancy Mode [2]
2
What is the root user password?
The AppDynamics Controller is a full multi-tenant platform where applications created and managed in one account are only visible to users and agents assigned to that specific account. You will need to use this password to log into the Account Management Screen to manage accounts. You can change this password later from the Account Management Screen.

Password
Reenter Password

- the account name for the initial account on the Controller

AppDynamics generates an account key for this initial account.

What is the initial account information?
The AppDynamics Controller is a full multi-tenant platform where applications created and managed in one account are only visible to users and agents assigned to that specific account. You must create an initial account. The account name and access key must be set in the agent controller-info.xml file for an agent to connect to the Controller under this initial account.

Account Name
[default]

Account Access Key be33f9a7-33a1-4cb6-8be4-8a44439c2361

- the user name and password of the admin user for the initial account
This user is able to create other users in this account.

Create an initial admin user for this account. You will need this user to log in for the first time into the Controller UI.

User Name

[]

Password

Reenter Password

Please read the following important information before continuing.

The following account information must be set in the agent controller-info.xml file for an agent to connect to the Controller under the initial account. Please retain the account name and access key for later use. We will save this information in a file named initial_account_access_info.txt in the Controller installation directory.

Read the paragraph about recording these settings, then press the Enter key.

10. Type the number that represents the Controller performance profile that matches your requirements, then press the Enter key.

Select the Controller performance profile based on the expected maximum number of instrumented nodes and applications.

Demo (up to 5 nodes assigned to 1 application, not for production use) [1], Small (up to 10 nodes assigned to 1 application) [2], Medium (up to 50 nodes assigned to at most 5 applications) [3], Large (up to 250 nodes assigned to at most 5 applications) [4], Extra Large (above 250 nodes assigned to 1 application) [5]

11. Type the path to data directory or accept the default. Then press the Enter key.

The data directory is where the Controller mysql data is stored.

Enter the path of the AppDynamics Controller data directory
Enter path to the data directory:
[/Applications/AppDynamics/Controller/db]

12. Type **2** to configure a high availability primary Controller, **3** to configure a high availability secondary Controller or **1** if you are not enabling HA for this Controller. For more information see [Controller High Availability](#).

Select the Controller high availability configuration (if applicable).
Not Applicable (Not HA Enabled) [1], HA Primary Controller [2], HA Secondary Controller [3]

13. Press the Enter key to start the installer.

To install the Controller on Linux in GUI Mode

1. In the Welcome screen click **Next**.

The license agreement displays.

2. Scroll down to the bottom and accept the license agreement, then click **Next**.

Setup – AppDynamics Controller 3.7.0

License Agreement

Please read the following important information before continuing.

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

USER DOES NOT AGREE TO ALL OF THE TERMS AND CONDITIONS OF THIS AGREEMENT, DO NOT SELECT THE "I ACCEPT" BOX AND DO NOT USE THE SOFTWARE. BY CHECKING THE "I ACCEPT" BOX, END USER AGREES TO BE BOUND BY THE AGREEMENT. END USER IS NOT AUTHORIZED TO USE THE SOFTWARE UNLESS AND UNTIL IT HAS AGREED TO THE TERMS AND CONDITIONS OF THIS AGREEMENT. IN THE EVENT OF UPDATES TO THIS AGREEMENT, THE LATEST VERSION IS APPLICABLE TO THE USE OF THE SOFTWARE THIRTY (30) DAYS AFTER LICENSOR'S POSTING OF THOSE CHANGES TO http://www.appdynamics.com/EULA/AppDynamics_EULA.pdf. BY CONTINUING TO USE THE SOFTWARE AFTER THE POSTING OF ANY CHANGES, END USER AGREES THAT IT SHALL BE BOUND BY THE UPDATED TERMS AND CONDITIONS OF SUCH AGREEMENT.

IF END USER DOES NOT AGREE TO THE TERMS AND CONDITIONS OF THIS AGREEMENT, CLICK "I DO NOT ACCEPT" AND EXIT THE PROGRAM.

- I accept the agreement
 I do not accept the agreement

3. Click **Yes** to grant or **No** to deny AppDynamics permission to collect usage data statistics from your controller. The purpose of collecting data is to improve the quality of AppDynamics products and services.



AppDynamics would like your help improving quality and performance of its products and services. AppDynamics can automatically collect usage data statistics from your controller and send it to AppDynamics for analysis. Information is sent only with your explicit consent, and submitted to AppDynamics.

Do we have your consent to collect usage data statistics?

4. Set the path of the Controller installation directory in the Destination directory field, then click **Next**.

Setup – AppDynamics Controller 3.7.0

Select Destination Directory

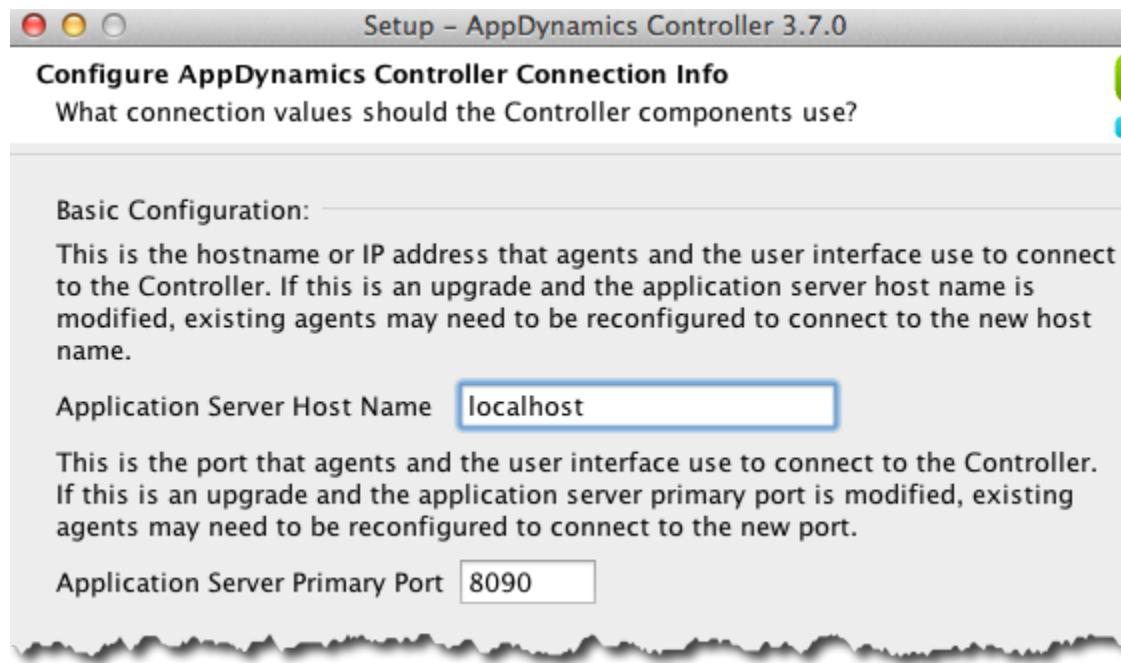
Where should AppDynamics Controller be installed?

Select the folder where you would like AppDynamics Controller to be installed, then click Next.

Destination directory

/Applications/AppDynamics/Controller

5. Configure the host name (or IP address) and primary port that the AppDynamics agents and user interface will use to connect to the Controller.



6. Scroll down to examine the default configuration of the internal Controller ports and modify the settings as necessary if other services are bound to these ports. Then click **Next**.

Advanced Configuration:

These are internal Controller ports that should be modified only if other services are already bound to one or more of these ports.

Database Server Port	3388
Application Server Admin Port	4848
Application Server JMS Port	7676
Application Server IIOP Port	3700
Application Server SSL Port	8181

7. Verify the displayed configured settings and click **Back** if you want to change any of them. Click **Next** to continue.

8. Choose single-tenancy or multi-tenancy mode for the Controller, then click **Next**.

Setup – AppDynamics Controller 3.7.0

AppDynamics Controller Tenancy Mode Configuration

Select the Controller tenancy mode configuration.

You can configure the AppDynamics Controller to run as a single or multi-tenant platform.

Select multi-tenancy mode if you want to set up access controls to managed applications at the account level. In this mode each application must be created within a specific account and only those agents and users assigned to that account will have visibility into that application. You will be required to configure additional account information in this mode for each user and agent connecting to the Controller.

Select single-tenancy mode if you do not need to use accounts to restrict access to managed applications. You will not need to configure account information for users and agents you want to connect to a Controller in single-tenancy mode.

You can always switch later on from single-tenancy to multi-tenancy mode, so if you are unsure about the mode, you should start with single-tenancy.

Single Tenancy Mode
 Multi Tenancy Mode

< Back Next > Cancel

See [Controller Tenant Mode](#) for more information about these modes.

9. Configure Controller access and authentication. This procedure depends on which tenancy mode you selected in the previous step.

If you selected single-tenancy mode:

- a. Enter admin user name and password. This user will be able to access the single-tenant controller and create other users. The admin user is the user described as the [AppDynamics Administrator](#).

Setup – AppDynamics Controller 3.7.0

Admin User Setup

What is the admin user info required to access the Controller UI?

User Name

Password

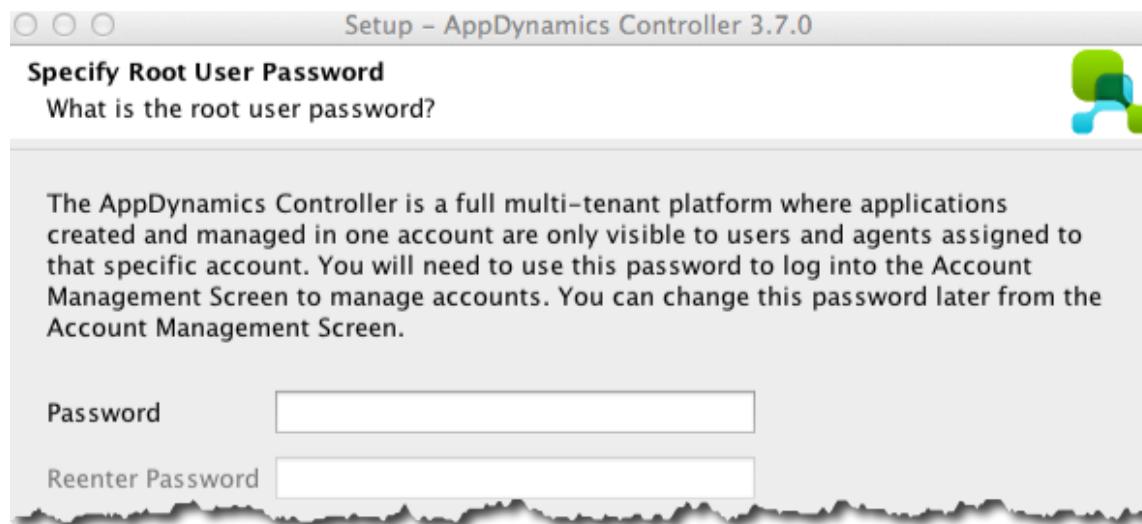
Reenter Password

- b. Click **Next**.

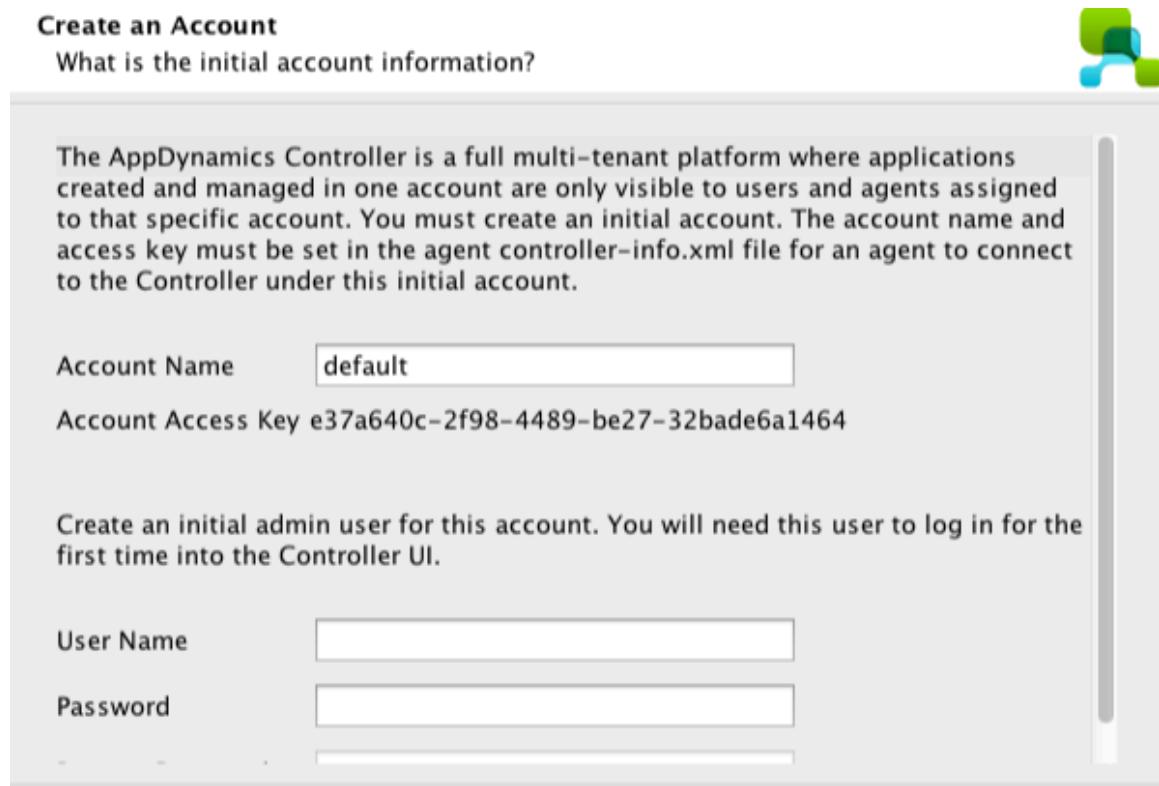
If you selected multi-tenancy mode:

- a. Enter a password. This password is used to log into the AppDynamics administration console, where a user can create and manage

AppDynamics accounts and configure various controller settings. See [Access the Administration Console](#) for more information about this console.



- b. Click **Next**.
- c. In the Account Name field, type the account name for the initial account on the Controller. AppDynamics generates an account key for this initial account.



- d. In the User Name field enter the user name of the admin user for the initial account. This user is able to create other users in this account.
- e. In the Password field enter the user name of the admin user for the initial account. Re-enter the password, then click **Next**.
- f. Review and save the settings for the initial account settings and click **Next**.

Setup – AppDynamics Controller 3.7.0

Information

Please read the following important information before continuing.

When you are ready to continue with Setup, click **Next**.

The following account information must be set in the agent controller-info.xml file for an agent to connect to the Controller under the initial account. Please retain the account name and access key for later use. We will save this information in a file named initial_account_access_info.txt in the Controller installation directory.

Account Name: Account1

Account Access Key: e37a640c-2f98-4489-be27-32bade6a1464

10. Select the Controller performance profile that matches your requirements and click **Next**.

Setup – AppDynamics Controller 3.7.0

AppDynamics Controller Performance Profile

Select the Controller performance profile based on the expected maximum number of instrumented nodes and applications.

Hardware requirements for the machine hosting the AppDynamics Controller have changed, so before selecting the performance profile please refer to the AppDynamics Controller Hardware Requirements to confirm that the Controller host is able to support that performance profile. These requirements may be found at:

<http://help.appdynamics.com/entries/204649-controller-hardware-requirements>

Demo (up to 5 nodes assigned to 1 application, not for production use)

Small (up to 10 nodes assigned to 1 application)

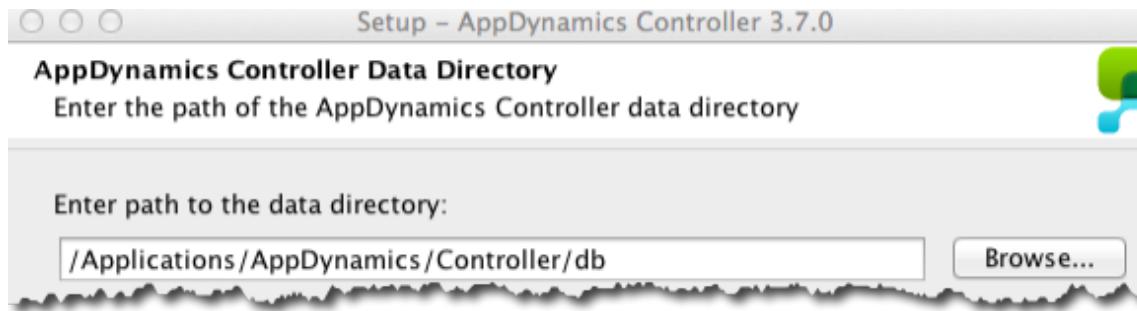
Medium (up to 50 nodes assigned to at most 5 applications)

Large (up to 250 nodes assigned to at most 5 applications)

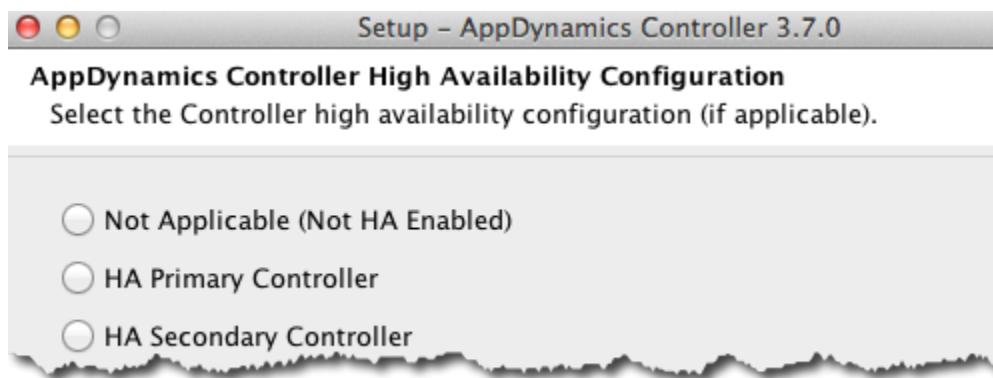
Extra Large (above 250 nodes assigned to 1 application)

< Back Next > Cancel

11. In the path to the data directory field, type the path to the directory where the Controller's mysql data will be stored or select the default, then click **Next**.



12. Select the high availability configuration for the Controller or Not Applicable if you are not enabling HA for this Controller. For more information see [Controller High Availability](#).



The installer installs the Controller.

To install the Controller on Linux in Silent Mode

See [Install an On-Premise Controller Silently](#).

Verifying Controller Installation

Wait for the installation status screen to show that the installation is complete. Then the Controller will automatically start.

To verify the success of your installation, open a browser to the URL that you configured for the Controller.

```
http://<application_server_host_name>:<http-listener-port>/controller
```

Post-installation Checklist for Linux

- Apply the License File
- Configure swappiness if the Linux kernel is newer than 2.1.10
- Configure Backups

Apply the License File

The license file (license.lic) was provided in your Welcome email from AppDynamics. Copy the downloaded license file to the Controller installation directory.

Allow up to 5 minutes for the license change to take effect.

Configure swappiness if the Linux kernel is newer than 2.1.10

For better performance, configure swappiness. For details see [Configure Swappiness on Linux](#). This feature is available only for Linux kernel versions greater than 2.1.10.

Configure Backups.

See Controller Data Backup and Restore.

Learn More

- Controller System Requirements
- Install an On-Premise Controller Silently
- Controller Tenant Mode
- Controller High Availability
- Install libaio on Linux
- Configure File Descriptor Limits on Linux
- Configure Swappiness on Linux
- Controller Data Backup and Restore

Install libaio on Linux

- Red Hat and CentOS
- Ubuntu
- Fedora
- Debian
- Learn More

Linux libaio is a library that works at the kernel level. This section provides instructions on how to install libaio for some common flavors of Linux operating system.

Red Hat and CentOS

Use the following command to install the libaio package:

```
yum install libaio
```

Ubuntu

Use the following command to install the libaio package:

```
sudo apt-get install libaio
```

Fedora

Install the rpm for the libaio package from the [Fedora website](#).

Debian

Use the following command to install the libaio package:

```

mkdir /usr/libbaio

cd /usr/libbaio

wget
http://http.us.debian.org/debian/pool/main/liba/libaio/libaio1_0.3.106-3_i386.deb
-x libaio1*.deb

tar zxvf data.tar.gz

sudo cp lib/* /emul/ia32-linux/lib/

```

Learn More

- [Install the Controller on Linux](#)

Configure File Descriptor Limits on Linux

- [Limiting the Number of File Descriptors](#)
 - To check the limits for number of file descriptors
 - To configure limits on a per-user basis in the limits.conf file
 - To configure limits for number of file descriptors in the profile
- [Learn More](#)

This topic explains the importance of checking file descriptor limits and provides the steps to modify the limit for Controller installations on Linux.

Limiting the Number of File Descriptors

If you do not set the limits for number of file descriptors, AppDynamics may produce warnings such as:

- Warning in database log: "Could not increase number of max_open_files to more than xxxx".
- Warning in server log: "Cannot allocate more connections".

These warnings indicate that the Controller database is attempting to open more file descriptors than is permitted by the operating system. This issue can affect product integrity in larger Controller installations.

AppDynamics recommends that the maximum number of file descriptors be set to **65535** for installations in Linux environments.

To check the limits for number of file descriptors

As the root user, execute following command:

```
ulimit -S -n
```

If this value is less than 65535, change it to 65535 either in the profile or limits.conf file.

On Fedora as well as Ubuntu (and probably other modern Linux distributions) using /etc/security/limits.conf is the preferred method. You should put the ulimit command in /etc/profile only if the /etc/security/limits.conf does not exist.

To configure limits on a per-user basis in the limits.conf file

1. Open the limits.conf file for edit:

```
/etc/security/limits.conf
```

2. Add the following lines:

```
appdynamics    hard    nofile 65535  
appdynamics    soft    nofile 65535
```

Where "appdynamics" is the username of the Linux user who runs the Controller.

3. Enable these limits by editing:

```
/etc/pam.d/common-session
```

4. Add the line:

```
session required pam_limits.so
```

5. When you log in again as the "appdynamics" user, the limits should apply.

To configure limits for number of file descriptors in the profile

1. As the root user, add the following entry to the /etc/profile:

```
ulimit -n 65535
```

2. Reboot the host server and check the limit again.

Learn More

- Install the Controller on Linux
- Controller System Requirements

Configure Swappiness on Linux

- The Swappiness Parameter in Linux
 - To configure swappiness
 - Learn More

The Swappiness Parameter in Linux

The swappiness parameter controls how often the Linux kernel moves processes out of physical memory and onto the swap disk. Because disks are much slower than RAM, this can lead to slower response times for system and applications if processes are too aggressively moved out of memory.

The default value is usually "60". For better AppDynamics performance, the recommended value is "0" (zero).

To configure swappiness

1. Check the current value for swappiness.

```
/sbin/sysctl -a | grep swappiness
```

2. Configure the swappiness parameter.

```
echo 0 > /proc/sys/vm/swappiness
```

Edit the /etc/sysctl.conf file and add following line:

```
vm.swappiness = 0
```

Learn More

- [Install the Controller on Linux](#)

Upgrade the Controller

- [Prior to Upgrade](#)
- [Upgrade](#)
 - To upgrade the Controller
 - To upgrade Controllers configured in HA mode
- [Learn More](#)

During an upgrade, the Controller will not process metrics from the agents because it is not running.

You are not required to stop the agents during the Controller upgrade.

Prior to Upgrade

- Check the most recent Controller System Requirements to determine whether you need to update your systems and/or change your performance profile. See [Controller System Requirements#Hardware Requirements Per Performance Profile].
- If you have changed any global (Controller-wide) properties using the Controller administration console, note your changes. You will have to reset them after upgrade. Controller upgrade does not persist properties set in the Admin interface.
- Before upgrade it is necessary to update the response.varfile with the mysql password, as the upgrade process does not do this automatically. The response.varfile must be up-to-date to enable the controller to log in to its database, even if you are not using this file to perform silent installs.

Update the mysql password in

<controller_installation_directory>/install4j/response.varfile

to match the mysql password in

<controller_installation_directory>/bin/controller.sh or <controller_installation_directory>\bin\controller.bat

In the case of an upgrade, the installer creates a response.varfile and copies it to <controller_installation_directory>/install4j/response.varfile. This is different from a fresh new install, in which case you create the response.varfile in the directory of your choice when you want to install a Controller silently.

Upgrade

To upgrade the Controller

1. Download the latest release from AppDynamics Download Center.
If you prefer to use the Linux shell, see [Downloading from the Linux Shell](#).

2. Stop the Controller application server and database.
You must stop the Controller before performing a Controller upgrade. No data is collected from the time you shut down and start the new version of the Controller.

For Linux:

```
<controller_installation_directory>/controller.sh stop-appserver
```

```
<controller_installation_directory>/controller.sh stop-db
```

For Windows:

```
<controller_installation_directory>\bin\controller.bat stop-appserver
```

```
<controller_installation_directory>\bin\controller.bat stop-db
```

For Windows as a service: Stop and uninstall the service. Choose the same values used when you originally installed the Controller such as ip address, port, performance profile, etc.

```
<controller_installation_directory>\bin\stopControllerSvcs.bat
```

```
<controller_installation_directory>\bin\uninstallControllerSvcs.bat
```

3. Back up the current controller installation directory, in particular the following files:

<controller_installation_directory>/db/db.cnf
<controller_installation_directory>/db/data
<controller_installation_directory>/appserver/domains/domain1/config/domain.xml
<controller_installation_directory>/appserver/domains/domain1/config/ (any other customizations in this directory)
<controller_installation_directory>/appserver/domains/domain1/appagent
<controller_installation_directory>/appserver/domains/domain1/applications/j2ee-apps/controller/controller-web_war/WEB-INF/flex/service (if using SSL)

4. Launch the Controller installer.

Confirm that the installation directory is the current location. The Controller will automatically migrate the data only when the existing installation directory is specified.

The installer completes the upgrade and restarts the Controller.

5. (Optional for Windows) If you want to install the Controller as a Windows service, see [Install the Controller as a Windows Service](#).

6. If, on the advice of AppDynamics Support, you previously manually changed the default profile configurations, .conf or .xml files, re-apply those changes to the newly upgraded files. Contact Technical Support if you have questions.

7. Open a browser and access the AppDynamics user interface:

```
http://<Controller_Host>:<Controller_Port>/controller
```

If the UI does not display the new Controller, do a hard refresh of your browser cache to pick up the new UI.

To upgrade Controllers configured in HA mode

If you configure the high availability mode for the AppDynamics Controller, you must upgrade both the primary and the secondary Controller.

1. On the secondary Controller, stop the application server:

```
<controller_installation_directory>/controller.sh stop-appserver
```

2. Upgrade the primary Controller using the instructions provided in the [To upgrade the Controller](#). Be sure to back up the files.

When you upgrade the primary Controller, the schema changes are replicated to the secondary Controller. Then, when you launch the Controller installer on the secondary Controller, the installer checks the schema version and, because the schema version is the same as the new version, the installer skips any schema upgrades and only upgrades software for the secondary Controller.

3. Verify that on the secondary Controller all the data has been replicated correctly.

- Open a command line utility and go to the <controller_installation_directory>/bin directory.
- Log into the secondary Controller database:

For Linux:

```
controller.sh login-db
```

For Windows:

```
controller.bat login-db
```

Execute following command:

```
SHOW SLAVE STATUS\G
```

This step should provide you following result:

```
Seconds_Behind_Master: $Number_Of_Seconds_Behind_Master
```

If you get a non-zero number for this test, wait until the number becomes zero.

4. Start the application server for the secondary Controller.

- Use the command-line utility and navigate to the <controller_installation_directory>/bin directory.
- Use following command to start the application server for the Controller:

For Linux:

```
<controller_installation_directory>/bin/controller.sh start-appserver
```

For Windows:

```
<controller_installation_directory>\bin\controller.bat start-appserver
```

5. Upgrade the secondary Controller using the instructions provided in the [To upgrade the Controller](#).

6. Stop the application server for the secondary Controller.

- Use the command line utility and go to the secondary Controller's <controller_installation_directory>/bin directory.
- Use following command to stop the application server for the secondary Controller:

For Linux:

```
controller.sh stop-appserver
```

For Windows:

```
controller.bat stop-appserver
```

This will upgrade both the primary and secondary Controllers.

Learn More

- Controller Data Backup and Restore
- Manage Controller High Availability
- Access the Administration Console
- Install an On-Premise Controller Silently

Uninstall the Controller

- To uninstall the Controller
- Learn More

The Controller uninstaller is bundled within the Controller installer as a shell script. This script is in the Controller installation directory.

To uninstall the Controller

1. Open a command shell.
2. Navigate to <Controller_Installation_Directory>/bin.
3. Stop the Controller. See [Start or Stop the Controller](#).
4. Navigate to <Controller_Installation_Directory>.
5. Execute the uninstaller script file to uninstall the Controller.

- On Linux:

```
uninstall.sh
```

- On Windows:

If you run the Controller as a Windows service first use:

```
run bin\uninstallControllerSvcs.bat
```

Then use:

```
run uninstallController.exe
```

Learn More

- Administer the Controller
- Install the Controller
- Install the Controller as a Windows Service

Migrate the Controller between Machines

- To migrate a Controller to new physical machine
- To copy an entire Controller directory to new machine
- **Controller Migration FAQ**
 - After moving the data to the new machine, why do I get the 443, permission denied error?
 - Can I take the hot backup and use that for the new machine?
 - Do I need a new license while moving the Controller from a virtual to a physical environment?

This topic describes how to migrate an existing Controller from a physical or virtual machine to a physical machine. Prior to migrating consider [Backing up the existing Controller](#).

To migrate a Controller to new physical machine

Follow these steps to install the Controller on a new physical machine and migrate the Controller data from the old location.

1. Ensure that the performance profile of the new machine matches the old machine. For profile information see [Controller System Requirements](#).
2. Ensure that the new Controller version matches the old Controller version. If necessary, [upgrade the old Controller first](#).
3. Install a new Controller on the destination machine. **⚠** The old and new Controllers must be the same version.
4. Stop the Controller after installation completes. For details see [Start or Stop the Controller](#).
5. Rename the old Controller machine to a new name and IP and IP address.
6. Rename the new Controller machine to the old Controller name and IP address.
7. Get the new MAC address, provide it to the [AppDynamics Support Team](#) and request a new license file.
8. Export the App Tier configurations from the old Controller and save them to the new machine. For details see [Export and Import Business Application Configurations](#).
9. Shut down the old Controller. By shutting down the old Controller before copying its data, you ensure that all the data is moved to the new machine. For details see [Start or Stop the Controller](#).
10. On the new machine, rename the data directory at <Controller_Installation_Directory>/db; for example, rename "data" to "data_default".
11. Copy the data directory from the original Controller machine to the new machine.
12. Start the Controller on the new machine. For details see [Start or Stop the Controller](#).
13. Import the configuration files for the App Tiers. For details see [Export and Import Business Application Configurations](#).

To copy an entire Controller directory to new machine

You can also copy the Controller directory to the new machine. However, there are two requirements:

- The new machine should have the same IP address and host name as the old machine.
- The new machine should have the exact same directory structure as the old installation location.

Follow the steps listed below for directly copying the Controller directory to the new machine:

1. Shut down the old Controller. By shutting down the old Controller before copying its data, you ensure that all the data is moved to the new machine. For details see [Start or Stop the Controller](#).
2. Copy the <Controller_Installation_Directory> from the old machine to the new machine. **⚠** Copy the directory into the exact same directory structure.
3. Start the Controller on the new machine. For details see [Start or Stop the Controller](#).

⚠ **IMPORTANT:** Do not ever start the Controller on the old machine again!

Controller Migration FAQ

After moving the data to the new machine, why do I get the 443, permission denied error?

This can happen if port 443 is bound to some other application. Change the HTTPS port, for example from 443 to 5443.

Can I take the hot backup and use that for the new machine?

You can do either hot or cold backup, but it is important to get the backup of the data directory from <Controller_Installation_Directory>/db. However, AppDynamics strongly recommends that you perform a cold backup of the data.

Hot backup will not bring the Controller down for a long time, but you will still lose the data when you migrate. This is because hot backup will only have the data from the point that the hot backup started and not when it ended.

Do I need a new license while moving the Controller from a virtual to a physical environment?

If the physical machine has same MAC address, you can reuse the license. If that is not the case, please contact the [AppDynamics](#)

Support Team for a new license.

Controller Install and Admin FAQ

- Controller Installation FAQ
 - Q. What should I do if I forget the credentials during Controller installation?
 - Can I install the Controller on a virtual machine?
 - Do I need a multi-tenant installation?
 - Q. What is Controller High Availability mode and when should I choose it?
 - Q. Is the Controller upgrade seamless or will the existing data get purged?
 - Q. Will my current license work during an upgrade?
 - Q. How do I upgrade a Controller installed as a Windows service?
 - Q. Can a 32-bit Controller installation be upgraded to a 64-bit installation?
- Controller Administration FAQ
 - Q. How do I start and stop the Controller?
 - Q. After I log out, why can't I see the Controller on the Windows machine?
 - How do I uninstall the Controller?
 - Q. How do I access the Controller's application server?
 - Q. After the Controller shuts down, why is there no increase in the amount of free memory on Linux system?
 - Q. Why am I getting "Caused by: java.net.ConnectException: Connection refused" message in the server.log file?
 - Q. Why do I get a stack overflow exception when installing the Controller installation on a Windows machine?
 - Q. What troubleshooting information should I collect if I observe heavy load on the Controller machine?
 - Q. How to trigger automatic collection of Controller logs?
 - Q. Why is there no data in the Metrics Browser?

Controller Installation FAQ

Q. What should I do if I forget the credentials during Controller installation?

The information that you provide during Controller installation creates an "admin" user in the system. This is the information that you use to access the AppDynamics User Interface (UI) for the first time. If you lose this information, contact [AppDynamics Support Team](#) to reset the password for the admin user.

Can I install the Controller on a virtual machine?

The AppDynamics Controller can be installed on virtual machines. However, make sure that the virtual machine meets the hardware resource requirements equivalent to physical machines. The virtualized storage subsystem must provide the required I/O capacity. To verify the I/O capacity, you can run Disk I/O rate tests. For details see [Controller System Requirements](#).

Do I need a multi-tenant installation?

You can configure the AppDynamics Controller in either single-tenant (single account) or multi-tenant (multiple account) modes. AppDynamics recommends single-tenant mode for most installations. For more details, see [Controller Tenant Mode](#).

You choose the tenancy mode when you run the Controller installer. If you need to modify the tenancy mode later, see [Controller Tenant Mode](#).

Q. What is Controller High Availability mode and when should I choose it?

When using AppDynamics you may need to ensure the availability of the Controller. Availability refers to the ability to cope with and if necessary recover from hardware or software failures.

A high availability mode configuration is two machines, both running the AppDynamics Controller. If the primary machine is brought down, the secondary machine assumes the workload.

Configure HA mode for the Controller in either of the following situations:

- When you want to ensure that any critical software or hardware failure cannot disrupt Controller operations.
- When you want to avoid the complexity of performing hot backups.

For more information see [Manage Controller High Availability](#).

Q. Is the Controller upgrade seamless or will the existing data get purged?

The Controller upgrade is seamless and retains all the existing client configurations, data, and reports. See [Upgrade the Controller](#).

Q. Will my current license work during an upgrade?

If you are moving from an older version to a newer version and have a license for the older version, then that license should work when upgrading the Controller to a new version.

However, if you had a temporary license for the old version and now have a new license, then this new license will not work on the old Controller. In this case, you should upgrade the Controller to the latest version.

Q. How do I upgrade a Controller installed as a Windows service?

To upgrade a Controller that is installed as a Windows service:

1. Uninstall the Windows service by running the <Controller_Installation_Directory>/bin/uninstallControllerSvcs script.

2. Restart the host system.

3. Install the latest version of the Controller and then configure the Controller as a Windows service.

For more information see [Install the Controller as a Windows Service](#).

Q. Can a 32-bit Controller installation be upgraded to a 64-bit installation?

Yes. Follow the usual upgrade procedure at [Upgrade the Controller](#).

Controller Administration FAQ

Q. How do I start and stop the Controller?

See [Start or Stop the Controller](#).

Q. After I log out, why can't I see the Controller on the Windows machine?

The Controller is not installed as a Windows service by default. If you log out, in most cases you will shut down the Controller.

To verify whether the Controller is shut down, check for the status of Controller Glassfish and MySQL processes. If these processes are active, that means the Controller is not shut down.

For more information see [Install the Controller as a Windows Service](#).

How do I uninstall the Controller?

See [Uninstall the Controller](#).

Q. How do I access the Controller's application server?

The AppDynamics Controller is a J2EE application that uses the GlassFish Application Server. Use the following URL to access the Controller's application server:

`http://Controller_Host:4848/login.jsf`

The default port for the Controller Application Server is 4848 and is configured during Controller installation. If you set a different value for this port, use that value in the URL.

To log in to the application server administrative console, use "admin" as the username and the password that is located in the <controller_install_directory>/.passwordfile file.

Q. After the Controller shuts down, why is there no increase in the amount of free memory on Linux system?

You should not worry about the "free memory" value as it will always trend towards zero. The Linux kernel tries to keep its cache as large as possible. As a result, the Linux kernel does not release the memory even after process termination. The memory is freed only if it is required by another process. For more detailed information, refer to following article: [Memory Management on Linux Systems](#).

Q. Why am I am getting "Caused by: java.net.ConnectException: Connection refused" message in the server.log file?

AppDynamics recommends that you verify the embedded MySQL database for the Controller when you get following exception in

server.log file of the Controller.

```
*Server log exception:* "Caused by: java.net.ConnectException: Connection refused"
```

To verify that the Controller database is running properly use one of the following commands on Linux environment:

Linux	Windows	Description
lsof -i:3388	SysInternals Process Explorer, will provide you list of files opened by process with pid 3388.	List open files opened by process with pid 3388.
netstat -anp grep 3388	netstat -ano find "3388"	List all networking ports opened by process with pid 3388.
ps -aef grep mysql	tasklist /v find "mysql"	Lists all processes and then checks if the process with name "mysql" is active and alive.

If no processes are found, it indicates that the Controller database was incorrectly terminated. Start the Controller database again and verify the Controller server.log file for any error messages. To start the database process again see [Controller Database Scripts](#).

Q. Why do I get a stack overflow exception when installing the Controller installation on a Windows machine?

This exception is usually caused when you set the -Xss option to a lower value. We recommend changing this value to 96000.

Q. What troubleshooting information should I collect if I observe heavy load on the Controller machine?

If the Controller machine experiences heavy load, [contact the AppDynamics Support Team](#) with the following information:

- Send all the <Controller_Installation_Directory>/logs/files, in particular the server.log files. You can also use the log file utility to collect the Controller logs.
- If the Controller runs out of memory, it generates a heap dump. Send all the <Controller-Installation-Directory>/appserver/domains/domain1/config/hprof files.
- Send all the <Controller-Installation-Directory>/appserver/domains/domain1/config/gc.log files.
- Send information about the hardware and operating system configuration of the machine that is currently hosting the Controller, including operating system, bit version, CPU cores, clock speed, disk configuration, and RAM.
- Indicate the Performance profile of Controller. See [Controller Performance Profiles](#).

Q. How to trigger automatic collection of Controller logs?

Use the following console commands to trigger automatic capture of Controller log files:

- On Linux:

```
controller.sh zip-logs
```

- On Windows:

```
controller.bat zip-logs
```

Q. Why is there no data in the Metrics Browser?

Sometimes the agents are not correctly configured. Begin troubleshooting by looking at the server.log file.

All log files for Controller are located in the <Controller_Installation_Directory>/logs folder.

Error Message	Solution
---------------	----------

Error receiving metrics (node not properly modeled yet: Could not find component for node:	This error means the app agent tried to upload metric data for a specific node, but the node does not belong to any tier. Nodes must belong to tiers and these tiers must belong to a business application in order to receive metric data for that node. See Logical Model .
Received Metric Registration request for a machine that is NOT registered to any nodes. Sending back null!	This error indicates that the Controller received a registration request for metrics for a Machine Agent that listed a machine ID not yet associated with any node. Configure the Machine Agent to associate with the correct application, tier, and node. See Install the Machine Agent .

Install the Controller on Windows

- The AppDynamics Controller Installer for Windows
- Supported Windows Operating Systems for Controller
- Pre-installation Checklist for Windows
- Download and Launch the Windows Installer
 - To download and launch the Controller on Windows
 - To install the Controller on Windows
- Verifying Controller Installation
- Post-installation Checklist for Windows
 - Apply the License File
 - Configure Backups
 - Install the Controller as a Windows Service (optional)
 - Restart the Controller If You Reboot
- [Learn More](#)

This topic describes how to install the AppDynamics Controller in a Windows environment.

For an overview of the Controller and its requirements, including supported operating systems, see [Controller System Requirements](#)

The AppDynamics Controller Installer for Windows

The AppDynamics Controller installer for Windows is an executable binary that provides a graphical wizard to guide you through the installation. You do not require any additional software; all software components required for Controller installation are bundled as part of the installer binary.

Supported Windows Operating Systems for Controller

The following operating systems are supported:

- Windows 2003 Server
- Windows 2008 Server

Pre-installation Checklist for Windows

- For optimum stability and performance, AppDynamics strongly recommends that you install the Controller on a dedicated machine.
- The disk space requirements differ for each of the [Controller Performance Profiles](#). Verify that there is enough disk space for the performance profile that describes your environment.
- Confirm the correct CPU, RAM and Disk I/O capacity for the [Controller Performance Profile](#) you plan to install. Run the Disk I/O measurement tests described at [To measure available Disk I/O](#).
- Verify that a ZIP/UNZIP utility is installed on the machine to which you will download the installer.
- Open the HTTP port that the Controller will use to communicate with agents and the AppDynamics UI.
- Verify that you have administrative privileges on the Windows machine to launch the Controller installer.

Download and Launch the Windows Installer

To download and launch the Controller on Windows

1. Download the Windows Controller binary, controller_windows_<version>.exe, from the [AppDynamics Download Center](#).
2. Execute the downloaded file.

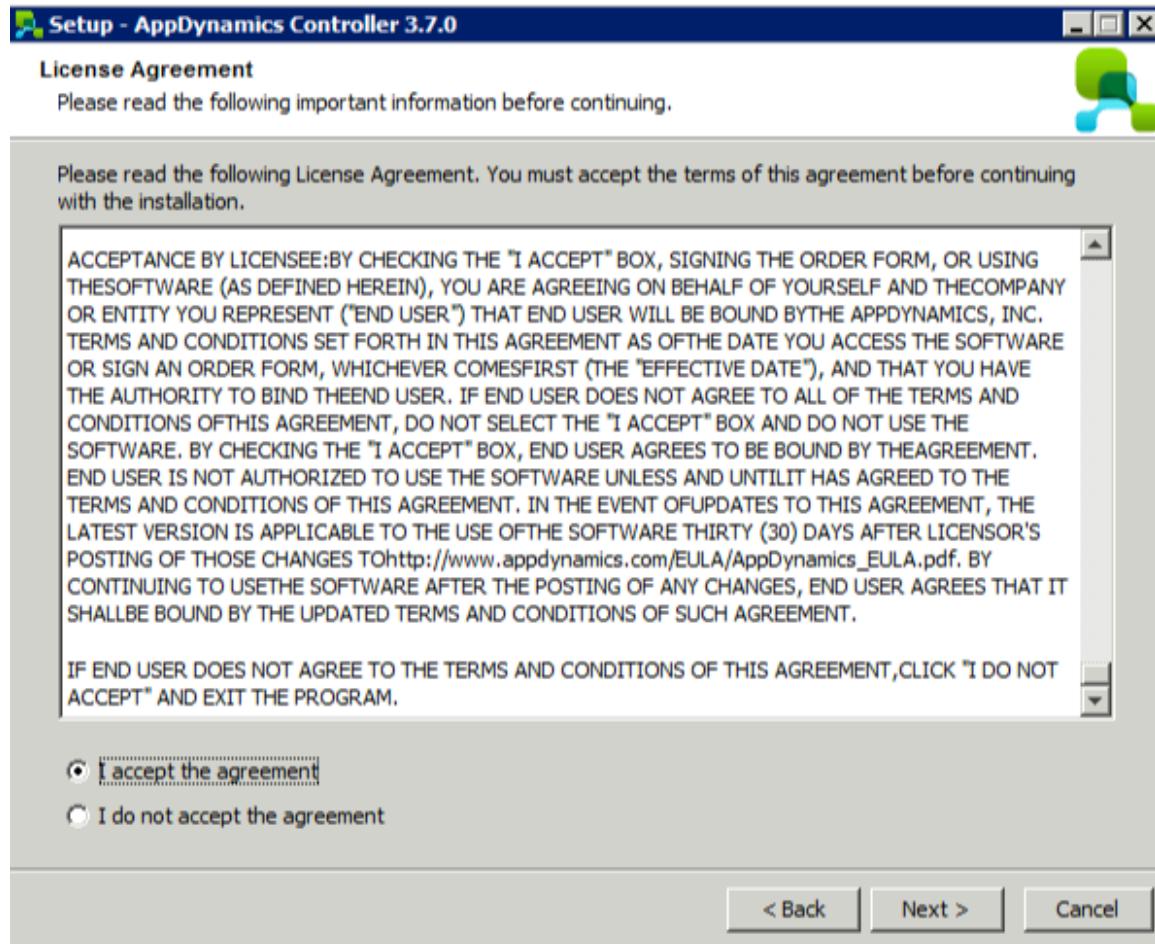
It is also possible to execute the installer silently without any user interaction. See [Install an On-Premise Controller Silently](#).

To install the Controller on Windows

1. In the Welcome screen click **Next**.

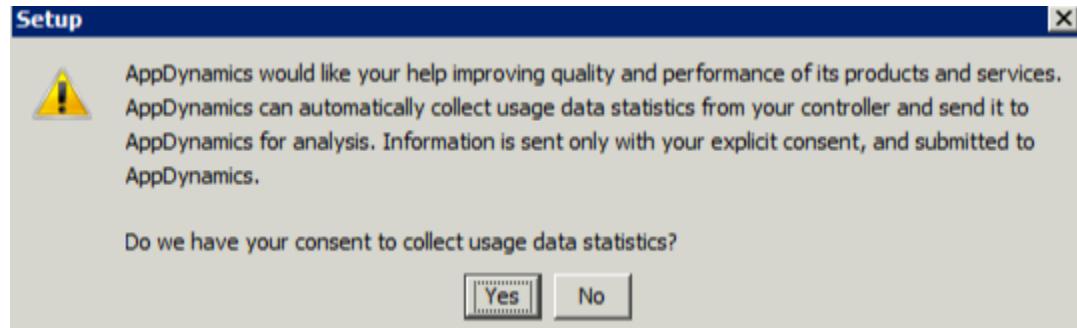
The license agreement displays.

2. Scroll down to the bottom and accept the license agreement, then click **Next**.

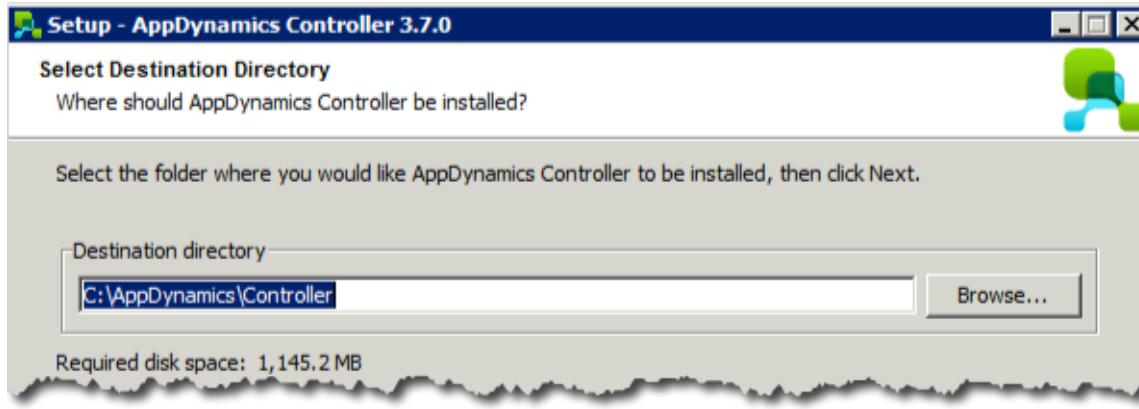


3. Click **Yes** to grant or **No** to deny AppDynamics permission to collect usage data statistics from your controller.

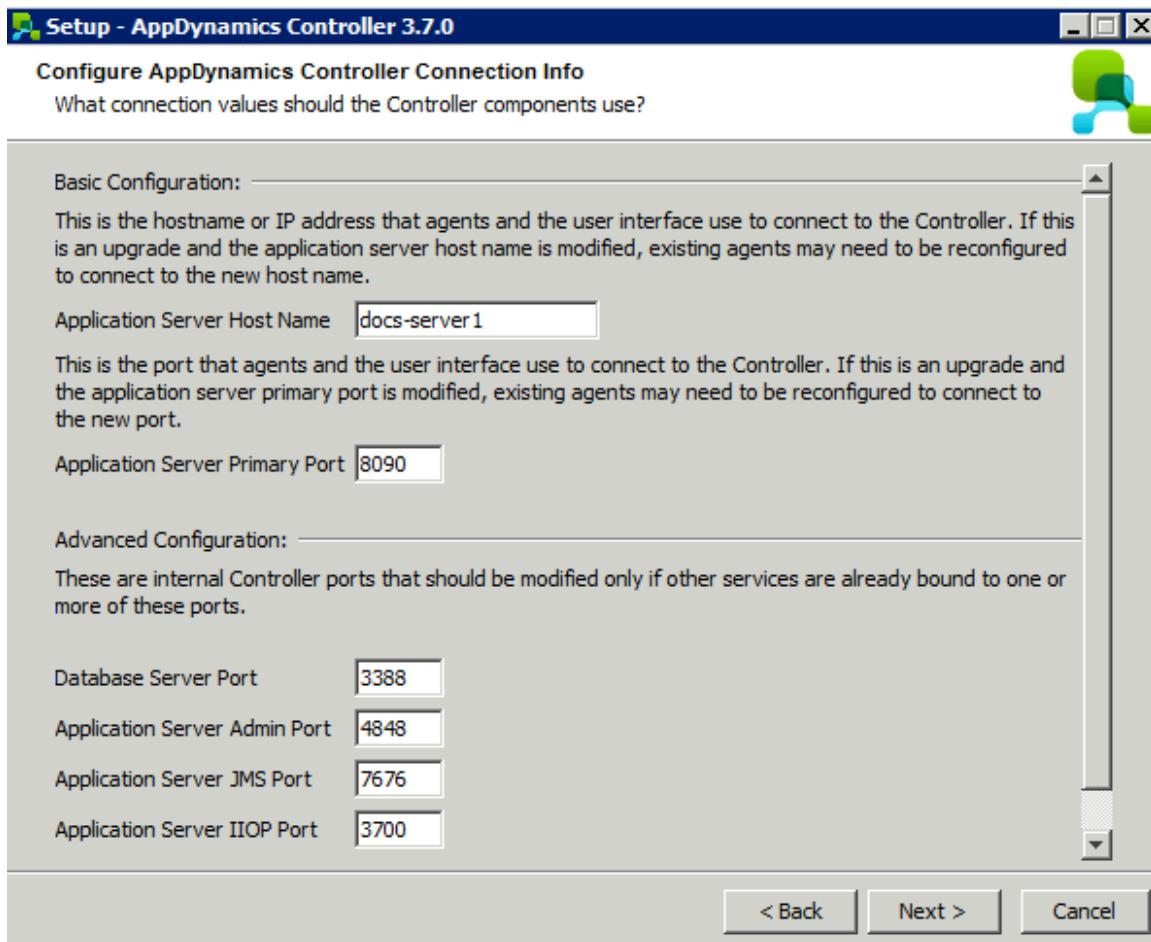
The purpose of collecting data is to improve the quality of AppDynamics products and services.



4. Set the path of the Controller installation directory in the Destination directory field, then click **Next**.



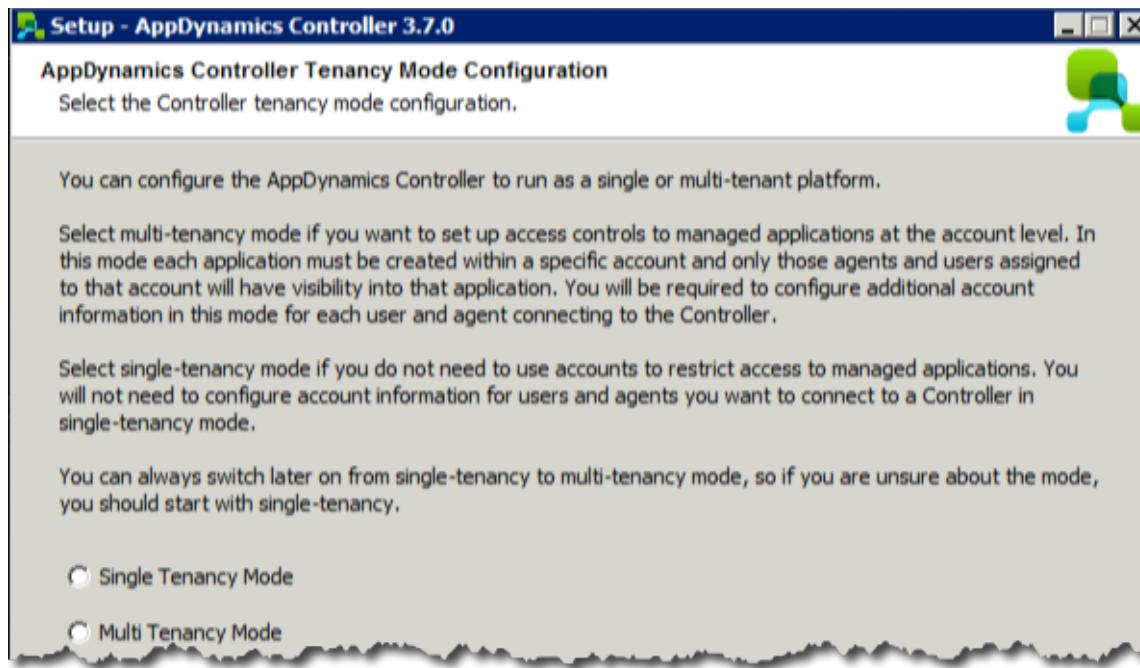
5. Type in the Application Server Host Name field the host name (or IP address) and in the Application Server Primary Port field the primary port that the AppDynamics agents and user interface will use to connect to the Controller.



6. Examine the default configuration of the internal Controller ports and modify the settings as necessary if other services are bound to these ports. Then click **Next**.

7. Verify the displayed configured settings and click **Back** if you want to change any of them. Click **Next** to continue.

8. Choose single-tenant or multi-tenant mode for the Controller, then click **Next**.

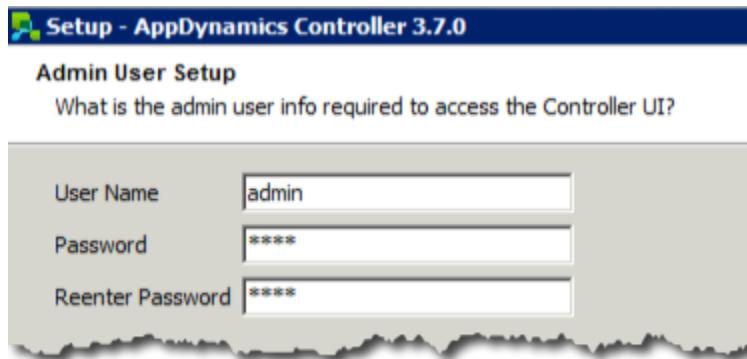


See Controller Tenant Mode for more information about these modes.

9. Configure Controller access and authentication. Which procedure you use depends on which tenancy mode you selected in the previous step.

If you selected single-tenant mode:

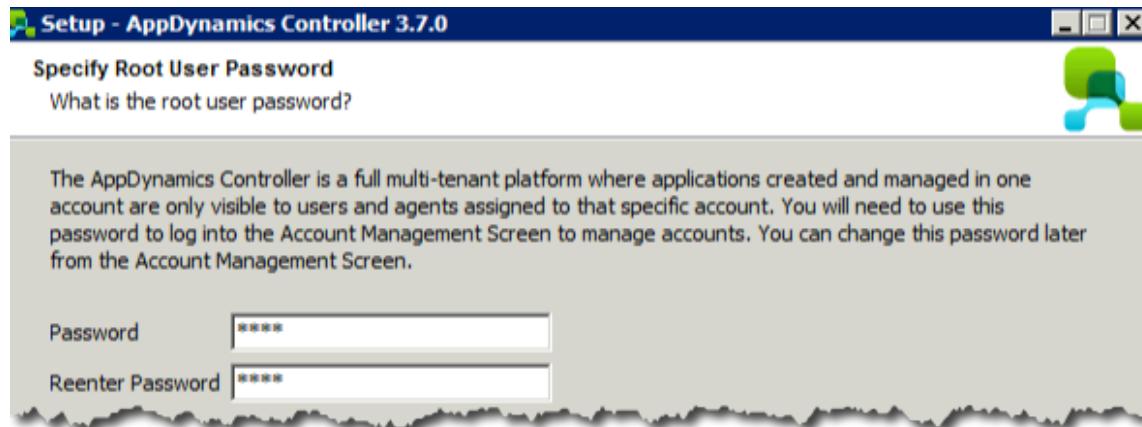
a. Enter admin user name and password. This user will be able to access the single-tenant controller and create other users. The admin user is the user described as the [AppDynamics Administrator](#).



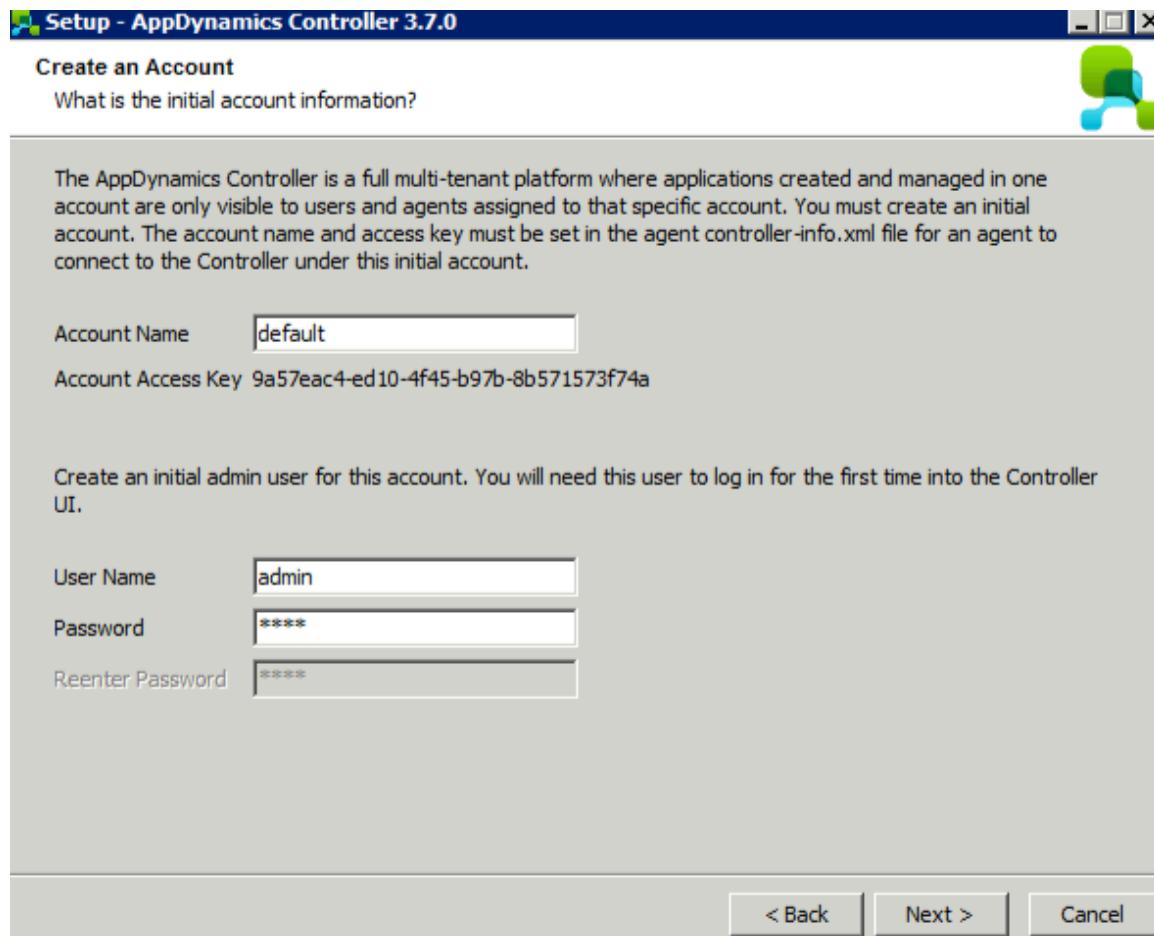
b. Click **Next**.

If you selected multi-tenant mode:

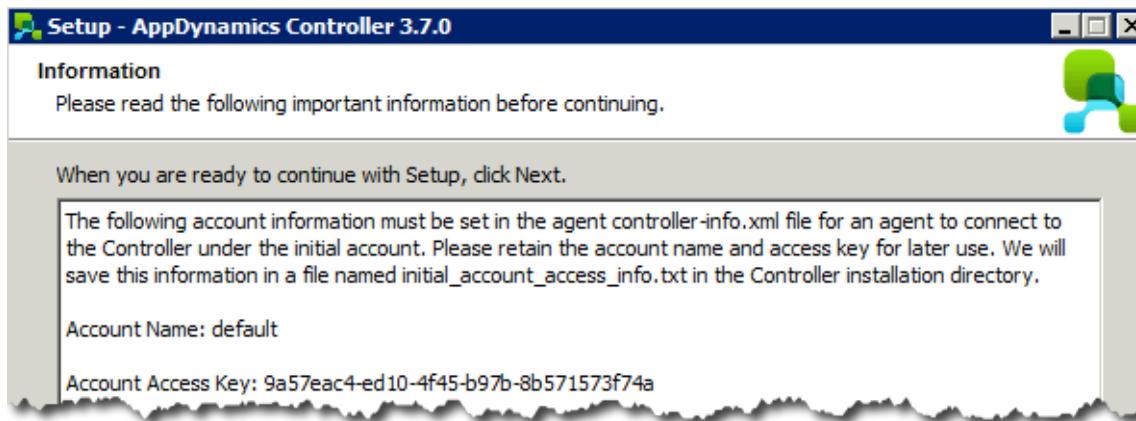
a. Enter a password. This password is used to log into the AppDynamics administration console, where a user can create and manage AppDynamics accounts and configure various controller settings. See [Access the Administration Console](#) for more information about this console.



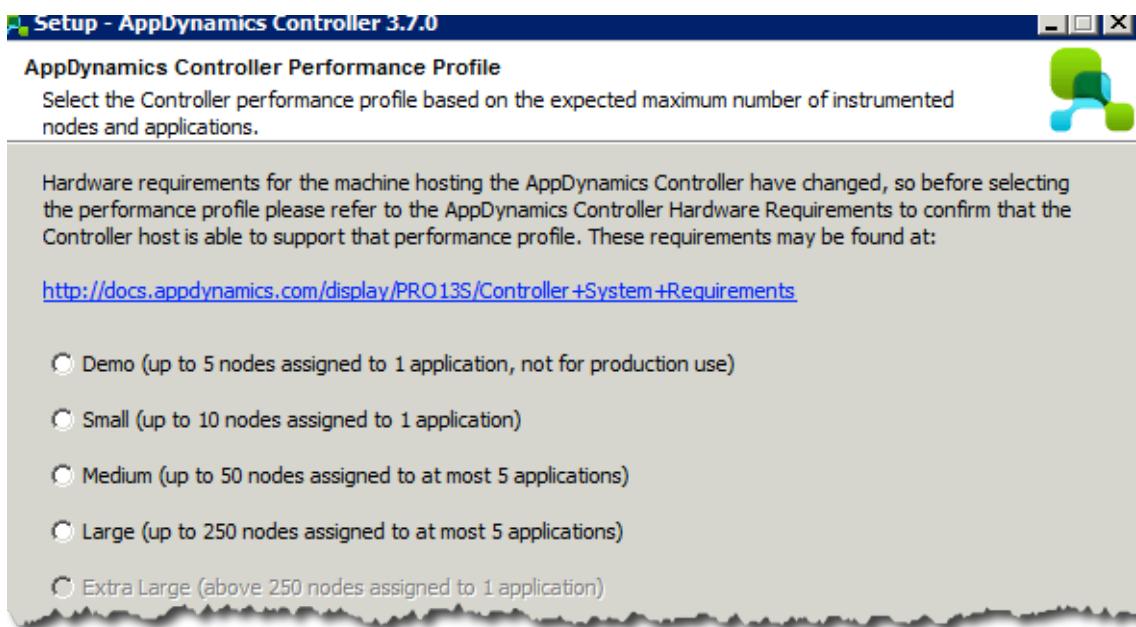
- b. Click **Next**.
- c. In the Account Name field, type the account name for the initial account on the multi-tenant Controller. AppDynamics generates an account key for this initial account.
- d. In the User Name field enter the user name of the admin user for the initial account. This user is able to create other users in this account.
- e. In the Password field enter the user name of the admin user for the initial account. Re-enter the password, then click **Next**.



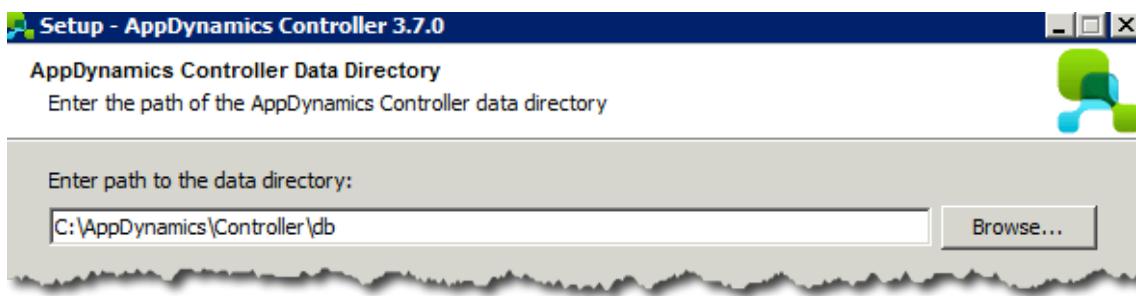
- f. Review and save the settings for the initial account settings and click **Next**.



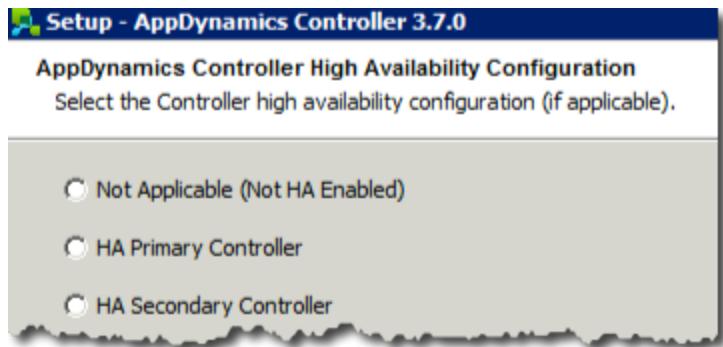
10. Select the Controller performance profile that matches your requirements, then click **Next**.



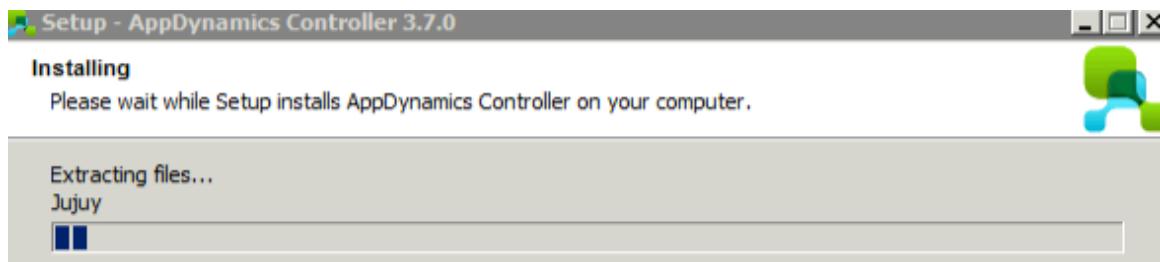
11. In the path to the data directory field, enter the path to the directory where the Controller's mysql data will be stored or select the default, then click **Next**.



12. Select the high availability configuration for the Controller or Not Applicable if you are not enabling HA for this Controller. Then click **Next**.
For more information see [Controller High Availability](#).



Installation begins.



Verifying Controller Installation

Wait for the installation status screen to show that the installation is complete. Then the Controller will automatically start.

To verify the success of your installation, do one of the following:

- Click the link to the Controller provided in the installer's "Finish" screen.
- Open a browser to the URL that you configured for the Controller.

```
http://<application_server_host_name>:<http-listener-port>/controller
```

Post-installation Checklist for Windows

- Apply the License File
- Configure Backups
- Install the Controller as a Windows Service (optional)
- Restart the Controller If You Reboot

Apply the License File

The license file (license.lic) was provided in your Welcome email from AppDynamics. Copy the downloaded license file (license.lic) to the Controller installation directory:

Allow up to 5 minutes for the license change to take effect.

Configure Backups

See Controller Data Backup and Restore.

Install the Controller as a Windows Service (optional)

By default the Controller is not installed as a Windows service. In most cases if you log out the Controller will shut down.

See [Install the Controller as a Windows Service](#).

Restart the Controller If You Reboot

If you reboot the Controller machine you need to restart the Controller, unless it is installed as a Windows service that is configured to start automatically. See [Start or Stop the Controller](#).

Learn More

- Controller System Requirements
- Install an On-Premise Controller Silently
- Install the Controller as a Windows Service
- Configure Controller VIP
- Controller Tenant Mode
- Controller High Availability
- Controller Data Backup and Restore
- Start or Stop the Controller

Install the Controller as a Windows Service

- To install the Controller as a Windows service
- Learn More

You can run the AppDynamics Controller as a background process by installing it as a Windows service. As a Windows service, the Controller will continue running after you exit its console or after the user running the Controller logs out.

To install the Controller as a Windows service

1. If Controller is not already installed, install the Controller. See [Install the Controller on Windows](#).
2. Shut down the Controller.
 - Open a command shell.
 - Navigate to the /bin directory of the Controller installation directory.
 - Stop the Controller processes:

stopController.bat
3. Execute the <Controller-Installation-Directory>/bin/installControllerSvcs script.
4. Reboot the host machine after the Controller is shut down.
5. Start the Controller as a service by executing the <Controller-Installation-Directory>/bin/startControllerSvcs script.

Learn More

- [Install the Controller on Windows](#)
- [Uninstall the Controller](#)
- [Configure Controller VIP](#)

Install an On-Premise Controller Silently

- To install a Controller silently
- [Settings in response.varfile](#)
- [Sample response.varfile](#)
- [Learn More](#)

This topic describes how to perform a silent installation of the Controller. This installation procedure does not require user interaction and does not display any indications of its progress.

Before performing a silent install of the Controller, read the system requirements and prerequisites:

- For all platforms, see [Controller System Requirements](#).
- For Linux, see [Pre-installation Checklist for Linux](#)
- For Windows, see [Pre-installation Checklist for Windows](#).

To install a Controller silently

1. Create the response.varfile file, which contains all the installation settings.
There is no response.varfile in a new installation.

2. Run the installation command passing the response.varfile as an argument:
On both Linux and Windows, the command to start the silent installation is:

```
./controller_64bit_linux_v3.2.3.sh -q -varfile /root/response.varfile
```

Settings in response.varfile

The values listed below are configured in response.varfile.

serverHostName	Server host name or IP address that AppDynamics agents and the AppDynamics UI use to connect to the Controller. Note that "localhost" and "127.0.0.1" are not valid settings.
serverPort	8090
databasePort	3388; can change if another process is bound to this port
adminPort	4848; can change if another process is bound to this port
jmsPort	7676; can change if another process is bound to this port
iiopPort	3700; can change if another process is bound to this port
sslPort	8181; can change if another process is bound to this port
controllerTenancyMode	single multi; Single tenant recommended for most installations. See Controller Tenant Mode .
haControllerType	demo small medium large huge See Controller Hardware Requirements for information about how the types correspond to sizes. If not specified, defaults to NotApplicable which means that HA is not enabled.
accountName	your AppDynamics account name. Ignored if controllerTenancyMode=single.
accountAccessKey	your AppDynamics account access key. Ignored if controllerTenancyMode=single.
rootUserPassword	root password of the server. Ignored if controllerTenancyMode=single.
rootUserRePassword	root user repeat password Ignored if controllerTenancyMode=single.
userName	admin This creates the AppDynamics admin user. Ignored if controllerTenancyMode=multi. See AppDynamics Administrator
password	admin user password Save this as you will need it to log into the AppDynamics, where you can create other administrative users and change the password. Ignored if controllerTenancyMode=multi.
rePassword	admin user repeat password. Ignored if controllerTenancyMode=multi.
sys.installationDir	Directory in which the Controller is installed. Can be blank. Ignored if controllerTenancyMode=multi.
sys.languageId	en
disableEULA	true
enableUDC	true false
mysql-root-user-password	root password to the Controller's mysql database.
actualDbDir	path to your data directory

Sample response.varfile

The following is a sample response.varfile.

```
#sampleresponse file for AppDynamics Controller 3.7
iiopPort=3700
serverPort=8090
serverHostName=demoserver
haControllerType=demo
controllerTenancyMode=single
adminPort=4848
sys.languageId=en
jmsPort=7676
sys.installationDir=/Applications/AppDynamics/Controller
mysql-root-user-password=DRvYYv9eq6
databasePort=3388
userName=admin
accountName=customer99
sslPort=8181
actualDbDir=/Applications/AppDynamics/Controller/db
```

Learn More

- [Install the Controller](#)

Install the Self-Service Controller on Linux

- [Pre-installation Checklist for Linux](#)
- [Run the Linux Installer](#)
 - To install the Controller on Linux in GUI Mode
- [Verify Controller Installation](#)
- [Post-installation Checklist for Linux](#)
 - Apply the License File
 - Configure swappiness if the Linux kernel is newer than 2.1.10
- [Learn More](#)

This topic describes how to install the AppDynamics Self-service Trial Controller in a Linux environment. If you are installing a standard controller see [Install the Controller on Linux](#).

Pre-installation Checklist for Linux

- Confirm operating system requirements:
 - Redhat Enterprise Linux (RHEL) 6.1, 6.2
 - CentOS 6.1, 6.2
 - Fedora 14
 - Ubuntu 8, 12
 - Open SUSE 11.x
 - SUSE Linux Enterprise Server
 - Cloud: Amazon EC2, Rackspace, Azure
- Confirm browser and Flash support:
 - Mozilla FireFox v 3.x, 4.x, 5.0
 - Internet Explorer 6.x, 7.x, 8.x, 9.0.
 - Safari 4.x, 5.x
 - Google Chrome 10.x, 11.x, 12.x
 - The Controller UI requires Flash Player 10 or greater; AppDynamics recommends version 11.
- Confirm disk space requirements:
 - 50 GB
- Confirm CPU, RAM and Disk I/O capacity:
 - 2 GB RAM
 - 2 CPU Cores
 - 1.5 GHz minimum
 - Disk I/O: 50 MB/sec write, 50 MB/sec read, 1.5 MB/sec random
- The Controller requires that libaio be installed on the same machine. For more information see [Install libaio on Linux](#).
- Make sure the file descriptor limit is set to at least 65535. For more information see [Configure File Descriptor Limits on Linux](#).

- Verify that you have Read/Write/Execute permissions on the directory where you install the Controller.
- Verify that a ZIP/UNZIP utility is installed on the machine to which you will download the installer.
- Open the HTTP port that the Controller will use to communicate with agents and the AppDynamics UI.
- Unzip the downloaded file. You should have a .lic license file and a .sh installer.
- Assign execute permissions to the downloaded Controller .sh installer.

Run the Linux Installer

1. Execute the Controller installer .sh file.

By default the installer tries first to start in GUI mode. If the system is not set up to display the GUI, it will start in console mode. You can force the installer to start in console mode by passing the **-c** option to the installer.

2. When the installer starts, type **o**, then press the Enter key.

```
This will install AppDynamics Controller on your computer.
OK [o, Enter], Cancel [c]
o
Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.
```

3. Press the Enter key several times to get the entire license agreement. At the end, type **1** to accept the license agreement.

```
I accept the agreement
Yes [1], No [2]
1
```

4. Type **1** to grant AppDynamics permission to collect usage data statistics from your controller, then press the Enter key.

```
Do we have your consent to collect usage data statistics?
Yes [1], No [2]
```

5. Set the path of the Controller installation directory or accept the default, then press the Enter key.

```
Where should AppDynamics Controller be installed?
[/Applications/AppDynamics/Controller]
```

6. Type the host name or IP address that the AppDynamics agents and user interface will use to connect to the Controller, then press the Enter key.

```
What connection values should the Controller components use?
Basic Configuration:
This is the hostname or IP address that agents and the user interface use to connect to the Controller. If this is an upgrade and the application server host name is modified, existing agents may need to be reconfigured to connect to the new host name.
Application Server Host Name
[osxltldavi.local]
```

7. Type the primary HTTP port that the AppDynamics agents and user interface will use to connect to the Controller, or accept the default. Then press the Enter key.

```
This is the port that agents and the user interface use to connect to the Controller. If this is an upgrade and the application server primary port is modified, existing agents may need to be reconfigured to connect to the new port.
Application Server Primary Port
[8090]
```

8. Accept the default internal controller port settings or configure different ports if the defaults are already in use. Then press the Enter key.

These are internal Controller ports that should be modified only if other services are already bound to one or more of these ports.

Database Server Port
[3388]

Application Server Admin Port
[4848]

Application Server JMS Port
[7676]

Application Server IIOP Port
[3700]

Application Server SSL Port
[8181]

9. Type **1** to configure the Controller in single-tenancy mode.

Select the Controller tenancy mode configuration.
You can configure the AppDynamics Controller to run as a single or multi-tenant platform.

Select multi-tenancy mode if you want to set up access controls to managed applications at the account level. In this mode each application must be created within a specific account and only those agents and users assigned to that account will have visibility into that application. You will be required to configure additional account information in this mode for each user and agent connecting to the Controller.

Select single-tenancy mode if you do not need to use accounts to restrict access to managed applications. You will not need to configure account information for users and agents you want to connect to a Controller in single-tenancy mode.

You can always switch later on from single-tenancy to multi-tenancy mode, so if you are unsure about the mode, you should start with single-tenancy.

Single Tenancy Mode [1], Multi Tenancy Mode [2]

10. Type the admin user name and password and confirm the password and press Enter.
This user will be able to access the single-tenant Controller and create other users. The admin user is the user described as the AppDynamics Administrator.

Single Tenancy Mode [1], Multi Tenancy Mode [2]
1
What is the admin user info required to access the Controller UI?
User Name
[]
admin
Password
Reenter Password

Press the Enter Key.

Select the Controller performance profile based on the expected maximum number of instrumented nodes and applications.

11. Type **1** for the Demo performance profile and press Enter.

Demo (up to 5 nodes assigned to 1 application, not for production use) [1], Small (up to 10 nodes assigned to 1 application) [2], Medium (up to 50 nodes assigned to at most 5 applications) [3], Large (up to 250 nodes assigned to at most 5 applications) [4], Extra Large (above 250 nodes assigned to 1 application) [5]

12. Type the path to data directory or accept the default. Press Enter.
The data directory is where the Controller mysql data is stored.

Enter the path of the AppDynamics Controller data directory
Enter path to the data directory:
[/Applications/AppDynamics/Controller/db]

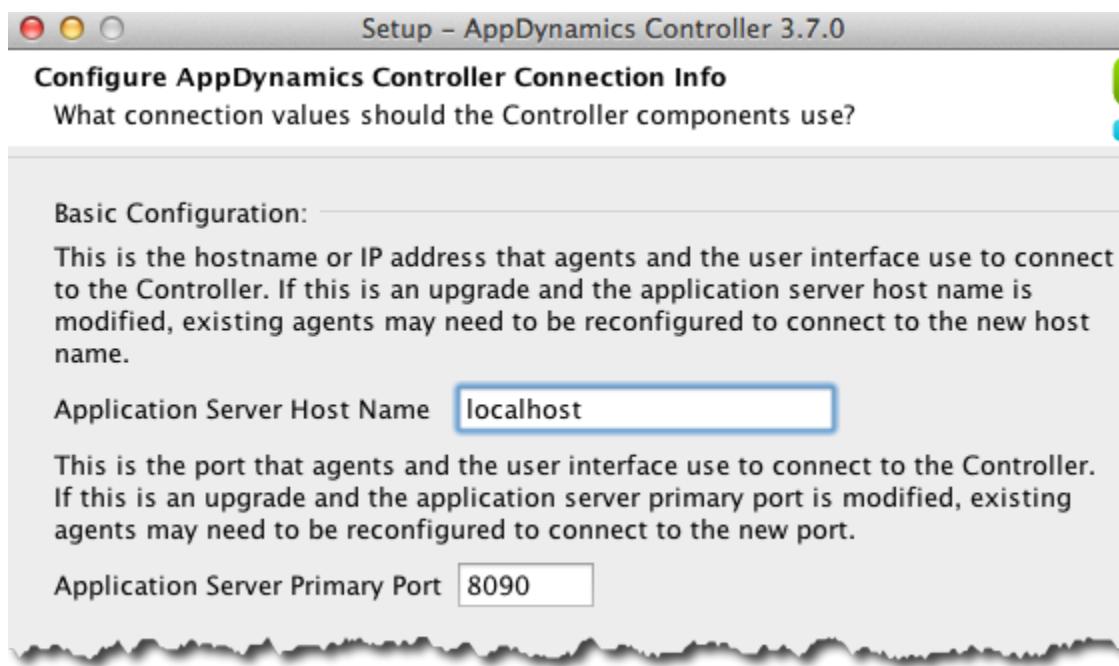
13. Type 1 as you do not need to enable HA for this Controller.

Select the Controller high availability configuration (if applicable).
Not Applicable (Not HA Enabled) [1], HA Primary Controller [2], HA Secondary Controller [3]

14. Press Enter to start the installation.

To install the Controller on Linux in GUI Mode

1. In the Welcome screen click **Next**.
2. Scroll down to the bottom and accept the license agreement then click **Next**.
3. Click **Yes** to grant AppDynamics permission to collect usage data statistics from your controller.
The purpose of collecting data is to improve the quality of AppDynamics products and services.
4. Set the path of the Controller installation directory in the Destination directory field then click **Next**.
5. Configure the host name (or IP address) and primary port that the AppDynamics agents and user interface will use to connect to the Controller.



6. Scroll down to examine the default configuration of the internal Controller ports and modify the settings as necessary if other services are bound to these ports. Then click **Next**.

Advanced Configuration:

These are internal Controller ports that should be modified only if other services are already bound to one or more of these ports.

Database Server Port	3388
Application Server Admin Port	4848
Application Server JMS Port	7676
Application Server IIOP Port	3700
Application Server SSL Port	8181

7. Verify the displayed configured settings and click **Back** if you want to change any of them. Click **Next** to continue.
8. Choose single-tenancy mode for the Controller, then click **Next**.
9. Enter an admin user name and password then click **Next**.
10. Select the Demo performance profile.
11. Select the default data directory path then click **Next**.
12. Select **Not Applicable** as you are not enabling HA for this Controller.

The installer installs the Controller.

Verify Controller Installation

When the installer completes the Controller automatically starts.

To verify the success of your installation, open a browser to the URL that you configured for the Controller.

```
http://<application_server_host_name>:<http_listener_port>/controller
```

Post-installation Checklist for Linux

- Apply the License File
- Configure swappiness if the Linux kernel is newer than 2.1.10

Apply the License File

The license file (license.lic) was provided in the downloaded ZIP file. Copy the downloaded license file to the Controller installation directory.

Allow up to 5 minutes for the license change to take effect.

Configure swappiness if the Linux kernel is newer than 2.1.10

For better performance, configure swappiness. For details see [Configure Swappiness on Linux](#). This feature is available only for Linux kernel versions greater than 2.1.10.

Learn More

- Start or Stop the Controller

Install the Self-Service Controller on Windows

- Pre-installation Checklist for Windows
- Run the Windows Installer
- Verify Controller Installation
- Post-installation Checklist for Windows
 - Apply the License File
 - Install the Controller as a Windows Service (optional)
 - Restart the Controller If You Reboot
- [Learn More](#)

This topic describes how to install the AppDynamics Self-Service Trial Controller in a Windows environment.

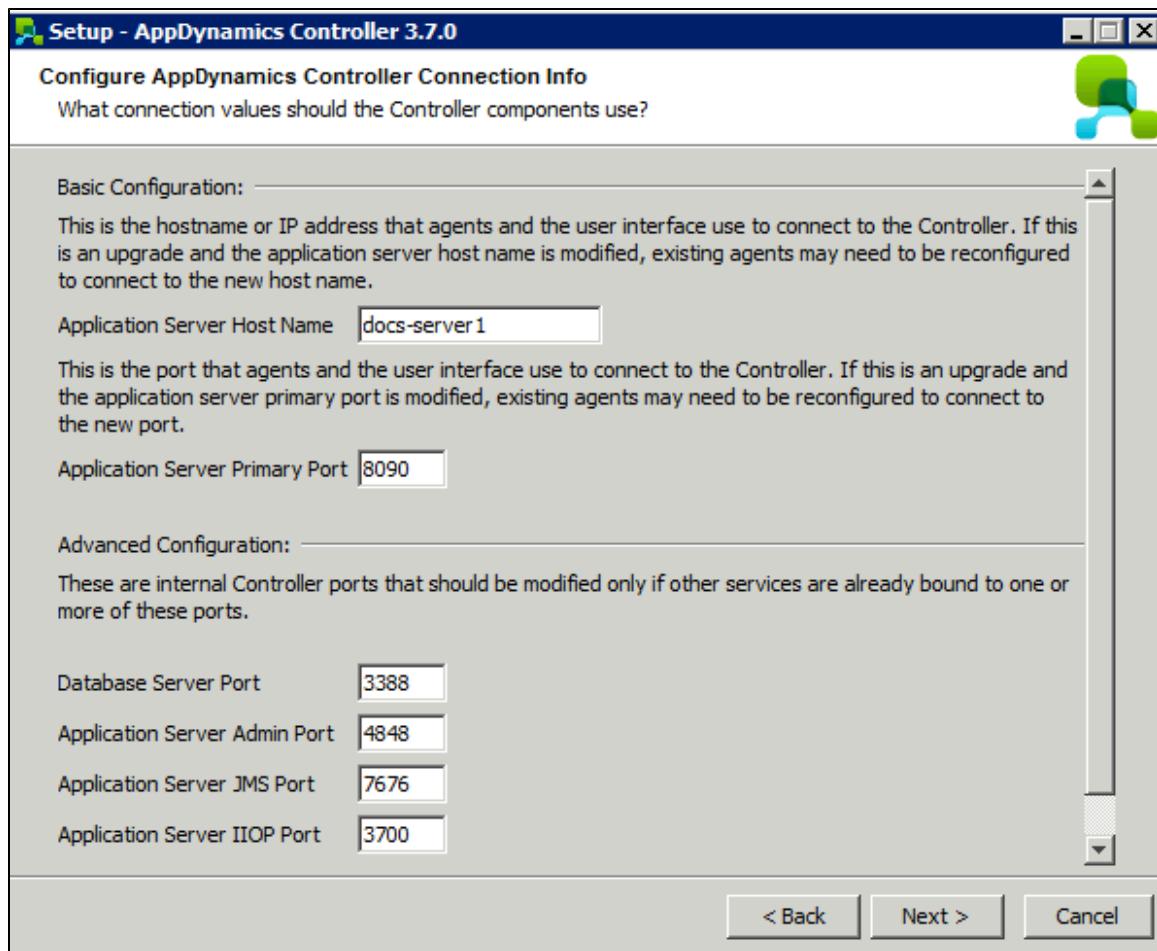
For an overview of the Controller and its requirements, including supported operating systems, see [Controller System Requirements](#)

Pre-installation Checklist for Windows

- Confirm the operating system requirements:
 - Windows 2003 Server
 - Windows 2008 Server
 - Windows 2012 Server
 - Windows 8
- Confirm browser and Flash support:
 - Mozilla FireFox v 3.x, 4.x, 5.0
 - Internet Explorer 6.x, 7.x, 8.x, 9.0
 - Safari 4.x, 5.x
 - Google Chrome 10.x, 11.x, 12.x
 - The Controller UI requires Flash Player 10 or greater; AppDynamics recommends version 11.
- Confirm disk space requirements:
 - 50 GB
- Confirm CPU, RAM and Disk I/O capacity:
 - 2 GB RAM
 - 2 CPU Cores
 - 1.5 GHz minimum
 - Disk I/O: 50 MB/sec write, 50 MB/sec read, 1.5 MB/sec random
- Verify that a ZIP/UNZIP utility is installed on the machine to which you will download the installer.
- Open the HTTP port that the Controller will use to communicate with agents and the AppDynamics UI.
- Verify that you have administrative privileges on the Windows machine to launch the Controller installer.
- Download and unzip the controller_<32bit or 64bit>_windows.zip file. You should have a .lic license file and a Controller installer .exe file.

Run the Windows Installer

1. Execute the Controller installer .exe file.
2. In the Welcome screen click **Next**.
The license agreement displays.
3. Scroll down to the bottom and accept the license agreement, then click **Next**.
4. Click **Yes** to grant AppDynamics permission to collect usage data statistics from your controller.
The purpose of collecting data is to improve the quality of AppDynamics products and services.
5. Accept the default installation directory by clicking **Next**.
6. Type in the Application Server Host Name field the host name (or IP address) and in the Application Server Primary Port field the primary port that the AppDynamics agents and user interface will use to connect to the Controller.



7. Examine the default configuration of the internal Controller ports and modify the settings as necessary if other services are bound to these ports. Then click **Next**.

8. Verify the displayed configured settings and click **Back** if you want to change any of them. Click **Next** to continue.

9. Choose single-tenant mode for the Controller, then click **Next**.

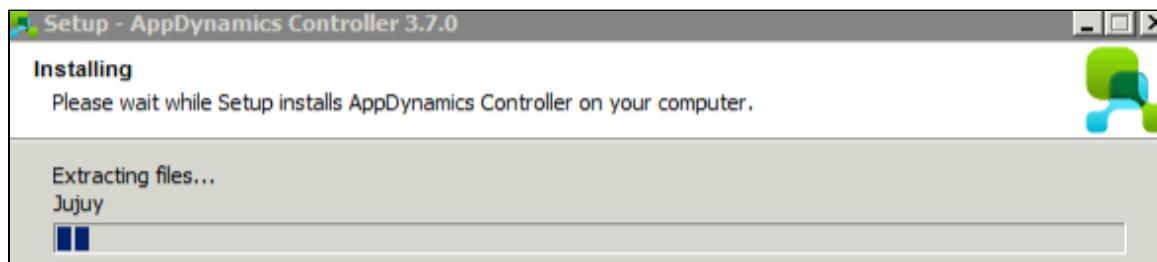
10. Enter admin user name and password then click **Next**. This user will be able to access the single-tenant controller and create other users. The admin user is the user described as the **AppDynamics Administrator**.

11. Select the Demo performance profile then click **Next**.

12. Select the default data directory path then click **Next**.

13. Select **Not Applicable** as you are not enabling HA for this Controller then click **Next**.

The installation begins.



Verify Controller Installation

Wait for the installation status screen to show that the installation is complete. Then the Controller will automatically start.

To verify the success of your installation, do one of the following:

- Click the link to the Controller provided in the installer's "Finish" screen.
- Open a browser to the URL that you configured for the Controller.

```
http://<application_server_host_name>:<http_listener_port>/controller
```

Post-installation Checklist for Windows

- Apply the License File
- Install the Controller as a Windows Service (optional)
- Restart the Controller If You Reboot

Apply the License File

The license file (license.lic) was provided in the downloaded ZIP file. Copy the license file to the Controller installation directory. Allow up to 5 minutes for the license change to take effect.

Install the Controller as a Windows Service (optional)

By default the Controller is not installed as a Windows service. In most cases if you log out the Controller will shut down.

See [Install the Controller as a Windows Service](#).

Restart the Controller If You Reboot

If you reboot the Controller machine you need to restart the Controller, unless it is installed as a Windows service that is configured to start automatically. See [Start or Stop the Controller](#).

Learn More

- [Install the Controller as a Windows Service](#)
- [Start or Stop the Controller](#)

Verify App Agent - Controller Communication

Follow these instructions to verify that the Java or .NET App Agent is reporting to the AppDynamics Controller.

1. Access the AppDynamics UI.

For on-premise Controller installations, open a browser at:

```
http://<Application_Server_Host_Name>:<HTTP_Listener_Port>/controller
```

For the SaaS Controller Service, open a browser at the URL provided to you by AppDynamics.

```
http(s)://<customer>.saas.appdynamics.com/controller
```

2. Provide user credentials.

For on-premise Controller installations, provide the credentials for the "admin" user as configured during AppDynamics Controller installation.

For the SaaS Controller Service, use the credentials provided to you by AppDynamics.

Upon login, the [All Applications Dashboard](#) lists the application.

3. In the left navigation panel, click **Servers -> App Servers -> <TierName>**.
AppDynamics displays the [Tier Dashboard](#) for the selected tier.

4. Click the Nodes tab.

When an App Agent is reporting to the Controller, the App Agent Status column shows a green up arrow icon.

Name	Health	App Agent Status	App Agent Version
Node_8002	!	100%	Server Agent v3.6.0.0 D

If the agent is not reporting, see the troubleshooting information.

- [Troubleshoot App Agent for Java](#)
- [Troubleshoot App Agent for .NET Installation and Configuration](#)

Install the Machine Agent

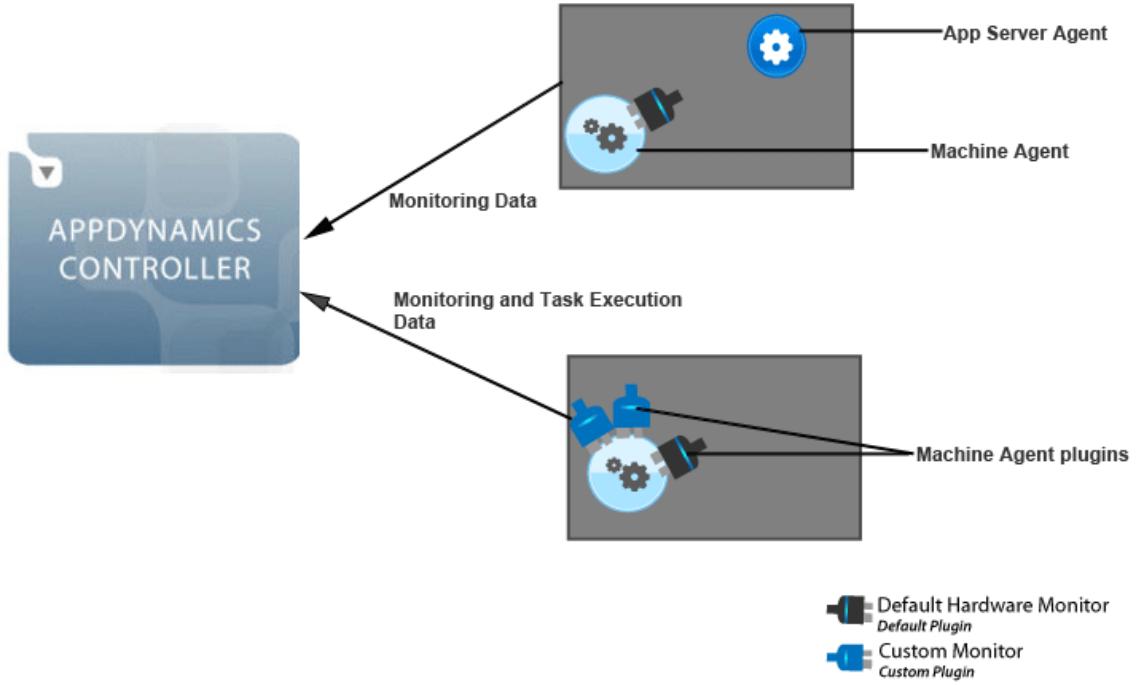
- [The Machine Agent](#)
- [Prerequisites for Installing the Machine Agent](#)
 - [To install the Machine Agent for use with the App Agent for Java](#)
- [Troubleshooting Machine Agent Installation](#)
 - [To verify that the Machine Agent is running](#)
- [Learn More](#)

The Machine Agent

The AppDynamics Machine Agent is a standalone Java program that collects performance statistics about your environment. It can be deployed on any machine that hosts application servers, database servers, messaging servers, Web servers, etc. It has an extensible architecture.

The AppDynamics Machine Agent supports [hardware monitoring](#) on most platforms. You can also [write custom plugins](#) for monitoring particular hardware metrics.

The following illustration shows the architecture for the Machine Agent. A Machine Agent collects hardware metrics using the default and custom plugins and sends task execution data to the Controller.



Prerequisites for Installing the Machine Agent

The Machine Agent requires Java 1.5 or later to be installed on the machine. You can download Java 1.6 from [the Oracle website](#).

A machine can have only one active Machine Agent installation at a time. Make sure you do not have any previous running installation processes before you install.

Read [Logical Model](#) to learn about these important concepts.

To install the Machine Agent for use with the App Agent for Java

1. Download the Machine Agent.

Download the Machine Agent from [AppDynamics Download Center](#).

You can also download from the Linux command line.

The Machine Agent is packaged as a zip file.

2. Extract the zip file to the destination directory.

⚠️ Do not use spaces in the destination directory path.

For Windows environments, unblock the zip file before you extract it. To unblock the zip file, right-click on the zip file and select the "Properties" tab. On the properties tab, choose "unblock" to unblock the file.

3. Configure how the agent connects to the Controller.

Configure properties for the Controller host name and port number using either the <Machine_Agent_Installation_Directory>/conf/controller-info.xml file or by adding system properties to the JVM startup script file.

Configure using controller-info.xml	Configure using System Properties	Required	Default
<controller-host>	-Dappdynamics.controller.hostName	Yes	None

<controller-port>	-Dappdynamics.controller.port	Yes	<p>For On-premise Controller installations: By default, port 8090 for HTTP and 8181 for HTTPS communication.</p> <p>For SaaS Controller service: By default, port 80 for HTTP and port 443 for HTTPS communication.</p>
-------------------	-------------------------------	-----	---

If you start a Machine Agent on a machine that already has an App Agent for Java installed, then the Machine Agent will automatically associate itself with the app agent's application, tier, and node settings.

To configure agent to use SSL see [Enable SSL for Communicating with the Controller](#).

To configure agent to use proxy settings see [Proxy Settings for the Controller](#).

4. (For Multi-tenant mode or SaaS installations only.) Configure the agent account information.

This step is required only when the AppDynamics Controller is configured in [Controller Tenant Mode](#) or when you [Use a SaaS Controller](#). Skip this step if you are using single-tenant mode.

Specify the properties for Account Name and Account Key. This information is in the welcome email sent by AppDynamics Support Team.

Configure using controller-info.xml	Configure using System Properties	Required	Default
<account-name>	-Dappdynamics.agent.accountName	Required only if your Controller is configured for multi-tenant mode or if your Controller is hosted.	None.
<account-access-key>	-Dappdynamics.agent.accountAccessKey	Required only if your Controller is configured for multi-tenant mode or if your Controller is hosted.	None.

5. Configure the business application, tier, and node.

If there is an App Agent already installed on the machine, AppDynamics automatically makes the association.

If there is no App Agent installed on the same machine, specify the application, tier, and node names.

Edit the Machine Agent <Machine_Agent_Installation_Directory>/conf/controller-info.xml file and specify the following elements:

Configure using controller-info.xml	Configure using System Properties
<application-name>	-Dappdynamics.agent.applicationName
<tier-name>	-Dappdynamics.agent.tierName
<node-name>	-Dappdynamics.agent.nodeName

If you do not provide configuration details the Machine Agent will not be associated with a business application. You can manually associate the agent in the UI at a later time. See [Administer Machine Agents](#).

6. (Optional) Configure the Machine Agent to run automatically when the machine starts.

See [Machine Agent Install and Admin FAQ](#).

7. Start the Machine Agent

In a command line console, execute the following command to start the Machine Agent:

```
java -jar machineagent.jar
```

Alternatively, in a Linux environment, you can execute the following command in the background:

```
nohup java -jar machineagent.jar &
```

8. Verify the Machine Agent installation.

Open the <Machine_Agent_Installation_Directory>/logs/machine-agent.log file. After successful installation, this file should contain the following message:

```
Started AppDynamics Machine Agent Successfully
```

This message is also printed on the STDOUT of the process.

9. Verify that the Machine Agent is reporting to the Controller

In the left navigation panel, click *Servers -> App Servers -> <Tier> -> <Node>. AppDynamics displays the Tier Dashboard for the selected tier.

When a Machine Agent is reporting to the Controller, the Machine Agent Status column shows a green up arrow icon.

You should see an "up" arrow symbol listed for the Machine Agent.

Troubleshooting Machine Agent Installation

To verify that the Machine Agent is running

Use the following command to verify that the Machine Agent script is running:

Linux:

```
ps -ef | grep machine
```

Windows:

1. Open a command line console.
2. Execute taskmgr and go to the process tab.
3. The agent process should be running. If the script is not running, terminate the Machine Agent Java process and restart it.

Learn More

- Monitor Hardware
- Machine Agent Install and Admin FAQ
- Machine Agent Configuration Properties
- Metric Browser
- Add Metrics Using Custom Monitors

Configure the Machine Agent to Automatically Start on Linux

- To create a start script for the Machine Agent
- [Learn More](#)
- To create a start script for the Machine Agent
- [Learn More](#)

You can configure the Machine Agent to run automatically when the machine starts.

To create a start script for the Machine Agent

1. Create a script file as shown below:

```

#!/bin/sh
# -----
# Typical AppDynamics Machine Agent Startup
# -----

#CHANGE ME: Set to the agent's install directory
JAVA=java

#CHANGE ME: Set to the agent's install directory
AGENT_HOME=/opt/appdynamics/machineagent
AGENT="$AGENT_HOME/machineagent.jar"

# Agent Options
# Uncomment and make available as needed
# -Dappdynamics.agent.applicationName : application that the agent participates in
# -Dappdynamics.agent.logging.dir : directory to put logs (agent "user" must have write
permissions
# -Dmetric.http.listener=true | false : open a kill port
# -Dmetric.http.listener.port : the port to send kill messages

AGENT_OPTIONS=
#AGENT_OPTIONS="$AGENT_OPTIONS -Dappdynamics.agent.applicationName=<application-name>"
#AGENT_OPTIONS="$AGENT_OPTIONS -Dappdynamics.agent.logging.dir="
#AGENT_OPTIONS="$AGENT_OPTIONS -Dmetric.http.listener=true | false"
#AGENT_OPTIONS="$AGENT_OPTIONS -Dmetric.http.listener.port=<port>"

#Consolidation Step - allows you to easily add or remove the agent
nohup $JAVA $AGENT_OPTIONS -jar $AGENT &

```

2. Add the script file to the rc utility

For Red Hat and most Linux Operating Systems	For Ubuntu and Debian Operating Systems
<p>Add the script file created in Step-1 in /etc/rc.local For example: echo '/path/to/startup/script.sh start' >> /etc/rc.local</p>	<p>Add the script file created in Step-1 as shown below: For example: update-rc.d -f script.sh start 99 2 3 4 5. where</p> <ul style="list-style-type: none"> • start is the argument given to the script (start, stop). • 99 is the start order of the script (1 = first one, 99= last one) • 2 3 4 5 are the runlevels to start <p> IMPORTANT: Do not forget to add dot(.) at the end.</p>

Learn More

- [Install the Machine Agent](#)
- [Machine Agent Configuration Properties](#)

Administer Agents

Metrics Limits

There are limits on the total number of metrics per app agent and per machine agent.

For the total number of metrics per app agent, the default limit is 5000 metrics.

For the total number of metrics per machine agent, the default limit is 200 metrics.

If the agent exceeds the limit, a single MetricOverflowException is generated and no new metrics are created until the agent restarts. The message is:

```
com.singularity.ee.agent.commonservices.metricgeneration.metrics.MetricOverflowException:  
Maximum metrics limit reached <maxMetrics> no new metrics can be created. This exception will  
not repeat until restart.
```

On Java platforms you can modify the limit using the agent.maxMetrics system property. For example:

```
\-Dappdynamics.agent.maxMetrics=10000
```

On .NET platforms set the maxMetrics property as an environment variable. For example:

```
appdynamics.agent.maxMetrics=10000
```

App Agent Node Properties

- [To edit a registered node property](#)
- [To register a node property](#)
- [To apply the configuration to all nodes](#)
- [Learn More](#)

Node properties affect AppDynamics behavior for monitoring a node, for example enabling or disabling features or setting maximum values for the number of discovered business transactions or the minimum number of requests to evaluate before triggering a diagnostic session. Node properties are inherited from the parent application or tier or customized for a specific node.

You can edit the values of the default node properties either in the app-agent-config.xml file or from the UI. To see the default node properties, follow steps 1 through 3 in [To edit a registered node property](#).

In addition to the default node properties, AppDynamics Support may recommend that you set a custom node property to solve a particular problem. The custom node properties are not available unless you register them.

To edit a registered node property

1. In the left navigation pane, access the node dashboard for the node for which you want to configure a property. For example, click **Servers -> App Servers -> <tier> -> <node>**. The Node Dashboard opens.
2. In the Node Dashboard, click **Actions -> Configure App Server Agent**. The agent configuration properties window opens.
3. In the left panel, select the application, tier, or node for which you want to edit a node property.
If you are editing at the node application level, click the **Java Agent Configuration** or **.NET Agent Configuration** tab. The list of node properties appears.
4. From the list of node properties, select the property that you want to edit. You can use the search box on the right to find the property by name.
5. Click the edit icon.
6. In the Edit Agent Property window, update the value of the property.
7. Click **Save**.



You can copy the configuration to other nodes, to the tier, or apply it to the entire application.

To register a node property

1. In the left navigation pane, click **Servers -> App Servers -> <tier> -> <node>**. The Node Dashboard opens.
2. In the Node Dashboard, click **Actions -> Configure App Server Agent**. The agent configuration properties window opens.
3. In the left panel, select the application, tier, or node for which you want to register a node property. If you are editing at the node application level, click the **Java Agent Configuration** or **.NET Agent Configuration** tab. The list of node properties appears.
4. Click the + plus icon.
5. In the Create Agent Property window, enter the values for:
 - the property name
 - a description of the property
 - the type from the drop-down menu
 - the value of the property
6. Click **Save**.

You can copy the configuration to other nodes, to the tier, or apply it to the entire application.

To apply the configuration to all nodes

If you just **Save** but do not click **Apply to All Nodes**, the new configuration applies to new nodes going forward but does not overwrite custom configuration for existing nodes.

If you are configuring at the application level, click **Apply to All Nodes** if you want the new configuration to overwrite all nodes in the entire application, including nodes with custom configuration. This erases the previous custom configurations for the application.

If you are configuring at the tier level, click **Apply to All Nodes in this tier** if you want the new configuration to overwrite all nodes in the tier including nodes with custom configuration. This erases the previous custom configurations for the tier.

Learn More

- [App Agent Node Properties Reference](#)
- [Hierarchical Configuration Model](#)

App Agent Node Properties Reference

Reference

This is a reference page containing information about app agent node properties. The properties are listed in alphabetical order.

- Reference
 - adaptive-callgraph-granularity
 - api-thread-activity-timeout-in-seconds
 - api-transaction-timeout-in-seconds
 - async-transaction-demarcator
 - bci-log-config
 - callgraph-granularity-in-ms
 - capture-404-urls
 - capture-error-urls
 - capture-raw-sql
 - collect-user-data-sync
 - collection-capture-period-in-minutes
 - disable-custom-exit-points-for
 - disable-exit-call-correlation-for
 - disable-exit-call-metrics-for
 - dont-show-packages
 - downstream-tx-detection-enabled
 - enable-collection-monitoring
 - enable-default-http-error-code-reporter
 - enable-info-point-data-in-snapshots
 - enable-instance-monitoring
 - enable-json-bci-rules
 - enable-object-size-monitoring
 - enable-startup-snapshot-policy
 - enable-transaction-correlation
 - enable-xml-bci-rules
 - end-to-end-message-latency-threshold-millis
 - find-entry-points
 - jdbc-callable-statements
 - jdbc-connections
 - jdbc-prepared-statements
 - jdbc-statements
 - leak-diagnostic-interval-in-minutes
 - log-request-payload
 - max-business-transactions
 - max-call-elements-per-snapshot
 - max-concurrent-snapshots
 - max-jdbc-calls-per-callgraph
 - max-jdbc-calls-per-snapshot
 - min-duration-for-jdbc-call-in-ms
 - min-load-per-minute-diagnostic-session-trigger
 - minimum-age-for-evaluation-in-minutes
 - minimum-number-of-elements-in-collection-to-deep-size
 - minimum-size-for-evaluation-in-mb
 - on-demand-snapshots
 - show-packages
 - slow-request-deviation
 - slow-request-monitor-interval
 - slow-request-threshold
 - thread-correlation-classes
 - thread-correlation-classes-exclude

adaptive-callgraph-granularity

Description

This property enables adaptive sampling. The call graph granularity for adaptive sampling is based on the average response time for the business transaction during the last one minute. The following distribution is used:

- Granularity of 10 ms for average response time of <= 10 seconds
- 50 ms for 10 to 60 seconds
- 100 ms for 60 to 600 seconds
- 200 ms for > 600 seconds

Type

Boolean

Default value

false

Platform

- Java
- .NET

api-thread-activity-timeout-in-seconds**Description**

This property provides a time-out value that comes into play when you have added global transactions to your application using APIs from the AppDynamics SDK. In the event that the added transaction spawns additional threads that do not return or complete, this property provides a safety valve time-out value. The value is in seconds. The removeCurrentThread method is invoked after the specified time out period.

Type

Integer

Default value

300 seconds

Range

Minimum =1; Maximum=3600

Platform

- Java

api-transaction-timeout-in-seconds**Description**

This property provides a time-out value that comes into play when you have added global transactions to your application using APIs from the AppDynamics SDK. In the event that the added transaction does not return or complete, this property provides a safety valve time-out value. The time-out value is in seconds. The endTransaction method is invoked after the specified time-out period.

Type

Integer

Default value

300 seconds

Range

Minimum=1; Maximum=3600

Platform

- Java

async-transaction-demarcator**Description**

This class name and method name combination marks the end of an asynchronous distributed transaction.
Use the format ClassName/MethodName. For example, foo/bar where foo is the class name and bar is the method name.

Type

String

Default value

none

Platform

- Java

bci-log-config

Description

Use this property to configure the bytecode transformer log (BCT log). This log shows what AppD instruments and what classes are loaded in the JVM. The initial file (the *0.log*) is saved to preserve the context of the server start up and is not rolled over. The subsequent files rotate. The format of the file name is ByteCodeTransformer.<timestamp>.<N>.log. The time stamp is represented as YYYY_MM_DD_HH_mm_ss. N increments starting from zero.

For example:

```
ByteCodeTransformer.2012_09_12_20_17_57.0.log
```

Because the file size is checked every 15 seconds, the files may be a bit larger than the specified threshold value before they are rolled over.

The first log file generated is named as follows:ByteCodeTransformer.<timestamp>.0.log
The value for this property is of the format: "20,5,4".

Type

String

Default Value

20,5,4

Example

For value = 20,5,4

20 = size, in MB, of the first log file, the .0 version

5 = size in MB for each subsequent rolling file

4 = the number of ByteCodeTransformer log files to keep

Platform

Java

callgraph-granularity-in-ms

Description

Specifies the granularity for call graphs for this node. The global configuration is ignored if this property is used. This value is ignored if the adaptive-callgraph-granularity property is set to true. A value of zero (default) means the global configuration (from Configuration --> Callgraph Settings) is used. Does not need a restart.

Type

Integer

Default value

zero

Range

Minimum=0;Maximum=5000

Platform

- Java
- .NET

capture-404-urls

Description

This property disables or enables the capture of the URLs causing 404 errors. The URLs are reported as ERROR events every 15 minutes and are viewable through the Event Viewer. The JVM needs a restart if retransformation is not supported for the JVM version.

404 errors usually mean that no application code is executed, so there is nothing to snapshot.
You can get insight into the 404 error by setting this property to true. It reports all the URLs which caused 404 error.

The capture-404-urls node property is **deprecated** in AppDynamics v. 3.6 and replaced with capture-error-urls.

Type

Boolean

Default value

false

Platform

- Java

capture-error-urls

Description

This property enables or disables the capture of the following HTTP errors:

- 401 - Unauthorized
- 500 - Internal Server Error
- 404 - Page Not Found
- All other error codes are put in a generic HTTP error code bucket.

For these 4 categories, the agent collects URLs, limited to 25 per category per minute, and sends an event out every 5 minutes.

You can see these URLs when you drill down on an error code by clicking Troubleshoot -> Errors -> Exceptions tab -> HTTP Error Codes.

Type

Boolean

Default value

true

Platform

- Java

capture-raw-sql

Description

Capture raw SQL calls for this node. For example:

```
select * from user where ssn = '123-123-1234'
```

If you change this node property in an environment that is using the IBM JVM, you need to restart the JVM. This is because the feature requires retransformation of certain JDBC classes, which is not possible with our IBM agent.

Type

Boolean

Default value

false

Platform

- Java
- .NET

collect-user-data-sync

Description

Collect user data from diagnostic POJO data collectors synchronously. Does not require a restart.

Type

Boolean

Default value

true

Platform

- Java

collection-capture-period-in-minutes

Description

Total interval in minutes since server restart for which collections are captured for leak evaluation. The property takes effect only after the node restart.

Type

Integer

Default value

30

Range

Minimum=5; Maximum=N/A

Platform

- Java

disable-custom-exit-points-for

Description

There is a limited set of "custom" exit points configured by default. You can disable these preconfigured custom exit points by specifying the type here using this property. Allowed values are: SAP, Mail, LDAP, MongoDB. For multiple values, use a comma separated list.

Type

String

Default value

none

Platform

- Java

disable-exit-call-correlation-for

Description

Disable exit call correlation for a specific type of call. For example, HTTP, JMS, RMI. By default all exit call correlations are enabled.

For a list of the backends that are automatically discovered see [Backend Monitoring](#).

Type

String

Default value

none

Platform

- Java
- .NET

disable-exit-call-metrics-for

Description

Disables exit call monitoring for a specific type of exit call; for example, HTTP, JMS, WEB_SERVICE. If this property is set, the average data (calls/min, avg response time) for the specific exit call type is not collected. However, for a snapshot all details are collected. Set this property if the application makes a large number of exit calls per transaction and the avg metrics are not important.

For lists of backends that are automatically discovered see [Backend Monitoring](#).

Type

String

Default value

By default all exit call metrics are enabled.

Platform

- Java
- .NET

dont-show-packages

Description

Do not show these packages / class names in addition to the ones configured in the global call graph configuration, for the call graphs captured on this node. Does not need a restart.

Type

String

Default value

none

Platform

- Java
- .NET

downstream-tx-detection-enabled

Description

If the agent cannot reach the controller for a prolonged period, it turns off most services and notifies the continuing tiers that upstream transaction was detected and is not being monitored. Set this property to true to enable the continuing tiers to detect their own transactions in the event of network failure on the upstream tiers.

Type

Boolean

Default value

false

Platform

- Java
- .NET

enable-collection-monitoring

Description

This property enables or disables Collection monitoring on this node. This property is related to Automatic Leak Detection, a feature that is available with JVM versions 1.6 upward. Changing this property requires a restart.

Type

Boolean

Default value

false

Platform

- Java

enable-default-http-error-code-reporter

Description

This property disables or enables automatic HTTP error code reporting for error codes between 400 - 505.

Type

Boolean

Default value

true

Platform

- Java
- .NET

enable-info-point-data-in-snapshots

Description

This property disables or enables the capture of information point calls in snapshots. When this property is set to true, information point calls appear in the User Data section of the call graph.

Type

Boolean

Default value

false

Platform

- Java
- .NET

enable-instance-monitoring**Description**

This property enables or disables Instance tracking on this node. Does not need a JVM restart.

Type

Boolean

Default value

false

Platform

- Java

enable-json-bci-rules**Description**

Set this property to true to enable JSON bytecode instrumentation rules. AppDynamics instruments the `get` and `getString` methods within the package/class `org.json.JSONObject` when you set this value to true. Needs a JVM restart.

Type

Boolean

Default value

false

Platform

- Java

enable-object-size-monitoring**Description**

This property is related to Automatic Leak Detection (ALD) and enables or disables Object Size monitoring on this node. Changing this property does not need a JVM restart. ALD is supported for JVM version 1.6 and up.

Type

Boolean

Default value

false

Platform

- Java

enable-startup-snapshot-policy**Description**

This property disables or enables the policy for start-up transaction snapshot. This means snapshots are collected for all BTs for all invocations for the first 15 minutes of an application server start.

Type

Boolean

Default value

false

Platform

- Java
- .NET

enable-transaction-correlation**Description**

This property disables or enables transaction correlation. It does not require a restart.

Type

Boolean

Default value

true

Platform

- Java
- .NET

enable-xml-bci-rules**Description**

This property enables Java XML Binding and DOM Parser bytecode instrumentation rules. Set to true to enable. The change takes effect after a JVM restart.

Type

Boolean

Default value

false

Platform

- Java

end-to-end-message-latency-threshold-millis**Description**

Enables end-to-end message latency monitoring for distributed asynchronous systems by setting up a threshold. Any message taking more time than the threshold is viewable through the Event Viewer.

Type

Integer

Default value

0

Range

Minimum=0; Maximum=36000

Platform

- Java

find-entry-points**Description**

Set this property to true to log all potential entry points that are hitting instrumented exit points or loggers to the Business Transactions log file.

Use this property when you suspect that some traffic is not being detected as business transactions.

Note that this property should be enabled for debugging only. It is strongly recommended to disable this property in production setup by changing the value from true to false.

Type

Boolean

Default value

false

Platform

- Java
- .NET

jdbc-callable-statements**Description**

Use this property to indicate the implementation classes of the *java.sql.CallableStatement* interface that should be instrumented. For example, to instrument calls to Times Ten (an unsupported database), you could set this property to the following:

```
com.timesten.jdbc.JdbcOdbcCallableStatement
```

Type

String

Default value

none

Platform

- Java

jdbc-connections**Description**

Use this property to indicate the implementation classes of the *java.sql.Connection* interface that should be instrumented. For example, to instrument calls to Times Ten (an unsupported database), you could set this property to the following:

```
com.timesten.jdbc.JdbcOdbcConnection
```

Type

String

Default value

none

Platform

- Java

jdbc-prepared-statements**Description**

Use this property to indicate the implementation classes of the `java.sql.PreparedStatement` interface that should be instrumented. For example, to instrument calls to Times Ten (an unsupported database), you could set this property to the following:

```
com.timesten.jdbc.JdbcOdbcPreparedStatement
```

Type

String

Default value

none

Platform

- Java

jdbc-statements**Description**

Use this property to indicate the implementation classes of the `java.sql.Statement` interface that should be instrumented. For example, to instrument calls to Times Ten (an unsupported database), you could set this property to the following:

```
com.timesten.jdbc.JdbcOdbcStatement
```

Type

String

Default value

none

Platform

- Java

leak-diagnostic-interval-in-minutes**Description**

Interval at which diagnostic data, content summary and activity trace, is captured for leaking collections.

Type

Integer

Default value

30

Range

Minimum=2; Maximum=N/A

Platform

- Java

log-request-payload**Description**

Set this property to true to log the request payload(HTTP parameters/cookies/session keys etc) as part of a transaction snapshot.

Type

Boolean

Default value

false

Platform

- Java
- .NET

max-business-transactions**Description**

Sets a limit on the number of business transactions discovered once an agent is started. This is done to prevent business transaction metric explosion because an unsuitable discovery scheme can potentially produce thousands of transactions.



Warning: Contact AppDynamics Support before changing this setting.

See also [The Business Transaction Limit](#).

Type

Integer

Default value

50

Range

Minimum=N/A; Maximum=200

Platform

- Java
- .NET

max-call-elements-per-snapshot**Description**

This property represents the maximum number of elements that are collected for any call graph for a snapshot. When the limit is reached, the agent stops collecting more data for this call graph, reports what has been collected to that point, and marks the call graph with a warning that the limit was reached. It is not recommended to dramatically increase this number as it may have overhead implications.

Type

Integer

Default value

5000

Platform

- Java
- .NET

max-concurrent-snapshots**Description**

The maximum number of total snapshots that are allowed, including continuing transactions. When the queue goes over the value set, additional snapshots are dropped.

Type

Integer

Default value

20 (v.3.5.x)

Range

Values must be positive integers. No other constraints.

Platform

- Java
- .NET

max-jdbc-calls-per-callgraph**Description**

Maximum number of JDBC/ADO.NET exit-call stack samples per call graph. Only queries taking more time than the value of min-duration-for-jdbc-call-in-ms are reported. Changing the value does not require a restart.

Type

Integer

Default value

100

Range

Minimum=1; Maximum=1000

Platform

- Java
- .NET

max-jdbc-calls-per-snapshot**Description**

Maximum number of JDBC/ADO.NET exit calls allowed in a snapshot. Calls after the limit are not recorded. Changing the value does not require a restart.

Type

Integer

Default value

500

Range

Minimum=1; Maximum=5000

Platform

- Java
- .NET

min-duration-for-jdbc-call-in-ms

Description

A JDBC/ADO.NET call taking more time than the specified time (in milliseconds) is captured in the call graph. The query continues to show up in a transaction snapshot. Setting this value too low (< 10ms) may affect application response times. Changing the value does not require a restart

Type

Integer

Default value

10

Range

Minimum=0; Maximum=N/A

Platform

- Java
- .NET

min-load-per-minute-diagnostic-session-trigger

Description

Indicates the number of requests per Business Transaction to evaluate before triggering a diagnostic session. This is useful to prevent diagnostic sessions when there is not enough load.

Type

Integer

Default value

10

Range

Minimum=N/A; Maximum=N/A

Platform

- Java
- .NET

minimum-age-for-evaluation-in-minutes

Description

Automatic Leak Detection (ALD) tracks all frequently used Collections. For a Collection object to be identified and monitored it must meet the conditions defined by the ALD properties. This property is the first criteria that needs to be met. The value is the minimum age of the Collection in minutes. The property takes effect after node restart.

From the point the collection is captured, it is monitored if it is still available for the specified period without getting garbage collected. If it survives then it is evaluated for size checks and if it meets the criteria then it is monitored for long term growth in

size.

 Warning: If you reduce the default there may be a performance hit on the CPU and memory because AD needs to process more collections.

Type

Integer

Default value

30

Range

Minimum=5; Maximum=N/A

Platform

- Java

minimum-number-of-elements-in-collection-to-deep-size

Description

Automatic Leak Detection (ALD) tracks all frequently used Collections. For a Collection object to be identified and monitored for it must meet the conditions defined by the ALD properties. This property sets the number of elements threshold.

 Warning: If you reduce the default there may be a performance hit on the CPU and memory because AD is processing more collections.

Type

Integer

Default value

1000

Platform

- Java

minimum-size-for-evaluation-in-mb

Description

Automatic Leak Detection (ALD) tracks all frequently used Collections. For a Collection object to be identified and monitored it must meet the conditions defined by the ALD properties. This property sets the minimum initial size in megabytes for a collection to qualify for monitoring. The collection must also survive for the period specified in the minimum-age-for-evaluation-in-minutes property.

 Warning: If you reduce the default there may be a performance hit on the CPU and memory because AD is processing more collections.

Type

Integer

Default value

5

Range

Minimum=1; Maximum=N/A

Platform

- Java

on-demand-snapshots

Description

Collect snapshots for all business transactions executed in this node. Does not need a restart.

Type

Boolean

Default value

false

Platform

- Java
- .NET

show-packages

Description

For the call graphs captured on this node, show the specified packages or class names in addition to the ones configured in the global call graph configuration. Does not need a restart.

Type

String

Default value

none

Platform

- Java
- .NET

slow-request-deviation

Description

The value in milliseconds for the deviation from the current average response time. This setting is used for evaluation of slow in-flight transactions. Also see the slow-request-threshold property for more details.

Type

Integer

Default value

200

Range

Minimum=10; Maximum=3600

Platform

- Java
- .NET

slow-request-monitor-interval

Description

In-flight requests are checked for slowness in the interval specified by this property. The value is specified in milliseconds.

Type

Integer

Default value

100

Range

Minimum=0; Maximum=3600

Platform

- Java
- .NET

slow-request-threshold

Description

In-flight requests taking more time than this threshold (in ms) with a deviation greater than the slow-request-deviation property from the current average response time are monitored to capture hot spots.

Type

Integer

Default value

500

Range

Minimum=0; Maximum=3600

Platform

- Java
- .NET

thread-correlation-classes

Description

For multi-threaded applications, use this property to configure classes to be included in Java thread correlation when simple prefix-matching (matching on STARTSWITH) is sufficient to identify the classes.

The thread-correlation-classes property specifies the classes to include for thread correlation. This property can be used together with the thread-correlation-classes-exclude.

The configured correlation takes effect without requiring a restart of the managed application.
Also see: [Configure Multi-Threaded Transactions \(Java only\)](#).

Type

Comma-separated string of fully-qualified class names or package names

Default value

none

Platform

- Java

thread-correlation-classes-exclude

Description

For multi-threaded applications, use this property to configure classes to be excluded in Java thread correlation when simple prefix-matching (matching on STARTSWITH) is sufficient to identify the classes. This property specifies the classes to exclude from thread correlation. This property can be used in conjunction with the thread-correlation-classes node property.

The configured correlation takes effect without requiring a restart of the managed application.
Also see: [Configure Multi-Threaded Transactions \(Java only\)](#).

Type

Comma-separated string of fully-qualified class names or package names

Default value

none

Platform

- Java

App Agent Node Properties Reference By Type

- [Learn More](#)

This topic lists the app agent node properties by type.

Category	Property
Automatic Leak Detection	minimum-age-for-evaluation-in-minutes
	minimum-number-of-elements-in-collection-to-deep-size
	minimum-size-for-evaluation-in-mb
Bytecode injection (BCI)	bci-log-config
	enable-json-bci-rules
	enable-xml-bci-rules
Object Instance Tracking (OIT)	collection-capture-period-in-minutes
	enable-collection-monitoring
	enable-instance-monitoring
	enable-object-size-monitoring
	leak-diagnostic-interval-in-minutes
Transaction Monitoring	api-thread-activity-timeout-in-seconds
	api-transaction-timeout-in-seconds
	async-transaction-demarcator
	capture-404-urls
	capture-error-urls
	capture-raw-sql
	disable-custom-exit-points-for
	disable-exit-call-correlation-for
	disable-exit-call-metrics-for
	downstream-tx-detection-enabled
	enable-default-http-error-code-reporter

	enable-transaction-correlation
	end-to-end-message-latency-threshold-millis
	find-entry-points
	jdbc-callable-statements
	jdbc-connections
	jdbc-prepared-statements
	jdbc-statements
	log-request-payload
	max-business-transactions
	min-load-per-minute-diagnostic-session-trigger
	slow-request-deviation
	slow-request-monitor-interval
	thread-correlation-classes
	thread-correlation-classes-exclude
Transaction Snapshots	adaptive-callgraph-granularity
	callgraph-granularity-in-ms
	dont-show-packages
	enable-startup-snapshot-policy
	max-call-elements-per-snapshot
	max-concurrent-snapshots
	max-jdbc-calls-per-callgraph
	max-jdbc-calls-per-snapshot
	min-duration-for-jdbc-call-in-ms
	on-demand-snapshots
	show-packages
	slow-request-threshold
Miscellaneous	collect-user-data-sync

Learn More

- [App Agent Node Properties Reference](#)
- [Hierarchical Configuration Model](#)

Manage App Agents

- [Managing App Agents](#)
 - To access the App Server Agents window
- [Resetting App Agents](#)
 - To reset an app agent
- [Enabling and Disabling App Agents](#)
 - [Disabled Agents Use Licenses](#)
 - To enable (activate) or disable (deactivate) an app agent without restarting the app server
- [Deleting App Agents](#)
 - To delete an app agent
- [Other Actions](#)

- To view the Node Dashboard of an agent
- To view the machine agent on the same node as an app agent
- To find the app agent version number
- [Learn More](#)

This topic discusses managing app agents from the Controller UI.

Managing App Agents

To access the App Server Agents window

- Click **Setup -> AppDynamics Agents** in the upper right corner of the console.

AppDynamics lists all agents for all business applications.

Resetting App Agents

The reset operation purges all old data including metrics, discovered transactions, etc. for an app agent. This action does not require that you restart the JVM or CLR.

After you have reconfigured AppDynamics Business Transaction and/or backend detection, you may need to reset the agents. If new detection rules are not detecting newer Business Transaction or backends, you can delete all of the existing ones and have the agent start detecting from scratch using the **Reset** option. See [Business Transactions List Operations](#) for information about the Delete operation.

Resetting an app server agent causes AppDynamics to:

- Purge all the data such as metrics, discovered Business Transactions, etc. gathered by that agent.
- Reset the Business Transaction limit counter to zero for that agent.

Resetting an agent flushes the in-memory Business Transactions, exit calls, and registration information. It does not re-instrument or remove existing instrumentation.

After the reset, the Business Transactions and backends are re-registered and new metrics are created.

Approximately a few minutes worth of data may be missed between the reset operation and the re-registration.

To reset an app agent

1. In the AppDynamics Agents window, select an app agent.

2. Click **Reset Selected App Agent**.

Alternatively:

1. In the left navigation panel, click **Servers -> App Servers -> <Tier> -> <Node>**.

2. In the Node Dashboard, click the Agents tab.

3. At the top of the Agents tab click **Reset**.

Enabling and Disabling App Agents

An app server agent polls the Controller every minute to check whether the agent is enabled or disabled.

If an app agent for Java is disabled, it stops capturing all data except for JVM JMX metrics such as heap memory, memory pools, garbage collection, thread count, etc.

Using this feature you can compare the difference in the agent's run-time overhead when it does and does not capture data.

Disabled Agents Use Licenses

The Controller counts as licenses all agents connected to the Controller. A disabled agent is connected to the Controller and therefore is counted as a license.

To enable (activate) or disable (deactivate) an app agent without restarting the app server

This action takes effect in less than a minute and you do not need to restart the app server.

1. In the Controller left navigation pane, click **Servers -> App Servers -> <Tier> -> <Node>**.
2. In the Node Dashboard, click the Agents tab.
3. To deactivate the agent, click Agent is **Off**.
To activate the agent, click Agent is **On**.

Deleting App Agents

This action will not change the instrumentation of the application server at the JVM or CLR level.

To completely remove an App Agent for Java, and free up its license, see [Uninstall the App Agent for Java](#).

To delete an app agent

1. In the AppDynamics Agents window, select an app agent.
2. Click **Delete Agent from System**.

Other Actions

To view the Node Dashboard of an agent

1. In the AppDynamics Agents window, select an app agent.
2. Click **View Node Dashboard**.

See [Node Dashboard](#).

To view the machine agent on the same node as an app agent

1. In the AppDynamics Agents window, select an app agent.
2. Click **View Machine**.

Alternatively:

1. In the navigation pane, click **Servers -> App Servers -> <Tier> -> <Node>**.
2. In the Node Dashboard, click the Agents tab.
3. Click the Machine Agent subtab.

To find the app agent version number

1. In the Controller left navigation pane, click **Servers -> App Servers -> <Tier> -> <Node>**.
2. In the Node Dashboard, click the Agents tab.
3. Click the App Server Agent subtab.

Learn More

- [Administer Machine Agents](#)

Administer Machine Agents

- Performance Data Collected by the Machine Agent
- Machine Agents and the Flow Map
- Machine Agents and the Server Health Indicator
- Common Tasks for the Machine Agent
 - To access the Machine Agents window
 - Resetting Machine Agents

- To reset (stop/restart) a machine agent
- Associate a Machine Agent with an Application
 - To associate a Machine Agent with a business application
 - To associate multiple Machine Agents with multiple business applications
- Learn More

This topic covers the standalone Machine Agent. It does not cover the embedded machine agent included with the App Agent for .NET.

The Machine Agent uses a built-in hardware monitor to report metrics and hardware utilization data. The hardware monitor is a script that writes data to STDOUT of a process. The Machine Agent parses and sends this data to the Controller every minute.

Performance Data Collected by the Machine Agent

The Machine Agent automatically collects the following performance data:

1. CPU
 - a. Busy, Free
2. Memory
 - a. Used, Free, Total
3. Disk
 - a. Network mounted disks (Free, Used, IO Writes/Reads in KB/s and packets)
 - b. Local disks (Free, Used, IO Writes/Reads in KB/s and packets)
4. Network interfaces
 - a. No Loopback
 - b. No disabled or unconnected
 - c. Incoming KB, KB/s, packets, packets/s
 - d. Outgoing KB, KB/s, packets, packets/s

Machine Agents and the Flow Map

The Machine Agent monitors a particular machine and not a particular application server or JVM. The Machine Agent can therefore refer to multiple Nodes running on the same machine. A Flow Map, on the other hand, displays the communication between different Nodes during application execution, or the Business Transaction flow from Tier to Tier. A Machine Agent cannot be a part of the flow and therefore is not shown in the Flow Map.

Machine Agents and the Server Health Indicator

Metrics monitored by the Machine Agent are included in the Infrastructure Health indicator in the dashboards.

The health indicator is driven by health rule violations in the given time period and health rule violations are configured on hardware metrics collected by the Machine Agent. Health rules for all possible metrics are not configured out-of-the-box, so you might want to configure additional health rule according to your requirements. For details see [Configure Health Rules](#).

Common Tasks for the Machine Agent

To access the Machine Agents window

- Click **Setup -> AppDynamics Agents** in the upper right corner of the console.

AppDynamics lists all agents for all business applications.

Resetting Machine Agents

The reset operation purges all existing metrics for an agent and starts gathering them again. It stops the agent and starts it again.

To reset (stop/restart) a machine agent

1. In the Machine Agents window, select a machine agent.
2. Click **Reset Selected Machine Agent**.

Associate a Machine Agent with an Application

If no configuration details are provided during installation, or if the node has been moved to another application, then the Machine Agent will appear in the **System -> Agents** tab as "not associated with any applications". To have the Machine Agent start sending metrics or executing workflow tasks, manually associate it with an application.

To associate a Machine Agent with a business application

1. In the AppDynamics Agents window, select a machine agent.
2. Click **Associate with an Application**.

To associate multiple Machine Agents with multiple business applications

If the machine is hosting servers that belong to multiple business applications, you may need multiple Machine Agents.

If there are nodes belonging to multiple business applications, you can run multiple Machine Agents each configured to report metrics for a different node, tier, and application.

Background information: You cannot "assign" a single Machine Agent to multiple business application per se. A Machine Agent on a specific machine is automatically associated with all nodes running on that machine. A node is associated with a single business application. Therefore an application is associated with the Machine Agents of its nodes. By default Machine Agents inherit the application/tier/node names of the App Agents installed on the same hardware.

Learn More

Add Metrics Using Custom Monitors

- Custom Monitoring Using Scripts
 - Supported Operations on the Metrics
 - Average
 - Aggregation
 - Roll Up
 - Adding a Custom Monitor
 - Step 1: Create a directory under the Machine Agent monitors directory
 - Step 2: Create the script file
 - Step 3: Copy the script file to the directory created in Step 1
 - Step 4: Create the monitor.xml file
 - Step 5: Copy the monitor.xml file to the directory created in Step 1
 - Step 6: Restart the Machine Agent
 - Step 7: Verify execution of the custom monitor script
 - Example: Create Custom Monitor for open files
 - Learn More

Custom Monitoring Using Scripts

You can write a script (also known as a custom monitor) to add custom metrics to the metric set that AppDynamics already collects and reports to the Controller. Your script reports the custom metrics every minute to the Machine Agent. The Machine Agent passes these metrics on to the Controller. For example, to avoid a potential JVM crash in a production environment, it is critical to manage the number of open files. You can configure a custom monitor to generate alerts when the monitoring threshold is violated. This topic describes the steps for adding custom metrics using a shell script and includes an example.

For more information, see [Integration Basics#Add Metrics with Custom Monitors](#).

Supported Operations on the Metrics

Average

AVERAGE is the default operation. For example:

```
Hardware Resources|Page File Size,value=271
Hardware Resources|Sockets,value=161
Hardware Resources|My Metric,value=13, aggregator=SUM, time-rollup=AVERAGE
```

Aggregation

- The **aggregator** qualifier specifies how the machine agent aggregates the values reported during a one-minute period.
- This value is an enumerated type. Valid values are:

Aggregator	Description
AVERAGE	Average of all reported values in that minute. The default operation.
SUM	Sum of all reported values in that minute (This operation behaves like a counter).
OBSERVATION	Last reported value in the minute. If no value is reported in that minute, the value from the last time it was reported is used.

Roll Up

- The **time-rollup** qualifier specifies how the controller rolls up the values when it converts from one-minute granularity tables to 10-minute granularity and 60-minute granularity tables over time.
- The value is an enumerated type. Valid values are:

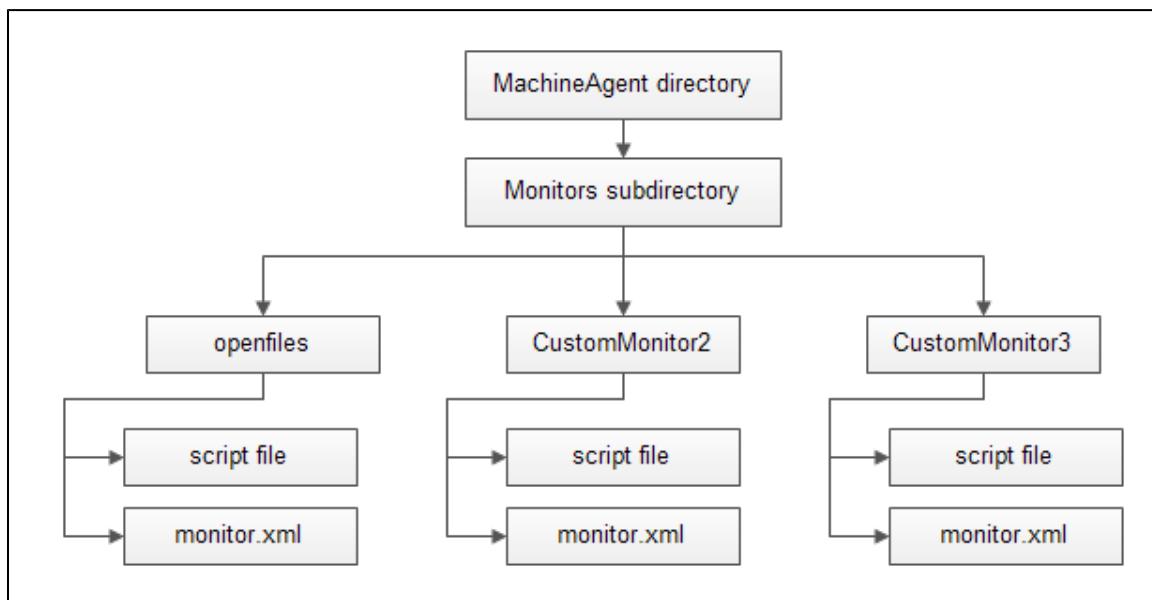
Roll up Strategy	Description
AVERAGE	Average of all one-minute data points when adding it to the 10-minute or 60-minute granularity table.
SUM	Sum of all one-minute data points when adding it to the 10-minute or 60-minute granularity table.
CURRENT	Last reported one-minute data point in that 10-minute or 60-minute interval.

Adding a Custom Monitor

Step 1: Create a directory under the Machine Agent monitors directory

The monitors directory in the Machine Agent installation directory is the repository for all default and custom monitors. For each new custom monitor, create a sub-directory under the monitors directory.

For example, to create a custom monitor for monitoring open files in the JVM, create a sub-directory named "openfiles" under the <Machine_Agent_installation/monitors> directory.



Step 2: Create the script file

A script writes data to STDOUT. The Machine Agent parses STDOUT and sends information to the Controller every minute. Use the

following instructions to create the script file.

1. Specify a name-value pair for the metrics.

Each metric has a name-value pair that is converted to a java 'long' value. A typical metric entry in the script file has the following structure:

```
name=<metric name>,value=<long value>
```

Use the following format:

	Format
Standard Form	Hardware Resources Instrument Name=Instrument Value
Fully Qualified Form	Hardware Resources <metric name>,value=<long value>

2. Define the category of the metric, for example:

- Infrastructure (for the default hardware metrics, see [Monitor Hardware](#))
- JVM (for the default metrics, see [Monitor JVMs](#))
- Custom Metrics

You can use the "|" separator to introduce further hierarchy in the metric name similar to the following:

```
Hardware Resources|Disks|Total Disk Usage %
Hardware Resources|Disks|Disk 1|Current Disk Usage %
Custom Metrics|MySQL|Avg Query Time
Custom Metrics|Apache|Avg Wait Time
```

3. To monitor multiple metrics in the same script file, print a different line for each one. For example:

```
name=Hardware Resources|Disks|Total Disk Usage %, value=23
name=Hardware Resources|Disks|Disk 1|Current Disk Usage %, value=56
name=Custom Metrics|MySQL|Avg Query Time, value=500
name=Custom Metrics|Apache|Avg Wait Time, value=$count
```

Step 3: Copy the script file to the directory created in Step 1

Ensure that the Agent process has execute permissions not only for the script file but also for the contents of the file.

Step 4: Create the monitor.xml file

Follow the steps listed below to create your monitor.xml file:

1. For each custom monitor create a monitor.xml file.

The monitor.xml file executes the script file created in Step 2. You can edit the following sample file to create your file.

```

<monitor>
    <name>HardwareMonitor</name>
    <type>managed</type>
    <description>Monitors system resources - CPU, Memory, Network I/O, and Disk
I/O.</description>
    <monitor-configuration>      </monitor-configuration>
    <monitor-run-task>
        <!-- Edit execution-style as needed. -->
        <execution-style>continuous</execution-style>
        <name>Run</name>
        <type>executable</type>
        <task-arguments></task-arguments>
        <executable-task>
            <type>file</type>
            <!-- Use only one file element per os-type. -->
            <file os-type="linux">linux-stat.sh</file>
            <file os-type="mac">macos-stat.sh</file>
            <file os-type="windows">windows-stat.bat</file>
            <file os-type="solaris">solaris-stat.sh</file>
            <file os-type="sunos">solaris-stat.sh</file>
            <file os-type="aix">aix-stat.sh</file>
        </executable-task>
    </monitor-run-task>
</monitor>

```

⚠ The os-type attribute is optional for the executable-task file element when only one os-type is specified. One monitor.xml file executes one script per os-type.

2. Select the execution style from one of the following:

Execution Style	Description	Example
Continuous	Choose "Continuous", if you want data collection averaged over time. (For example: Averaging CPU over minute). This ensures that the script keeps running until the machine agent process is terminated.(!) For the monitor to be declared as 'continuous', the script should also run in an infinite loop.	while [1]; do ... the actual script goes here ... sleep 60 done
Periodic	Choose periodic, if you want data to be reported from system performance counters periodically.	The periodic task runs every minute by default. Use following parameter in the XML file, if you want the periodic task to run with any other frequency: <monitor-run-task> element. <execution-frequency-in-seconds>120</execution-frequency-in-seconds> For all the other frequency settings, the data is aggregated.

3. Add the name of this script file to the <file> element in the monitor.xml file. Be sure to use the correct os-type attribute. The os-type value should match the value returned from calling System.getProperty("os.name"). See <http://docs.oracle.com/javase/1.5.0/docs/api/java/lang/System.html#getProperties%28%29>

```
<file os-type="your-os-type">{script file name}</file>
```

i You can use either the relative or absolute path of the script.

Step 5: Copy the monitor.xml file to the directory created in Step 1

Step 6: Restart the Machine Agent

After restarting the Machine Agent, you should see following message in your log file:

```
Executing script [<script_name>] on the console to make sure your changes work with the  
machine agent.
```

Step 7: Verify execution of the custom monitor script

To verify the execution of custom monitor, wait for at least one minute and check the metric data in the [Metric Browser](#).

You can now create alerts based on any of these metrics.

Example: Create Custom Monitor for open files

This section provides instructions to create a custom monitor for monitoring all the open files for JVMs.

1. Create a new directory in the custom monitor repository.
2. Create the script file.

 This is a sample script. Modify this script to plug-in the specific process name (for example: Author, Publish, and so on).

```
lookfor=<process name 1>  
pid=`ps aux | grep "$lookfor" | grep -v grep | tr -s " " | cut -f2 -d' '|`  
count1=`lsof -p $pid | wc -l | xargs`  
  
lookfor=<process name 2>  
pid=`ps aux | grep "$lookfor" | grep -v grep | tr -s " " | cut -f2 -d' '|`  
count2=`lsof -p $pid | wc -l | xargs`  
  
echo "name=JVM|Files|<process name 1>,value=\"$count1"  
echo "name=JVM|Files|<process name 2>,value=\"$count2"
```

3. Create the monitor.xml and point this XML file to the script file created in step 2.

```
<monitor>  
  <name>MyMonitors</name>  
  <type>managed</type>  
  <description>Monitor open file count </description>  
  <monitor-configuration>  
  </monitor-configuration>  
  <monitor-run-task>  
    <execution-style>continuous</execution-style>  
    <name>Run</name>  
    <type>executable</type>  
    <task-arguments>  
    </task-arguments>  
    <executable-task>  
      <type>file</type>  
      <file>openfilecount.sh</file>  
    </executable-task>  
  </monitor-run-task>  
</monitor>
```

Learn More

- [Integration Basics](#)
- [Monitor Hardware](#)
- [Administer Machine Agents](#)
- [Notification Actions](#)
- [Machine Agent Install and Admin FAQ](#)
- [Supported Environments and Versions](#)

Configure Custom Metrics for the z-OS Machine Agent

- Start the Resource Management Facility (RMF)
 - To initialize and start the RMF
 - To install the scripts
 - To confirm that the scripts are successful
- Learn More

This topic describes how to configure custom metrics in a z-OS environment.

Start the Resource Management Facility (RMF)

AppDynamics requires the RMF (Resource Management Facility) to collect the data for the required metrics.

To initialize and start the RMF

1. Connect to the EPTDFRH user and initialize the ETPGZCK user, if not already done.
2. Connect to TSO and provide the details of IBMUSER.
3. Upon a successful connection, choose the **SD (System Display and Search Facility)** option.
4. Use the following commands to initialize and start the RMF:

```
/S RMF
```

```
/F RMF,START III
```

```
/S GPMERVE, MEMBER=01
```

5. Access the following URL to confirm the RMF startup:

```
http://192.86.32.72:8803/
```

After successful RMF startup, the URL should display a valid HTML page.

6. Click **Explore** and then **Metrics** to display all the available metrics for SVSCPLEX, SYSPLEX.

To install the scripts

1. Download and unzip the attached **zos-machine-agent.zip** file to the <machine-agent-install-directory>/monitors/ directory.
2. Select the required metric locations, and add them to urls.list file in the zos-monitor directory. For example:

```
% CPU utilization          =
http://192.86.32.72:8803/gpm/perform.xml?resource=S0W1,* ,PROCESSOR&id=8D0460
% users                   =
http://192.86.32.72:8803/gpm/perform.xml?resource=%22,SVSCPLEX,SYSPLEX%22&id=8D0D50
% using for i/o by MVS image =
http://192.86.32.72:8803/gpm/perform.xml?resource=%22,SVSCPLEX,SYSPLEX%22&id=8D1DA0
% CSA utilization by MVS image =
http://192.86.32.72:8803/gpm/perform.xml?resource=%22,SVSCPLEX,SYSPLEX%22&id=8D2410
% users by MVS image      =
http://192.86.32.72:8803/gpm/perform.xml?resource=%22,SVSCPLEX,SYSPLEX%22&id=8D0D60
```

To confirm that the scripts are successful

Once the above steps are performed successfully, start the Machine Agent in debug mode. The following information in the Machine Agent log confirms that the processing is successful.

```
[Worker-7] 28 Mar 2012 19:59:02,128 INFO ExecTask - Started Executable Command  
[[G:\AppDynamics\64bit\3.3.4\  
MachineAgent-3.3.4.0RC\monitors\CustomMonitor\metrics.bat]]  
[Worker-7] 28 Mar 2012 19:59:02,128 DEBUG ExecTask - Will wait for process exit  
, before sending execution status.  
[Worker-1] 28 Mar 2012 19:59:04,651 DEBUG MonitorOutputHandler - Monitor line  
parsed:name=Custom Metrics|zos|% CPU utilization (CP), value=3  
[Worker-1] 28 Mar 2012 19:59:04,653 DEBUG MonitorOutputHandler - Reporting  
metric after reading metric [Custom Metrics|zos|% CPU utilization (CP)]  
[Worker-1] 28 Mar 2012 19:59:04,653 DEBUG MonitorOutputHandler - Reporting  
Metric Name [Custom Metrics|zos|% CPU utilization (CP)] Value [3]  
[Worker-7] 28 Mar 2012 19:59:04,654 DEBUG ExecTask - Process exited with code:  
0
```

Learn More

- App Agent for Java on z-OS or Mainframe Environments Configuration

Machine Agent Configuration Properties

- Where to Configure Machine Agent Properties
- Example Machine Agent controller-info.xml File
- Example Startup Configuration Using System Properties
- Machine Agent Properties
 - Agent-Controller Communication Properties
 - Controller Host Property
 - Controller Port Property
 - Machine Agent Identification Properties
 - Application Name Property
 - Tier Name Property
 - Node Name Property
 - Multi-Tenant Mode Properties
 - Account Name Property
 - Account Access Key Property
 - Proxy Properties for the Controller
 - Proxy Host Property
 - Proxy Port Property
 - Other Properties
 - Controller SSL Enabled Property
 - Enable Orchestration Property
 - Force Agent Registration Property
 - Unique Host ID Property
- Learn More

Where to Configure Machine Agent Properties

You can configure Machine Agent properties:

- in the controller-info.xml file located in the <Machine_Agent_Installation_Directory>/conf directory
- in the system properties (-D options) in the JVM start-up script

The system properties override the settings in the controller-info.xml file.

Example Machine Agent controller-info.xml File

```

<?xml version="1.0" encoding="UTF-8"?>
<controller-info>

<controller-host>192.10.10.10</controller-host>

<controller-port>8090</controller-port>

<controller-ssl-enabled>false</controller-ssl-enabled>

<enable-orchestration>false</enable-orchestration>

<account-name></account-name>
<account-access-key></account-access-key>

<application-name></application-name>
<tier-name></tier-name>
<node-name></node-name>

<force-agent-registration>false</force-agent-registration>

</controller-info>

```

Example Startup Configuration Using System Properties

A bash example:

```

-Dappdynamics.controller.hostName=192.168.1.20 -Dappdynamics.controller.port=8090
-Dappdynamics.agent.applicationName=ACMEOnline -Dappdynamics.agent.tierName=Inventory
-Dappdynamics.agent.nodeName=inventory1 org.tomcat.TomcatServer

```

Machine Agent Properties

This section describes the Machine Agent configuration properties, including their controller-info-xml elements and their system property options.

Agent-Controller Communication Properties

Controller Host Property

Description: This is the host name or the IP address of the AppDynamics Controller, e.g. 192.168.1.22 or myhost or myhost.abc.com. This is the same host that you use to access the AppDynamics browser-based user interface.

Element in controller-info.xml: <controller-host>

System Property: -Dappdynamics.controller.hostName

Type: String

Default: None

Required: if the Enable Orchestration property is false.

If Enable Orchestration is true, and if the agent is deployed in a compute cloud instance created by an AppDynamics workflow, do not set the Controller host unless you want to override the auto-detected value. See [Enable Orchestration Property](#).

Controller Port Property

Description: This is the HTTP(S) port of the AppDynamics Controller. This is the same port that you use to access the AppDynamics browser-based user interface. If the Controller SSL Enabled property is set to true, specify the HTTPS port of the Controller; otherwise specify the HTTP port. See [Controller SSL Enabled Property](#).

Element in controller-info.xml: <controller-port>

System Property: -Dappdynamics.controller.port

Type: Positive Integer

Default: The default values are 8090 for HTTP and 8181 for HTTPS.

Required: Yes, if the Enable Orchestration property is false.

If Enable Orchestration is true, and if the agent is deployed in a compute cloud instance created by an AppDynamics workflow, do not set the Controller port unless you want to override the auto-detected value. See [Enable Orchestration Property](#).

Machine Agent Identification Properties

If the machine agent is installed on a machine that does not have an App Server agent, configure the application name, tier name and the node name. Otherwise these configurations are not required for the machine agent.

Application Name Property

Description: This is the name of the logical business application that this JVM node belongs to. Note that this is not the deployment name(ear/war/jar) on the application server.

If a business application of the configured name does not exist, it is created automatically.

Element in controller-info.xml: <application-name>

System Property: -Dappdynamics.agent.applicationName

Type: String

Defaults: None

Required: If a registered app server agent is already installed on the same host as this machine agent, this configuration is not required.

Tier Name Property

Description: This is the name of the logical tier that this JVM node belongs to. Note that this is not the deployment name (ear/war/jar) on the application server.

If a tier of the configured name does not exist, it is created automatically.

Element in controller-info.xml: <tier-name>

System Property: -Dappdynamics.agent.tierName

Type: String

Defaults: None

Required: If a registered app server agent is already installed on the same host as this machine agent, this configuration is not required.

Node Name Property

Description: This is the name of the JVM node.

Element in controller-info.xml: <node-name>

System Property: -Dappdynamics.agent.nodeName

Type: String

Defaults: None

Required: If a registered app server agent is already installed on the same host as this machine agent, this configuration is not required.

Multi-Tenant Mode Properties

If the AppDynamics Controller is running in multi-tenant mode or if you are using the AppDynamics SaaS Controller, specify the account name and account access key for this agent to authenticate with the Controller.

If the Controller is running in single-tenant mode (the default) there is no need to configure these values.

Account Name Property

Description: This is the account name used to authenticate with the Controller.

If you are using the AppDynamics SaaS Controller, the Account Name is provided in the Welcome email sent by AppDynamics.

Element in controller-info.xml: <account-name>

System Property: -Dappdynamics.agent.accountName

Type: String

Default: None

Required: Yes for AppDynamics SaaS Controller and other multi-tenant users; no for single-tenant users.

Account Access Key Property

Description: This is the account access key used to authenticate with the Controller.

Element in controller-info.xml: <account-access-key>

System Property: -Dappdynamics.agent.accountAccessKey

Type: String

Default: None

Required: Yes for AppDynamics SaaS Controller and other multi-tenant users; no for single-tenant users.

Proxy Properties for the Controller

These properties route data to the Controller through a proxy.

Proxy Host Property

Description: This is the proxy host name or IP address.

Element in controller-info.xml: Not applicable

System Property: -Dappdynamics.http.proxyHost

Type: String

Default: None

Required: No

Proxy Port Property

Description: This is the proxy HTTP(S) port.

Element in controller-info.xml: Not applicable

System Property: -Dappdynamics.http.proxyPort

Type: Positive Integer

Default: None

Required: No

Other Properties

Controller SSL Enabled Property

Description: This property specifies whether the agent should use SSL (HTTPS) to connect to the Controller. If SSL Enabled is true, set the Controller Port property to the HTTPS port of the Controller. See [Controller Port Property](#).

Element in controller-info.xml: <controller-ssl-enabled>

System Property: -Dappdynamics.controller.ssl.enabled

Type: Boolean

Default: False

Required: No

Enable Orchestration Property

Description: When set to true, this property enables machine agent workflow task execution. It also enables auto-detection of the controller host and port when the app server is a compute cloud instance created by an AppDynamics orchestration workflow. In a cloud compute environment, auto-detection is necessary for the Create Machine tasks in the workflow to run correctly.

See [Controller Host Property](#) and [Controller Port Property](#).

The machine agent polls for task executions only when orchestration is enabled.

If the host machine on which this agent resides is not created through AppDynamics workflow orchestration, this property should be set to false.

Element in controller-info.xml: <enable-orchestration>

System Property: Not applicable

Type: Boolean

Default: False

Required: No

Force Agent Registration Property

Description: Set to true only under the following conditions:

- The Agent has been moved to a new application and/or tier from the UI and
- You want to override that move by specifying a new application name and/or tier name in the agent configuration.

If there is already a registered app server agent installed on the same host as this machine agent, this override does not work. If you want to override the UI in this case, you must force the agent registration change from the app server agent configuration.

Element in controller-info.xml: <force-agent-registration>

System Property: Not applicable

Type: Boolean

Default: False

Required: No

Unique Host ID Property

Description: This property logically partitions a single physical host or virtual machine.

You can use the unique host id when you want to use the same node name for multiple nodes on the same physical machine.

Set the value to a string that is unique across the entire managed infrastructure. The string may not contain any spaces.

If this property is set on the machine agent, it must be set on the app agent as well.

Note that if more than one app agent is running on the host, to see machine agent metrics it is necessary to run a new machine agent instance every time you specify a different unique host id on that host.

System Property: -Dappdynamics.agent.uniqueHostId

Type: String

Default: None

Required: No

Learn More

- [Install the Machine Agent](#)
- [Machine Agent Install and Admin FAQ](#)

Machine Agent HTTP Listener

- To activate the HTTP listener
- To send metrics
- To send events
- To upload metrics
- To upload events
- To shut down the machine agent

You can send metrics to the Machine Agent using its HTTP listener. You can report metrics through the Machine Agent by making HTTP calls to the agent instead of piping to the agent through sysout.

To activate the HTTP listener

- Restart the Machine Agent using metric.http.listener:

```
java -Dmetric.http.listener=true -Dmetric.http.listener.port=<port_number>  
-jar machineagent.jar
```

If you do not specify the optional metric.http.listener.port, it defaults to 8293.

To send metrics

```
GET | POST /machineagent/metrics
```

To send events

```
GET /machineagent/event
```

To upload metrics

You can use GET or POST to upload metrics to the Metric Browser under **Application Performance -> <Tier>** where the tier is the one defined for the Machine Agent. For example:

```
http://host:port/machineagent/metrics?name=foo|bar&value=42&type=average
```

Valid values for type are:

- average

- sum
- current

To upload events

2. Send events using HTTP get requests to:

```
http://localhost:8293/machineagent/event?type=<event_type>&summary=<summary_text>
```

Event_type is one of the following:

- error
- info
- summary
- warning

To shut down the machine agent

```
GET /machineagent/shutdown
```

Machine Agent Install and Admin FAQ

- Why is the Machine Agent not displayed on the Flow Map?
- Are the metrics monitored by the Machine Agent included in the Infrastructure Health indicator in the Main Dashboard?
- How can I configure the Machine Agent to run automatically when the machine starts?
 - Instructions for Linux environments
 - Instructions for the Windows environment
- If the machine is hosting servers that belong to multiple business applications, how do I assign the Machine Agent to multiple applications?
- Why are all the metrics for disk reads and writes and/or network I/O statistics zero?
- [Learn More](#)

Why is the Machine Agent not displayed on the Flow Map?

The Machine Agent monitors a particular machine and not a particular application server or JVM. The Machine Agent can therefore refer to multiple Nodes running on the same machine. A Flow Map, on the other hand, displays the communication between different Nodes during application execution, or the Business Transaction flow from Tier to Tier. A Machine Agent cannot be a part of the flow and therefore is not shown in the Flow Map.

Are the metrics monitored by the Machine Agent included in the Infrastructure Health indicator in the Main Dashboard?

Yes, the health indicator is driven by health rule violations in the given time period and health rule violations are configured on hardware metrics collected by the Machine Agent. Health Rules for all possible metrics are not configured out-of-the-box, so you might want to configure additional health rule according to your requirements. For details see [Configure Health Rules](#).

How can I configure the Machine Agent to run automatically when the machine starts?

Use the instructions listed below:

- Instructions for Linux environments
- Instructions for the Windows environment

Instructions for Linux environments

1. Create a script file as shown below:

```

#!/bin/sh
# -----
# Typical AppDynamics Machine Agent Startup
# -----

#CHANGE ME: Set to the agent's install directory
JAVA=java

#CHANGE ME: Set to the agent's install directory
AGENT_HOME=/opt/appdynamics/machineagent
AGENT="$AGENT_HOME/machineagent.jar"

# Agent Options
# Uncomment and make available as needed
# -Dappdynamics.agent.applicationName : application that the agent participates in
# -Dappdynamics.agent.logging.dir : directory to put logs (agent "user" must have write
permissions
# -Dmetric.http.listener=true | false : open a kill port
# -Dmetric.http.listener.port : the port to send kill messages

AGENT_OPTIONS=
#AGENT_OPTIONS="$AGENT_OPTIONS -Dappdynamics.agent.applicationName=<application-name>"
#AGENT_OPTIONS="$AGENT_OPTIONS -Dappdynamics.agent.logging.dir="
#AGENT_OPTIONS="$AGENT_OPTIONS -Dmetric.http.listener=true | false"
#AGENT_OPTIONS="$AGENT_OPTIONS -Dmetric.http.listener.port=<port>"

#Consolidation Step - allows you to easily add or remove the agent
#Call the nohup $JAVA $AGENT_OPTIONS -jar $AGENT &

```

2. Add the script file to the rc utility

For Red Hat and most Linux Operating Systems	For Ubuntu and Debian Operating Systems
Add the script file created in Step-1 in /etc/rc.local For example: echo '/path/to/startup/script.sh start' >> /etc/rc.local	Add the script file created in Step-1 as shown below: For example: update-rc.d -f script.sh start 99 2 3 4 5. where <ul style="list-style-type: none"> • start is the argument given to the script (start, stop). • 99 is the start order of the script (1 = first one, 99= last one) • 2 3 4 5 are the runlevels to start ⚠️ IMPORTANT: Do not forget to add dot(.) at the end.

Instructions for the Windows environment

See [The Java-Based Machine Agent](#).

If the machine is hosting servers that belong to multiple business applications, how do I assign the Machine Agent to multiple applications?

You can configure the Machine Agent with its own Application/Tier/Node properties. If there are Nodes belonging to multiple business applications, you can run multiple Machine Agents each configured to report metrics for a different Node, Tier, and Application. For details see [Install the Machine Agent](#).

Background information: You cannot "assign" a Machine Agent to multiple business application per se. A Machine Agent on a specific machine is automatically associated with all Nodes running on that machine. A Node is associated with a single business application. Therefore an application is associated with the Machine Agents of its Nodes. By default Machine Agents inherit the Application/Tier/Node names of the App Agents installed on the same hardware.

Why are all the metrics for disk reads and writes and/or network I/O statistics zero?

This situation may occur when a 32-bit JRE is used with 64-bit operating system. To solve this problem, use a 64-bit JRE with the 64-bit

operating system.

Learn More

- [Install the Machine Agent](#)
- [Machine Agent Configuration Properties](#)
- [Logical Model](#)

Modify Machine Agent Data Collection Metrics

- [Machine Agent Metrics](#)
 - To customize default Machine Agent metric collection

Machine Agent Metrics

By default, AppDynamics Machine Agent report metrics for only 'external' network and 'real' disks (network and local). Also, only the external network traffic is aggregated (to ensure backward compatibility with previous versions of AppDynamics).

However, you can customize this default behavior by modifying the auto-generated configuration file, task-template.xml. The task-template.xml file provides information about the current configuration of the Machine Agent.

To customize default Machine Agent metric collection

Step 1: Open the task-template.xml file.

AppDynamics creates this file in the <Machine_Agent_Directory>/monitors/JavaHardwareMonitor/ directory when the Machine Agent starts up for the first time.

Step 2: Modify the task-template.xml file.

A sample task-template.xml file is given below:

```
<config>
    <disk aggregate="false" enabled="false">sunrpc</disk>
    <disk aggregate="true" enabled="true">/dev/sdb1</disk>
    <disk aggregate="false" enabled="false">proc</disk>
    <disk aggregate="false" enabled="false">none</disk>
    <disk aggregate="false" enabled="false">devpts</disk>
    <disk aggregate="true" enabled="true">/dev/sda1</disk>
    <disk aggregate="false" enabled="false">nfsd</disk>
    <disk aggregate="true" enabled="true">/dev/mapper/saas4-binlog</disk>
    <disk aggregate="false" enabled="false">sysfs</disk>
    <disk aggregate="false" enabled="false">tmpfs</disk>

    <network aggregate="true" enabled="true">lo</network>
    <network aggregate="true" enabled="false">sit0</network>
    <network aggregate="true" enabled="true">eth0:1</network>
    <network aggregate="true" enabled="true">eth0</network>
    <network aggregate="true" enabled="false">eth1</network>
</config>
```

- To enable aggregation operation for localhost (lo) network metrics, change the value of the aggregate attribute (for the network element "lo") to "true".
- To enable monitoring for a virtual disk, set the value of the enabled attribute to "true" for that disk.

Step 3: Rename the task-template.xml file to task.xml.

- It is important to rename the task-template.xml file (else it will be overwritten by Machine Agent). The task-template.xml file is automatically generated by Machine Agent.

- If you want the Machine Agent to monitor a special device that is not enabled, add a file named "task.xml" in the <Machine_Agent_Directory>/monitors/JavaHardwareMonitor/ directory. The format of the task.xml file must be exactly the same format as the task-template.xml file.
- Not all disks and networks have to be listed in task.xml. If the machine agent finds a disk or a network that is not listed in task.xml, default properties are applied.

Step 4: Restart the Machine Agent.

Restart the machine agent for the changes to take effect.

Configure Multiple Machine Agents for One Machine

- [Configuring Multiple Machine Agents on Single Machine \(Java Only\)](#)
 - To configure two machine agents for two applications running on the same machine
- [Sample Configuration](#)
- [Learn More](#)

Configuring Multiple Machine Agents on Single Machine (Java Only)

If you have different applications running on the same machine, to get hardware metrics for each application, run separate machine agents on the same machine.

To do this, create multiple copies of the machine agent. Then configure each machine agent and each app agent pair to use the same application name and Unique Host ID. The Unique Host ID makes it appear to the Controller that the application is running on different machines. See [Application Name Property](#) and [Unique Host ID Property](#).

The following instructions assume two applications and two machine agents on a single machine, but they can be interpolated to cover more than two.

To configure two machine agents for two applications running on the same machine

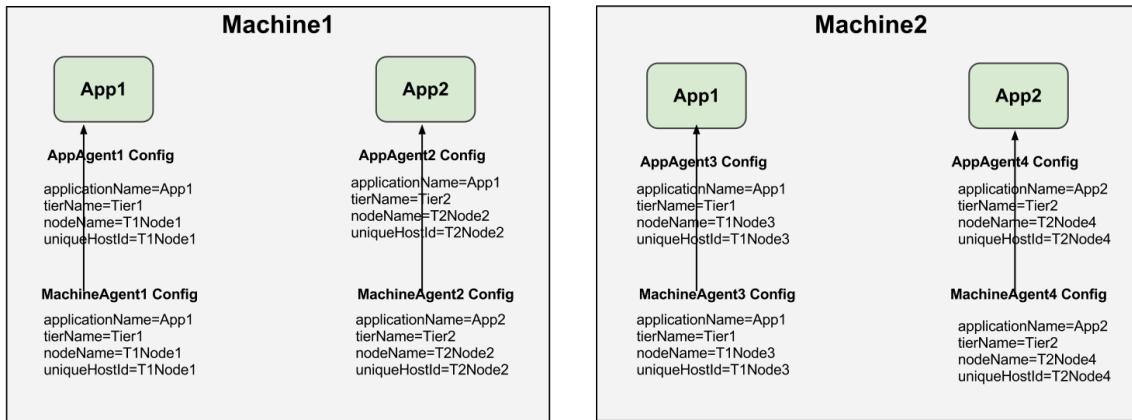
1. Download two copies of the machine agent, one for each application.
2. Assign the machine agents different names, for example: "MachineAgent1" and "MachineAgent2". If there are custom scripts running on the machine agents, they must not use the same resources.
3. Configure the first application/machine agent pair for the first application:
 - Delete the app agent node from the Controller UI.
 - Configure the applicationName and UniqueHostID properties for the app agent.
 - Configure the applicationName and UniqueHostID properties for the machine agent using the same application name and unique host id values that you used for the app agent configuration.
4. Configure the second application/machine agent pair for the second application:
 - Delete the app agent node from the Controller UI.
 - Configure the applicationName and UniqueHostID properties for the app agent. These values must be different from the values used for the first application.
 - Configure the applicationName and UniqueHostID properties for the machine agent using the same application name and unique host id values that you used above for the app agent configuration.
5. Restart all the JVMs.

Sample Configuration

The following sample configures two physical machines.

Machine1 runs App1, which is instrumented with one app agent (AppAgent1) and one machine agent (MachineAgent1). Machine1 also runs App2, which is instrumented with one app agent (AppAgent2) and one machine agent (MachineAgent2).

Machine2 runs App1, which is instrumented with one app agent (AppAgent3) and one machine agent (MachineAgent3). Machine2 also runs App2 is instrumented with one app agent (AppAgent4) and one machine agent (MachineAgent4).



In the Controller, separate metrics are reported for:

Machine1_App1
Machine1_App2
Machine2_App1
Machine2_App2

Learn More

- App Agent for Java Configuration Properties
- Machine Agent Configuration Properties

Administer the Controller

For Controller installation topics see:

Modify Glassfish JVM Options

- Synopsis
- Description
- Restart
- Examples
 - Modify a startup parameter for the Java Application Launcher
 - Modify Multiple Startup Parameters for the Java Application Launcher
 - Modify Multiple Java System Properties

AppDynamics provides a modifyJvmOptions utility for adding and deleting JVM options for the Java application launcher in a Glassfish environment.

Invoke the utility from the /bin subdirectory of the Controller installation directory.

On Linux use modifyJvmOptions.sh.
On Windows use modifyJvmOptions.bat.

Synopsis

modifyJvmOptions [add | delete] (jvm-option-name=jvm-option-value) [@jvm-option-name=jvm-option-value*]

Separate multiple JVM options using the @ sign.

On Microsoft Windows it is necessary to enclose the entire JVM option names and values string in double-quotation marks. This is a Microsoft Windows requirement. See the examples.

Description

The modifyJvmOptions utility adds or deletes command-line options that are passed to the Java application launcher when Glassfish Server is started. These are in addition to the options that are preset in the Glassfish Server.

Use modifyJvmOptions to add or delete the following types of options:

- **Java system properties:** These options are set using the **-D** option of the Java Application launcher
- **Startup parameters for the Java application launcher:** These options are set using the dash (-) character

The utility does not validate your settings, so make sure you use valid values. Invalid options can cause startup to fail.

After the server starts, the options are written to server.log before any other information is logged.

Restart

The addition of some options requires a server restart for changes to become effective. Other options are set immediately in the environment of the domain administration server (DAS) and do not require a restart. Whether a restart is required depends on the type of option.

Server restart is required when you add or delete the following types of properties:

- Java system properties with names that start with **-Djava.** or **-DJavax.** (including the trailing period)
For example, you must restart the server when you modify this property: **-Djava.security.manager**.
- All startup parameters for the Java application launcher.
For example, you must restart the server when you modify these properties: **-client, -Xmx1024m, -d64**.

Server restart is not required when you add or delete the following types of properties:

- Java system properties with names that do not start with **-Djava.** or **Djavax.** (including the trailing period)
For example, you do not need to restart the server when you modify this property: **-Denvironment=Production**.

Examples

Modify a startup parameter for the Java Application Launcher

The following command sets the maximum available heap size to 1024MB using modifyJvmOptions.sh:

```
modifyJvmOptions.sh add "-Xmx1024m"
```

The following command deletes the maximum available heap size parameter set in the previous command:

```
modifyJvmOptions.sh delete "-Xmx1024m"
```

Modify Multiple Startup Parameters for the Java Application Launcher

The following command sets the maximum available heap size to 1024 and requests details about garbage collection using modifyJvmOptions.bat:

```
modifyJvmOptions.bat add "-Xmx1024m@-XX\:+PrintGCDetails"
```

The following command deletes the startup parameters set in the previous command

```
modifyJvmOptions.bat delete "-Xmx1024m@-XX\:+PrintGCDetails"
```

Modify Multiple Java System Properties

The following commands add and delete multiple Java system properties using modifyJvmOptions.bat. Note that the quotation marks

enclosing the argument string are required.

```
modifyJvmOptions.bat add  
"-Dunixlocation=/root/example@-Dvariable=\$HOME@-Dwindowslocation=d:\\sun\\appserver@-Doption1=v  
delete  
"-Dunixlocation=/root/example@-Dvariable=\$HOME@-Dwindowslocation=d:\\sun\\appserver@-Doption1=v
```

The following commands add and delete multiple Java system properties using modifyJvmOptions.sh. Quotation marks enclosing the argument string are not required.

```
modifyJvmOptions.sh add  
-Dunixlocation=/root/example@-Dvariable=\$HOME@-Dwindowslocation=d:\\sun\\appserver@-Doption1=v  
delete  
-Dunixlocation=/root/example@-Dvariable=\$HOME@-Dwindowslocation=d:\\sun\\appserver@-Doption1=v
```

Access the AppDynamics UI

Once you have installed the Controller and agents, launch your web browser and connect to the AppDynamics User Interface (UI).

On-Premise Controller UI URL

```
http://<controller-host>:<controller-port>/controller
```

Login Credentials

On-premise Controller admin login credentials are configured during installation.

Account Name in the URL

In Multi-Tenant Mode you can have multiple account names. You can pass the account name to the Controller UI using the URL:

```
http://<account-name>.saas.appdynamics.com/controller?account=Customer1
```

When you pass the account name in the URL, the login screen Account field will contain the name.

Learn More

- [Install and Upgrade AppDynamics](#)
- [Use a SaaS Controller](#)

Start or Stop the Controller

- To start the Controller
- To stop the Controller
- To stop the Controller database
- [Learn More](#)

The scripts to start or stop the Controller are located in the <Controller_Installation_Directory>/bin folder.

To start the Controller

Open a command line console and execute following command:

- For Linux:

```
./controller.sh start
```

- For Windows:

```
controller.bat start
```

To stop the Controller

Open a command line console and execute following command:

- For Linux:

```
./controller.sh stop
```

- For Windows:

```
controller.bat stop
```

To stop the Controller database

Often when you stop the Controller you also need to stop its database. Open a command line console and execute following command:

- For Linux:

```
./controller.sh stop-db
```

- For Windows:

```
controller.bat stop-db
```

Learn More

- [Controller Database Scripts](#)

Administrative Users

- Account Owner or Account Administrator
 - To create other users
- AppDynamics Administrator
 - Administrator Password
- [Learn More](#)

This topic describes the different types of administrative users in AppDynamics.

Account Owner or Account Administrator

Every account including SaaS accounts has an account owner, also called the account administrator. Account owners are created by

the AppDynamics administrative user. Account owners can create other users with different levels of access within the account.

To create other users

1. Log in to the Controller with an account administrator password.
2. In the upper right of the window, click **Setup -> Administration**.
3. See [Configure Users](#).

AppDynamics Administrator

The AppDynamics administrator is created during on-premise Controller installation. For most SaaS installations an AppDynamics employee is the AppDynamics administrator.

The AppDynamics administrator can log into the Administration console with the administrative password to create, edit, and delete AppDynamics accounts and configure Controller settings. In Linux environments this user is sometimes called "root" or the "root user".

When creating accounts, the AppDynamics administrator creates account owners for those accounts.

Create Account Administrator
The Account Administrator will be able to login and create other users.
Once it is created, this user can be edited by logging in as this user and selecting Settings -> My Settings from the main menu.

Username:

Password:

Repeat Password:

See [Access the Administration Console](#).

Administrator Password

To change the Administrator password:

1. Login to the Controller UI with the `enableAccounts` parameter:

```
<host>:<port>/controller/?enableAccounts=true
```
2. In the upper right of the window, click **Setup -> My Preferences**.
3. In the My Account panel, click **Change Password**.
4. Enter the new administrator password.
5. Click **Save**.

Learn More

- [Install the Controller on Linux](#)
- [Install the Controller on Windows](#)

Access the Administration Console

- [Access the Administration Console](#)
- [Learn More](#)

To configure particular Controller settings, log into the AppDynamics Administration console. You will be prompted for the password that was created for the AppDynamics admin user when the Controller was installed. For a Controller installed on Linux this is called the "Root Password".

AppDynamics recommends that you contact AppDynamics Support for more information before changing Controller settings.

Access the Administration Console

The URL of the Administration Console is:

```
<host>:<port>/controller/admin.html
```

Log in with the default administrator/root password, which you should change immediately. See [Administrative Users](#).

Learn More

- [Administrative Users](#)

Configure an On-Premise Controller

This section describes how to configure the Controller for best results.

Controller Licenses

- [The License File for On-Premise Installations](#)
 - [To add the Controller license file on Windows](#)
 - [To add the Controller license file on Linux](#)

This topic discusses how to manage Controller licenses.

The License File for On-Premise Installations

Download the license file, license.lic, from the Welcome email that you receive when you sign up for the trial or when you purchased AppDynamics.

You do not need to deploy the license file for SaaS installations.

To add the Controller license file on Windows

1. Download the license file.
2. Copy and paste this license.lic file to the <Controller_Installation_Directory>.

To add the Controller license file on Linux

1. Download the license file.
2. Use the cp command to copy the downloaded license file to the Controller installation directory.

```
>> cp license.lic <Controller_Installation_Directory>
```

3. Wait at least 30 seconds for the Controller to pick up the new license file.

Controller Tenant Mode

ays

- [Tenant Mode and the Controller Accounts with Access to Business Applications](#)
 - [Switching Between Single- and Multi-Tenant Mode](#)
 - [To switch between single- and multi-tenant mode](#)
- [Accounts for Multi-Tenant Mode](#)
 - [To create accounts in multi-tenant mode](#)
 - [Agent-Controller Communication Settings](#)
 - [To update agents with the new Controller Account information](#)
- [Learn More](#)

Tenant Mode and the Controller Accounts with Access to Business Applications

You can configure the AppDynamics Controller in either single-tenant (single account) or multi-tenant (multiple account) modes. By default, the Controller runs in single-tenant mode.

The differences are:

In single-tenant mode

- There is only one account (tenant) in the Controller system.
- All users are part of this single built-in account. Similarly, all Applications are part of that single built-in account.
- All the users will have access to all monitored Applications in this mode.
- AppDynamics recommends single-tenant mode for most installations.

In multi-tenant mode

- You can create multiple accounts (tenants).
- Each account will have its own set of users and Applications. The users in an account will only have access to Applications in that account.
- Multi-tenant mode allows secure partitioning of users and data among different parts of your organization.
- If you install the Controller in multi-tenant mode, the agents must specify additional information about the account to which they connect. For details see [App Agent for Java Configuration Properties](#), [App Agent for .NET Configuration Properties](#), and [Machine Agent Configuration Properties](#).

Switching Between Single- and Multi-Tenant Mode

You can choose the tenancy mode when you run the Controller installer. You can switch between these two modes after you have installed the Controller.

 Note: these instructions are applicable only if you have installed the Controller on-premise.

To switch between single- and multi-tenant mode

1. Use the following URL for accessing the account management information.

```
http://<controller-installer-host>:<port>/controller/admin.html
```

 Note: You have to be logged in as the root user. The default password for the root user is "changeme", for both single- and multi-tenant mode.

2. Click **Controller Settings**.

3. If you want to switch from single to multi-tenant mode, set the value for the multi-tenant.controller property to "true".

4. Save the settings.

5. Log out and refresh your browser.

Accounts for Multi-Tenant Mode

In multi-tenant mode you create accounts using the "Accounts" setting.

To create accounts in multi-tenant mode

1. Use the following URL to access the account management information:

```
http://<controller-installer-host>:<port>/controller/admin.html
```

 Note: You have to be logged in as the root user. The default password for the root user is "changeme", for both single- and multi-tenant mode.

2. Click **Accounts**.

If you are switching from single to multi-tenant mode, you will see the default accounts used for single-tenant mode.

This window also provides information on the number of licenses provisioned for the agents in your environment.

3. Click **Add** (the + icon).

4. Provide the details about the Account.

Users will log into the AppDynamics UI using this Account Name.

5. Create credentials for the Controller administrator.

The user with administrator privileges can then create users for that Account. For details see [Controller Users](#).

6. Click **Create account**.

Agent-Controller Communication Settings

When you change tenant modes, you must also update the app server and machine agents with the new Account information.

To update agents with the new Controller Account information

For details see [App Agent for Java Configuration Properties](#), [App Agent for .NET Configuration Properties](#), and [Machine Agent Configuration Properties](#).

Learn More

- [Controller Users](#)
- [App Agent for Java Configuration Properties](#)
- [App Agent for .NET Configuration Properties](#)
- [Machine Agent Configuration Properties](#)
- [Install the App Agent for Java](#)
- [Install the App Agent for .NET](#)
- [Install the Machine Agent](#)

Controller Port Settings

- [Controller Ports](#)
- [Changing the Controller Port for On-Premise Installations](#)
 - [Reinstalling the Controller with New Port Settings](#)
 - [To reinstall the Controller with new port settings](#)
 - [Editing Controller Port Configurations](#)
 - [To edit configuration files and change the Controller port settings](#)

Controller Ports

The Controller requires the following port connection settings in the configuration files:

Port Name	Default
Application server primary port	Port 8090 is used for HTTP and port 8181 is used for HTTPS.
Database server port	3388
Application server admin port	4848
Application server JMS port	7676
Application server IIOP port	3700

Changing the Controller Port for On-Premise Installations

If you have installed the Controller on-premise, you can change the port settings using either of the following procedures:

- [Reinstalling the Controller](#)
or

- Editing Controller Port Configurations

Reinstalling the Controller with New Port Settings

Use the reinstall procedure if you want to modify connection settings without editing configuration files.

To reinstall the Controller with new port settings

1. Shut down the Controller. For instructions see [Start or Stop the Controller](#).
 2. Create a complete backup of the Controller installation directory and all of its sub-directories.
 3. Re-install the Controller in the same installation directory as your original Controller and choose the new port settings while running the installer. For instructions see [Install the Controller on Linux](#) or [Install the Controller on Windows](#).
-  This step will be treated as an upgrade by the installer for the Controller.

Editing Controller Port Configurations

Use this procedure to manually edit the port connection setting without reinstalling the Controller.

To edit configuration files and change the Controller port settings

1. Shut down the Controller. For instructions see: [Start or Stop the Controller](#).
2. **Important:** Shut down the Controller before making edits to configuration files. AppDynamics stops collecting data when you shut down the Controller and resumes data collection after the Controller is restarted.
3. Open the config.properties file located at <Controller-Installation-Directory>/appserver/domains/domain1/imq/instances/imqbroker/props.
 - Set the "imp.persist.jdbc.mysql.property.url" variable (the JDBC connection string) to your new port setting.
 - Save your changes.
4. Open the db.cnf file located at <Controller-Installation-Directory>/db.
 - Set the "port=" variable to your new port setting.
 - Save your changes.
5. Open the controller.bat (.sh) file located at <Controller-Installation-Directory>\bin.
 - Change the "DB_PORT" variable to your new port setting.
 - Save your changes.
6. Open the domain.xml file located at <Controller-Installation-Directory>/appserver/domains/domain1/config.
 - Find the <http-listener> element that has following attribute: id="admin-listener".
 - Set the port (port=) value to your new port setting.
 - Save your changes.
7. Open the asadminenv.conf file located at <Controller-Installation-Directory>/appserver/config.
 - Change the "AS_ADMIN_PORT" variable to your new port setting.
 - Save your changes.
8. Open the domain.xml file located at <Controller-Installation-Directory>/appserver/domains/domain1/config.

- Find the <jms-host> element that has the attribute name="default_JMS_host".
- Set the value of "port=" variable to your new port setting.
- Save your changes.

9. Open the domain.xml file located at <Controller-Installation-Directory>/appserver/domains/domain1/config. Find the <iop-listener> element that has the attribute: id="orb-listener-1".

- Set the value of the "port=" variable to your new port setting.
- Save your changes.

10. Restart the Controller. For instructions see [Start or Stop the Controller](#).

Controller SSL and Certificates

- [Using SSL with the Controller](#)
- [To install a trusted certificate for the Controller](#)

Using SSL with the Controller

By Default, the Controller ships with a self-signed certificate, which is not recommended for use in production. We recommend that you replace this certificate with a proper CA signed certificate.

To install a trusted certificate for the Controller

1. Generate a key pair.

```
keytool -genkey -keyalg rsa -keystore keystore.jks -alias slas
```

 The default keystore password is "changeit". The alias is "slas".

2. Change the password.

```
<Controller_Installation_Directory>/appserver/bin/asadmin
change-master-password --savemasterpassword=true
```

3. Generate a Certificate Signing Request (CSR).

```
keytool -certreq -alias slas -file controller-ca.csr -keystore keystore.jks
```

4. Make a service request to a trusted Certificate Authority (CA) to issue a valid certificate.

5. Import the root and intermediate certificates into the keystore.jks: keytool

```
keytool -import -alias slas -file <CA-signed-cert>.cert -keystore keystore.jks
```

6. Copy the keystore.jks into the config directory for the Controller located at <Controller_Installation-Directory>/appserver/domains/domain1/config.

Controller Logs

- [Log Files Generated by the Controller](#)
 - To change the location of the Controller log directory
- [Debug Mode Logging For the Controller](#)
 - To configure logging levels for the Controller
- [Learn More](#)

Log Files Generated by the Controller

The Controller creates two log files in the <Controller_Installation_Directory>/logs directory:

- database.log
- server.log

To change the location of the Controller log directory

1. Stop the Controller and its database. See [Start or Stop the Controller and Controller Database Scripts](#).
2. Open the <Controller_Installation_Directory>/db/db.cnf file.
3. Set the value of the log-error property to the new location of the database.log file.
4. Save the db.cnf file.
5. Open the <Controller_Installation_Directory>/appserver/domains/domain1/config/domain.xml file.
6. Locate the <log-service> element.
7. Set the value of the file attribute to the new location of the server.log file.
8. Copy (or move) the pre-existing logs directory (<Controller_Installation_Directory>/logs) to the new location.
- 9 Start the Controller. See [Start or Stop the Controller](#).
10. Verify that the database.log and server.log are being written to the new locations.

Debug Mode Logging For the Controller

Debug logs provide information about possible errors in Controller operations.

To configure logging levels for the Controller

1. Access the Controller application server.

- Open a browser at the following URL:

```
http://<Controller-Host>:4848
```

The administrative console for the Controller Application Server opens. Port 4848 is the default port number for the Controller application server.

To log in to the Controller application server administrative console, use "admin" as the username and the password that is located in the <controller_install_directory>/.passwordfile file.

2. Go to "Application Server" present in the left hand side tree of the administrative console.
3. Click **Logging -> Logging Levels**.
4. Click **Additional Properties**.
5. Modify only those elements that start with "com.singularity". INFO is the default logging mode.

To enable debug mode logging for any component, change the logging level from INFO to "FINE" mode.

The screenshot shows the Sun GlassFish Enterprise Server v2.1 Administration Console. The top navigation bar includes 'Home' and 'Version' buttons, and displays 'User: admin | Domain: domain1 | Server: localhost'. The main title is 'Sun GlassFish™ Enterprise Server v2.1'. On the left, a sidebar menu has 'Application Server' selected (marked with a yellow circle 1). The main content area shows the 'Logging' tab of the 'Application Server' configuration (marked with a yellow circle 2). Within the 'Logging' tab, the 'Log Levels' sub-tab is selected (marked with a yellow circle 3). A large blue arrow points down to the 'Additional Properties (17)' table (marked with a yellow circle 4), which lists log levels for various components. The table has columns for Name and Value.

	Name	Value
<input type="checkbox"/>	com.singularity.ORCHESTRATION.ENGINI	INFO
<input type="checkbox"/>	com.singularity.INCIDENTS.WRITE	INFO
<input type="checkbox"/>	com.singularity.ORCHESTRATION.AGENT	INFO
<input type="checkbox"/>	com.singularity.INCIDENTS.READ	INFO
<input type="checkbox"/>	com.singularity.EVENTS.WRITE	INFO
<input type="checkbox"/>	com.singularity.SNAPSHOTS.WRITE	INFO
<input type="checkbox"/>	com.singularity.SNAPSHOTS.READ	INFO
<input type="checkbox"/>	com.singularity.RULES.PROCESSING	INFO
<input type="checkbox"/>	com.singularity.IPS	INFO
<input type="checkbox"/>	com.singularity	INFO
<input type="checkbox"/>	com.singularity.AGENT	INFO
<input type="checkbox"/>	com.singularity.BTS	INFO
<input type="checkbox"/>	com.singularity.METRICS.READ	INFO
<input type="checkbox"/>	com.singularity.METRICS.WRITE	INFO
<input type="checkbox"/>	com.singularity.EVENTS.READ	INFO
<input type="checkbox"/>	org.hibernate.engine.StatefulPersistence	SEVERE
<input type="checkbox"/>	javax.enterprise.system.stream.out	WARNING

You can control logging information for following components:

- Orchestration
- Incidents
- Events
- Snapshots
- Rules
- AGENT
- Business Transactions (BTS)
- Metrics
- Information Points (IPS)

Learn More

- Start or Stop the Controller

- Controller Disk Space and the Database
- Controller Database Scripts

Controller Performance

- Monitoring Controller Performance
 - To monitor heap usage
 - To check the server.log for any errors
- Performance Issues
 - To troubleshoot Controller performance issues
 - To troubleshoot Controller issues

Monitoring Controller Performance

To monitor Controller performance:

- Monitor heap usage
- Review the server log file

To monitor heap usage

- **On a Windows machine:** Use the Task Manager to measure the memory usage for the Controller.
- **On a Linux machine:** Use the **top** command to get statistics for the memory data.

```
ps -elf (expect to see a "java" process and a "mysql" process)

top (expect to see java and mysql with cpu greater than 0)
```

To check the server.log for any errors

1. Open a console and navigate to the <Controller_Installation_Directory>/logs/server.log file.
2. Open the log file and look for any anomalies.

Performance Issues

If you observe degradation in Controller performance it may be due to one of the following:

- The hardware resources for the Controller might not match the correct Controller profile.
- The Controller performance profile may be incorrectly configured.

To troubleshoot Controller performance issues

1. Confirm that the hardware matches the Controller profile you use. For details see [Controller System Requirements](#).
2. Confirm that your disk performance matches the recommended thresholds for minimum disk performance. For details see [Controller System Requirements](#).
3. Confirm that the Java SDK version is exactly the same as the Java version on the Controller. To display the version of Java used by the Controller:
 - Open the command line utility.
 - Go to <Controller_Installation_Directory>/jre/bin
 - Run java -version.

To troubleshoot Controller issues

If problems persist, contact Support. See AppDynamics Support for information about how to contact Support and how to collect troubleshooting information.

Configure the SMTP Server

- To configure the SMTP server
- To troubleshoot notifications
- Learn More

This topic describes how to configure the SMTP server for email and SMS notifications. In order for the email or SMS notifications to work, the SMTP server settings must be accurate.

To configure the SMTP server

1. Click **Settings -> Email / SMS Configuration**.
2. Provide connection information about the SMTP host, port, connection information, etc.

Email / SMS Configuration

Specify the SMTP server that the AppDynamics Controller will use to send email/SMS alert notifications.

SMTP Host: []

SMTP Port: []

Use Secure Connection:

- Never
- TLS, if avail.
- TLS
- SSL

Use Secure Authentication:

Notification Header Text: []

SMS Carrier: []

Authentication

Authentication required:

Username: []

Password: []

Confirm Password: []

Buttons: Test Email, Cancel, Save

3. Save the settings.

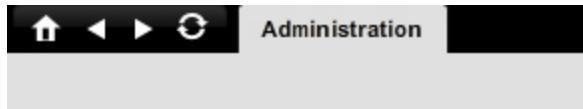
To troubleshoot notifications

If you do not receive notifications on health rule violations, it could be because the SMTP server timeout setting is too short a period of time. To troubleshoot this problem:

1. Log in to the Admin console using the admin account:

```
http://<controller-installer-host>:<port>/controller/admin.html
```

2. Select the Controller Settings option.



AppDynamics Administration

Accounts

Create and Manage Accounts

Controller Settings

Configure the Controller

3. Increase the mail.smtp.sockettimeout global configuration property. The default setting is 30 seconds.

Learn More

- [Notification Actions](#)
- [Use a SaaS Controller#SMTP Service for SaaS](#)

Controller High Availability

See also: [Upgrade the Controller#To upgrade Controllers configured in HA mode](#).

Manage Controller High Availability

- [High Availability Mode](#)
- [Environment for Controller High Availability](#)
 - [TTL \(Time To Live\) of DNS A Records](#)
 - [Setting Up HA Controllers](#)
 - [Common Commands Used to Set Up Controller HA](#)
 - [To set up HA Controllers](#)
 - [Failover and Failback Procedures](#)
 - [Configuring Agents for an HA Controller](#)
- [Learn More](#)

High Availability Mode

AppDynamics recommends that you use a [high availability cluster](#) for continued Controller operations if the server fails or becomes inoperable. You can configure the Controller in high availability (HA) mode to minimize any disruptions to Controller operations.

HA mode also helps you to avoid the complexity of performing hot backups.

Environment for Controller High Availability

To use HA mode, you need two servers for two Controllers which are almost exactly the same. Each Controller uses its own MySQL database.

- Primary Controller, with the primary (master) MySQL database
- Secondary Controller, with the secondary (slave) MySQL database

When HA is configured, the primary (master) MySQL database replicates data to the secondary (slave) MySQL database. HA mode uses a MySQL Master-Master replication type of configuration. For more information see [Master-Master replication for MySQL](#).

Use the information at [Controller System Requirements](#) to determine the sizing and resource needs for your environment. The two servers should be functionally equivalent. If you have to use one server that is more robust than the other, use the more powerful server for the primary Controller.

TTL (Time To Live) of DNS A Records

The Sysadmin of the DNS server should set the TTL (time to live) of the DNS A records to a short time (about 60 seconds), to more frequently refresh the IP address of the HA Controller in the agent DNS caches. This ensures that the agent gets a new name to IP translation for the HA Controller every minute or so. Otherwise the agent will continue to send messages to the failed controller until the TTL expires. If the TTL is set to the default of 84600, the agents keep using the old IP for a whole day and may not failover to the secondary controller in a timely fashion.

Agents and Controllers in an HA Scenario

Normally, before any failure, the App Server and Machine Agents communicate with the primary Controller. If the primary Controller becomes unavailable, the Agents start communicating with the secondary Controller.

The agents require information about both Controllers to ensure that this communication occurs.

Each Controller and MySQL database has a unique IP address, for example:

- Primary Controller has an IP address of 101.xxx.xxx.xxx
- Primary (master) MySQL Database has an IP address of 201.xxx.xxx.xxx
- Secondary Controller has an IP address of 102.xxx.xxx.xxx
- Secondary (slave) MySQL Database has an IP address of 202.xxx.xxx.xxx

When configuring the agents, AppDynamics recommends using the Controller DNS address (controller.company.com) instead of the IP address. For example, initially controller.company.com will have the IP address of 101.xxx.xxx.xxx. When the primary fails you can change the DNS to 102.xxx.xxx.xxx without changing each of the agents.

The other option is to use a load balancer that will switch Controllers with the agent knowledge. For example, the load balancer has an IP address of 103.xxx.xxx.xxx and it balances the load between 101.xxx.xxx.xxx and 102.xxx.xxx.xxx.

Setting Up HA Controllers

The following set-up procedures cover configuring a primary and a secondary Controller and their databases.

When executing SQL statements, use the IP address for the PRIMARY-HOSTNAME and SECONDARY-HOSTNAME.

The following table lists the Linux and Windows commands used during the set-up procedures.

Common Commands Used to Set Up Controller HA

Action	Commands for Windows	Commands for Linux
Start the Controller database	controller.bat start-db	controller.sh start-db
Log into the Controller database	controller.bat login-db	controller.sh login-db
Stop Controller database	controller.bat stop-db	controller.sh stop-db
Reset Controller database	controller.bat reset-db	controller.sh reset-db
Start Controller	startController.bat	startController.sh
Stop Controller	stopController.bat	stopController.sh

To set up HA Controllers

1. Obtain the free license for the secondary Controller from the [AppDynamics Support Team](#).
2. Launch the Controller installation and select HA Mode.
 - On the primary Controller machine select **Primary HA Mode**.
 - On the secondary Controller machine select **Secondary HA Mode**.

3. Stop both Controllers.

4. On the primary Controller machine perform the following actions:

- Start the primary Controller database and log in to the database.
- Execute the following command:

```
RESET MASTER
```

- Stop the primary Controller database.

5. Copy the <Controller_Installation_Directory>/db/data directory from the primary to the secondary machine.

For example, on Linux:

Primary Controller Machine

```
tar czvf data.tgz /mnt/appdynamics2/db/data  
scp data.tgz user@server2:/tmp
```

Secondary Controller Machine

```
cd /mnt/appdynamics/db/data  
rm \--fr *  
tar xzvf /tmp/data.tgz
```

6. Copy the copy controller.sh or controller.bat file from the primary machine to the secondary machine.

This is to make sure that the secondary machine uses the same mysql root user password as that of primary. Otherwise you cannot log into the secondary database for setting up replication.

7. On the primary Controller machine, log in to the Controller database and execute the following command:

```
GRANT ALL ON *.* TO 'controller_repl'@'SECONDARY-HOSTNAME' IDENTIFIED BY 'controller_repl';
```

8. Perform following actions on the SECONDARY machine:

- Start the secondary Controller database.
- Reset the database.
- Log in to the database and execute the following commands:

```
RESET MASTER  
  
GRANT ALL ON *.* TO 'controller_repl'@'PRIMARY-HOSTNAME' IDENTIFIED BY 'controller_repl';  
  
STOP SLAVE  
  
CHANGE MASTER TO MASTER_HOST='PRIMARY-HOSTNAME' , MASTER_USER='controller_repl' ,  
MASTER_PASSWORD='controller_repl' , MASTER_PORT=YOUR-MYSQL-PORT# ;  
  
START SLAVE
```

 DO NOT start the Controller yet.

9. On the primary Controller machine, execute following commands:

```
STOP SLAVE

CHANGE MASTER TO MASTER_HOST='SECONDARY-HOSTNAME' , MASTER_USER='controller_repl' ,
MASTER_PASSWORD='controller_repl' , MASTER_PORT=YOUR-MYSQL-PORT# ;

START SLAVE
```

10. Verify HA mode setup.

- On each machine, log into the Controller database.
- Execute the following command to display any set-up errors:

```
SHOW SLAVE STATUS\G
```

11. Start the primary Controller.

12. Set the appserver.mode property to "active" on the primary Controller.

```
http://<controller-host>:<controller-port>/controller/changeappservermode?activate=true
```

The Controller should return the message:

```
200: "App Server started in ACTIVE mode..."
```

13. Start the secondary Controller.

14. Set the appserver.mode property to "passive" on the secondary Controller.

```
http://<controller-host>:<controller-port>/controller/changeappservermode?activate=false
```

The Controller should return the message:

```
200: "App Server moved to PASSIVE mode..."
```

 Previous documentation instructed that you do not need to start the secondary Controller. This has changed and it is OK to run both Controllers.

Failover and Failback Procedures

If the Controller crashes on the primary machine, follow these two processes to recover:

1. Failover Process: use the appserver.mode property to switch operations to the secondary Controller machine.
2. Failback Process: once it is recovered and stable, switch operations back to the primary Controller machine

For details see [Controller HA Failover and Failback Process](#).

Configuring Agents for an HA Controller

When configuring the agents for a Controller in HA mode, you set the <controller-host> properties in the controller-info.xml (Linux) or Web.config (.NET) files for an App Agent and in the controller-info.xml file for a Machine Agent.

As previously discussed, use the DNS name for the <controller-host> property. For example:

```
<controller-host>controller.company.com</controller-host>
```

For more information see [Install and Upgrade AppDynamics](#).

Learn More

- Controller Data Backup and Restore
- App Agent for Java Configuration Properties
- App Agent for .NET Configuration Properties

Common Commands Used to Set Up Controller HA

Common Commands Used to Set Up Controller HA

Action	Commands for Windows	Commands for Linux
Start the Controller database	controller.bat start-db	controller.sh start-db
Log into the Controller database	controller.bat login-db	controller.sh login-db
Stop Controller database	controller.bat stop-db	controller.sh stop-db
Reset Controller database	controller.bat reset-db	controller.sh reset-db
Start Controller	startController.bat	startController.sh
Stop Controller	stopController.bat	stopController.sh

Controller HA Failover and Fallback Process

- Switching Operations between the Primary and Secondary Controllers
 - Common Commands Used to Set Up Controller HA
 - The appserver.mode Property and changeappservermode Command
- Controller Failover
 - To stop data replication on the secondary Controller
 - Increasing the Time for Storing the Replicated Data
 - To increase the data storage time to 10 days
- Controller Fallback
 - To sync up the database
 - Learn More

Switching Operations between the Primary and Secondary Controllers

If the primary Controller experiences a failure, use the process described in this topic to switch all operations to your secondary Controller on the backup machine. When the primary Controller is restored, switch back.

Common Commands Used to Set Up Controller HA

Action	Commands for Windows	Commands for Linux
Start the Controller database	controller.bat start-db	controller.sh start-db
Log into the Controller database	controller.bat login-db	controller.sh login-db
Stop Controller database	controller.bat stop-db	controller.sh stop-db
Reset Controller database	controller.bat reset-db	controller.sh reset-db
Start Controller	startController.bat	startController.sh
Stop Controller	stopController.bat	stopController.sh

The appserver.mode Property and changeappservermode Command

These instructions use the appserver.mode property. You must invoke this property using the system account. For example:

```
curl --user root@system:changeme http://<host>/controller/changeappservermode?activate=true)
```

Controller Failover

To stop data replication on the secondary Controller

1. Confirm that the secondary Controller database is running properly on your secondary machine.

- For Linux:

```
ps -aef | grep mysql
```

- For Windows:

```
tasklist /v | find "mysql"
```

2. Log in to the secondary Controller database on the secondary Controller machine and stop replicating data from the primary Controller machine. Execute following command:

```
STOP SLAVE;
```

3. Set the appserver.mode property to "true" on the secondary Controller.

```
http://<controller-host>:<controller-port>/controller/changeappservermode?activate=true
```

The Controller should return the message:

```
200: "App Server started in ACTIVE mode..."
```

Increasing the Time for Storing the Replicated Data

 **Important:** If it will take more than two days to repair and failback the primary Controller, AppDynamics recommends that you increase the number of days for which the secondary Controller stores the replicated data.

To increase the data storage time to 10 days

1. Login to the Controller database.

2. Execute following command:

```
SET GLOBAL expire_logs_days=10;
```

Controller Failback

Once the primary Controller machine is back up and restored to a stable state, you switch processing back to it and restore the secondary Controller to backup or passive state.

To sync up the database

1. On the primary Controller machine, start up only the database if it is not already started by running:

```
controller.sh start-db
```

⚠ Important: Do not start the application server during this phase as it will assume the appserver mode it was last in, which is likely to be "active". Running two active application servers on the same dataset can lead to corruption in the data.

2. Log in to the Controller database on the primary Controller machine and execute following command:

```
START SLAVE
```

To verify that the slave is running:

```
SHOW SLAVE STATUS \G
```

i Wait until the value for "Seconds_Behind_Master" column reaches zero. This step can be 20X faster than the time since failover event. For example, if the primary Controller machine has been down for 24 hours, it may take more than one hour for the sync process.

4. Shut down the Controller and start the Controller database on the secondary machine.

5. Log in to the Controller database for the primary Controller machine and perform following actions:

- Check that there is no database corruption on the disk or in the tables. There are third-party utilities available for this or you can use MySQL's [mysqlcheck](#).

For example, to copy the data directory on Linux:

On the Secondary Controller machine:

```
tar cvf data.bak /mnt/appdynamics2/db/data
scp user@server2:/tmp
```

On the Primary Controller machine:

```
cd /mnt/appdynamics/db/data
rm \--fr *
tar xvf /tmp
```

- Replace the data directory on the primary Controller machine with the one that was copied from the secondary Controller machine.
 - Start the Controller database on the primary Controller machine.
7. Start the Controller database on the secondary Controller machine.
 8. Reset the Controller database on the primary Controller machine.
 9. On the primary Controller machine, log into the Controller database and execute following command:

```
"RESET MASTER"
```

- If you have copied data directory from the secondary Controller to the primary Controller, also invoke following command:

```
"CHANGE MASTER"
```

10. On the secondary Controller machine, log in to the Controller database and execute following commands:

```
"CHANGE MASTER TO MASTER_HOST='PRIMARY-HOSTNAME' , MASTER_USER='controller_repl' ,
MASTER_PASSWORD='controller_repl' , MASTER_PORT=YOUR-MYSQL-PORT# ;"
"START SLAVE"
```

11. Start the Controller on the primary Controller machine.
12. Set the appserver.mode property to "passive" on the secondary Controller.

```
http://<controller-host>:<controller-port>/controller/changeappservermode?activate=false
```

The Controller should return the message:

```
200: "App Server moved to PASSIVE mode..."
```

13. Set the appserver.mode property to "active" on the primary Controller.

```
http://<controller-host>:<controller-port>/controller/changeappservermode?activate=true
```

The Controller should return the message:

```
200: "App Server started in ACTIVE mode..."
```

 **Warning:** Never set both Controllers to "active" at the same time.

[Learn More](#)

Automating HA Controller Failover and Failback

- Failover and Failback Scripts
 - Failover Script
 - To Failover
 - Failback Script
 - To Failback
- Moving Traffic between Controllers
 - Using Virtual IP (VIP)
 - Using a Load Balancer
- Modifying and Testing the Scripts
 - To Modify and Test the Scripts
- Learn More

This topic describes how to use scripts provided by AppDynamics to perform failover and failback operations for your AppDynamics Controller. It provides a simpler alternative to the manual failover and failback procedures described at [Controller HA Failover and Failback Process](#).

It assumes that you have already provisioned a primary and secondary controller in High Availability (HA) mode. For information on how to do this see [Provisioning Controllers in High Availability Mode](#). It also assumes that you also have a provisioning server, which has SSH access to both controller machines, from which to run the scripts.

Failover is the operation to perform if the primary controller experiences a failure; it switches all operations to your secondary controller on the backup machine. Failback is the operation that switches operations from the secondary controller back to the primary controller after the primary controller has been restored.

Failover and Failback Scripts

AppDynamics provides a set of scripts that you can run to perform failover and failback. Click [failover_scripts.zip](#) to download the zip file containing the scripts. Unzip all the scripts into the same directory on the provisioning server.

The scripts that you run directly are:

- failover.sh: to perform failover
- fallback.sh: to perform failback

These two scripts invoke the other scripts to perform support tasks.

These scripts do not handle switching traffic between the primary and secondary controllers. See [Moving Load Between Controllers](#) for information on how to do this.

Failover Script

This script performs the following tasks:

1. Stops traffic to the primary controller.
2. Sets the primary controller to passive mode.
3. Turn off replication.
4. Records where the primary controller last stopped replicating.
5. Stops the controller on the primary machine.
6. Makes the secondary controller the active controller.
7. Directs traffic to the secondary controller.

To Failover

- Run failover.sh from the provisioning server to failover from the primary to the secondary controller.

Fallback Script

This script performs the following tasks:

1. Stops traffic to the secondary controller
2. Sets the secondary controller to passive mode.
3. Turn off the database on the primary controller and lets it replicate from the secondary controller until the primary controller has all the data collected since the failover.
4. Sets up replication again.
5. Makes the primary controller the active controller so that it can receive traffic.
6. Directs traffic to the primary controller

To Fallback

- Run fallback.sh from the provisioning server to fallback from the secondary to the primary controller after the primary controller has been restored.

Moving Traffic between Controllers

To move traffic between controllers, choose from the common options listed below:

- [Using Virtual IP \(VIP\)](#)
- [Using a Load Balancer](#)

Based on the option you choose, make the appropriate changes to the failover and fallback scripts.

Using Virtual IP (VIP)

To use the VIP approach, assign a single IP which can be grabbed and released by either machine. During failover, the machine with the primary controller releases the VIP and the machine with the secondary controller takes it by binding it to eth1.

You implement this solution using the Linux ifup and ifdown commands.

Using a Load Balancer

To use the load balancer approach, point agent and UI traffic to a loadbalancer IP which then redirects the traffic to the active controller

During failover, you can manually direct the loadbalancer to switch traffic to the other controller.

Modifying and Testing the Scripts

To Modify and Test the Scripts

Modify the scripts as needed and then test them.

1. Edit the export entries in setup.sh with the information for your own environment.

2. Edit the failover.sh script as needed based on your failover strategy.

At the start of the script, stop traffic to the primary controller. At the end of the script, direct traffic to the secondary controller.

3. Edit the failback.sh script as needed based on your failback strategy. At the start of this script, stop traffic to the secondary controller. At the end of the script, direct traffic to the primary controller.

4. Test these scripts simulating a failover and failback to make sure everything works as expected. Then just execute failover.sh and failback.sh when needed.

Learn More

- Provisioning Controllers in High Availability Mode

Controller High Availability FAQ

- Q. What do I do when the database size between the primary and the secondary Controller does not match?
- Q. Why is there an error 1045?
- Learn More

This topic lists frequently asked questions about Controller high availability configurations.

Q. What do I do when the database size between the primary and the secondary Controller does not match?

To troubleshoot this problem, gather information about the system and open an [AppDynamics support ticket](#). Follow these steps:

1. For each Controller, locate the <Controller-Installation-Directory>/db/db.cnf file and make a copy of each.

Send these files to the AppDynamics Support Team.

2. Capture the secondary database status.

- Log in to the secondary Controller database.
- Execute the following command on secondary Controller database and save the results:

```
SHOW SLAVE STATUS\G
```

Send these results to the AppDynamics Support Team.

3. Capture the disk usage information on each machine.

- From a console, navigate to the <Controller-Installation-Directory>/db/data directory.
- Execute the following command and save the output:

For Linux:

```
du -h
```

For Windows:

```
du -l
```

Send these results to AppDynamics Support Team.

i You can download a Windows disk usage tool from [sysinternals](#) site. You can also use tools such as [WinDirStat](#) to get the disk usage statistics.

4. Capture the file system information on each machine.

- From a console, navigate to the <Controller-Installation-Directory>/db/data/controller directory.
- Execute following command and save the output:

For Linux:

```
ls -l
```

For Windows:

```
dir
```

- Execute following command and save the output:

For Linux:

```
ls -s |sort -n
```

For Windows:

```
dir /s /os
```

Send these results to AppDynamics Support Team.

Q. Why is there an error 1045?

MySQL may return a "cannot authenticate" error code 1045 if the domain name is used for the PRIMARY-HOSTNAME and SECONDARY-HOSTNAME instead of the IP address. Use the IP address.

[Learn More](#)

Provisioning Controllers in High Availability Mode

- Required Machines
- Required Scripts
- Required AppDynamics License and Account
 - To get license keys
 - To get a monitor account
- Set Up the Machines
 - To set up the HA machines and the provisioning server
- [Learn More](#)

This topic describes how to provision a controller in high availability (HA) mode.

For successful operations of a large scale Controller AppDynamics strongly recommends that your Controller be allowed to upload its performance data to the AppDynamics monitoring system.

Required Machines

The Controller setup for HA requires three machines:

- Primary HA Controller
- Secondary HA Controller
- Provisioning Server

Required Scripts

The setup uses the following scripts:

- external_provision_controller.sh
- controller_call.sh

- external_Install_controller.sh
- keyval_21.template

Click [provision_controller.zip](#) to download the zip file containing these scripts.
Unzip them in any directory on the provisioning server.

You will edit only external_provision_controller.sh. This script invokes the other three scripts.

Required AppDynamics License and Account

To get license keys

1. Send AppDynamics Support a request for a license key for your primary and secondary controller hosts. Include with your request the MAC address of the each machine. See [AppDynamics Support](#) for information on how to contact support.

AppDynamics will generate a license key for each of your controller machines and email them to you.

2. Name the license keys in a logical way, such as "primary_license.lic" and "secondary_license.lic" and save them in a directory on the provisioning server. Make a note of the full path of these directories, as you will need to enter them as primary_license and secondary_license when you edit the external_provision_controller.sh script.

To get a monitor account

1. Send AppDynamics Support a request for an account on SaaS Monitor.
AppDynamics will send you an account name and password for the SaaS Monitor.

2. Save these values.

You will need to enter them as monitor_acctname and monitor_pass when you edit the external_provision_controller.sh script.

Set Up the Machines

To set up the HA machines and the provisioning server

1. Allow port 80 on both HA machines to be an open outbound port through which the Controllers will send data to the AppDynamics monitoring system.

2. Set up a third machine to be the provisioning server. The machine resources can be very small, and the machine can be virtual.

The provisioning server must have SSH access with an SSH key to both HA machines.

The provisioning server must have its SSH key set up to automatically ssh into the HA machines without prompting. See [Setting Up an SSH Key for Controller Provisioning](#) if you need detailed instructions on how to do this.

3. Confirm that all three machines (provisioning server, Primary HA Controller, and Secondary HA Controller) have the following Linux commands installed and in the path:

sed, awk, scp, ssh, ls, mkdir, unzip, java (64-bit)

4. Download the latest Machine Agent from the AppDynamics download server at <http://download.appdynamics.com/> and put it in a directory of your choosing on the provisioning server. You will need to enter the full path of the machine agent installer when you edit the external_provision_controller.sh script.

5. Download the latest Controller installer from the AppDynamics download server at <http://download.appdynamics.com/> and put it in a directory of your choosing on the provisioning server. You will need to enter the full path of the Controller installer when you edit the external_provision_controller.sh script.

6. Test the three machines to confirm that you can SSH from the provisioning server to either Controller machine without being prompted.

7. Open the external_provision_controller.sh script that you stored on the provisioning server and change the variables based on the machine IP numbers, machine hostnames, directories, users, etc.

8. Run the ./external_provision_controller.sh script to install the Controller in HA mode on the provisioning server.

9. Verify that the Machine Agent on the provisioning server will restart after any future reboot by adding the following line in /etc/rc.local:

```
nohup <path_to_64bit_java>/java -jar machineagent.jar &
```

Learn More

- Setting Up an SSH Key for Controller Provisioning
- AppDynamics Support

Setting Up an SSH Key for Controller Provisioning

- To set up SSH Key Pairs Using DSA
- To set up SSH Key Pairs Using RSA

You can set up SSH (Secure Shell) with public/private key pairs so that you do not have to type the password each time that you ssh into the Controller machines. This allows scripts and automation processes to access the necessary systems easily. You can generate DSA or, if you want stronger encryption, RSA keys.

To set up SSH Key Pairs Using DSA

1. Run the ssh command that sets up the key pair:

```
% ssh-keygen -t dsa
```

2 To the prompt:

```
Generating public/private dsa key pair.  
Enter file in which to save the key (~/.ssh/id_dsa):
```

just type RETURN.

3. To the prompt:

```
Enter passphrase (empty for no passphrase):
```

just type RETURN.

4. To the prompt:

```
Enter same passphrase again:
```

just type RETURN.

You should see the following information:

```
Your identification has been saved in ~/.ssh/id_dsa  
Your public key has been saved in ~/.ssh/id_dsa.pub  
The key fingerprint is: <Some really long string>
```

If SSH continues to prompt you for your password, verify your permissions in your remote .ssh directory. It should have only your own read/write/access permission (octal 700):

```
% chmod 700 ~/.ssh
```

5. Open the local ~/.ssh/id_dsa.pub file and paste its contents into the ~/.ssh/authorized_keys file on the remote host.

6. Update the permissions on the authorized_keys file on the remote host as follows:

```
% chmod 600 \~/.ssh/authorized_keys
```

To set up SSH Key Pairs Using RSA

Run the ssh command that sets up the key pair:

```
% ssh-keygen \-t rsa
```

The generated files will be named id_rsa and id_rsa.pub, instead of id_dsa and id_dsa.pub.

Otherwise, the remaining steps are identical to those beginning with step 2 in [To set up SSH Key Pairs Using DSA](#).

Controller Data and Backups

This section discusses Controller data administration and backups.

Controller Data Backup and Restore

- [Controller Backups](#)
 - Best Practices for Backup
 - Regular Hot Backups
 - Using a Hot Backup for the New Physical Server
 - System Settings and Tools
 - Why AppDynamics recommends binary backup
 - Using XtraBackup
 - To backup Controller data using XtraBackup
 - To restore Controller data using XtraBackup
 - Backup and Restore Procedures when Running in High Availability (HA) Mode
 - To backup when using High Availability mode
 - To restore when using HA mode
- [Backing Up Metadata Only](#)
- [Learn More](#)

This topic describes the backup and restore procedure for the Controller database.

Controller Backups

The AppDynamics Controller uses MySQL to store information about following components:

- Design of your applications (all meta-data about business transactions, tiers, policies etc.)
- History of the performance of your applications (metric data)
- Transaction Snapshot Data and Events
- History of incidents that occurred (both resolved and unresolved incidents are stored)

Best Practices for Backup

 **Important:** AppDynamics strongly recommends that you **perform routine data backups** as a part of disaster recovery preparedness.

There are other situations when you should do a backup. Back up the Controller under the following conditions:

- During an upgrade
- When migrating your installation from one server to another
- If the Controller disk space is running low

Regular Hot Backups

A "hot" backup of Controller data is recommended when you do not want to stop the Controller.

 **Important:** AppDynamics strongly recommends that you **perform nightly backups** of your data. If you cannot do nightly backups

do backups at least once a week.

The AppDynamics Controller uses the MySQL default storage engine, InnoDB. Therefore, you cannot copy the data files directly while the Controller is running. Copying data files works only if the Controller is stopped. For details see [Start or Stop the Controller](#).

 **Warning:** Do not use tools like **mysqldump** because these tools convert the data into text format and are time-consuming, especially if you have gigabytes of data in your system.

Using a Hot Backup for the New Physical Server

You can perform a hot or a cold backup, but it is important to get the backup of the data directory, located in <Controller_Installation_Directory>/db.

However, we recommend a cold backup of the data. Hot backup will not bring the Controller down for a long time, but you will still lose the data when you migrate. This is because hot backup will only have the data from the time that the hot backup started, and not when the backup was completed.

System Settings and Tools

Ensure that <controller_install_dir>/db/bin is in your \$PATH variable and it should be ahead of any other path that has another MySQL. (Typically Linux has a standard instance of MySQL installed, which will conflict with the Controller's path.)

There are chances of data loss since your last backup and so more frequent backups are recommended.

It is recommended that you use only those tools that perform binary copies of the data.

Why AppDynamics recommends binary backup

A binary data backup saves a copy of your directory data that you can use if the database files later become corrupted or deleted. For details see [Binary Backup](#).

- You can use the following backup tools for Linux:
 - mylvmbackup
 - Xtrabackup
 - INNODB Hot Backup
- You can use the following backup tool for Windows:
 - Zmanda Recovery Manager for MySQL

Additionally, you can also use the ZFS snapshot method to perform the backup for Controller database.

For details on the ZFS snapshot method, see: [Using ZFS methods for data backup](#).

Using XtraBackup

To backup Controller data using XtraBackup

You can use the Percona xtraBackup tool for the backup, available at: <http://www.percona.com>.

Use version 1.6.4-314 or greater. You can either use the Linux binary or the RPM install.

- Add the <xtrabackup_dir>/bin directory path to the \$PATH variable.
- Use the following two commands for each backup (scroll right to see the whole command line):

```
innobackupex-1.6.4 --user=root--password=singcontroller --defaults-file=$INSTALL_DIR/db/db.cnf <backup-dir> --no-lock

innobackupex-1.6.4 --user=root--password=singcontroller --use-memory=1GB
--defaults-file=/${INSTALL_DIR}/db/db.cnf --apply-log <backup-dir-from above>/<some-date-dir>
```

The first command creates a directory under your <Backup_Directory> using the time-stamp.

Use the full directory name as the final argument to the second command.

You must run both commands for a successful backup.

To restore Controller data using XtraBackup

1. Shut down the MySQL database.
2. Copy back the files in the data directory located at <Controller_Installation_Directory>/db.
3. Restart MySQL.

Backup and Restore Procedures when Running in High Availability (HA) Mode

After you have configured the Controller in [high availability mode](#), use the following data backups and restore procedures.

You can perform a hot backup on the secondary machine in an HA environment using mylvmbackup or Xtrabackup.

To backup when using High Availability mode

DO NOT backup the Controller database. Use the following steps to ensure less impact on your running system.

1. Log into the Controller database on the secondary machine.
2. For every backup period (which should be nightly), shut down the Controller database on the secondary machine.
3. Use any incremental copy tool to copy the Controller database to a safe location.
4. Start the Controller database.

To restore when using HA mode

Use the following steps when your primary MySQL server suffers a data loss, such as might happen after you switch your operations to the secondary MySQL:

1. Shutdown the Controller database on primary machine, after it is repaired.
2. Copy the data from the secondary machine to the primary machine.
3. Start the primary machine. The primary machine should now start replicating any data that it has lost since the time the last backup performed on the secondary server.
4. Verify that the primary and secondary machines are in sync.
5. Switch back to using the primary machine as your Controller.

Backing Up Metadata Only

You can perform a minimal backup that backs up only metadata. This is essentially a MYSQL dump. Click [MetadataBackupCommand.txt](#) to download the command script to back up the metadata.

Learn More

- [Manage Controller High Availability](#)
- [App Agent for Java Configuration Properties](#)
- [App Agent for .NET Configuration Properties](#)

Controller Database Scripts

Database Scripts

The scripts to start or stop the Controller are located in the <Controller_Installation_Directory>/bin directory.

Action	Commands for Windows	Commands for Linux
Log into the Controller database	controller.bat login-db	./controller.sh login-db
Start the Controller database	controller.bat start-db	./controller.sh start-db
Stop Controller database	controller.bat stop-db	./controller.sh stop-db

Reset Controller database	controller.bat reset-db	./controller.sh reset-db
---------------------------	-------------------------	--------------------------

Learn More

Controller Data Storage

- The Controller Data Directory
- Moving the Controller Data Directory
 - To relocate the Controller data directory

This topic describes moving the Controller data storage directory.

The Controller Data Directory

By default, the Controller's data directory is located at: <Controller_Installation_Directory>/db.

Moving the Controller Data Directory

You may decide to move the data directory to a new location under following conditions:

- When you want to store the Controller data on a SAN in order to get higher I/O performance and redundancy.
- If there is not enough disk space available during Controller installation.

 **Note:** If you are using symlinks, you must create the symlink outside of the root Controller install directory and move the data directory to the new volume after you install the Controller.

 **Warning:** Do not mount a file system on <Controller_Installation_Directory>/db/data. During Controller upgrade, the installer moves the data directory to data_orig. Upgrade will fail if the installer cannot complete this move.

To relocate the Controller data directory

1. Stop the Controller and its database. See [Start or Stop the Controller](#).
2. Modify following properties in the <Controller_Installation_Directory>/db/db.cnf file to point to the new location of the data directory.

```
datadir
tmpdir
log
slow_query_log_file
```

3. Copy (or move) the existing data directory <Controller_Installation_Directory>/db> to the new location.

For Linux:

```
>cd <Controller_Installation_Directory>/db/
>cp data <new-location>
```

4. Start the Controller. See [Start or Stop the Controller](#).
5. Check the database.log and server.log for any database connection related errors.

Controller Disk Space and the Database

- Disk Space Considerations

- Disk Space Warnings
- Managing Disk Space
- Automatic Shutdown When Disk Space is Low
 - To disable automatic shutdown for a running Controller
- Learn More

This topic discusses best practices for managing disk space for the MySQL database used by the Controller.

Disk Space Considerations

If disk space runs out the Controller will not function and data may become corrupted.

Disk Space Warnings

When disk space starts getting low, the AppDynamics UI displays a Controller Disk Space Low alert. The warning is triggered based on the Controller profile. See [Controller System Requirements](#).

AppDynamics also logs error in the server.log.

Managing Disk Space

If the disk space is low, you need to reduce the size of the Controller database.

To manage how much disk space the Controller database uses, you change the amount of data retained in the Controller database. See [Database Size, Data Retention, and Metric Resolution](#).

Automatic Shutdown When Disk Space is Low

Starting with Controller release 3.4.2, if the space on the disk the Controller is running on falls below 1 GB, the Controller automatically shuts down to avoid DB corruption.

To disable automatic shutdown for a running Controller

1. Shut down the Controller. See [Start or Stop the Controller](#).
2. Open the <Controller-Installation-Directory>/appserver/domains/domain1/config/domain.xml file for edit.
3. Add the following:

```
<jvm-options>-Dignore.disk.space=true</jvm-options>
```

4. Save the file.
5. Restart the Controller.

Learn More

- [Database Size, Data Retention, and Metric Resolution](#)
- [Controller System Requirements](#)
- [Start or Stop the Controller](#)

Database Size, Data Retention, and Metric Resolution

- [Information Stored by the Controller](#)
- [The Controller Database and Data Retention Settings](#)
 - To change the Controller data retention settings
 - Troubleshooting Controller Database Growth Issues
- [Default Data Retention Policy for Metrics](#)
 - Modifying Default Data Retention Policies
 - To change the data retention period for metrics
- [Learn More](#)

This topic explains the default data retention settings for data stored by the AppDynamics Controller, and how often AppDynamics resolves metric data.

Information Stored by the Controller

The Controller holds information about following components:

- The design of your applications (all metadata about business transactions, tiers, policies, etc.)
- Several years of historical data about application performance (metric data)
- Several weeks of transaction snapshot data and events
- History of incidents that occurred (whether they were resolved or not)

The Controller uses MySQL to store this data.

The Controller Database and Data Retention Settings

You can change the amount of data you retain in the Controller database by changing the retention period. You can also change the data retention policy for snapshots and hourly metric data.

You should consider changing the data retention settings when you see an increase in the size of the Controller database. See [Controller Disk Space and the Database](#).

You can also purge old data that is no longer needed by changing the retention settings.

To change the Controller data retention settings

1. Log in to the Controller administration console using the root account password. See [Access the Administration Console](#).

2. Click **Controller Settings**.

3. Change the data retention period by configuring the following parameters:

- `events.retention.period`
The default value is 336 hours, which is 2 weeks.
- `snapshots.retention.period`
The default value is 336 hours, which is 2 weeks.

The size of the database should drop between 30-60 minutes after you make this change. You should see fewer eventdata_ files. If you do not see this result [restart the Controller](#).

Troubleshooting Controller Database Growth Issues

If changing the data retention settings does not restrict the database growth, send a directory listing of your data directory, in particular the eventdata_ files, to the [AppDynamics Support Team](#).

Use the following command to get the directory listing of the data directory:

```
ls <install_dir>/db/data/controller eventdata_detail* | sort
```

There should be one file per hour of data.

Default Data Retention Policy for Metrics

AppDynamics triggers data retention for metrics after the data has existed for 60 minutes.

Over time, AppDynamics resolves the data in the following manner:

- Initially, for up to 4 hours, all data is captured at full resolution, minute-by-minute, for each metric.
- After 4 hours, the Controller resolves the data every 10 minutes.
- After 48 hours, the Controller resolves the data every 60 minutes.
- Finally, daily averages are used as resolution.

AppDynamics resolves the metrics using averages on the data and discards the remaining metric data points.

The data retention policy aggregates the minute-by-minute data into ten-minute resolutions after 4 hours. After 48 hours, the ten-minute data is further aggregated to 60-minute resolutions.

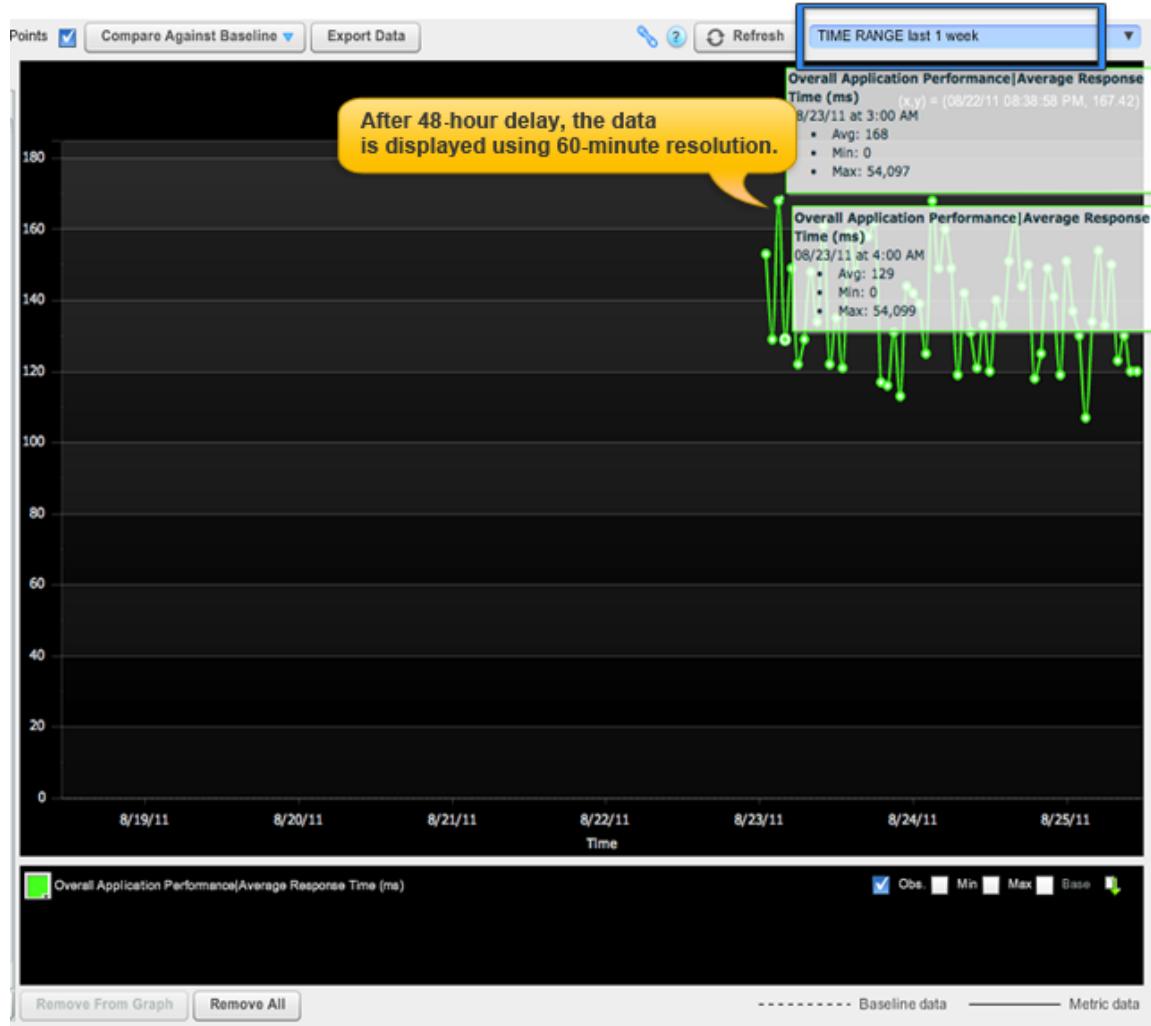
For example, if you select a four-hour time-range between 10.00 a.m. to 12.00 noon for a data which is 4 hours old, your data points will be determined by the following resolution: 10.00 a.m.-10.09 a.m., 11.00 a.m.-11.09 a.m., 12.00 noon - 12.09 p.m.

Important: In the above example, if you select a custom time range of 10.00 a.m. to 12.01 p.m., it is likely that the data might not capture all the data points.

The following screenshot shows how the data retention policy determines the metric resolution. AppDynamics displays metric data after 4 hours, and any two data points are separated by a 10-minute time interval.



After 48 hours, AppDynamics displays the data using the 60-minute resolution.



To get the required visibility, select only those custom time ranges that follow the "data-resolution" rule.

Modifying Default Data Retention Policies

You can modify the default data retention policies. However, due to performance reasons, AppDynamics strongly recommends that you exercise caution if you plan to increase the default limits.

The data retention policies are determined by three properties that are accessible from the Controller administration console.

Property name	About the property	Default	Allowed Values
metrics.min.retention.period <i>specified in hours</i>	This property determines the number of hours your minute-by-minute data will be retained.	4 hours	Allowed value is between 1 hour to 1 week (168 hours).
metrics.ten.min.retention.period <i>specified in hours</i>	This property determines the number of hours your ten-minute data will be retained.	48 hours	Allowed value is between 2 hours to 3 weeks (504 hours).
metrics.retention.period <i>specified in days</i>	This property determines the number of days the hour-by-hour metrics will be retained in the Controller database.	365 days	Allowed value is between 30 days to 2 years (730 days).

The value for metrics.min.retention.period should always be less than the value for metrics.ten.min.retention.period.

The value for metrics.ten.min.retention.period should always be less than the value for metrics.retention.period.

⚠ Warning: If you specify the value for minute data (metrics.min.retention.period) either less than 36 hours or greater than 36 hours (or vice versa) the Controller requires special repartitioning and therefore you will lose all the minute data for the first time after you save the settings.

To change the data retention period for metrics

1. Log in to the Controller administration console.

```
http://<controller-installer-host>:<port>/controller/admin.html
```

Use the root account password to access the Admin console. The root password is "changeme" and is the same even if the Controller is installed in single or multi-tenant mode.

2. Click **Controller Settings**.

3. Change the data retention period by configuring the following parameters. Save each property individually.

- metrics.min.retention.period
The default value is 4 hours.
- metrics.ten.min.retention.period
The default value is 48 hours.
- metrics.retention.period
The default value is 365 days.

After you modify these values, the time range selector in the UI that displays "resolution X minute" should change.

For example, if the metrics.min.retention.period property is changed to 3, in the AppDynamics UI all time ranges <= 3 hours should have resolution of 1 minute. If the metrics.ten.min.retention property is changed to 168, all time ranges <= 168 hours (1 week) and greater or equal to min.retention should have resolution of 10 minutes.

Learn More

- Controller Disk Space and the Database
- Controller Database Scripts
- Infrastructure Metrics

Controller Version

To determine the version of the Controller

1. Launch the AppDynamics UI in a browser.

2. Click **Help -> About AppDynamics**.

AppDynamics displays the version number for the Controller.

Learn More

- Access the AppDynamics UI

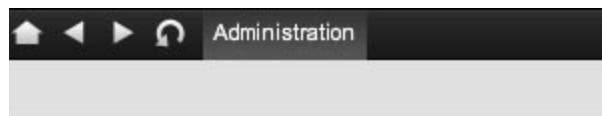
Remove Unused Nodes

- To remove dead nodes
- Learn More

To remove dead nodes

In a typical environment, the nodes are recycled and new nodes get generated. AppDynamics strongly recommends that you remove the dead nodes both from the AppDynamics user interface (UI) as well as from the system.

1. Log in to the Controller administration console.
2. Go to "Controller Settings" section to see the advanced properties for the Controller.



AppDynamics Administration

Accounts

Create and Manage Accounts

Controller Settings

Configure the Controller

3. Set the retention and deletion properties, based on the requirements for your environment. AppDynamics recommends that you set the permanent deletion period at least an hour more than the retention period.

- **node.permanent.deletion.period:** Time (in hours) after which a node that has lost contact with the Controller is deleted permanently from the system.
- **node.retention.period:** Time (in hours) after which a node that has lost contact with the Controller is deleted. In this case, the AppDynamics UI will not display the node, however the system will continue to retain it.

Learn More

- [Access the Administration Console](#)

Controller Info Configuration for Agents

- [Learn More](#)

The Controller gets information from app agents and machine agents about how the agent connects to the Controller from the agent's controller-info.xml file.

For app agents, the controller-info.xml file is in the <Agent_Installation_Directory>/conf directory.

For machine agents, the controller-info.xml file is in the <Machine_Agent_Installation_Directory>/conf directory.

The properties set in the controller-info.xml file include:

- controller-host
- controller-port
- controller-ssl-enabled
- application-name
- tier-name
- node-name
- agent-runtime-dir
- enable-orchestration
- account-name
- account-access-key
- force-agent-registration

System properties in the app server startup script override the settings in controller-info-xml.

For information about the controller-info settings, as well as their corresponding system properties, see [App Agent for Java](#)

Configuration Properties, App Agent for .NET Configuration Properties and Machine Agent Configuration Properties.

Learn More

- App Agent for Java Configuration Properties
- App Agent for .NET Configuration Properties
- Machine Agent Configuration Properties

Controller Infrastructure

- Embedded Glassfish Server
- MySQL Database

The Controller uses an embedded Glassfish application server and an embedded MySQL database.

Embedded Glassfish Server

The Glassfish server version for the 3.6 Controller is 2.1.1.

The Glassfish server version for the 3.7 Controller is 3.1.2.2.

The Glassfish server is installed in <AppDynamics_install_dir>/appserver.

MySQL Database

The AppDynamics Controller uses MySQL to store information about:

- Design of your applications (all meta-data about business transactions, tiers, policies etc.)
- History of the performance of your applications (metric data)
- Transaction Snapshot Data and Events
- History of incidents that occurred (both resolved and unresolved incidents are stored)

The MySQL version for the 3.6 and 3.7 Controllers is MySQL 5.5.16.

The MySQL database is installed in <AppDynamics_install_dir>/db.

Controller Audit Log

- What is Audited
- Structure of Audit Log

An audit log for the Controller is available in the controller_audit table for monitoring logins and configuration.

Starting with release 3.6 audit logging is enabled by default.

What is Audited

The following actions are logged during user authentication:

- LOGIN*
- LOGIN_FAILED*

The following entities are audited for creation, modification and removal:

- APPLICATION_COMPONENT
- APPLICATION_COMPONENT_NODE*
- USER
- GROUP
- ACCOUNT*
- ACCOUNT_ROLE
- WORKFLOW*
- GLOBAL_CONFIGURATION
- NOTIFICATION_CONFIG
- DASHBOARD
- JMX_CONFIG*
- BUSINESS_TRANSACTION*

- BUSINESS_TRANSACTION_GROUP*
- CUSTOM_EXIT_POINT_DEFINITION*
- CUSTOM_MATCH_POINT_DEFINITION
- TRANSACTION_MATCH_POINT_CONFIG
- BACKEND_DISCOVERY_CONFIG
- ERROR_CONFIGURATION
- DOT_NET_ERROR_CONFIGURATION
- CALL_GRAPH_CONFIGURATION
- SQL_DATA_GATHERER_CONFIG
- MEMORY_CONFIGURATION
- EUM_CONFIGURATION
- AGENT_CONFIGURATION
- APPLICATION_DIAGNOSTIC_DATA
- POJO_DATA_GATHERER_CONFIG
- HTTP_REQUEST_DATA_GATHERER_CONFIG
- POLICY*
- RULE*

When a user or agent modifies one of these objects, one of the following actions is logged:

- OBJECT_CREATED
- OBJECT_UPDATED
- OBJECT_DELETED

Structure of Audit Log

The structure of the controller_audit table is:

column	value
ts_ms	creation time in milliseconds
account_name	user account of user performing the action
account_id	account id of user performing the action
user_name	user name of user performing the action
user_id	user id of user performing the action
object_name	entity type of modified object
object_id	id of modified object
action	audited action, one of: OBJECT_CREATED, OBJECT_UPDATED, OBJECT_DELETED, LOGIN, LOGIN_FAILED

Configure Controller VIP

If you want to use a virtual IP (VIP) for the Controller, perform the following edits in the <Controller-Installation-Directory>/appserver/domains/domain1/config/domain.xml file:

1. Set the appdynamics.com.hostname element to the internal IP address that you used when you installed the Controller.
2. Set the appdynamics.com.server.hostname to the virtual IP address.

Controller Error Messages

- Name Not Specified Error
 - Symptom
 - Response

This topic describes error messages that might be seen in the Controller logs.

Name Not Specified Error

Symptom

The following WARN Message is seen in the log even with a good Application Name, Tier Name and Node name.

```
[Thread-0] 21 Feb 2013 20:41:29,747 INFO ConfigurationChannel - Sending  
Registration request with: Application Name [Invoicing], Tier Name  
[tsc-349756], Node Name [wlsstress4], Host Name [wlsstress4.ce.fedex.com] Node  
Unique Local ID [wlsstress4], Version [Server Agent v3.6.1.0 GA  
#2012-12-11_14-59-42 r2346f2ff54e68a0be000f975f9d0ebc828f8b660 42]  
[Thread-0] 21 Feb 2013 20:41:30,668 WARN ConfigurationChannel -  
ResponseReadException creating Response Wrapper [pb], :  
com.singularity.ee.rest.c: Error in controller in processing binary request  
AppAgent Config Data - name is not specified.
```

Response

Carefully examine the arguments in the JVM startup script. This message can be seen when there is an extraneous | -D | argument. Remove that argument and it should work.

```
os.version=2.6.9-89.0.9.ELsmp  
[Thread-0] 21 Feb 2013 20:41:27,773 INFO AgentKernel - JVM Args :  
-Dsbm.home=/opt/fedex/tsc/SBM763 | -server | -Xms256m | -Xmx1024m |  
-XX:PermSize=128m | -XX:MaxPermSize=256m | -Dsbm.server.name=ejbnode_stress4 |  
-Dsbm.server.type=ejb | -Dweblogic.Name=ejbnode_stress4 |  
-Dbea.home=/opt/weblogic/wl10.3.0 |  
-Dweblogic.management.server=http://wlsstress4.ce.fedex.com:7590 |  
-Djava.security.policy=/opt/fedex/tsc/domains/tscdomain/sbm.policy |  
-Djava.io.tmpdir=/var/tmp/tsc | -Dweblogic.ProductionModeEnabled=false |  
-Dweblogic.system.BootIdentityFile=/opt/fedex/tsc/domains/tscdomain/security/boot  
| -Dweblogic.system.nativeIO.enable=true * | -Dsun.java.launcher=SUN_STANDARD |  
-javaagent:/opt/wily/appdynamics/appagent/javaagent.jar |  
-Dappdynamics.http.proxyHost=internet.proxy.fedex.com |  
-Dappdynamics.http.proxyPort=3128 |  
-Dappdynamics.agent.applicationName=Invoicing |  
-Dappdynamics.agent.tierName=tsc-349756 |  
-Dappdynamics.agent.nodeName=wlsstress4 | -Dsun.java.launcher=SUN_STANDARD |
```

Tuning for Large Enterprise Environments

- Linux Settings
- File System and RAID Recommendations
- Glassfish Configuration
 - To configure Glassfish
 - Glassfish Settings
- MySQL Configuration
 - To configure MySQL
 - MySQL Settings
- Controller Configuration
 - To Change the Controller Settings
- Additional Setup for Environments with More than 500 Nodes
 - To create a second HTTP listener
 - Terminating SSL Traffic at a Reverse Proxy
- Learn More

This topic contains recommendations for how to configure AppDynamics Controllers and their underlying platforms for Controllers that are monitoring 250 or more nodes.

Linux Settings

Use the following settings for the operating system:

- Use the Deadline scheduler.
- Set swappiness to 0.
- Set the open file limit to 819200 or greater.
- Set the per-process open file limit for soft and hard limits to 819200 or greater.
- Allow Web server to retry longer during stalls by setting higher TCP timeouts.

File System and RAID Recommendations

- Recommended file system: XFS.
- Recommended RAID version: There are only minor performance differences between RAID 5, 6, and 10. The reasons to use one over another are purely for level of safety and space requirements. The more disks you have in any RAID configuration, the better the performance, since you have more stripes to work concurrently in doing disk I/O.
- Recommended RAID Configuration:
 - Use a RAID Controller with a Battery Backup Unit (BBU) to allow DB to use O_DIRECT sync mode for faster disk writes. Never set O_DIRECT without a BBU.
 - Enable WriteBack caching mode on the RAID Controller.
 - Disable individual hard drive from using onboard cache. All caching is to be done by RAID Controller.
- On Board disks are the easiest option to assure performance.
- When you use remote disks (ie. SAN, NAS, etc.) the configuration must be able to support very heavy disk I/O, with very low latency. Requirements are similar to any production database that you run.
- LVM partitions can be useful especially for doing quick hot backups with LVM snapshots.

Glassfish Configuration

The Controller uses Glassfish version 3.1.2.2 as its embedded application server. For better scalability and performance, you should tune Glassfish with settings based on your final hardware sizing.

To configure Glassfish

1. Shut down the controller.
2. Edit the \$install/appserver/domains/domain1/config/domain.xml file with the settings described in [Glassfish Settings](#).
3. Restart the Controller.

Glassfish Settings

- Set the thread count (XX) to a value of 12 X # of CPU cores.

```
<request-processing header-buffer-length-in-bytes="8192" initial-thread-count="16"
request-timeout-in-seconds="300" thread-count="XX" thread-increment="1"/>?
```

- Set the depth of the connection pool queue to 32K allowing for connections to queue up instead of being dropped during peak load bursts.

```
<connection-pool max-pending-count="32768" queue-size-in-bytes="32768"
receive-buffer-size-in-bytes="32768" send-buffer-size-in-bytes="32768"/>
```

- Set the Heap size to approximately 20 to 40% of the RAM size on your machine: higher Heap size for lower RAM sizes, lower Heap for high-end RAM sizes. Set Xmn to 1/3 of the Xmx setting.

```
<jvm-options>-Xmx30g</jvm-options>
<jvm-options>-Xms30g</jvm-options>
<jvm-options>-Xmn10g</jvm-options>
```

- Replace the garbage collection settings with these specially tuned settings:

```

<jvm-options>-XX:+UseConcMarkSweepGC</jvm-options>
<jvm-options>-XX:+UseParNewGC</jvm-options>
<jvm-options>-XX:+ScavengeBeforeFullGC</jvm-options>
<jvm-options>-XX:TargetSurvivorRatio=80</jvm-options>
<jvm-options>-XX:SurvivorRatio=6</jvm-options>
<jvm-options>-XX:+UseBiasedLocking</jvm-options>
<jvm-options>-XX:MaxTenuringThreshold=15</jvm-options>
<jvm-options>-XX:ParallelGCThreads=16</jvm-options>
<jvm-options>-XX:+OptimizeStringConcat</jvm-options>
<jvm-options>-XX:+UseStringCache</jvm-options>
<jvm-options>-XX:MaxPermSize=256m</jvm-options>
<jvm-options>-XX:+UseCompressedOops</jvm-options>
<jvm-options>-XX:+UseCMSInitiatingOccupancyOnly</jvm-options>
<jvm-options>-XX:CMSInitiatingOccupancyFraction=95</jvm-options>

```

MySQL Configuration

The Controller uses MySQL as its the database server. You should tune MySQL for scalability and better performance.

To configure MySQL

1. Shut down the controller.
2. Edit the \$install/db/db.cnf file with the settings described in [MySQL Settings](#). If the name in the name/value pair already exists in your configuration, just replace the current value with the recommended value. If your current configuration does not include the name/value pair, then add the pair to the file.
3. Restart the Controller.

MySQL Settings

```

thread_cache_size=120
table_definition_cache=500
open_files_limit=40960
innodb_open_files=3000
table_open_cache=4000
lock_wait_timeout=300
query_cache_size=0
long_query_time=4
innodb_flush_method=O_DIRECT # ONLY DO IF YOU HAVE BATTERY BACKED RAID !
innodb_buffer_pool_size= 11264M # This should be based on available RAM. This value should be
approximately 40 to 60% of the RAM size.

```

Controller Configuration

You need to adjust some Controller settings for configurations that handle large amounts of traffic. This includes increasing the events, snapshots, and buffer sizes, increasing the read and write thread counts, decreasing the node retention and node permanent deletion periods, etc.

To Change the Controller Settings

1. Make sure controller is running.
2. Log into the Controller Administration console at <host>:<port>/controller/admin.html using the root password. See [Access the Administration Console](#).
3. In the Administration Console select Controller Settings.
4. Modify the each of the following settings, clicking **Save** to save each update.
 - disable.historic.transactional.flow = true
Allows dashboards to draw quickly by using cache values from the cache setting above. For large deployments this should

- always be set this to true to avoid timeout errors.
- business.transaction.retention.period = 48
- events.buffer.size = 50
- snapshots.buffer.size = 200
- metrics.buffer.size = 600
- node.permanent.deletion.period = 168
- node.retention.period = 6
- read.thread.count = 3
Number of threads to use simultaneously to do READS from the database. Set this to 20% of the number of CPU cores but not greater than 4.
- write.thread.count = 4
Number of threads to use simultaneously to load data into the database. Set this to the same value as read.thread.count but not greater than 4.

Additional Setup for Environments with More than 500 Nodes

If your environment consists of more than 500 nodes, AppDynamics recommends that you create a second http-listener.

If you are using SSL connections for agents, you need a solution to convert the SSL into HTTP traffic.

To create a second HTTP listener

Use the original HTTP-listener (from the installer) for agents to connect to the Controller. Create a second listener for GUI users, so that they are not affected by traffic from the agents.

To create the second listener:

1. edit the domain.xml file. There are already two listeners; look for "http-listener" tags. One of them is being used by your agents.
Find the other one, and use it as your GUI listener.
2. Change the port number and remove the SSL line if needed. The GUI listener should look similar to the following example:

```
<http-listener acceptor-threads="1" address="0.0.0.0" blocking-enabled="false"
default-virtual-server="server" enabled="true" family="inet" id="http-listener-1" port="8080"
security-enabled="false" server-name="" xpowered-by="true">
<property name="proxiedProtocols" value="ws/tcp"/>
</http-listener>
```

Terminating SSL Traffic at a Reverse Proxy

If you want to terminate SSL traffic at reverse proxy, such as Nginx or Apache, contact AppDynamics Support for help with configuring SSL environments.

Learn More

- [Controller System Requirements](#)
- [Measuring Disk Performance Using a Script](#)

Configure Authentication, User Permissions and Integrations

You can configure AppDynamics Controller user authentication and permissions in the Administration setup menu.

Information on Configuring Authentication, Permissions, and Integrations

See [Configure Authentication Provider](#) for information on enabling an authentication provider (external or AppDynamics).

See [Configure Authentication Using SAML](#) if you are using SAML to authenticate users.

See [Configure Authentication Using LDAP](#) if you are using LDAP to authenticate users.

See [Configure Users](#) if you are using AppDynamics to authenticate users.

See [Configure Groups](#) to create groups to assign roles to multiple users whom you want to have identical privileges.

See [Configure Roles](#) to configure user permissions, for example which applications or custom dashboards a user can view, delete, or edit and which types of configurations a user can perform.

AppDynamics provides a set of predefined roles that you cannot modify. To create custom roles that assign permissions according to your organization's needs, see [Configure Custom Roles](#).

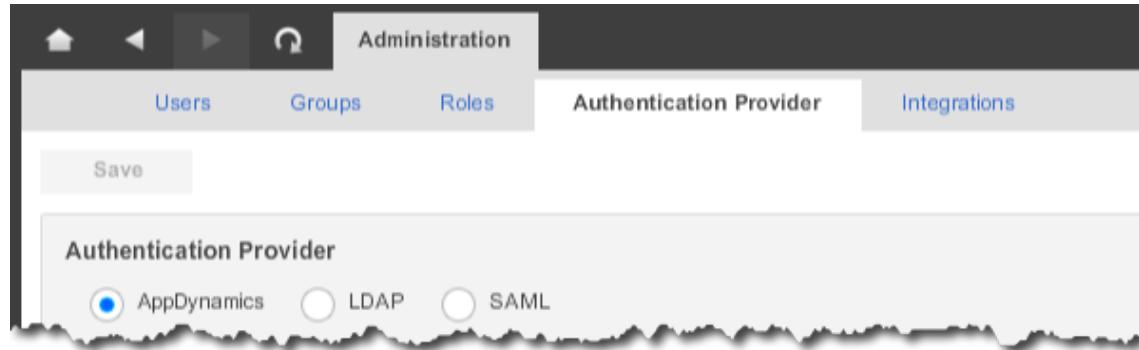
See [Configure Integrations](#) to enable and configure integrations with external products.

To access the configuration screen

1. Click the Setup menu in the upper right section of the screen.

2. From the drop-down menu, click **Administration**.

The Administration screen opens with tabs for configuring the various settings.



Configure Authentication Provider

- [To Enable an Authentication Provider](#)
- [Learn More](#)

AppDynamics Controller users can be authenticated by the AppDynamics Controller or by a third-party authentication system that is integrated with the Controller, such as LDAP or SAML.

The most efficient way to add and configure a large number of users is to use a third-party authentication provider.

See [To access the configuration screen](#) if you do not know how to access the administration configuration screens.

To Enable an Authentication Provider

1. In the Administration screen, click the **Authentication Provider** tab.
2. Select radio button for the authentication provider to use.

If you select AppDynamics, the AppDynamics Controller is the authentication provider. In this case, users log in with their AppDynamics credentials. See [Configure Users](#) for information about creating AppDynamics users. You can also select one of the third-party authentication providers: LDAP or SAML.

Learn More

- [Configure Authentication Using LDAP](#)
- [Configure Authentication Using SAML](#)
- [Configure Roles](#)
- [Configure Users](#)
- [Configure Groups](#)

Configure Authentication Using SAML

- [Prerequisites](#)
- [Configuring SAML Settings](#)

- To Access Security Settings
- To Enable a Security Provider
- To configure AppDynamics SAML Settings.
- [Learn More](#)

This topic describes how to configure an AppDynamics account for single sign-on to an AppDynamics Controller using SAML.

The AppDynamics Controller uses this configuration to redirect authentication requests to an identity provider.

The AppDynamics SAML integration conforms to the Security Assertion Markup Language 2.0 (SAML 2.0) specification, so any SAML 2.0-compliant identity provider can be integrated with AppDynamics.

The specific configuration tasks depend on the SAML identity provider's implementation. See [SAML Configuration for OneLogin](#) for an example of AppDynamics' integration with One Login.

Prerequisites

To perform SAML configuration you must have:

1. An account with a supported identity provider

You need your SAML Login URL and your x.509 certificate supplied by your identity provider.

If OneLogin is your identity provider, use the widget described in [SAML Configuration for OneLogin](#) to configure OneLogin for AppDynamics. For other SAML identity providers, you need to provide the following SAML authentication assertion custom attributes to be sent by the identity provider:

- accountName - required only if the controller is multi-tenant
- emailAddress - the user's email address.

AppDynamics assumes that the SAML authentication assertion NameID will be set to the user name of the user logging into the controller.

2. An account on an AppDynamics SaaS or on-premise Controller

In the case of an on-premise controller, if the controller is within a closed network and the identity provider is a service external to that network, the identity provider must be able to connect to the SAML Consumer URL and the controller must be able to connect to the SAML Login URL. For examples of these URLs, see [To configure AppDynamics SAML Settings for OneLogin](#) and [To configure OneLogin Settings for AppDynamics](#).

3. Account Administrator privileges on the AppDynamics Controller, as described in [Administrative Users](#).

Before configuring the SAML settings in the AppDynamics Controller, verify that the Account Administrator user who will log into the Controller and set up the SAML authentication also exists as a user in the identity provider. After configuring the SAML settings in AppDynamics and logging out, this user will be forced to log in again as that Account Administrator this time using the identity provider for authentication.

Configuring SAML Settings

Follow the configuration steps specific to your identity provider to enable the integration.

To Access Security Settings

1. Click the Setup menu in the upper right section of the UI.
2. From the drop-down menu, select **Administration**.

The Administration screen opens with tabs for configuring the various settings.

To Enable a Security Provider

1. Click the **Authentication Provider** tab.
2. Select the **SAML** radio button for the authentication provider to use.

To configure AppDynamics SAML Settings.

See [To configure AppDynamics SAML Settings for OneLogin](#) for a sample configuration with an actual identity provider.

The screenshot shows the 'Security Configuration' interface with the 'Authentication Provider' tab selected. At the top, there are tabs for 'Users', 'Groups', 'Roles', and 'Authentication Provider'. Below the tabs is a 'Save' button. The 'Authentication Provider' section contains three radio buttons: 'AppDynamics' (unchecked), 'LDAP' (unchecked), and 'SAML' (checked). Underneath are fields for 'Login URL' and 'Logout URL', both of which are currently empty. A 'Certificate' field contains the text '-----BEGIN CERTIFICATE-----' and '-----END CERTIFICATE-----'. Below these fields is a table titled 'Default Roles' with columns for 'Member', 'Name', and 'Description'. The table lists five roles:

Member	Name	Description
<input type="checkbox"/>	Account Owner (Default)	Can access and modify global configuration and view/modify applications and dashboards
<input type="checkbox"/>	Administrator (Default)	Can view/modify applications and dashboards
<input type="checkbox"/>	Custom Dashboard Viewer (Default)	Can view dashboards only
<input type="checkbox"/>	Read Only User (Default)	Can view applications and dashboards but not modify their configuration
<input type="checkbox"/>	Workflow Executor (Default)	Can execute workflows only

In the SAML Configuration page:

1. In the Login URL field, enter the SAML Login URL. The SAML Login URL is the URL to the SSO service at the identity provider. The identity provider provides this URL to the Controller.
2. In the Logout URL field, enter the URL to which the browser should redirect when the user logs out. This is useful for redirecting the user back to the identity provider instead of to the AppDynamics login screen. This field is optional.
3. In the Certificate field, paste the x.509 certificate from your identity provider configuration between the BEGIN CERTIFICATE and END CERTIFICATE delimiters. Do not copy the BEGIN CERTIFICATE and END CERTIFICATE from certificate field.
4. In the Default Roles section, select the roles to grant to new users of the SAML-enabled controller by checking the Member check box for the role. You must grant at least one default role, and you can select multiple roles. See [Configure Roles](#) for information about roles and permissions.

The roles that you assign here will be granted to new users when they first log in to the SAML-enabled controller if those users have not been previously created directly in the Controller. Users created prior to SAML enablement or directly within the controller prior to the user's initial login retain their original roles.

Typically SAML users get the default roles assigned in this configuration. In exceptional cases an account owner may want to grant individual users different roles. See [To Assign A Role to a User](#).

5. Click **Save**.

Learn More

- [SAML Configuration for OneLogin](#)
- [Disable SAML Authentication for an Account](#)
- [Configure Users](#)

Disable SAML Authentication for an Account

- [To disable SAML Authentication](#)

You can disable SAML authentication for an account from the AppDynamics administration console.

To disable SAML Authentication

1. Log into the administration console.
See [Access the Administration Console](#).
2. In the left navigation panel click **Accounts**.
3. From the accounts list select the account for which you want to disable SAML authentication.
4. Click the Edit icon.
The account settings for the account are displayed. If SAML is enabled, the setting appears as follows.



5. Click **Disable**.

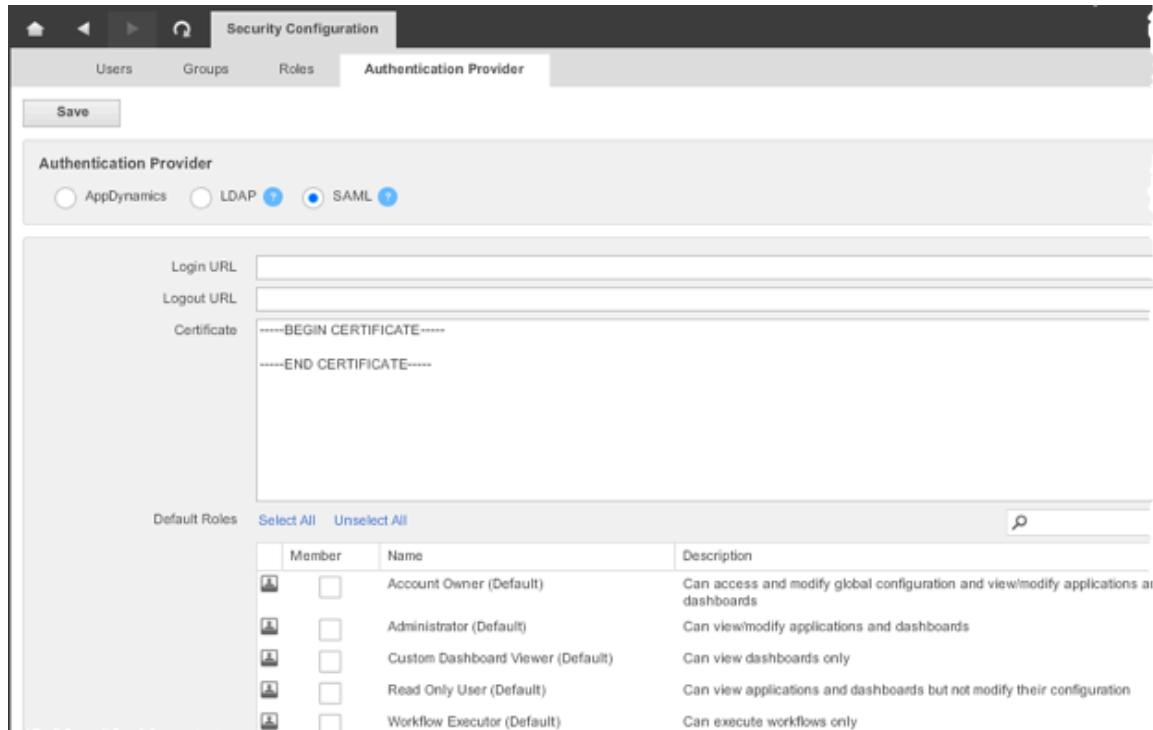
SAML Configuration for OneLogin

- To configure AppDynamics SAML Settings for OneLogin
- To configure OneLogin Settings for AppDynamics
- [Learn More](#)

This topic describes a sample SAML integration with the identity provider, OneLogin. See [Configure Authentication Using SAML](#) for general information about SAML integration.

To configure AppDynamics SAML Settings for OneLogin

1. Access the security settings configuration screen. See [To Access Security Settings](#).
2. Select **SAML** as the provider. See [To Enable a Security Provider](#).
The SAML Single Sign On Configuration page opens.



Member	Name	Description
<input type="checkbox"/>	Account Owner (Default)	Can access and modify global configuration and view/modify applications and dashboards
<input type="checkbox"/>	Administrator (Default)	Can view/modify applications and dashboards
<input type="checkbox"/>	Custom Dashboard Viewer (Default)	Can view dashboards only
<input type="checkbox"/>	Read Only User (Default)	Can view applications and dashboards but not modify their configuration
<input type="checkbox"/>	Workflow Executor (Default)	Can execute workflows only

3. In the Login URL field enter the SAML Login URL from your OneLogin configuration. The SAML Login URL is the URL to the SSO service at the identity provider. The identity provider provides this URL to the Controller.

If you do not know your SAML Login URL, you can locate it in your OneLogin configuration:

- a. Log in to your OneLogin account.
- b. Click the **Apps** tab in the first set of tabs.
- c. Click **edit** next to the application for which you want to view the Login URL.
Click the **Company Apps** tab in the second set of tabs if it is not already selected.
- d. Click **Single Sign-on** in the third set of tabs.
The SAML Login URL is the HTTP SAML Endpoint in the Sign-on method section.

The screenshot shows the AppDynamics configuration interface. At the top, there are tabs for Portal, Apps, People, Activity, Security, and Account. Below that, a sub-navigation bar has tabs for Company apps, Find apps, Tabs, and Custom connectors. The main content area shows the AppDynamics logo and navigation tabs for Portal, Configuration, Single Sign-on (which is selected), Access Control, and Logins. Under the Single Sign-on tab, the 'Sign-on method' is listed as SAML2.0. The 'Issuer URL' is https://app.onelogin.com/saml/metadata/46533. The 'SAML Endpoints:' section contains two entries: 'HTTP: https://app.onelogin.com/trust/saml2/http-post/sso/46533' and 'SOAP: https://app.onelogin.com/trust/saml2/soap/sso/46533'. Below this, under 'Credentials', it says 'Configured by admin' (radio button selected). Under 'Default values', there are fields for 'Login' (AD user name dropdown) and 'Email Address' (Email dropdown).

4. In the Logout URL field in the AppDynamics form, enter the URL to which the browser should redirect when the user logs out. This is useful for redirecting the user back to the identity provider instead of to the AppDynamics login screen. This field is optional. For example, the following logout URL redirects the user to the OneLogin application dashboard:

<https://app.onelogin.com/client/apps>

5. In the Certificate field in the AppDynamics form, paste the x.509 certificate from your OneLogin configuration between the BEGIN CERTIFICATE and END CERTIFICATE delimiters. Do not copy the BEGIN CERTIFICATE and END CERTIFICATE from the OneLogin x.509 certificate field.

To find your x.509 certificate in your OneLogin configuration:

- a. Log in to your OneLogin account.
- b. Click the **Security** tab in the first set of tabs.
- c. Click **SAML** in the second set of tabs.



The x.509 certificate is generated specifically for your OneLogin account and must be entered in any external application that you wish to use SAML with. Configuring SAML can be done in two steps.

1. Add an application that uses SAML, e.g. Google Apps or Salesforce
2. Configure the application with the x.509 certificate using the [Instructions here](#)

x.509 certificate

```
-----BEGIN CERTIFICATE-----
MIICMTCCAiBgAwIBAgIBATADBgEAMCcxCzAJBgNVBAYTAlVTMRMwEQYDVQQIDA
-----END CERTIFICATE-----
```

6. In the Default Roles section in the AppDynamics form, select the roles to grant to new users of the SAML-enabled controller by checking the Member check box for the role. You can select multiple roles in the list. See [Configure Roles](#) for information about roles and permissions.

The roles that you assign here will be granted to new users when they first log in to the SAML-enabled controller if those users have not been previously created directly in the Controller. Users created prior to SAML enablement retain their original roles.

You must grant at least one default role.

7. Click **Save**.

To configure OneLogin Settings for AppDynamics

In your OneLogin account, configure the SAML Consumer URL with the host, port and optional account name values from the AppDynamics Controller. The Consumer URL is where the identify provider posts the SAML Authentication Assertion.

1. Log in to your OneLogin account.
2. Click the **Apps** tab in the first set of tabs.
3. Click **edit** next to the AppDynamics Connector.
Click the **Company Apps** tab in the second set of tabs if it is not already selected.
4. In the third set of tabs click **Configuration**.

5. Enter the Consumer URL for the AppDynamics connector.
It has the format:

`http[s]://<controller-host>:<controller-port>/controller/saml-auth`

The host and port for your Controller account are supplied by AppDynamics.

Application settings	Consumer URL
	<code>https://test-huge4.saas.appdyn...</code>
	Account Name
	<small>needed only for multi-tenant AppDynamics controllers</small>

6. Provide the AppDynamics account name if your controller is configured in multi-tenant mode and if the user normally enters an account name on login. If your controller is configured in single-tenant mode or if the user does not supply an account name on login, you can leave the Account Name field blank.

See [Controller Tenant Mode](#) for information about controller tenant modes.

7. Save your settings.

Learn More

- [Configure Authentication Using SAML](#)
- [Disable SAML Authentication for an Account](#)

Configure Authentication Using LDAP

- [Prerequisites](#)
- [Configuring LDAP Settings](#)
 - [To Access the LDAP Configuration Screen](#)
 - [To Enable LDAP Authentication for AppDynamics](#)
 - [To Configure the Connection to the LDAP Server](#)
 - [To Configure the Query to Find Users](#)
 - [To Configure the Query to Find Groups](#)
- [Configuring the LDAP Synchronization Frequency](#)
 - [To Configure the LDAP Synchronization Frequency](#)
- [Configuring the Maximum Entries Returned from the LDAP Server](#)
 - [To Configure the Maximum LDAP Results Limit](#)
- [Using LDAP](#)
 - [To Enable LDAP as Your Authentication Provider](#)
 - [To Assign AppDynamics Permissions to an LDAP User](#)
 - [To Assign AppDynamics Permissions to an LDAP Group](#)
- [Learn More](#)

This topic describes how to configure AppDynamics account authentication using Lightweight Directory Access Protocol (LDAP).

The AppDynamics Controller uses this configuration to redirect authentication requests to an LDAP server.

Prerequisites

To perform LDAP configuration you must have:

- An LDAP server
There is a one-to-one correspondence between an AppDynamics account and an LDAP server.
- An account on an AppDynamics SaaS or on-premise Controller
- Account Administrator privileges on the AppDynamics Controller, as described in [Administrative Users](#).

Configuring LDAP Settings

Configuring LDAP settings includes:

- Enabling or disabling LDAP authentication for AppDynamics
- Configuring the connection to the LDAP server
- Configuring the query LDAP uses to find users
- Configuring the query LDAP uses to find groups

To Access the LDAP Configuration Screen

1. Click the Setup menu in the upper right section of the screen.
2. From the drop-down menu, click **Administration**.
3. Click **Authentication Provider**.
4. Select **LDAP**.
The LDAP configuration screen opens.

To Enable LDAP Authentication for AppDynamics

Select LDAP in the security settings configuration screen. See [To Enable a Security Provider](#).

To Configure the Connection to the LDAP Server

1. Access the LDAP configuration screen if you have not yet done so. See [To Access the LDAP Configuration Screen](#).
2. In the General section, you can optionally set the paging feature:
 - Enable Paging: Check this option if the number of entries that could be pulled in exceeds the limits on the LDAP server.
 - Page Size: Number of entries per round-trip from AppDynamics to LDAP; 500 is the default. A maximum size would depend on the LDAP server configuration.
3. In the Connection String section of the LDAP configuration screen, enter the information AppDynamics uses to connect to the LDAP server:
 - Host: Address of the LDAP server. Required.
 - Port: Port that the LDAP server listens on. Default is 636 for an SSL connection and 389 if not using SSL. Required.
 - Use SSL: Enabled by default to use a secure connection to the LDAP server. Clear if not using SSL.
 - Enable Referrals: Enabled by default to support LDAP referrals. For an overview of LDAP referrals see [Referrals in the LDAP](#).
 - Maximum Referral Hops: The maximum number of referrals that AppDynamics will follow in a sequence of referrals. Default is 5. See [referral hop limit](#).
 - Bind DN: Distinguished name of the AppDynamics administrative user to log into the LDAP Server. Required.
 - Password: Password of the AppDynamics administrative user. Required.
4. Click **Test Connection** to ensure that the connection string works properly.
5. When the connection test is successful, click **Save**.

To Configure the Query to Find Users

1. Access the LDAP configuration screen if you have not yet done so. See [To Access the LDAP Configuration Screen](#).
2. In the Users Query section of the LDAP configuration screen, enter the information to use to find LDAP users:
 - Base DN: Location in the LDAP tree to begin recursively searching for users. Required.
 - Filter: Optional LDAP search string that filters the items matched from the base DN. See [RFC2254](#) for information about LDAP search filters.
 - Login Attribute: User's LDAP login. Default is "uid".
 - Display Name Attribute: Optional user's display name.
 - Group Membership Attribute: Optional user group membership. Recommended for faster retrieval.
 - Email Attribute: Optional user email address.

The screenshot shows the 'Users Query' configuration screen. It includes the following fields:

- Base DN: cn=users, dc=ad, dc=example, dc=com
- Filter: (empty)
- Login Attribute: uid
- Display Name Attribute: displayName
- Group Membership Attribute: memberOf
- Email Attribute: mail

A 'Test Query' button is located at the bottom of the form.

3. To test the configuration, click **Test Query**.

This launches a screen displaying the first few users returned by the query.

4. Click **Save**.

To Configure the Query to Find Groups

LDAP Group configuration is optional.

1. Access the LDAP configuration screen if you have not yet done so. See [To Access the LDAP Configuration Screen](#).

2. In the Groups Query section of the LDAP configuration screen, enter the information to use to find LDAP groups:

- Base DN: Location in the LDAP tree to begin recursively searching for groups. Required.
- Enable Nested Groups: Option to include nested LDAP groups to a depth of 10.
- Filter: Optional LDAP search string that filters the items matched from the base DN. See [RFC2254](#) for information about LDAP search filters.
- Name Attribute: Name of the group. Default is "cn". Required.
- Description Attribute: Optional description of the group.
- User Membership Attribute: Identifies member of the groups. Optional.
- Referenced User Attribute: Optional Child attribute of the User Membership Attribute. Disabled if the parent is empty. Identifies property of the user that the user membership attribute contains.

The screenshot shows a configuration interface for a 'Groups Query'. The fields are as follows:
Base DN: ou=Groups,dc=corp,dc=appdynamics,dc=com
Filter: objectClass=groupOfNames
Name Attribute: cn
Description Attribute: description
User Membership Attribute: member
Referenced User Attribute: dn
A 'Test Query' button is located at the bottom right of the form.

3. To test the configuration, click **Test Query**.

This launches a screen displaying the first few groups returned by the query.

4. When the query test is successful, click **Save**.

Configuring the LDAP Synchronization Frequency

You can configure how often the Controller synchronizes with the LDAP server to capture new and removed users and groups. The default is 1 hour (3600000 milliseconds).

To Configure the LDAP Synchronization Frequency

1. Stop the Controller application server:

On Linux:

```
./controller.sh stop-appserver
```

On Windows:

```
controller.bat stop-appserver
```

2. Open the <Controller-Installation-Directory>/appserver/domains/domain1/config/domain.xml file for edit.
3. In the <jvm-options> element, set the appdynamics.ldap.sync.frequency system property to the desired synchronization frequency in milliseconds.
For example, to synchronize every 15 minutes (900000 milliseconds) enter:

```
<jvm-options>-Dappdynamics.ldap.sync.frequency=900000</jvm-options>
```

4. Save the file.
5. Restart the Controller app server:

On Linux:

```
./controller.sh start-appserver
```

On Windows:

```
controller.bat start-appserver
```

Configuring the Maximum Entries Returned from the LDAP Server

If you are importing a large number of LDAP users or groups into the Controller, it is possible to get a max_results_exceeded error. This occurs if your request exceeds the LDAP server's limit on the number of entries that an LDAP client can retrieve in a single operation.

If you do get the max_results_exceeded error, AppDynamics recommends that you first try to revise the filter in your user or group query to return fewer results. See [To Configure the Query to Find Users](#) and [To Configure the Query to Find Groups](#).

If you are unable to reduce the number of results returned so that you do not get the error, you can increment the appdynamics.controller.ldap.max.results system property to the number of entries that you expect to return, if that number is less than the LDAP server's hard limit on the number of entries that can be returned to the client. Work with your LDAP administrator to arrive at the right value.

If neither fine-tuning the user query filter nor setting the appdynamics.controller.ldap.max.results system property succeeds in stopping the max_results_exceeded error, call AppDynamics Support.

To Configure the Maximum LDAP Results Limit

1. Stop the Controller application server:

On Linux:

```
./controller.sh stop-appserver
```

On Windows:

```
controller.bat stop-appserver
```

2. Open the <Controller-Installation-Directory>/appserver/domains/domain1/config/domain.xml file for edit.
3. In the <jvm-options> element, set the appdynamics.controller.ldap.max.results system property to a value somewhat greater than the number of entries that you expect to retrieve.
For example, to set the maximum results returned to 800 enter:

```
<jvm-options>-Dappdynamics.controller.ldap.max.results=800</jvm-options>
```

4. Save the file.
5. Restart the Controller app server:

On Linux:

```
./controller.sh start-appserver
```

On Windows:

```
controller.bat start-appserver
```

Using LDAP

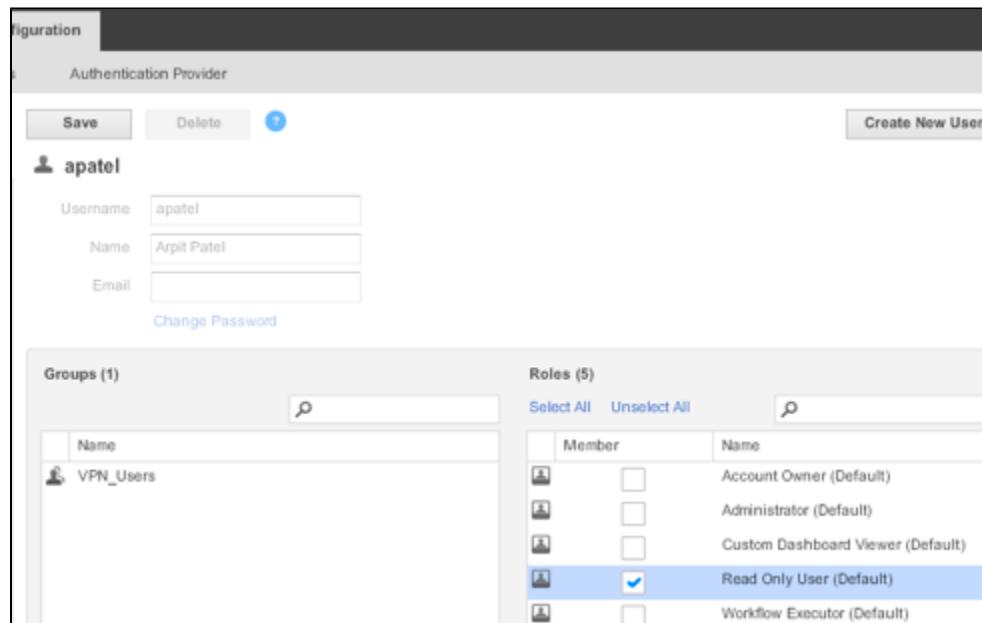
You can assign permissions to LDAP groups or users.

To Enable LDAP as Your Authentication Provider

In the **Authentication Provider** tab of the Security Configuration screen, select LDAP. See [Configure Authentication, User Permissions and Integrations](#).

To Assign AppDynamics Permissions to an LDAP User

1. In the Security Configuration window, click the **Users** tab.
If LDAP is enabled and correctly configured, AppDynamics fetches the usernames of the authenticated LDAP users.
2. Select the name of the user to whom you want to assign permissions.
3. In the Roles panel, check the roles that you want to assign to this user. You can assign multiple roles to a user.



4. When the query test is successful, click **Save**.

To Assign AppDynamics Permissions to an LDAP Group

LDAP Group configuration is optional. LDAP nested groups are not supported.

1. In the Security Configuration window, click the **Groups** tab.
If LDAP is enabled and correctly configured, AppDynamics fetches the group names of the authenticated LDAP users.
2. Select the name of the group to which you want to assign permissions.
3. In the Roles panel, check the roles that you want to assign to this group. You can assign multiple roles to a group.

4. Click **Save**.

Learn More

- Configure Roles
- Configure Groups
- Configure Users

Configure Users

- To Create an AppDynamics User
- To Assign a User to a Group
- To Assign A Role to a User
- To Modify A User's Settings
- To Delete a User
- To Duplicate a User
- Learn More

This topic describes how to configure an AppDynamics Controller user if you are not using an external provider, such as LDAP or SAML, to authenticate AppDynamics Controller users.

To Create an AppDynamics User

1. Click the Setup menu in the upper right section of the screen.
2. From the drop-down menu, click **Administration**.
3. Click the **Users** tab.
4. Click the Add icon.
5. Enter the information for the following fields:

- Username
- Name
- Email
- Password
- Repeat Password

6. Click **Save**.
7. Click **Create New User**.

To Assign a User to a Group

1. In the left panel, select the user or users whom you want to assign to one or more groups. You can enter a string in the filter field to locate a specific user.
2. In the Groups list in the right panel, check the Member check boxes of the groups to which you want to assign the selected user or users. You can enter a string in the filter field to locate a specific group. You can click **Select All** to check all the groups and **Unselect All** to uncheck all the groups. A single user can be assigned to multiple groups.
3. Click **Save**.

A single user can belong to multiple groups.

You can also assign users to a group from the **Groups** tab. See [Configure Groups](#).

To Assign A Role to a User

1. In the left panel, select the user or users to which you want to assign one or more roles. You can enter a string in the filter field to locate a specific user.
2. In the Roles list in the right panel, check the Member checkboxes of the roles that you want to assign to the selected user or users. You can enter a string in the filter field to locate a specific role. You can click **Select All** to check all the roles and **Unselect All** to uncheck all the roles. A single user can be assigned multiple roles.

3. Click **Save**.

You can also assign roles to a user from the **Roles** tab.

Warning: Do not remove yourself from all groups or from all roles. Also, if the only roles of which you are a member are custom roles, do not delete those custom roles or remove permissions from them. Doing these things can result in your being locked out of AppDynamics with no permissions at all. If this happens, contact AppDynamics Support.

To Modify A User's Settings

1. In the User list in the left panel, select the user to modify.
2. Click the Edit icon.
3. To change the username, name and password, enter the new values in the fields in the right panel.
4. To change the password, click **Change Password** and then enter the new password values in the Password and Repeat Password fields in the right panel.
5. Click **Save**.

To Delete a User

1. In the User list in the left panel, select the user to delete.
2. Click **Delete**.

To Duplicate a User

1. In the User list in the left panel, select the user to duplicate.
2. Click the Duplicate icon.

In the Duplicate User screen overwrite the user's name with a new name and enter the new password.

The user created by duplication gets the groups and roles of the user from which he was cloned.

Learn More

- [Configure Groups](#)
- [Configure Roles](#)

Configure Groups

- [To Create an AppDynamics Group](#)
- [To Assign a User to a Group](#)
- [To Assign A Role to a Group](#)
- [To Modify A Group's Settings](#)
- [To Delete a Group](#)
- [Learn More](#)

This topic describes how to configure an AppDynamics Controller group. A group is a set of users that can be granted sets of permissions through assignment of roles. It is more efficient to assign roles to a group than to assign the same roles to each user individually.

If you are using LDAP to authenticate all AppDynamics Controller users you do not need to create AppDynamics groups.

To Create an AppDynamics Group

1. Click the Setup menu in the upper right section of the screen.
2. From the drop-down menu, click **Administration**.
3. Click the **Groups** tab.
4. Click the Add icon.

5. Enter the information for the following fields:

- Name
- Description

6. Click **Save**.

7. Click **Create New Group**.

To Assign a User to a Group

1. In the left panel, select the group to which you want to assign one or more users. You can enter a string in the filter field to locate a specific group.

2. In the Users list in the right panel, check the Member check boxes of the users whom you want to assign to the selected group or groups. You can enter a string in the filter field to locate a specific user. You can click **Select All** to check all the users and **Unselect All** to uncheck all the users.

A single user can belong to multiple groups.

To Assign A Role to a Group

1. In the left panel, select the group or groups to which you want to assign roles. You can enter a string in the filter field to locate a specific group.

2. In the Roles list in the right panel, check the Member check boxes of the roles that you want to assign to the selected group or groups. You can click **Select All** to check all the roles and **Unselect All** to uncheck all the roles. You can enter a string in the filter field to locate a specific role.

A single group can be assigned multiple roles.

3. Click **Save**.

To Modify A Group's Settings

1. In the groups list in the left panel, select the group to modify.

2. Enter the new values in the fields in the right panel.

3. Click **Save**.

To Delete a Group

1. In the group list in the left panel, select group to delete.

2. Click **Delete**.

Learn More

- Configure Roles
- Configure Users
- Configure Authentication Using LDAP
- Configure Authentication Using SAML

Configure Roles

- Predefined Roles
- Custom Roles
- Permission Inheritance for Default Permissions
- Account Level Permissions
- Application Level Permissions
- Custom Dashboard Permissions
- Learn More

Roles define a set of privileges that AppDynamics Controller users have within the AppDynamics managed environment. This is also called "role-based access control", or "RBAC".

Roles provide an easy way to define and clone a set of permissions for a user or a group without having to configure every user's or

every group's permissions individually.

You can assign a role to a user or group either in the **Users** or **Groups** tabs or in the **Users and Groups with this Role** subtab of the Roles configuration screen.

A user or group can have multiple roles.

Predefined Roles

AppDynamics provides the following predefined roles:

- Account Owner: Can manage security settings (users, groups, roles) as well as view and modify applications and dashboards. This role is also known as the account administrator.
- Administrator: Can view and modify components that change state: applications, business transactions, dashboards, etc. Can view and edit all applications and all custom dashboards.
- Custom Dashboard Viewer: Can only view custom dashboards. Cannot do anything else.
- Read Only User: Can view but not edit all applications.
- Workflow Executor: Can execute workflows.

You can view the configurations for the predefined roles but you cannot change them. See [View Predefined Roles](#).

Although you cannot modify the predefined roles, you can add or remove users and groups from the predefined roles by checking or clearing the Member check box in the Roles panel. See [Configure Users](#) and [Configure Groups](#).

Custom Roles

You can create custom roles if the predefined roles do not fit your organization's needs.

You can create a custom role from scratch or you can duplicate a predefined role, save it under another name, and then modify it as a custom role.

You can assign permissions to a custom role at the application/tier level, the custom dashboard level, and at the account level.

See [Configure Custom Roles](#).

Permission Inheritance for Default Permissions

For application-level and custom dashboard permissions, AppDynamics provides a default set of permissions, named Default. The default permissions are inherited by new applications and new custom dashboards. The Default permissions are listed first in the **Application Permissions** subtab of the **Roles** tab.

Tier-level permissions are inherited from the containing application.

Account Level Permissions

Permissions that can be granted at the account level include:

- Administer: Users with this permission can edit users, groups, roles, and the authentication provider.
- Configure Email/SMS: Users with this permission can edit email and SMS settings used by AppDynamics to send alerts. See [Configure the SMTP Server](#) and [Notification Actions](#).
- Execute Workflows: Users with this permission can execute workflows. See [Workflow Automation](#).

Application Level Permissions

Permissions that can be granted at the application level and tier levels include the following. Follow the links if you want to see precisely the tasks that the configuration permissions allow the user to perform.

- [Configure Business Transaction Detection](#)
- [Configure Backend Detection](#)
- [Configure Error Detection](#)
- [Configure Data Collectors](#)
- [Configure Call Graphs](#)
- [Configure JMX Metrics from MBeans and Import or Export JMX Metric Configurations](#) - pertains to the **Configure JMX** permission
- [Configure Memory Monitoring \(Java\)](#)
- [Configure End User Experience](#)
- [Configure Code Metric Information Points](#)

- Configure Policies
- Configure Baselines
- Notification Actions
- SQL Capture Settings - pertains to the **Configure SQL Bind Variables** permission
- App Agent Node Properties

Custom Dashboard Permissions

Permissions that can be granted at the custom dashboard level include:

- View
- Edit
- Delete

Custom dashboards are a good way to present a selected set of metrics for a "guest" user, such as an executive, who does not normally use AppDynamics. You can grant such users permission to view custom dashboards created especially for them.

Learn More

- [Configure Custom Roles](#)

Access Role Configuration

- [To Access the Roles Configuration Screen](#)

To Access the Roles Configuration Screen

1. Access the security settings configuration screen. See [Configure Authentication, User Permissions and Integrations](#).

2. Click the **Roles** tab.

The Roles configuration screen opens.

View Predefined Roles

- [To View a Predefined Role](#)
- [Learn More](#)

You can view the predefined roles and assign them to users and groups if you have the required permissions.

If the checkbox for a permission in a role configuration is checked, the permission is granted to users with that role. If the checkbox is clear, the permission is denied.

See [Configure Roles](#) for general information about roles.

To View a Predefined Role

1. Access the Role Configuration screen. See [Access Role Configuration](#).

2. Click the **Roles** tab.

3. In the left panel select the role that you want to view.

4. In the right panel, click the **Account Level Permissions** tab to see the role's permission at the account level.

5. Click the **Custom Dashboards** tab to view the role's permissions for viewing, editing, and deleting custom dashboards.

6. Click the **Application Permissions** tab to see the role's permissions at application and tier levels. You need to expand the applications to see the tier-level permissions.

a. Select the application or tier for which you want to view the configuration. You need to expand the applications tabs to view the tiers.
b. Click the Edit icon.

The Edit Permissions window opens showing the edit permissions for the selected predefined role for the selected application or tier. Although you cannot edit these permissions, you can view which permissions have been granted and then determine whether the default role meets your needs for the selected application or tier or whether you need to configure a custom role. See [Configure Custom Roles](#).

c. Click **Close** to close the Edit Permissions window.

Learn More

- Configure Roles
- Access Role Configuration

Configure Custom Roles

- Configuring Custom Roles
 - To Create a Custom Role
- Configuring Application Level Permissions
 - To Configure the Default Application Permissions
 - To Configure Application Permissions
 - To Configure Tier-Level Permissions
- Configuring Custom Dashboard Permissions
 - To Configure the Default Custom Dashboard Permissions
 - To Configure Permissions for Individual Dashboards
- Configuring Account Level Permissions
 - To Configure Account Level Permissions
- Modifying a Custom Role
 - To Modify a Custom Role
- Learn More

Create custom roles if you want to grant permissions to users and groups using roles that are configured differently from the predefined roles. See [View Predefined Roles](#) to see exactly which permissions are granted and denied by the predefined roles.

You can configure custom roles very finely to grant users certain permissions in a single application or even a single tier or set of tiers or custom dashboard.

See [Configure Roles](#) for information about the types of permissions that roles grant.

Configuring Custom Roles

After you have created a custom role, select it and configure the three categories of permissions by clicking the tabs:

- Application Level Permissions
- Custom Dashboard Permissions
- Account Level Permissions

To Create a Custom Role

1. Open the role configuration screen. See [Access Role Configuration](#).
2. Either
 - a. In the left panel, click the Add icon to create a new role.
 - b. In the right panel, enter the name of the role and an optional description.
 - c. Click **Save**.
- or
 - a. Select an existing role in the left panel.
 - b. Click the Duplicate icon.
 - c. In the Duplicate Role window, overwrite the default name with your name for the new role.
 - d. Click **Duplicate**.
 - e. In the left panel, select the newly created role.
 - f. In the right panel, edit the name and description for the new role.
 - g. Click **Save**.

Configuring Application Level Permissions

In this tab you can configure:

- the role's default application-level permissions
- the role's custom permissions for specific applications and tiers

Custom roles can be very fine-tuned at the application level and tier levels. For example, you could create an AcmeManager role with certain delete and edit permissions that apply only to the Acme bookstore application and another AcmeUser role with only view permissions or possibly also with a more limited set of edit permissions for the Acme bookstore application. You can also create roles at the tier level; for example, an InventoryManager role or an InventoryUser role with certain permissions only the Inventory tier and not for any of the other tiers in the application.

Every role has a set of default application-level permissions that are inherited by all new applications in the account. For a custom role you can reconfigure these default permissions. You can also override the inherited permissions for custom roles by reconfiguring application-level and tier-level permissions.

To Configure the Default Application Permissions

1. With the custom role that you are configuring selected in the left panel, in the right panel click the **Application Permissions** tab.
2. To grant the role permission to create new applications, check the **Can Create Applications** check box. Otherwise leave it clear.
3. In the Default row do the following:
 - To permit users with this role to view applications check the **View** check box. To deny them view permission, clear the **View** check box.
 - To permit users with this role to delete the application check the **Delete** check box. To deny them delete permission, clear the **Delete** check box.
 - To permit users to edit AppDynamics configurations:
 - a. Click the **Edit** icon.
 - b. In the Edit Permissions window check the check boxes for the configurations that the role can perform and clear them for the configurations that it cannot perform.
- For information about the permissions that can be granted at the application level and tier levels, see [Application Level Permissions](#)
4. Click **OK** in the Edit Permissions window.
5. Click **Save** at the top of the pane to save the default configuration.

To Configure Application Permissions

You can configure a custom role to have different permissions in different applications.

1. With the custom role that you are configuring selected in the left panel, in the right panel click the **Application Permissions** tab.
2. In the row for the application for which you want to configure the role's permissions, do one of the following:
 - If you want the role to inherit the default application-level permissions, select **Inherit from Default** from the dropdown menu if it is not already selected.
or
 - If you do not want the role to inherit the default application-level permissions, select **Customized** from the dropdown menu and then follow steps 3 and 4 as described above in [To Configure the Default Permissions](#) for the application's row instead of for the Default row.
3. Repeat the previous step for every application that you want to configure.
4. Click **Save** at the top of the pane to save the configuration.

To Configure Tier-Level Permissions

By default, a role's permissions in a tier are inherited from the role's permissions in the tier's containing application. You can override this behavior to configure a custom role to have specific permissions in different tiers.

1. In the **Application Permissions** tab, expand the application in which you want to configure the role's tier-level permissions.
2. In the row for the tier for which you want to configure the role's permissions, do one of the following:

- If you want the role to inherit the application permissions for the tier, select **Inherit from Application** from the dropdown menu if it is not already selected.

or

- If you do not want the role to inherit the application permissions for the tier, select **Customized** from the dropdown menu and then follow steps 3 and 4 as described above in [To Configure the Default Permissions](#) for tier's row instead of for the Default row.

3. Repeat the previous step for every tier that you want to configure.

4. Click **Save** at the top of the pane to save the configuration.

Configuring Custom Dashboard Permissions

In this tab you can configure:

- the role's default custom dashboard permissions
- the role's permissions for specific custom dashboards

Every dashboard inherits the default custom dashboard permissions unless you override them by configuring separate permissions for individual dashboards. For example, you could have a custom dashboard called SalesDashboard and a custom role SalesRole, and another custom dashboard called FinanceDashboard and a custom role FinanceRole. The SalesRole could be configured to have permissions in the SalesDashboard but no permissions in the FinanceDashboard and vice-versa.

To Configure the Default Custom Dashboard Permissions

1. With the custom role that you are configuring selected in the left panel, in the right panel click the **Custom Dashboard Permissions** tab.

2. To grant the role permission to create new custom dashboards, check the **Can Create Custom Dashboards** check box. Otherwise leave it clear.

3. In the default row do the following:

- To permit users with this role to view custom dashboards, check the **View** check box. To deny them view permission, clear the **View** check box.
- To permit users with this role to edit custom dashboards, check the **Edit** check box. To deny them edit permission, clear the **Edit** check box.
- To permit users with this role to delete custom dashboards, check the **Delete** check box. To deny them delete permission, clear the **Delete** check box.

4. Click **Save** at the top of the pane to save the default configuration.

To Configure Permissions for Individual Dashboards

You can configure a custom role to have different permissions in different custom dashboards.

1. With the custom role that you are configuring selected in the left panel, in the right panel click the **Custom Dashboard Permissions** tab.

2. In the row for the custom dashboard for which you want to configure the role's permissions:

- To permit users with this role to view the custom dashboard, check the **View** check box. To deny them view permission, clear the **View** check box.
- To permit users with this role to edit the custom dashboard, check the **Edit** check box. To deny them edit permission, clear the **Edit** check box.
- To permit users with this role to delete the custom dashboard, check the **Delete** check box. To deny them delete permission, clear the **Delete** check box.

3. Repeat the previous step for every custom dashboard that you want to configure.

4. Click **Save** at the top of the pane to save the configuration.

Configuring Account Level Permissions

To Configure Account Level Permissions

1. With the role selected in the left panel, in the right panel click the **Account Level Permissions** tab.
2. Check the check boxes for the tasks that can be performed by the selected custom role. see [Account Level Permissions](#) for descriptions of these permissions.
3. Clear the check boxes for the tasks that cannot be performed by this role if they are not already clear.
4. Click **Save**.

Modifying a Custom Role

Modifying a custom role is similar to creating a new role.

To Modify a Custom Role

1. Select the role to modify in the left panel of the role configuration screen.
The role configuration is visible in the right panel.
2. Edit the role as you would for a creating a new custom role, as described in the preceding sections.

Learn More

- [Configure Roles](#)

Assign Roles to Users and Groups

- [To Assign a Role from the Role Configuration Screen](#)
- [Learn More](#)

This topic describes how to assign roles from the role configuration screen when the users and groups already exist.

You can also assign existing roles when you create users and groups from the user and group configuration screens. See [To Assign A Role to a User](#) and [To Assign A Role to a Group](#).

See [Configure Roles](#) for general information about configuring roles.

To Assign a Role from the Role Configuration Screen

1. Access the Role Configuration screen. See [Access Role Configuration](#).
2. In the left Role Name panel select the Role that you want to assign.
3. In the right panel, click the **Users and Groups with this Role** tab.
4. To assign a role to a user or withhold a role from a user:
 - a. Locate the user or users to whom you want to assign the selected role from the Users list. You can enter a string in the filter field to locate a specific user.
 - b. Check the Member checkboxes of the users to whom you want to assign the role. A single user can be assigned multiple roles. You can click **Select All** to check all the users.
 - b. Clear the Member check boxes of the users from whom you want to withhold the role. You can click **Unselect All** to clear all the users.
5. To assign a role to a group or withhold a role from a group:
 - a. Locate the group or groups in the Groups list. You can enter a string in the filter field to locate a specific group.
 - b. Check the Member check boxes of the groups to which you want to assign the role. A single group can be assigned multiple roles. You can click **Select All** to check all the groups.
 - b. Clear the Member check boxes of the groups from whom you want to withhold the role. You can click **Unselect All** to clear all the groups.

Learn More

- [Access Role Configuration](#)
- [Configure Groups](#)

- Configure Users

Configure Integrations

- To configure integrations

You can enable and perform basic configuration with external products from the Administration window.

To configure integrations

1. Click the Setup menu in the upper right section of the screen.
2. From the drop-down menu, click **Administration**.
3. Click the **Integration** tab.
4. In the left panel select the product for which you want to configure integration. The fields specific to the selected product appear in the right panel.
5. Check the Enabled check box to enable the integration. Clear this check box to disable the integration.
6. Provide the product-specific information in the text fields.
7. Click **Save**.

AppDynamics for Large Enterprises

This section links to topics useful for planning and managing AppDynamics in large, enterprise-grade environments.

The Controller in Large Environments

- Controller System Requirements
- Tuning for Large Enterprise Environments
- Measuring Disk Performance Using a Script
- Manage Controller High Availability
- Common Commands Used to Set Up Controller HA
- Provisioning Controllers in High Availability Mode
 - Setting Up an SSH Key for Controller Provisioning
- Controller HA Failover and Failback Process
 - Automating HA Controller Failover and Failback
- Controller High Availability FAQ

Agents in Large Environments

- Automate Multi-Agent Deployment

The Java Agent in Large Environments

- Best Practices for Failover Scenarios

Best Practice Guides

- Best Practices for Application Developers
- Best Practices for Performance and Quality Assurance Engineers
- Best Practices for Operations Professionals

Best Practices for Failover Scenarios

- Configure Applications for Production and Backup Data Centers
 - To configure the agents
- Flip Nodes During Datacenter Failover
 - To rename the properties

- To restart the servers

This topic describes how to set up AppDynamics monitoring in multiple data centers for failover scenarios. You configure AppDynamics so that you can monitor your application under normal circumstances and preserve your production-level baselines so you can continue monitoring in case of failover.

The most valuable time to have AppDynamics monitoring and alerting enabled is right after a failover, as this is the most likely time that significant problems have occurred. The procedure that we recommend sets up AppDynamics to use production-level baselines for your application after failover even when the failover node has no significant load history.

You accomplish this by performing the following tasks:

- Create the AppDynamics business application in each data center, using the same AppDynamics configuration settings in each application. Configure all nodes in one data center to point to the application for that data center.
- In case of failover of production load to another data center, flip the node configuration to point the nodes to the alternate application.

Configure Applications for Production and Backup Data Centers

Set up two data centers with exactly the same structure in terms of tiers, nodes and business transactions. One data center is for production, the other for backup.

Configure the agents for both applications to use the same Controller.

In addition, configure the Application Name, Tier Name, Node Name, and Unique Host ID properties for every app agent and every machine agent in both data centers. The Unique Host ID property provides identification for a node when you use the same node name for multiple nodes on the same physical machine. It is absolutely required to enable the model described here. For information on how to set it, see [Unique Host ID Property](#).

To configure the agents

1. Configure the Controller Host property and Controller Port property for the app and machine agents so that these settings are the identical for all the agents in both datacenters. This sets the agents in both datacenters to use the same controller.
2. Configure the app agent and the machine agent Application Name, Tier Name, Node Name, and Unique Host ID properties for the application that you will deploy at the production datacenter and at the backup datacenter.

The Application Name and Unique Host ID properties must have different settings on the production and backup datacenters. AppDynamics recommends that you use a string, such as "Prod", to identify one application as the production application and another string such as "Backup" to identify the other as the backup application. Whatever convention you use, it is absolutely necessary for the Unique Host ID properties to have different values on the production and backup servers. It does not matter whether the Tier Name and Node Name properties are different or the same in both datacenters.

The configuration below suggests an appropriate naming scheme.

Production Datacenter

Node Name: Node 1

App Name: Prod
Tier Name: Tier 1
Unique Host ID: ProdMachine1

Node Name: Node 2

App Name: Prod
Tier Name: Tier 1
Unique Host ID: ProdMachine2

Node Name: Node 3

App Name: Prod
Tier Name: Tier 2
Unique Host ID: ProdMachine3

Node Name: Node 4

App Name: Prod
Tier Name: Tier 2
Unique Host ID: ProdMachine4

Backup Datacenter

Node Name: Node 1

App Name: Backup
Tier Name: Tier 1
Unique Host ID: BackupMachine1

Node Name: Node 2

App Name: Backup
Tier Name: Tier 1
Unique Host ID: BackupMachine2

Node Name: Node 3

App Name: Backup
Tier Name: Tier 2
Unique Host ID: BackupMachine3

Node Name: Node 4

App Name: Backup
Tier Name: Tier 2
Unique Host ID: BackupMachine4

Flip Nodes During Datacenter Failover

If data center failover occurs:

1. Rename the Application name and Unique Host ID properties in the agent configurations in the production application to point to the backup application and vice-versa.
2. Restart the servers.

To rename the properties

1. Rename the Application Name and Unique Host ID properties of the app and machine agents for the application on the backup datacenter to the names that were originally configured for the application at the production datacenter.
2. Reconfigure the Application Name and Unique Host ID properties of the app and machine agents for the application on the production datacenter to the names that were originally configured for the application at the backup datacenter.

Now your app agent and machine agent configurations should look like this:

Production Datacenter

Node Name: Node 1

App Name: Backup
Tier Name: Tier 1
Unique Host ID: BackupMachine1

Node Name: Node 2

App Name: Backup
Tier Name: Tier 1
Unique Host ID: BackupMachine2

Node Name: Node 3

App Name: Backup
Tier Name: Tier 2
Unique Host ID: BackupMachine3

Node Name: Node 4

App Name: Backup
Tier Name: Tier 2
Unique Host ID: BackupMachine4

Backup Datacenter

Node Name: Node 1

App Name: Prod
Tier Name: Tier 1
Unique Host ID: ProdMachine1

Node Name: Node 2

App Name: Prod
Tier Name: Tier 1
Unique Host ID: ProdMachine2

Node Name: Node 3

App Name: Prod
Tier Name: Tier 2
Unique Host ID: ProdMachine3

Node Name: Node 4

App Name: Prod
Tier Name: Tier 2
Unique Host ID: ProdMachine4

To restart the servers

1. Kill each machine agent process and re-start it so that it will pick up its new properties.
2. Restart each application server.
This will restart the app agents with their new properties.
3. Verify that the agents that was originally sending data to the backup application are now sending the new load data to the production application and that the agents that was originally sending data to the production application are sending new load data to the backup application.

Implement SSL

- Controller
 - On-premise
 - SaaS
- App Agents
- Machine Agents

Controller

Implementing SSL requires configuring the agent-controller communication properties. Both the agent and the controller must use the correct HTTPS port setting. If you are using an AppDynamics SaaS controller you need to configure your agents to use the HTTPS port.

AppDynamics Controller SSL Port settings are as follows:

On-premise: port 8181
SaaS Controller: port 443

On-premise

To implement SSL for controller-agent communication do the following:

- Set the application server primary port to 8181, see [Controller Port Settings](#).
- Install a trusted certificate, see [Controller SSL and Certificates](#).
- Configure your agents for SSL, see [Agents](#).
- (Optional, but recommended) Set up a reverse proxy server inside your firewall.

SaaS

To implement SSL in a SaaS environment do the following:

- Configure your agents for SSL, see [Agents](#).
- Set the controller-port to 443
- Set the controller-ssl-enabled property to true.

App Agents

To configure your agents for SSL, set the SSL related properties:

- Set controller-ssl-enabled to true
- Set the controller-port to the correct value for either on-premise or SaaS controller

For more details:

See [App Agent for Java Configuration Properties](#).

See [App Agent for .NET Configuration Properties](#).

Machine Agents

For the Java environment, see [Machine Agent Configuration Properties](#).

For the .NET environment, see [Configure the Machine Agent for .NET Environments](#).

Export and Import Business Application Configurations

- [Copying Business Application Configurations](#)
 - To export an application configuration
 - To import configuration information
- [Learn More](#)

This topic describes the how to use the import and export functionality to copy configurations from one AppDynamics business application to a new business application.

Copying Business Application Configurations

You can export a business application configuration and import it as a new business application.

The two applications can be on the same or different Controllers. If on different Controllers they must be using the same major version of AppDynamics.

The configuration information is in an XML file that includes:

- Snapshot collection settings
- Call graph settings
- SLA configurations
- Error configuration
- Stall configuration and Business Transaction thresholds
- HTTP and SQL Data gatherer settings
- Tier information
- Custom entry point configuration for Business Transactions
- Memory configurations
- Metric baselines

The configuration information does not include data for:

- Events
- Policies
- Alerts
- Metrics

To export an application configuration

1. In the upper right menu bar, click **Applications -> Export Application**.
2. In the Application Export window, select the application to export from the drop down list.
3. Click **Export**.
AppDynamics creates an XML file containing application configurations.

Tip: If you receive 401 error while exporting the configuration, use "@customer1" after the username for the export application option to work. Customer1 is the default account name.

To import configuration information

 **Note:** Application import will not work over HTTPS if you use a self-signed SSL certificate. Use a customer certificate. See Controller SSL and Certificates.

1. In the upper right menu bar, click **Applications -> Import Application**.
3. Select the XML file that was exported.
4. Enter a name for the application.
5. Click **Import**.

AppDynamics creates a new application with the copied configuration.

Learn More

- Import or Export JMX Metric Configurations
- Hierarchical Configuration Model
- Upgrade the App Agent for Java
- Upgrade the App Agent for .NET

Internationalization

- Internationalization Support in AppDynamics
 - To enable double-byte characters in the UI
 - Learn More

Internationalization Support in AppDynamics

Starting with version 3.4, AppDynamics supports double- and triple-byte characters in the agents, Controller, and UI. You can:

- enter double- or triple-byte characters into text fields in the UI
- view double- or triple-byte characters from applications

This feature requires a complete reinstall of the Controller. There is no upgrade support for this feature.

By default the Controller UI uses single-byte fonts. After you install version 3.4 or later, you can change the UI font setting for each user.

To enable double-byte characters in the UI

1. In the upper right menu bar, click **Settings -> My Settings**.
2. Click **Use System Font**.

You do not have to explicitly save this preference; the setting takes effect when you click the option.

No restart is required.

Learn More

- [Install the Controller](#)
- [Set User Preferences](#)

Automation

Automation features in the AppDynamics UI are hidden by default. To show the automation features:

1. Click the Setup menu in the upper right section of the UI:
2. From the drop-down menu, click **My Preferences**.
3. In the Advanced Features panel check Show Cloud Auto-Scaling features.
The menu displays in the lower part of the left navigation bar.

Workflow Automation

- [Workflows](#)
- [Process for Creating Workflows](#)
- [Learn More](#)

Workflows

A workflow is an automated event triggered by a policy action. For more information about policy actions see [Policies and Actions](#).

A workflow is a sequence of steps, and each step consists of one or more tasks. These tasks are executed on a machine instrumented by the Machine Agent.

Workflow steps are predefined; you add your tasks to the steps. Steps include:

- Create and configure new machine.
- Terminate machines.
- Configure machines.
- Configure a specific machine.
- Add manual activities.

AppDynamics provides templates for common tasks, such as Ant or BZip. You can create your own custom tasks.

Process for Creating Workflows

To create a workflow, follow these instructions:

1. If you have not already done so, [Install the Machine Agent](#).
If you are using .NET, install the standalone Machine Agent, as the embedded machine agent does not support workflow automation.
See [Install Standalone Machine Agent in .NET environment](#).

2. (Optional) Create any operating system scripts required for a custom task.
3. [Create the workflow and its steps](#).

If you are orchestrating a compute cloud see [Workflow for Cloud Orchestration](#).

4. [Add tasks to the steps](#).
5. Save and execute the workflow. Verify that it works as intended.
6. Automate the workflow using a policy action.

Learn More

- Create a Workflow
- Workflow for Cloud Orchestration
- Policies

Create a Workflow

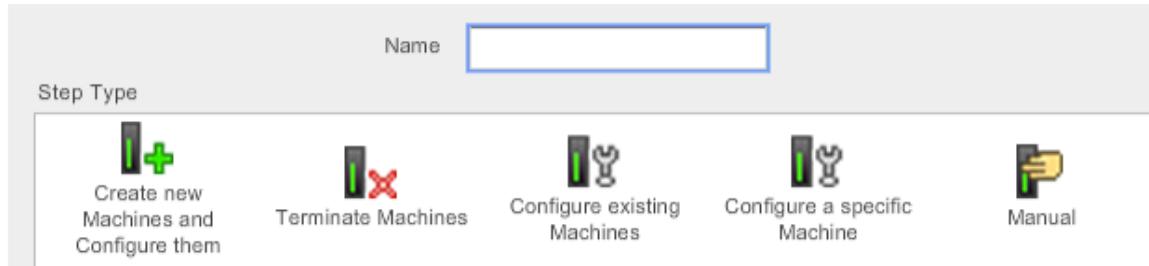
- To create a new workflow
- To add a "Create New Machine" step
- To add a "Terminate machines" step
- To add a "Configure existing machines" step
- To add a "Configure a particular machine" step
- To add a "Introduce Manual User Intervention" step

This topic describes how to create a workflow.

To create a new workflow

1. Click **Alert & Respond -> Cloud Auto-Scaling -> Workflows** the left navigation panel.
Automation features must be shown for this item to be visible. See [Automation](#) if you do not see this item.

2. Click **New +**.
3. Enter a name and provide a brief description.
4. Click **Create Workflow**.
3. Click the Steps + icon to add a step to the workflow.
4. Enter the name of the step.
5. Select the type of step:
 - Create and configure new machine
 - Terminate machines
 - Configure existing machines
 - Configure a specific machine
 - Manual



6. Add Tasks for Workflow Steps.

To add a "Create New Machine" step

1. Select the Compute Cloud for launching the virtual machines (VMs).
2. Select the registered Images and enter the number of virtual machines (VM) you want to create in your Compute Cloud.
3. Click **Create Workflow Step** to create the step for your workflow.

Create Workflow Step

Name	<input type="text" value="Create Machine"/>			
Step Type				
 Create new Machines and Configure them	 Terminate Machines	 Configure existing Machines	 Configure a specific Machine	 Manual
Create one or more machines and execute Tasks on each machine after it has started.				
<input checked="" type="checkbox"/> Setup Default Parameters for this Step ?				
Compute Cloud	<input type="button" value="Select a Compute Cloud: ▾"/>	Register a Compute Cloud		
Launch instances of Image	<input type="button" value="Select an Image: ▾"/>	Register an Image		
Number of Machines to Create	<input type="text" value="1"/>			
Timeout (1 - 60 min)	<input type="text" value="30"/>			

4. Add Tasks for Workflow Steps.

To add a "Terminate machines" step

Use this step to terminate a specific machine or a group of machines that were created using the "Create new Machines and Configure them" step.

1. Select the compute cloud and Image from where you launched the machines.
2. Select the tier or tiers on which you want to terminate the machines.
3. Click **Create Workflow Step** to create the step for your workflow.

Create Workflow Step

Name: Terminate machines

Step Type:

- Create new Machines and Configure them
- Terminate Machines**
- Configure existing Machines
- Configure a specific Machine
- Manual

Terminate machines. Tasks can be executed on each machine before it is shut down.

Setup Default Parameters for this Step [?](#)

Define which Machines to Terminate

Compute Cloud: Select a Compute Cloud: ▾

Image that Machines were launched from: Select an Image: ▾

Tiers running on the Machines:

 E-Commerce	 Inventory
 Order Processing	

Hold control/command to select multiple items, or to de-select an item.

Timeout (1 - 60 min): 30

Number of Machines to Terminate: Terminate all machines that match the above criteria

4. Add Tasks for Workflow Steps.

To add a "Configure existing machines" step

Use this step to run tasks on the virtual machines that were previously created as a "Create new Machines and Configure them" step.

1. Click **Create Workflow Step** to create this step for your workflow.

Create Workflow Step

Name: Configure machines

Step Type:

- Create new Machines and Configure them
- Terminate Machines
- Configure existing Machines
- Configure a specific Machine
- Manual

Execute tasks on all the machines which match the criteria below.

Setup Default Parameters for this Step [?](#)

Define which Machines to configure

Compute Cloud: Select a Compute Cloud: | ▾

Image that Machines were launched from: Select an Image: | ▾

Tiers running on the Machines:

 Java	E-Commerce	 Java	Inventory
 Java	Order Processing		

Hold control/command to select multiple items, or to de-select an item.

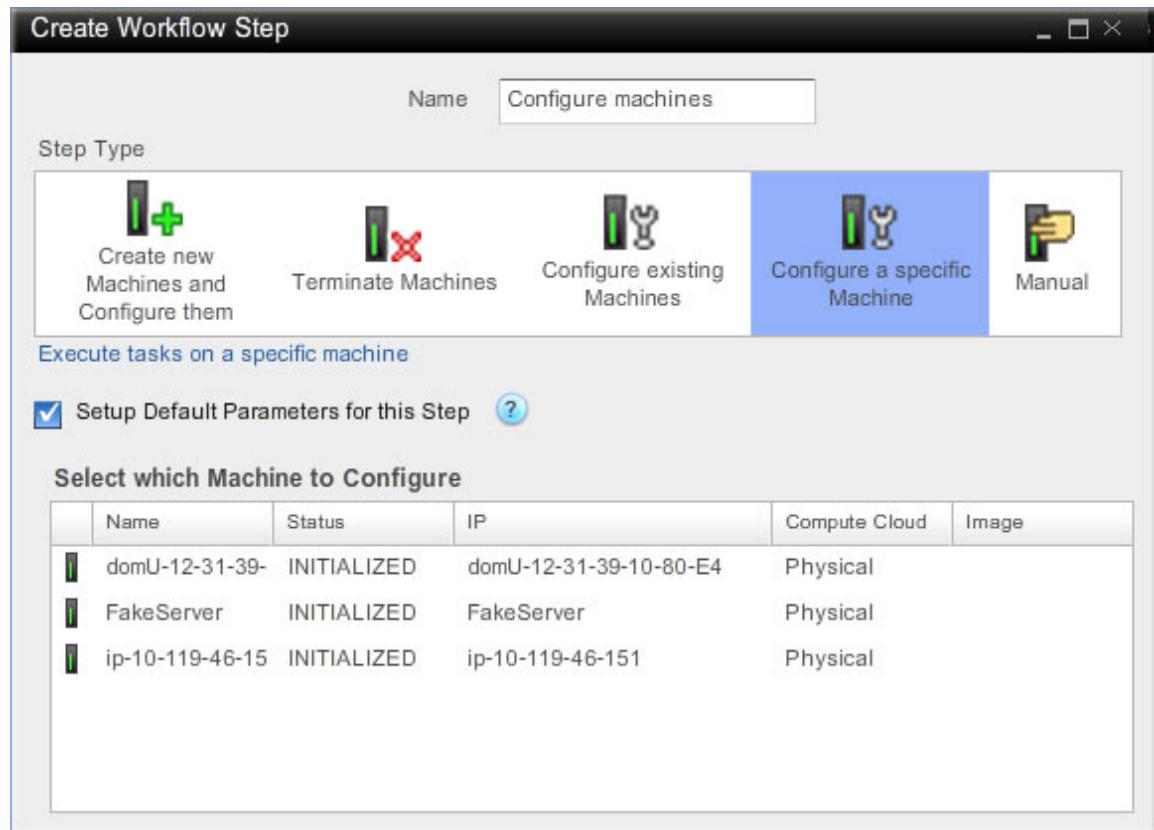
2. Add Tasks for Workflow Steps.

To add a "Configure a particular machine" step

Use this step to run tasks on a machine that has been "imported into the system". You can import the machines between different providers.

1. Click **Create Workflow Step** to create the step for your workflow.

2. Add Tasks for Workflow Steps.



3. Install the Machine Agent on the imported machine.

4. Register the Machine Agent with your application either through auto-agent registration or manually using **Systems -> AppDynamics Agents**.

To add a "Introduce Manual User Intervention" step

Use this step if your workflow needs manual user intervention. This step will pause workflow execution until the operator resumes execution.

1. Click **Create Workflow Step** to create this step for your workflow.

2. Add Tasks for Workflow Steps.

Add Tasks for Workflow Steps

Tasks are the basic unit of execution for a workflow and are added for each step of the workflow.

You must first create a workflow step before adding a task to the workflow.

See [Create a Workflow](#).

To add a task for a workflow step

1. Click **Alert & Respond -> Cloud Auto-Scaling -> Tasks**.

2. Click **New** (the plus sign). The Create Task window opens.

3. Name the task.

4. Select a task template from the task template library. See [Available Task Templates](#).

If there are no templates that suite your needs, you can click **Create Task Templates** to create a custom task template and add it to the library. See [Custom Tasks](#).

5. Enter in the input parameters for the task. Asterisks indicate the required parameters.

- If needed specify a workflow variable that will be evaluated during execution.

6. If needed specify the output parameter bindings.

7. Click **Create Task**.

The types of workflow variables include:

- **Shared Variables**

- **Tier IP Addresses:** This variable will return a comma separated list of IP addresses of nodes in a tier.
- **Machine IP:** This variable will return the IP address of the machine where the task is executing.
- **Task Output:** This variable will return the value which is output from another task. To use a task's output, the Task must define output parameters, and it must have executed before the Task that uses it.

Create Task

Name * Create Archive

Select Task Template:

Name	Description
<input checked="" type="checkbox"/> AddTomcatWorker	Adds a Tomcat worker to ar
<input checked="" type="checkbox"/> Ant	Runs Ant on a supplied bu
<input checked="" type="checkbox"/> BZip2	Zips to a bzip file using
<input checked="" type="checkbox"/> Cab	Creates Microsoft c

Input Param... Output Param...

source-file * [] destination-file *

Click Here to Add your workflow variable

2 Select Workflow Variable

Variable Type: Shared Variable | **Tier IP Addresses** | Machine IP | Task Output

This variable will return a comma separated list of IP addresses of Nodes in a Tier

Select Tier

Name	Description
<input checked="" type="checkbox"/> Ecommerce	
<input type="checkbox"/> Inventory	
<input type="checkbox"/> Order Processing	

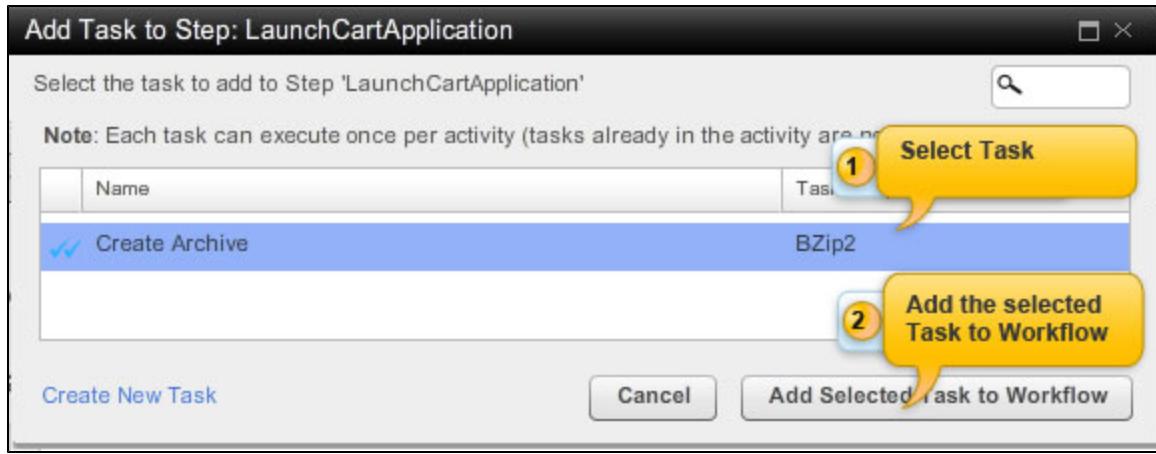
Select Nodes All Nodes in the Tier

Use Variable: \${context.tier-ip-addresses.E}

3 Add the Workflow Variable

Cancel Select

5. Select the newly created task to add to your workflow.



This action will add the task to your workflow.

Applications > ACME Online Bookstore > Workflows

Step 1: CreateMachine

Step Summary: Create 1 instances of image 'Cart App Image' in

1 Tasks

- Create Archive

To edit a task

- In the left navigation pane click **Alert & Respond** > **Cloud Auto-Scaling** > **Tasks**.
- Select the task.
- Click **Edit** or double-click to open the Task window.
- Change the parameters and/or bindings.
- Click **Save**.

Learn More

- Custom Tasks

Available Task Templates

The following task templates are available in the Task Template Library by default.

AddTomcatWorker
Ant
Apt
Attribu
BUnzip2
BZip2
Cab
CheckSum
Chgrp
Chmod
Chown
ConcatFiles
ControlOutcomeTask
CopyFile
CopyFiles
CreateDirectory
CreateMySqlSchema
DeleteDirectory
DeleteFile
DeleteFiles
DownloadFile
Ear
ExecuteShellScript
ExecuteShellScriptNonBlocking
ExecuteSql
FixCRLF
GUnzip
GZip
Jar
Javac
Jspc
MoveDirectory
MoveFile
MoveFiles
PatchFile
RemoveTomcatWorker
ReplaceRegEx
ReplaceToken
RestartApache
RestartMySQL
RestartTomcat
Rmic
Rpm
SecureCopy
SetProxy
SignJar
StartMySQL
StartServer
StartTomcat
StopMySql
StopTomcat
SynchDirectory
Tar
TouchFile
TruncateFile
UniqueKeyGenerator
UnTar
UnZip
War
Zip

Custom Tasks

- Custom Tasks and Task Templates
 - Custom Tasks Using XML
 - To create a task.xml file
 - To create a run.xml file
 - To package the XML files as a Zip archive

- To add the Zip archive as a task template
- Custom Tasks Using Shell or Batch Scripts
- Learn More

Custom Tasks and Task Templates

Tasks are the actions used in workflows to automate your system. For more information see [Workflow Automation](#).

AppDynamics provides task templates that you can modify for your use. You can create your own tasks using either:

- a Zip archive containing custom XML files
- a shell or batch script as an argument to the ExecuteShellScripts template

Custom Tasks Using XML

AppDynamics defines specific XML to use for custom tasks.

You use the specified XML format and create the following two XML files:

- task.xml
- run.xml

To create a task.xml file

The task.xml file must have the structure as shown below.

It must have the <java-task> child element.

Do not rename this file.

```
<task>
  <name>Name of the File</name>
  <display-name>Display Name</display-name>
  <description>Description</description>
  <type>java</type>
  <task-arguments>
    <argument name="Arg1_Specified_In_Run.XML_File" description="Description"
      is-required="Specify_Boolean_Value_true/false" type="Type_Of_Argument"/>
    <argument name="Arg2_Specified_In_Run.XML_File" description="Description"
      is-required="Specify_Boolean_Value_true/false"/>
  </task-arguments>
  <java-task>
    <impl-class>com.singularity.ee.agent.systemagent.task.ant.AntTask
    </impl-class>
  </java-task>
</task>
```

To create a run.xml file

The run.xml file contains the details for the action that will be executed when this task is triggered.

Use following sample structure to create the run.xml file.

```

<?xml version="1.0" encoding="UTF-8"?>
<project name="run" default="run" basedir=".">
    <property file="args.properties" />
    <target name="run">
        <"Operation_Name_that_the_task_will_perform" file="$\{Arg_1\}">
        todir="$\{arg_2\}"/>
    </target>
</project>

```

To package the XML files as a Zip archive

Add the task.xml and run.xml files to a Zip archive and name this Zip archive the name of the task.

To add the Zip archive as a task template

1. From the **Cloud Auto-Scaling** menu select **Task Library**.
2. Click **New**.
3. Provide the name and description for the task.
4. Upload the Zip file containing the XML task files.
5. Click **Create Task Template**.

AppDynamics adds the task template to the Task Library where you can use it in your workflows.

Name	Description	Task Zip File	Sample Task
AddTomcatWorker	Adds a Tomcat worker to an Apache server by modifying the AddTomcatWorker.zip	Yes	
CopyFiles_Demo	Copies specified files from source to destination directories	CopyFile.zip	No

Task Template: CopyFiles_Demo

- Task Properties** **Task Input Parameters** **Task Output Parameters** **task.xml**
- Name: CopyFiles_Demo
- Description: Copies specified files from source to destination directories
- Task Zip File: CopyFile.zip [Download File](#)
- Overwrite?

Custom Tasks Using Shell or Batch Scripts

1. Create the shell or batch script file on the system.
2. In the Workflows window, select the task template ExecuteShellScripts. It will accept the shell or batch script file as an argument to the task. For instructions see [Add Tasks for Workflow Steps](#).

Learn More

- Workflow Automation
- Add Tasks for Workflow Steps

Workflow for Cloud Orchestration

- Prerequisites for Cloud Computing
- Learn More

Before you create a workflow for cloud orchestration, you need to register the compute cloud and the machine image with AppDynamics and set the <enable-orchestration> Java App Agent and Machine Agent configuration properties.

These are in addition to the workflow tasks described in [Create a Workflow](#).

Prerequisites for Cloud Computing

Sign up for an account with a cloud service provider. The service provider will provide you with the credentials that you need to register the account with AppDynamics.

Register the cloud with AppDynamics. See [Compute Clouds](#). The registered cloud will appear as an item in the Select a Compute Cloud drop-down menu used for creating a machine as part of a workflow.

Register the machine image. See [Machine Images and Instances](#). The image will appear as an item in the Select an Image drop-down menu used for creating a machine as part of a workflow.

Learn More

- Compute Clouds
- Machine Images and Instances
- Use the ITask Interface for Cloud Orchestration

Compute Clouds

- To register a compute cloud

To use a compute cloud, first register your cloud account in AppDynamics.

To register a compute cloud

1. Click the **Automation** menu at the bottom of the left navigation panel.
Automation features must be shown. See [Automation](#) if you do not see this item.
2. From the Automation menu click **Compute Clouds**.
3. Enter the name and description of the compute cloud instance.
4. From the Type drop-down menu select the type of the cloud from among the solutions that AppDynamics supports for automation integration.
5. Enter the account credentials that are specific to the cloud type. These typically include an identifier or name, the access key and the secret access key.
6. Click **Register Compute Cloud**.

Machine Images and Instances

- Managing Images
 - To register an image
 - To launch an instance of a machine image
 - To restart or stop an instance of a machine image
- Learn More

This topic discusses managing cloud-based machine images.

Before you can register an image you register the cloud provider. For instructions see [Compute Clouds](#).

Managing Images

Before AppDynamics can manage machine images, you must register the image with the system.

To register an image

1. Click the Automation menu at the bottom of the left navigation panel.
See [Automation](#) about displaying the menu if you do not see this item.
2. In the menu click **Images**.
3. Click **Register Image**.
4. Enter the name and description of the image.
5. Select the operating system for the image.
6. Select the compute cloud that you have previously registered. For instructions about registering compute clouds see [Compute Clouds](#).
7. Select the format for the image.
8. Provide additional information required in the dynamically-generated fields.
9. Click **Register Image**.

To launch an instance of a machine image

After the image has been registered, you can launch the instances of it.

1. In the Images window, select the image that you want to use.
2. Click **Launch Instance**.

All instances that are launched for this image are listed under the details for that image or in the Machines window.

The following screenshot displays instances for the "Ecommerce Node V1" image. You can also restart or terminate the instances from this screen.

▼ Instances of this Image					
	 View Details	 Restart	 Terminate		
	Name	Status	IP	Image	Compute Cloud Type
	i-22d7ec4d	STARTED	ec2-50-17-80-66.compute	Ecommerce Node V1	Amazon EC2
	i-30af945f	STARTED	ec2-50-16-103-15.comput	Ecommerce Node V1	Amazon EC2

To restart or stop an instance of a machine image

1. In the Automation menu, click **Machines**.
AppDynamics lists all instances that were launched from the Images window.
2. Select the instance that you want to restart or stop.
3. Click **Restart** or **Terminate**.

Learn More

- [Compute Clouds](#)
- [Workflow for Cloud Orchestration](#)

Use the ITask Interface for Cloud Orchestration

Your task file must implement the com.singularity.ee.agent.systemagent.api.ITask interface that is bundled with the Machine Agent.

This interface has following two operations:

- execute()
- stop()

AppDynamics highly recommends creating shell script tasks when possible.

Contact the [AppDynamics Support Team](#) if you need additional help in creating a Java task.

AppDynamics Best Practices

AppMan Advice



"A good place to begin is with our mission – deliver maximum monitoring visibility thru minimal effort."

Best Practices for Performance and Quality Assurance Engineers

- Compare Performance Metrics
- Compare Application Flow
- Compare Critical Code
- Test Load
- Examine Errors and Exceptions
- Examine Memory Usage
- Learn More

Performance and QA engineers can use AppDynamics to help certify new versions of applications before they are released to production. You can quickly detect problems and errors and discover their root cause.

Specifically you can:

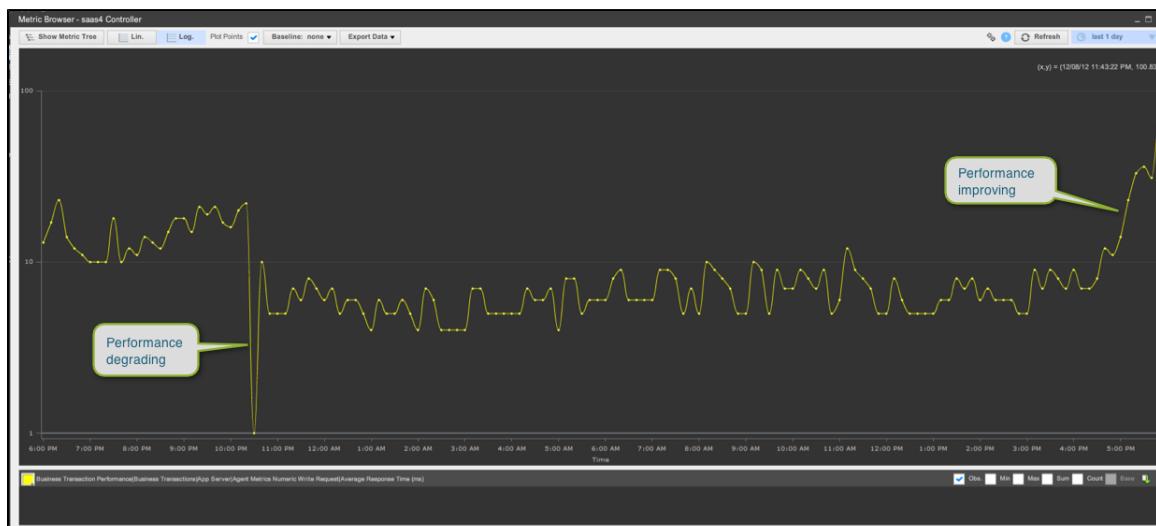
- Identify performance regression
- Identify functionality regression
- Compare critical sections of code
- Perform load testing to verify response to peak load
- Discover new errors being produced by the new release
- Identify changes in memory usage that could have negative effects on the system

You can create summary before and after performance reports in PDF format for management. See [Reports](#).

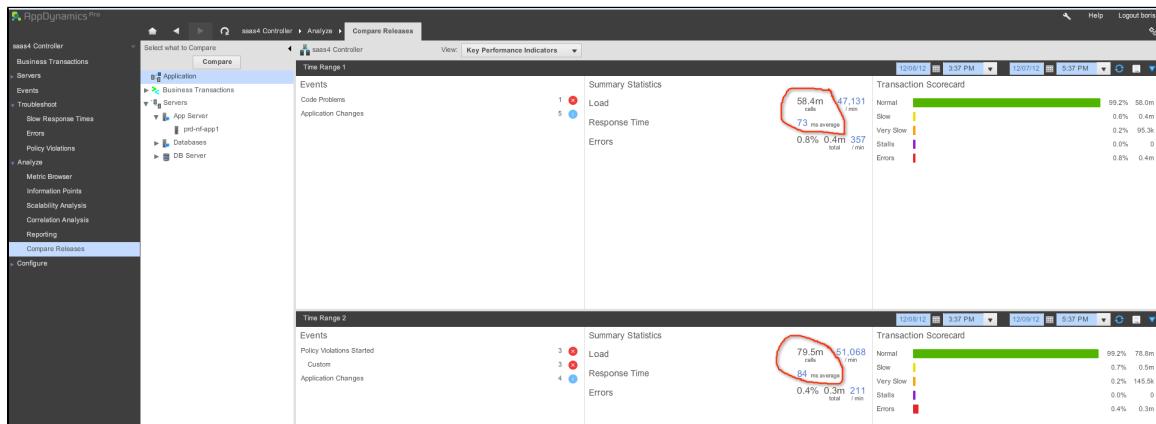
Compare Performance Metrics

Before and after the release, use the AppDynamics Metric Browser to graph key performance metrics for each business transaction.

Compare graphs taken before and after the new release. For example, the graph below shows average response time over a twelve hour period. See [Infrastructure Metrics](#).



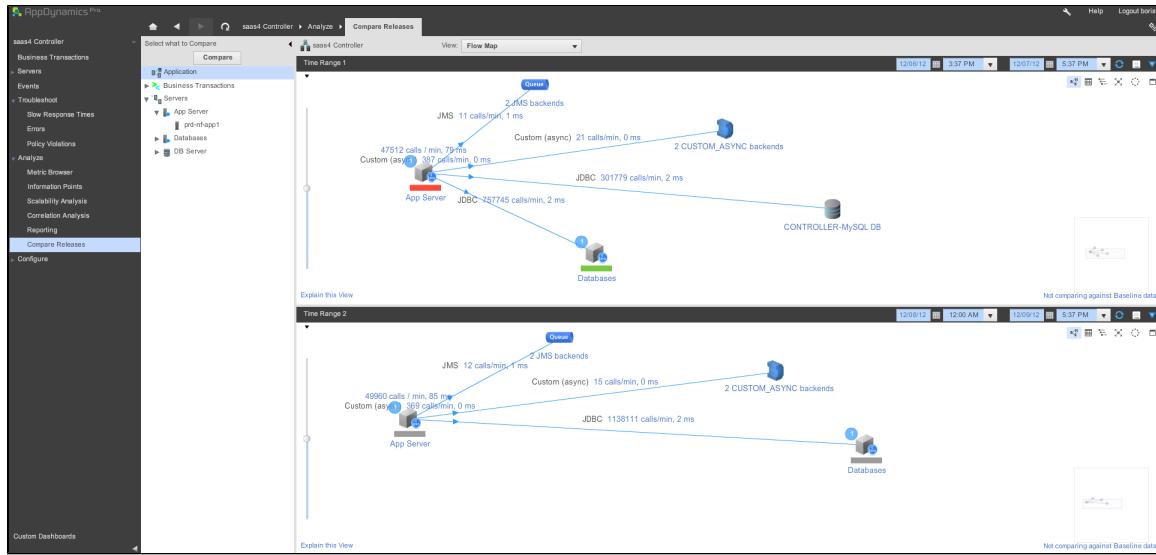
You can compare summary performance indicators (load, average response time, error rate) from two different time periods, by Business Transaction, by tier (application component) or by node (app server). Note the difference in load and response time between the two days in the Compare Releases window below. See [Compare Releases](#).



Compare Application Flow

You can see changes in application flow by comparing flow maps generated before and after the new release. You can view flow maps by Business Transaction in its Business Transaction Dashboard or by tier in each Tier Dashboard. See [Tier Dashboard](#) and [Business Transaction Dashboard](#).

Flow maps display the flow of traffic, the sequence of calls and their frequency, between tiers. They can tell you if a component is calling obsolete services and if it is not calling vital services. For example in the comparison below you can see that in addition to the difference in call rates the CONTROLLER-MySQL DB in the Dec 6 flow map is no longer present in the Dec 8 flow map.



Compare Critical Code

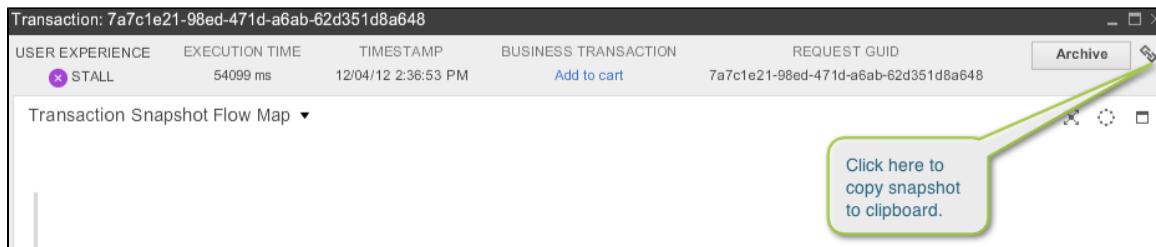
Transaction snapshots provide information about the executing code (using [call graphs](#)), the SQL that was executed, the HTTP that was sent, cookie values, etc. Configurable settings control when snapshots are collected. By default snapshots are collected on a periodic basis and when a Business Transaction is slow or has errors.

You can analyze differences in code paths before and after a new release. See [To Compare Snapshots](#).



Tip: Record any changes in code that affect performance in the new release and show them to developers. It is much faster to resolve problems if you have clearly identified some discrepancies than just to complain that operations have slowed. You can copy a link of the transaction snapshot to the clipboard by clicking the clipboard icon in the upper right corner of the snapshot.

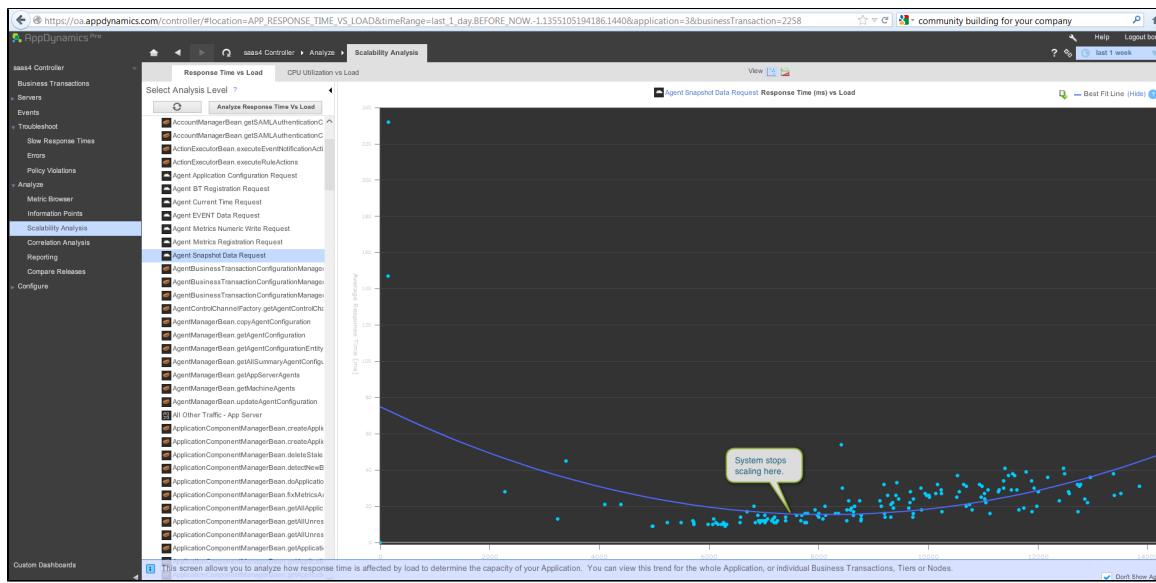
Collect snapshots of all key transactions that effect the performance in important systems. On Java platforms you can make sure to collect some full call graphs on key operations to capture the complete execution path of slow transactions starting from before the time that they started to slow. See [Configure Call Graphs](#).



Test Load

Apply maximum load on the test system while monitoring it with AppDynamics to see how your site performs under stress. You can use tools such as Apica Soasta or LoadRunner to simulate load on your system.

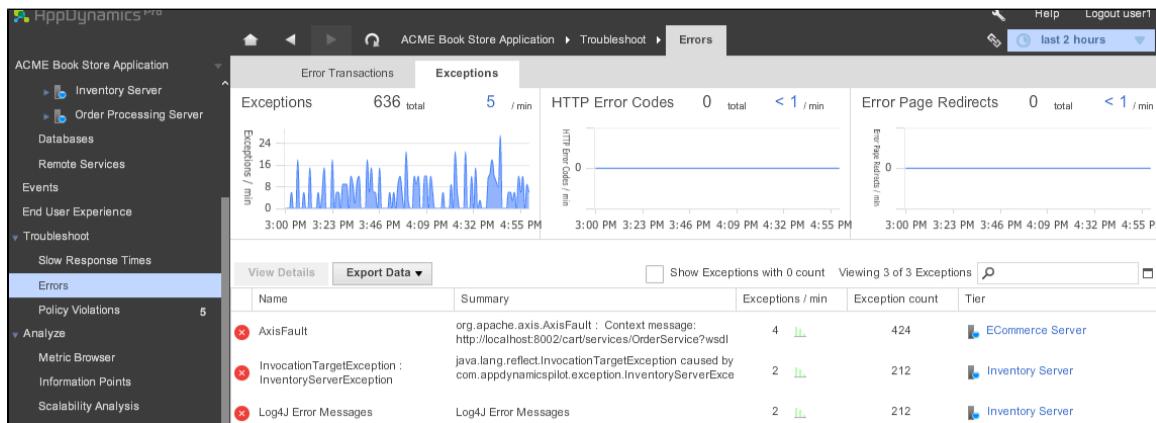
You can measure how response time and CPU utilization degrades with increased load. Scalability analysis helps you to determine the capacity that your system can support and to spot any reduction in performance introduced with new releases. See [Scalability Analysis](#).



Examine Errors and Exceptions

Click **Troubleshoot -> Errors** to view the volume of errors and exceptions that are being produced by various tiers.

Click the **Exceptions** subtab to look for new high-frequency exceptions that were introduced by the new code. Also look for exceptions that are not new but are becoming more frequent. Errors that are trending up are potentially just as dangerous as new errors.



Drill down to the exception detail to see the historical frequency of an exception that is new or trending up.

To drill down on an exception:

1. Double-click the exception.
2. Click the **Occurrences of this Exception** tab to see all the occurrences in the selected time range.
3. Click the **Stack Traces for this Exception** tab to see a stack trace.

The screenshot shows the AppDynamics Pro interface with the following details:

- Left Sidebar:** Shows the navigation tree for the ACME Book Store Application, including Business Transactions, Servers, Events, End User Experience, Troubleshoot, Slow Response Times, Errors (selected), Policy Violations (31), Analyze, Configure, Automate, Custom Dashboards, and Automation.
- Top Bar:** Includes Help, Logout user!, and a time range selector for "last 12 hours".
- Central Content:**
 - Tier:** ECommerce Server
 - Name:** AxisFault
 - Summary:** org.apache.axis.AxisFault : Context message: http://localhost:8002/cart/services/OrderService?wsdl
 - Chart:** Errors Per Minute (last 12 hours) showing fluctuating error counts between 1 and 5.
 - Tabs:** Occurrences of this Exception (selected) and Stack Traces for this Exception.
 - Stack Trace:** A detailed stack trace for org.apache.axis.AxisFault, listing numerous method calls across various Apache XML and SAX parser classes.

Examine Memory Usage

Another important regression to monitor is memory usage for each instrumented app server (node).

Click the Memory tab in the Node Dashboard to analyze:

- The type of garbage collection strategy being used
- How often collections occur
- How much memory is being used

The screenshot shows the AppDynamics Node Dashboard with the following details:

- Left Sidebar:** Shows the navigation tree for the ACME Book Store Application, including Business Transactions, Servers (selected), App Servers (selected), Databases, Remote Services, Events, End User Experience, Troubleshoot, Slow Response Times, Errors (selected), Policy Violations (16), Custom Dashboards, and Automation.
- Top Bar:** Includes Help, Logout user!, and a time range selector for "last 4 hours".
- Central Content:**
 - Memory Tab:** Sub-tabs include Heap & Garbage Collection (selected), Automatic Leak Detection, Object Instance Tracking, and Custom Memory Structures.
 - Heap Section:**
 - Average Utilization: 3% (last 4 hours)
 - Current Utilization: 3%
 - Current Usage: 64 MB
 - Current Committed: 113 MB
 - Max Available: 1712 MB
 - Free: 1648 MB
 - Graph:** A line graph titled "Heap Utilization %" showing memory usage over time from 2:20 PM to 6:04 PM. The Y-axis ranges from 1.8 to 4.2. The graph shows a peak around 3:00 PM and a dip around 3:30 PM.
 - Legend:** Current Usage RANGE (MB) and Max (MB).

The Node Dashboard is a good place to see any changes that may have degraded performance, such as lower heap size or a different garbage collection strategy. Any degradation in this area can lead to performance problems and to systems running out of memory and needing a restart.

For more information about memory see [Troubleshoot Java Memory Issues](#) and [Monitor CLRs](#).

Learn More

- Logical Model
- Metric Browser

- Business Transaction Monitoring
- Business Transactions List
- Business Transaction Dashboard
- Reports
- Compare Releases
- Tier Dashboard
- Transaction Snapshots
- Call Graphs
- Configure Call Graphs
- Node Dashboard
- Troubleshoot Java Memory Issues
- Monitor CLRs

Best Practices for Operations Professionals

- Monitor Overall Application Health
- Identify Failing Business Transactions
- Monitor Application Server Problems
 - General Node Health
 - Hardware Problems
 - Memory Problems
- Analyze Incidents
- Use Policies, Health Rules, and Actions
 - Deep Links in Notifications
- Share Deep Links
- Suggestions for Customizing AppDynamics
 - Custom Dashboards
 - Policies and Alerts
- Learn More

Network Operations Center (NOC) and other front-line support staff can use AppDynamics to:

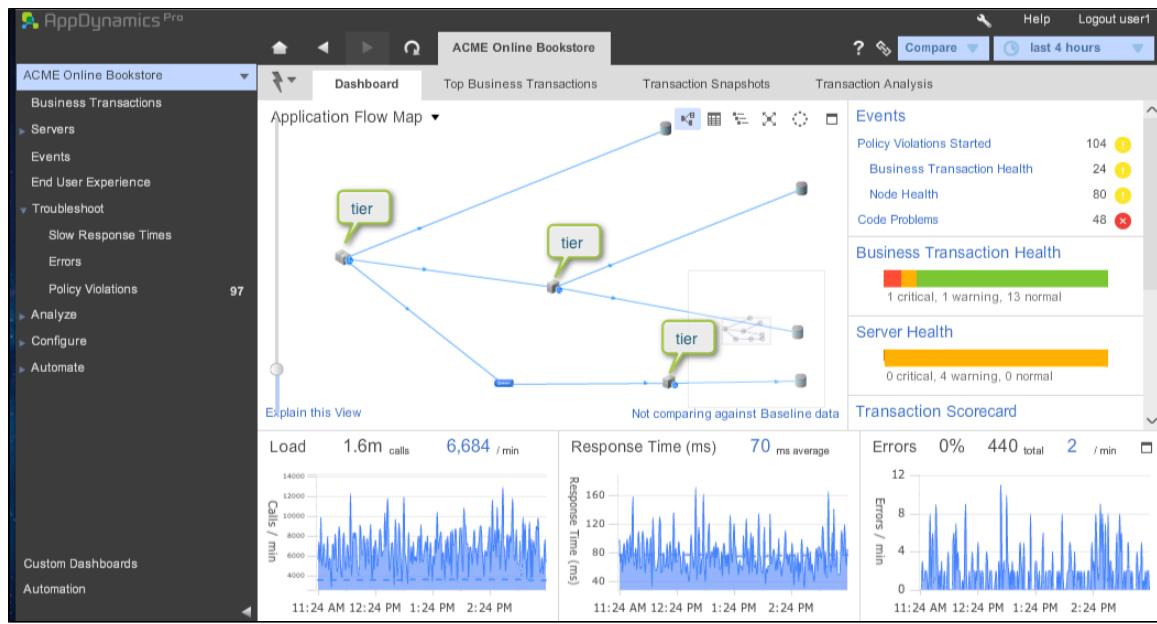
- Assess overall application health by viewing key performance metrics
- Accurately identify trouble spots
- Immediately triage problems and either repair them or alert the responsible team
- Help the team maintain low mean time to recovery (MTTR)

If you are new to AppDynamics, see [Use AppDynamics for the First Time with Java](#).

Monitor Overall Application Health

You can get an idea of overall application health by viewing the Application Dashboard.

The Application Dashboard shows all the tiers and nodes (representing app servers) in the business application and how traffic moves between them. The bottom of the dashboard shows key performance indicators. For details see [Application Dashboard](#).



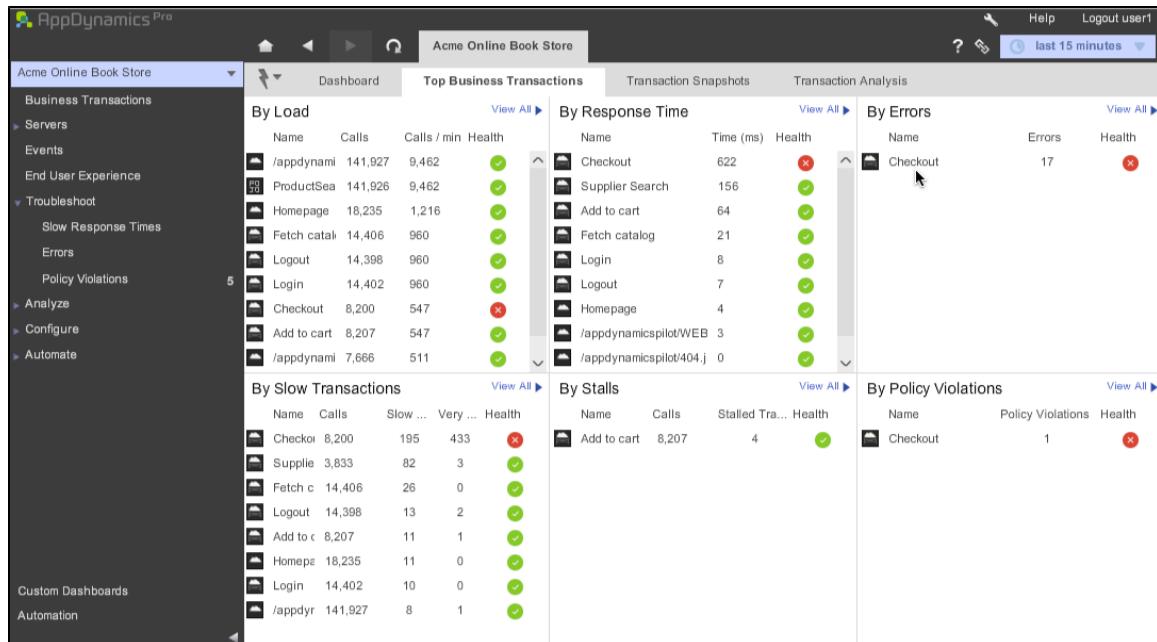
The **Time Range** pulldown in the upper right of the window sets the duration for the monitoring session. Change the time range to see a larger set of monitoring data.

Color indicators show the health of each subsystem; green indicates normal operation, yellow or orange indicates a warning condition, red indicates a critical condition. Click on yellow, orange, or more importantly red indicators to quickly identify trouble spots to investigate.

Identify Failing Business Transactions

Business Transactions organize application load into logical operations that represent user requests, such as Login, Search, Checkout, etc. Operations management can make sure that Business Transactions are properly configured based on the most important user operations. See [Business Transaction Monitoring](#).

In the Application Dashboard click the Top Business Transactions tab to quickly see business transactions sorted by various possible performance problems.



Often the same Business Transaction has problems in multiple performance areas.

Double-click a Business Transaction listing to open its **Business Transaction Dashboard**. Look at the Events panel to see if there are any code problems or the Transaction Scorecard for errors and click on them. For example the Events panel can quickly reveal a code deadlock.

The screenshot shows the AppDynamics Events panel for the 'Acme Online Book Store'. The left sidebar shows navigation options like Servers, Events, and Troubleshoot. The main area displays a grid of events with columns for Type, Summary, Time, Business Transaction, Tier, and Node. Three 'Code Deadlock' events are listed, each with a red error icon. A tooltip points to the first event: 'Code Deadlock' (JVM deadlock detected) at 12/18/12 6:51:45 PM, Tier: E-Commerce, Node: E-Commerce-Node-8000. Below the grid, a detailed view of the first event is shown with tabs for Summary, Details, and Comments (0). The summary details include Severity: Critical, Type: Code Deadlock, Time: 12/18/12 6:51:45 PM, Summary: JVM deadlock detected, Tier: E-Commerce, and Node: E-Commerce-Node-8000.



Tip: Immediately investigate any Business Transaction with an error rate over 10% or a red icon in the Health column. A red icon indicates that performance is abnormally slow compared to its baseline.

Monitor Application Server Problems

To see all the application servers (nodes) in a business application do one of the following to open the App Servers List:

- In the left navigation panel click **Servers -> App Servers**.
- In the Application Dashboard click **Server Health**.

There are three tabs in the App Server List: Health, Hardware and Memory. In any tab click the Grid View icon to see a list of nodes.

The screenshot shows the App Servers list for the 'Acme Online Book Store'. The left sidebar shows navigation options like Servers, App Servers, and Databases. The main area displays a grid of application servers with columns for Name, Tier, Health, App Agent Status, App Ag., JVM, Last JVM Restart, and Machine Agent Status. Four servers are listed: E-Commerce-Node-8000 (Tier: E-Commerce), E-Commerce-Node-8004 (Tier: E-Commerce), Inventory-Node-8002 (Tier: Inventory), and Order Processing-Node-8001 (Tier: Order Processing). A green callout points to the 'grid view icon' in the top right corner of the grid header.

Each node represents an application server. You can sort the listings in ascending or descending value by clicking a column header. See [App Server List](#).

Whenever you identify a node with problems, double-click its listing to open its Node Dashboard. From the Node Dashboard you can investigate problems in detail. You can forward the URL of the Node Dashboard to the team that is responsible for maintaining that app server. See [Node Dashboard](#).

General Node Health

In the App Server List click the Health tab. Then click the Health column header to sort the nodes so that the unhealthy nodes (red followed by yellow) are at the top of the list.

If the health status of a node is red or yellow, double-click the node to open its dashboard.

Hardware Problems

In the App Server List click the Hardware tab.

The screenshot shows the AppDynamics Pro interface with the following details:

- Top navigation bar: Help, Logout user1, last 3 hours.
- Left sidebar: Acme Online Book Store, Business Transactions, Servers, App Servers (selected). Sub-items include E-Commerce (with three nodes), Inventory (with two nodes), and Order Processing.
- Central pane: Title "App Servers". Sub-tabs: Health, Hardware (selected), Memory. Buttons: View Dashboard, Create Tier.
- Data table: Headers: Name, 1 (sorted ascending), Tier, CPU % (current), CPU .., Mem .., Mem .., Disk I., Disk I., Netw..., Network IO KB writes/sec. Data rows:
 - E-Commerce-Node-8000 (Tier: E-Commerce, CPU %: 60)
 - E-Commerce-Node-8004 (Tier: E-Commerce, CPU %: 60)
 - Inventory-Node-8002 (Tier: Inventory, CPU %: 60)
 - Order Processing-Node-800 (Tier: Order Processing, CPU %: 60)
- Bottom right: View: 3 Tiers, 4 Nodes, search icon.

Sort the nodes by CPU % (avg) in descending order to see where CPU usage is excessive.



Tip: In most cases AppDynamics recommends that you restart or kill all application servers that are over 80% CPU.

Sort the nodes by Mem % (avg) to identify servers that are running out of RAM.



Tip: In most cases memory usage 90% and above is considered dangerous and should probably be reduced by changing the cache settings. In the case of a memory leak restart the application running on that machine.

Double-click any node in which CPU % or Mem % is excessive to open its Node Dashboard.

Memory Problems

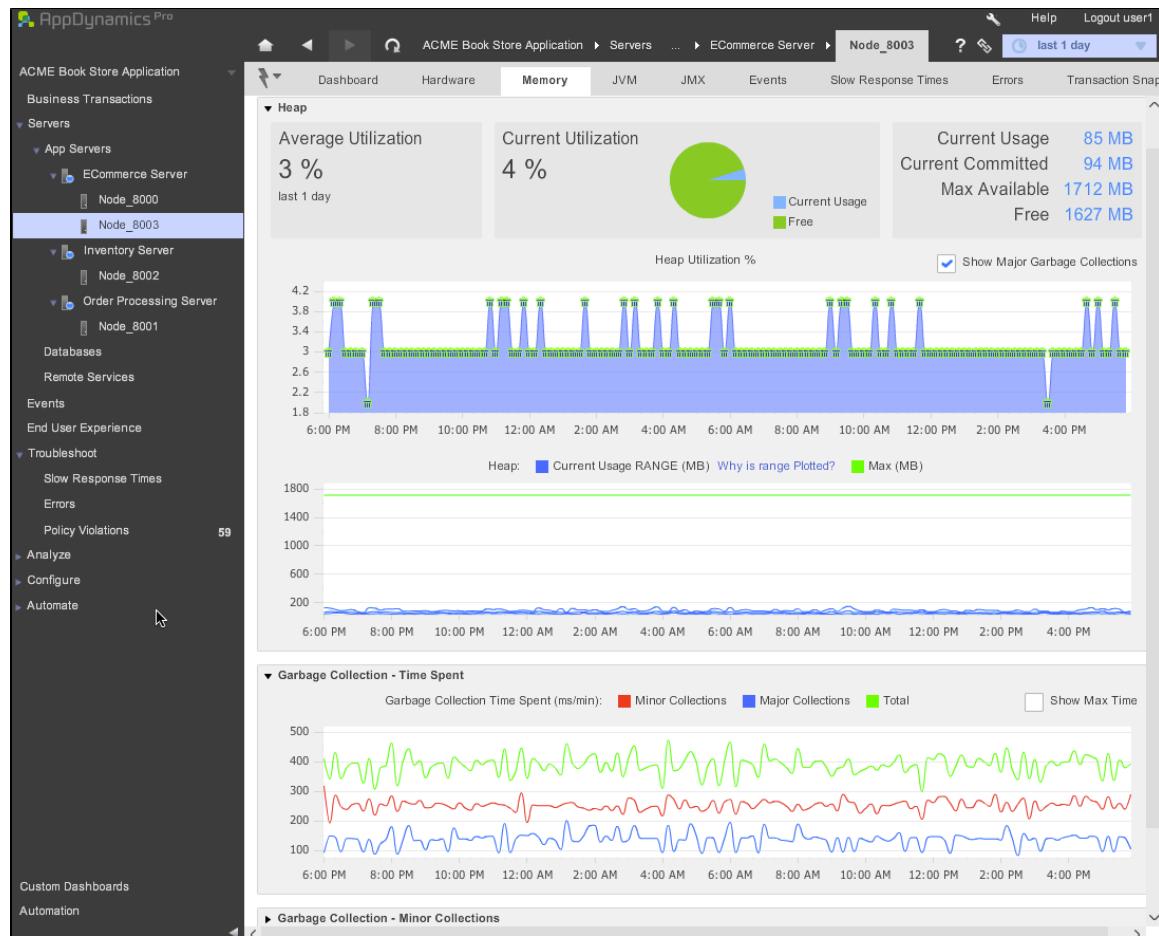
In the App Server List click the Memory tab.

Sort the nodes by GC Time Spent in descending order to identify where garbage collection is taking too long and hindering app performance. Sort the nodes by JVM % Heap in descending order.

The screenshot shows the AppDynamics Pro interface with the following details:

- Top navigation bar: Help, Logout user1, last 4 hours.
- Left sidebar: ACME Book Store Application, Business Transactions, Servers, App Servers (selected). Sub-item: ECommerce Server (selected).
- Central pane: Title "ECommerce Server". Sub-tabs: Dashboard, Nodes (2), Events, Slow Response Times, Errors, Transaction Snapshots, Transaction Analysis. Buttons: View Dashboard, Reload Nodes.
- Data table: Headers: Name, JVM % Heap, Max Heap, JVM CPU Burnt (ms/min), GC Time Spent (ms/min), Major Coll..., Major Col..., Minor Coll..., Minor Col... Data rows:
 - Node_8003 (JVM % Heap: 3.3, GC Time Spent: 393 ms/min)
 - Node_8000 (JVM % Heap: 3.5, GC Time Spent: 377 ms/min)
- Bottom right: View: 2 Nodes, Reload Nodes, search icon.

For nodes in which garbage collection time or heap percentage (%) is excessive, double-click the node to open its Node Dashboard. Click the Memory tab to see detailed information about the application server's memory usage.



Tips: If major garbage collection is occurring more than once an hour, or if the major collection time is more than five seconds, the app server may be experiencing unsustainable memory usage. If heap usage is over 90% for more than a few minutes, the app server may have a memory leak. In this case, restart the application and investigate leakage.

Analyze Incidents

AppDynamics reports all changes in application state as events. Click **Events** in the left navigation pane to see the list of events for the selected time range.

You can double-click any event in the list to get details about the event. In the details window that opens you can get more information about the event. The type of information depends on the type of event. For example, a health rule violation event produces an event summary with a button that links to the dashboard of the affected entity at the time that the event occurred.

See [Filter and Analyze Events](#) for information about viewing events.

Use Policies, Health Rules, and Actions

Policies enable proactive monitoring, allowing you to identify and escalate problems before they become critical. You can configure policies to alert operations staff when any event occurs. See [Policies](#).

There are default health rules based on rules of the Node Health-Hardware, JVM, CLR type that you can modify.

You can create additional health rules based on other critical infrastructure metrics. See [Health Rules](#). Then you match health rules violation events to actions to take when these events occur.

Operations may also receive notifications generated by policies related to application performance that have been created by other teams.

You can also configure actions for events that are not health rule violations. See [Notification Actions](#) for details about configuring notifications to alert staff about health rule violations and other events that you specify.

Deep Links in Notifications

Notifications sent by email contain deep links to details about the rule that was violated or the event that was generated. This is an example of an alert with a deep link:



saas4 Controller Application Policy Violation

Metric Buffer queue too high policy triggered at Tue Dec 11 13:35:55 PST 2012

This policy was violated because the following conditions were met for the **prd-nf-app1 Node** for the last 5 minute(s):

For Evaluation Entity: prd-nf-app1 Node

- condition 1 is greater than 325000. **Observed value = 332523**

[Click here to view details about this Policy Violation.](#) deep link

This is an auto-generated email detailing a policy violation on the *saas4 Controller* application. You are receiving this because you are configured as a recipient on the *Metric Buffer queue too high* policy.

Share Deep Links

In addition to deep links in alerts, any time you are observing a dashboard or troubleshooting an issue in AppDynamics, the URL in the browser is a deep link to a place where anyone can get a head start investigating. For example, if you are examining a Node Dashboard for a node that is experiencing problems, the deep link might be:

```
http://demo2.appdynamics.com/controller/#location=APP_NODE_MANAGER&timeRange=last_6_hours.BEFORE
```

If you are examining a transaction snapshot for a stalled request, the deep link might look like this:

```
http://demo2.appdynamics.com/controller/#location=APP_SNAPSHOT_VIEWER&timeRange=last_15_minutes.E
```

When you escalate a problem to other teams, such as developers, email them the deep link. These links are much more succinct and useful for troubleshooting and resolving issues than log files and long descriptions about where to look.

Conversely, when a problem is escalated to you by another team, request the deep link provided by AppDynamics to help you start troubleshooting.

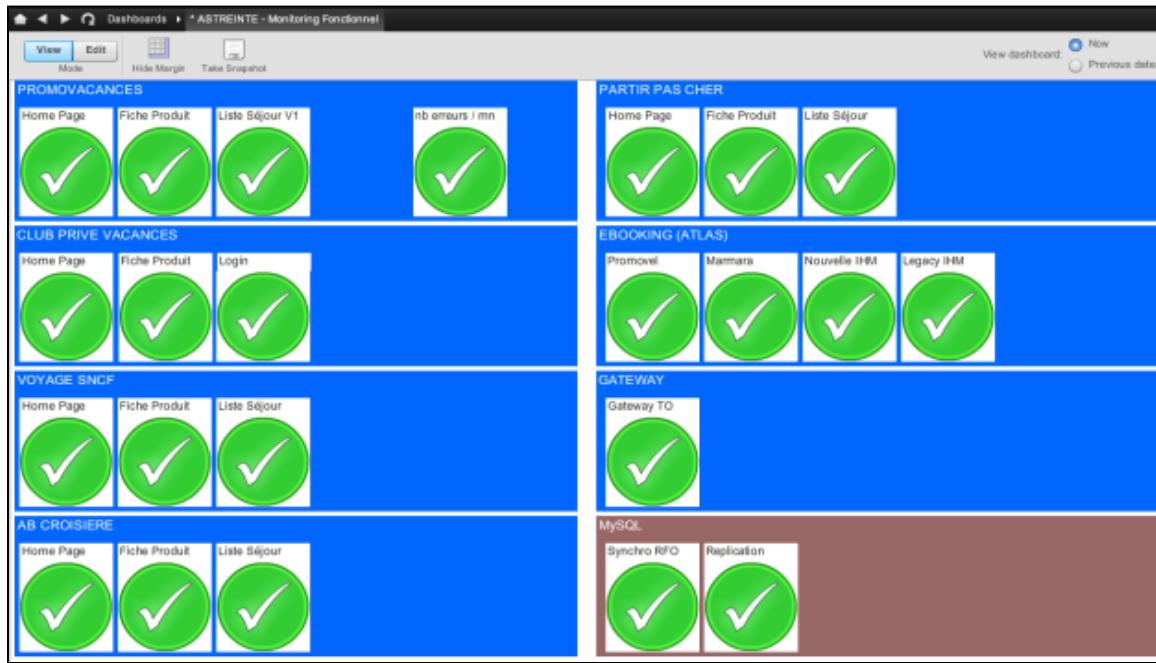
Suggestions for Customizing AppDynamics

Organizations can use custom dashboards, policies, and alerts to customize AppDynamics for NOC and Support staff. Customizations can focus on metrics that the organization deems important to track and the values that define good and bad performance.

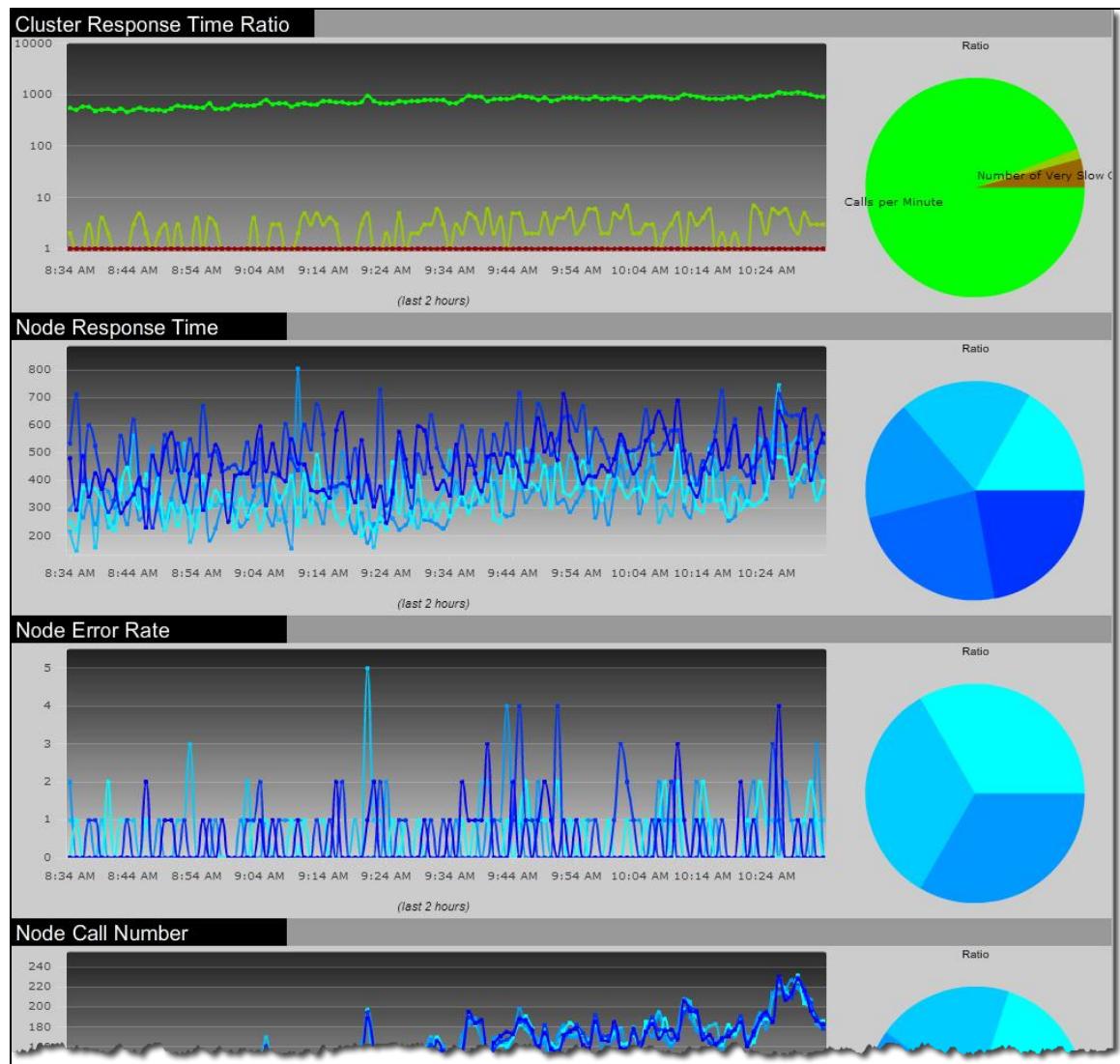
Custom Dashboards

Custom dashboards help you successfully identify problems without overloading you with all the details. Operations management can create custom dashboards for you that include only the key indicators that they want you to monitor.

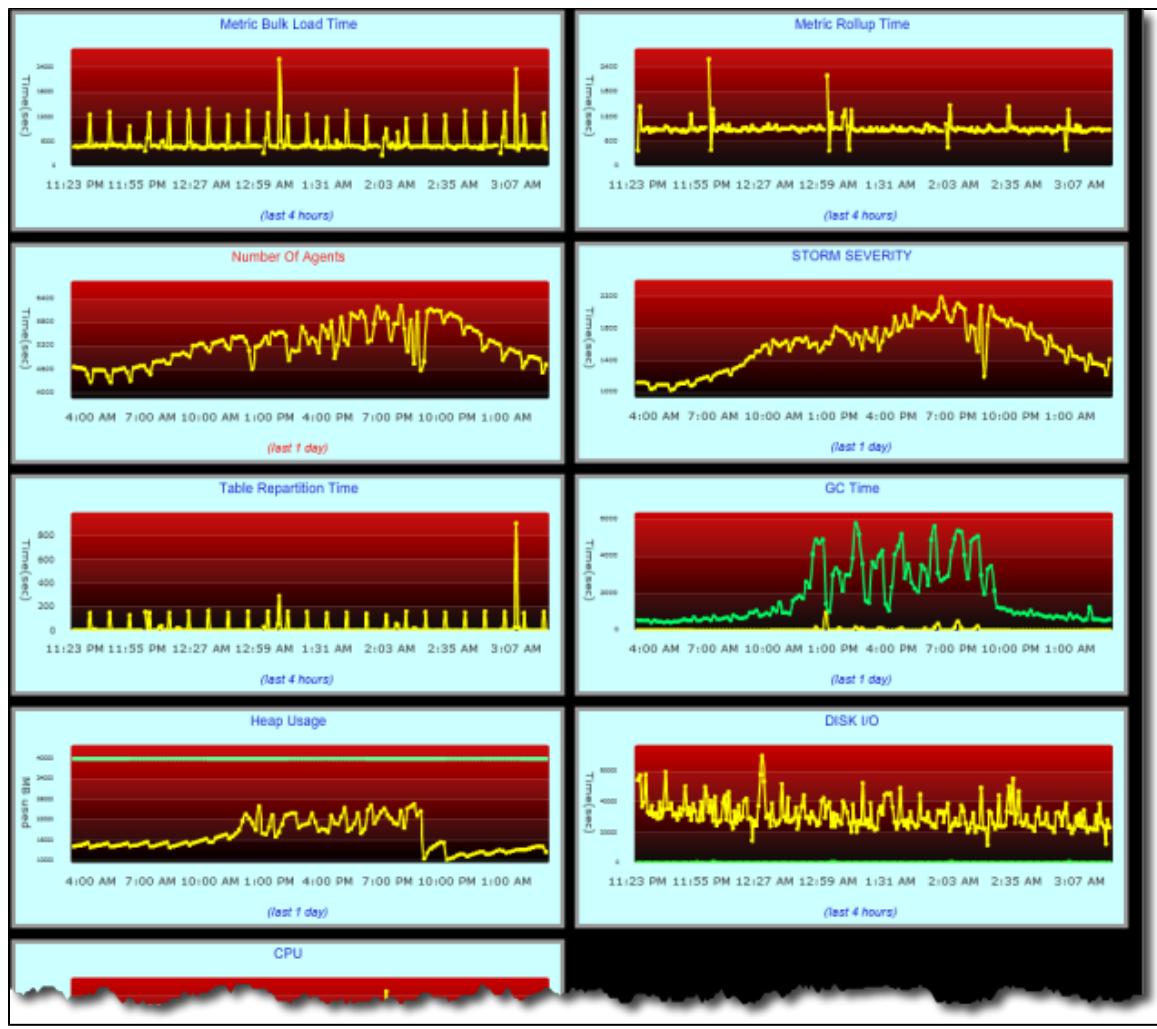
For example, here is a custom dashboard that summarizes application server health across the business application. It reports metrics for good and bad performance using green and red lights on the dashboard.



Here is a custom dashboard that graphs application server health metrics over time.



Here is a custom dashboard that graphs the most important metrics over specific time ranges.



See [Create a Custom Dashboard](#).

Policies and Alerts

Operations management can create policies that define thresholds for acceptable performance. Policies can automatically alert you when performance crosses those thresholds. See [Policies](#).



Tip: Send email alerts generated by health rule violations to the NOC.

You can click through on the deep link in the email alert, do basic triage, and then escalate the problem to the right team. Include the deep link in the notification so that the next team can quickly start working on it.

If the NOC already uses an alarming system, a more sophisticated approach is to integrate alerts from AppDynamics into that system using custom notifications. See [Integrate AppDynamics Alerting with Third Party Notification Systems](#).

Learn More

- Application Dashboard
- Business Transaction Dashboard
- Organizing Traffic as Business Transactions
- Business Transaction Monitoring
- Business Transactions List

- App Server List
- Node Dashboard
- Create a Custom Dashboard
- Administer Machine Agents
- System Integrations
- Integrate using Custom Action Scripts
- Monitor Events

Best Practices for Application Developers

- Identify Your Module
- Examine Flow
- Examine Errors in Your Module
- Examine Your Code
- Examine Server Problems
 - Memory Problems
 - Hardware Problems
- Examine by Business Transaction
- Learn More

Application developers can use AppDynamics to observe how a module they have designed and built functions in both test and production environments. You can see how a module interacts with other modules, both internal and external, and often observe conditions that you may not have anticipated during development.

Specifically, you can:

- Explore interactions between your module and other services
- View performance indicators for your module
- Discover stalls and code deadlocks
- Identify slow user requests, in your module or in another service that your module calls and find root cause
- Identify types and volumes of errors generated by the module
- Troubleshoot server problems, including memory usage and machine hardware issues



A developer or a developer team should be assigned to every module (tier) and to every business transaction. A tier or business transaction with no owner will not be maintained and will be visited only during an emergency outage. Best practice is to maintain ownership to monitor tiers and transactions on a regular basis so you can find and correct problems before they lead to outage.

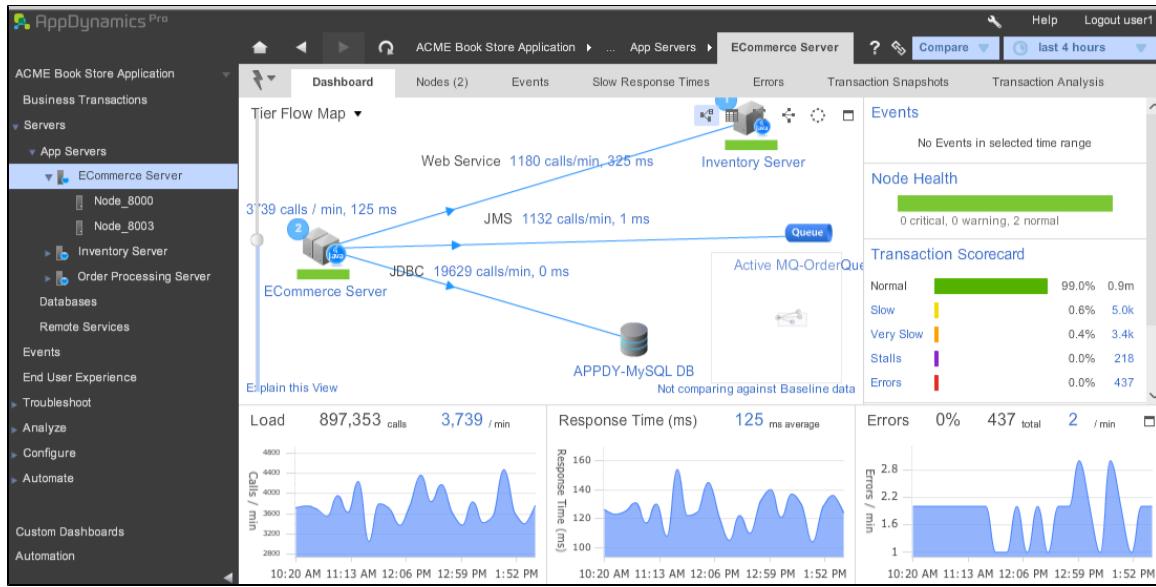
Identify Your Module

Identify your application module under the AppServers menu in the left navigation pane.

In AppDynamics nomenclature, an application module is modeled as a tier. See [Logical Model](#) to make sure you understand AppDynamics terminology for applications, tiers and nodes.

Examine Flow

Open the tier dashboard for your tier. For details about this dashboard, see [Tier Dashboard](#).

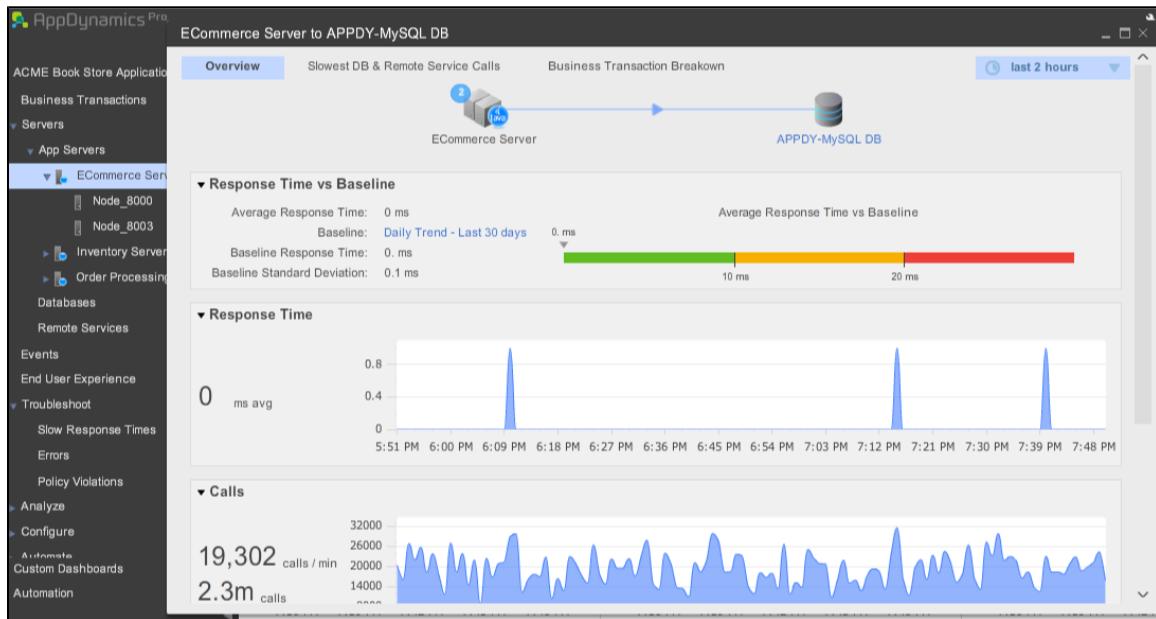


The tier dashboard is the starting point for viewing high-level information about your module's performance and its interactions with other modules.

In the flow map you can observe the flow of traffic between modules: the sequence of calls and their frequency.

If your module is calling obsolete services, or if it is not calling vital services, this will be evident on the flow map.

By clicking the flow line between your module and another, you can discover the locations of bottlenecks that affect your module.

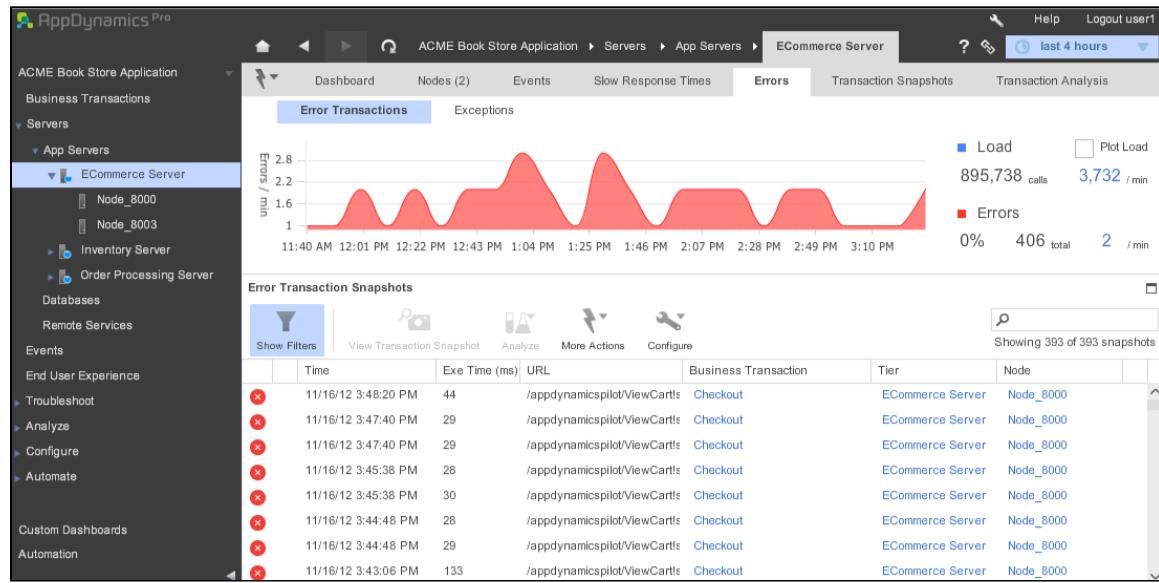


Some bottlenecks that start far downstream can eventually effect your module's performance. Many problems stem from backend servers such as databases, LDAP servers and caches. If you can identify these troubled servers, you can ask that they be fixed or work around them by removing them from the load balanced list.

In the lower part of the tier dashboard, you can examine key performance statistics such as average response times, load and errors. In the transaction scorecard on the right you see the number and percentage of requests to your module that are slow, very slow and stalled.

Examine Errors in Your Module

To view the errors that your module generates, click the **Errors** tab in the tier dashboard.

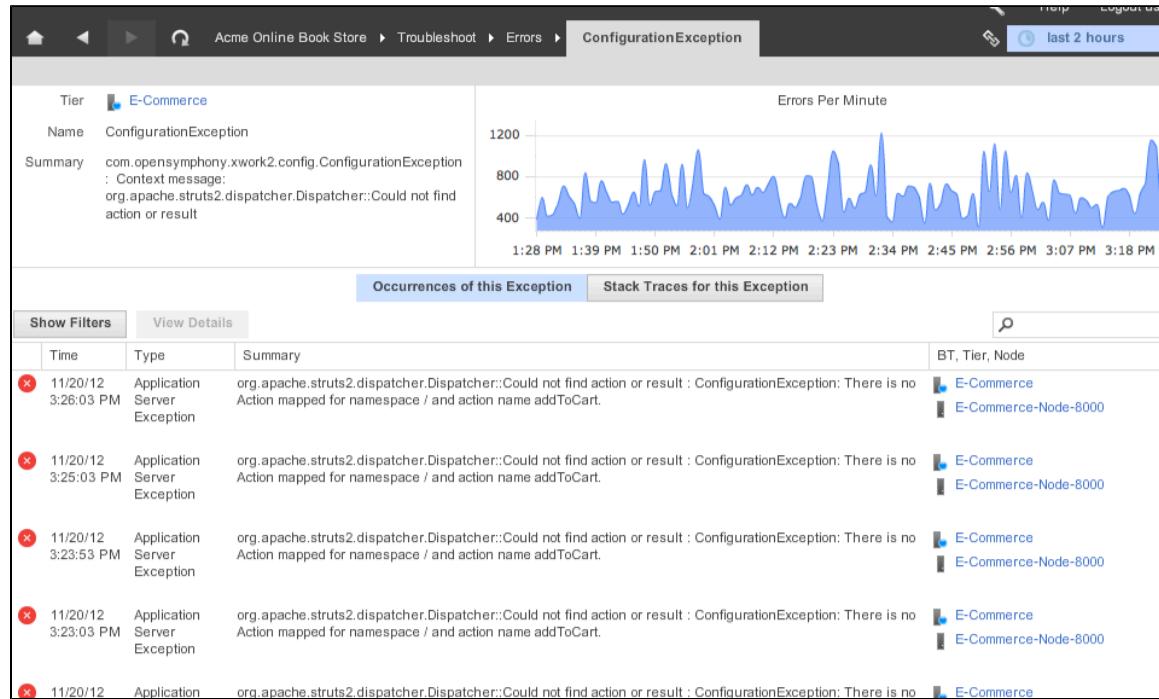


At the top is a graph of error rates over the selected time range where you can see when errors spiked.

Click the **Exceptions** subtab to see the exception counts and rates for the exceptions in the selected time range. You can decide which ones are acceptable in the short term and which ones have the potential to bring down a live system.

Monitor exceptions at least once a day, concentrating on the most frequently occurring types to see which are increasing. If some of the reported exceptions are not really a problem, change the error configuration to exclude them. See [Configure Error Detection](#).

For any exception that occurs frequently, double click it to see its graph. Examine the pattern.



There are three basic patterns:

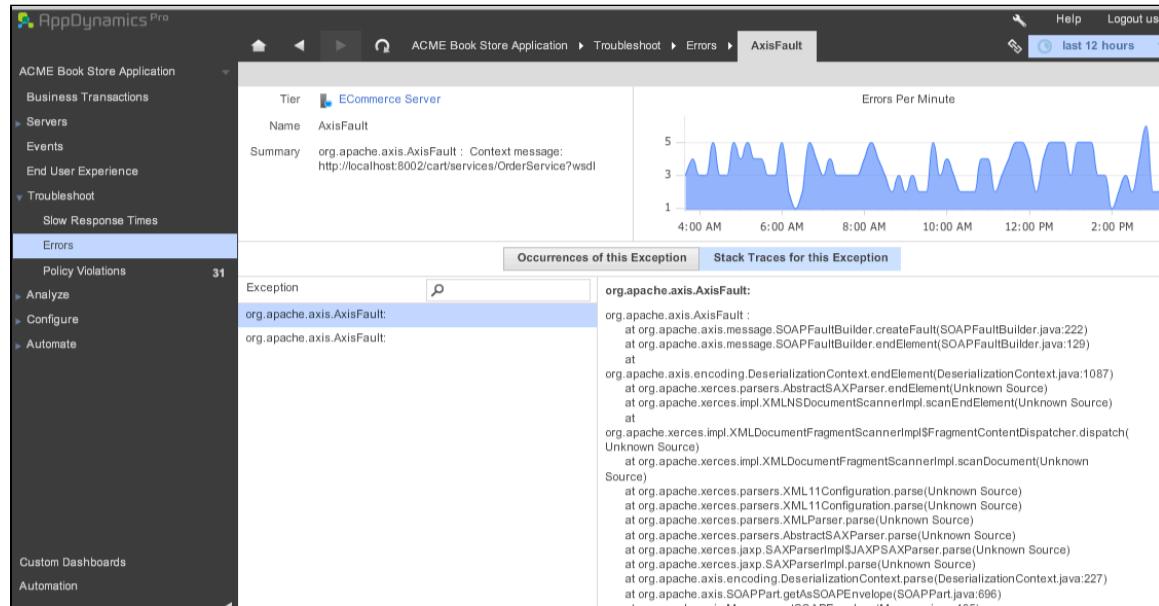
- Steady: Implies that this exception has been occurring for a long time, so it is unlikely to be urgent. But you can log a bug to get it fixed.
- Occasional: Maybe the exception spikes up every day at lunch time, or every 3 days. This pattern implies that some additional

load is failing and causing this exception. You should investigate and diagnose this type of pattern. Find out what is happening during those special times to make the exception rate increase.

- Trending up: This is the most dangerous pattern, because it has the potential to bring down the website. If an exception rarely or never happened and suddenly has increased and keeps increasing, this indicates a critical and potential emergency. Drill down and find root cause at once!

To drill down on an exception:

1. Double-click the exception.
2. Click the **Occurrences of this Exception** tab to see all the occurrences in the selected time range.
3. Click the **Stack Traces for this Exception** tab to see a stack trace.



After any major code change, re-examine the **Errors** tab to compare the results with the list of known errors to see if any new serious errors have emerged or if any known errors are occurring more frequently per time range.

Some errors may not originate from your own code, but may be a result of problems further downstream, for example, errors like ConnectionTimedOut. When you see those, the tier dashboard can help you identify which module is causing the error.

Examine Your Code

If you see a problem or just want to get a sample, view the transaction snapshots. These are collected based on various settings, but usually either because something is wrong (slow or error) or on a periodic basis.

Transaction snapshots offer you diagnostic information about your tier's transactions, including call graphs (callstacks), SQL that was executed, the HTTP that was sent, cookies, etc.

Click the **Transaction Snapshots** tab in the tier dashboard to see the list of transaction snapshots for the selected time range. See [Transaction Snapshots](#) and [Call Graphs](#) for information about filtering and drilling down into snapshots.

Time	Exec Time (ms)	URL	Business Transaction	Tier	Node
11/16/12 5:22:58 PM	10109	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8003
11/16/12 5:22:58 PM	10025	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8003
11/16/12 5:22:49 PM	10034	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:48 PM	10025	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:48 PM	10097	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8003
11/16/12 5:22:48 PM	10111	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:44 PM	10056	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:40 PM	10030	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:40 PM	10020	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:40 PM	34	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:40 PM	31	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:38 PM	10024	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8003
11/16/12 5:22:38 PM	10086	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:38 PM	10072	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000
11/16/12 5:22:34 PM	10111	/appdynamicspilot/ViewCart!	Checkout	ECommerce Server	Node_8000

The most common use of the snapshots is in conjunction with investigations into performance problems. If you know approximately when a problem occurred, you can find snapshots taken during that time range and analyze any concurrent deviations in the code performance. Also when examining a particular problem, such as an error, stall or deadlock, you can begin to see the code patterns that are associated with errors, stalls and deadlocks.

Examine Server Problems

To see all the application servers (nodes) that belong to your module, click the **Nodes** tab in the tier dashboard. The **Nodes** tab offers a rich set of details about how your servers are performing. Each node represents an application server.

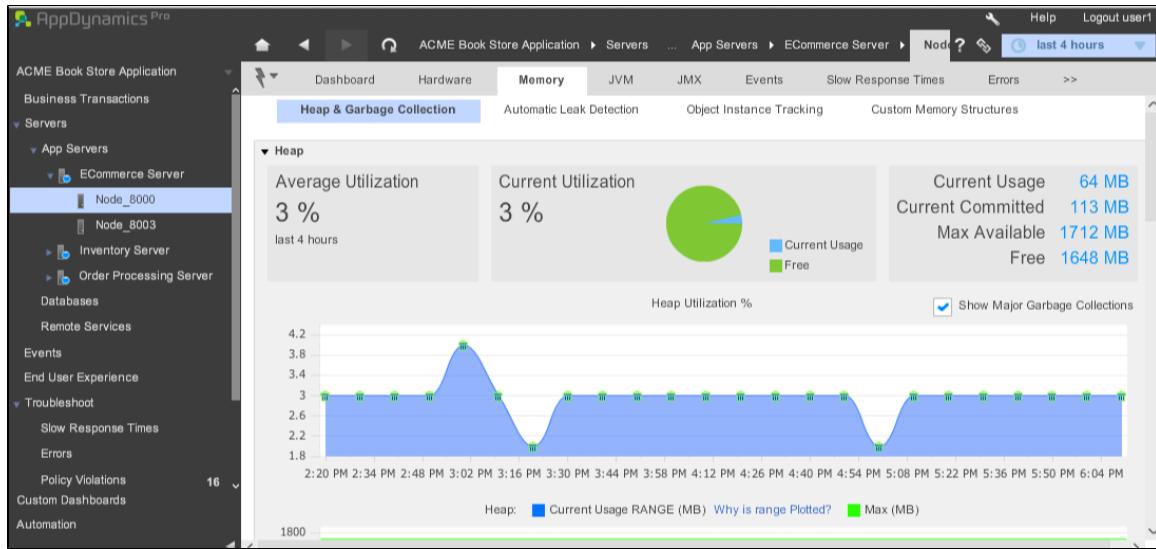
Name	Health	App Agent Status	App Agent Version	JVM	Last JVM Restart	Machine Agent Status
Node_8003	Green	100%	Server Agent v3.6.0.0 GA #	Java HotSpot(TM) 64-Bit Server VM 1.6.0_33 Sun Microsystems Inc.	11/12/12 4:48:56 PM	Red
Node_8000	Green	100%	Server Agent v3.6.0.0 GA #	Java HotSpot(TM) 64-Bit Server VM 1.6.0_33 Sun Microsystems Inc.	11/12/12 4:48:57 PM	Red

Memory Problems

Click the **Memory** subtab in the **Nodes** tab of the tier dashboard. Sort the nodes by **JVM % Heap** to identify machines that are running out of heap.

Name	JVM % Heap	Max Heap	JVM CPU Burnt (ms/min)	GC Time Spent (ms/min)	Major Col...	Minor Col...	Minor Col... tim...	
Node_8003	3.3	1712	12589	393	< 1 /min 76	143	35 /min 8.31	250
Node_8000	3.5	1712	13966	377	< 1 /min 63	123	34 /min 8.11	254

Double-click one of those nodes to get its node dashboard and in its **Memory** tab get detailed information about memory usage.



If heap is over 90% for a period of time more than a few minutes, you probably have a memory leak. In this case, restart the application and investigate leakage.

Then, in node tab of the tier dashboard, sort the nodes by GC Time Spent to find where garbage collection is taking too long and hindering app performance.



If major garbage collection is occurring more than once an hour, or if the major collection time is more than five seconds, you may be experiencing unsustainable memory usage. Investigate heap size settings, any potential memory leakage, and your GC settings.

Memory management problems can cause serious problems for developers. The information in the **Memory** tab of the node dashboard helps you to understand how memory pools are being used and how much time is being spent freeing memory.

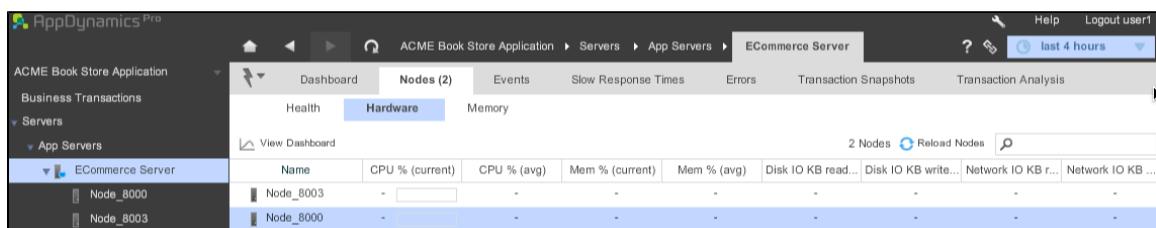
The most common problems occur when memory usage spikes. This can endanger the whole application. You can find the root cause of this problem by noting when the memory spiked, which is easy to see on the memory graphs, and then correlating the spike with other activity that you observe. For example:

- Specify a time range around the time of the spike and look in the business transaction dashboard to find out which business transactions deviated from normal.
- In the tier dashboard, examine any errors, stalls, deadlocks, or slow calls that occurred at this time.
- Examine a few transaction snapshots to see the code executing at this time and drill down to the root cause of the problem.

Sometimes it is difficult to catch a problem at the exact time it occurred. In those cases, you can be alerted any time a certain condition exists, so you can immediately investigate the issue. To do this is to create a policy based on a health rule for the condition (for example when Heap-Size > 80%) that sends an email to the developers. See **Policies**.

Hardware Problems

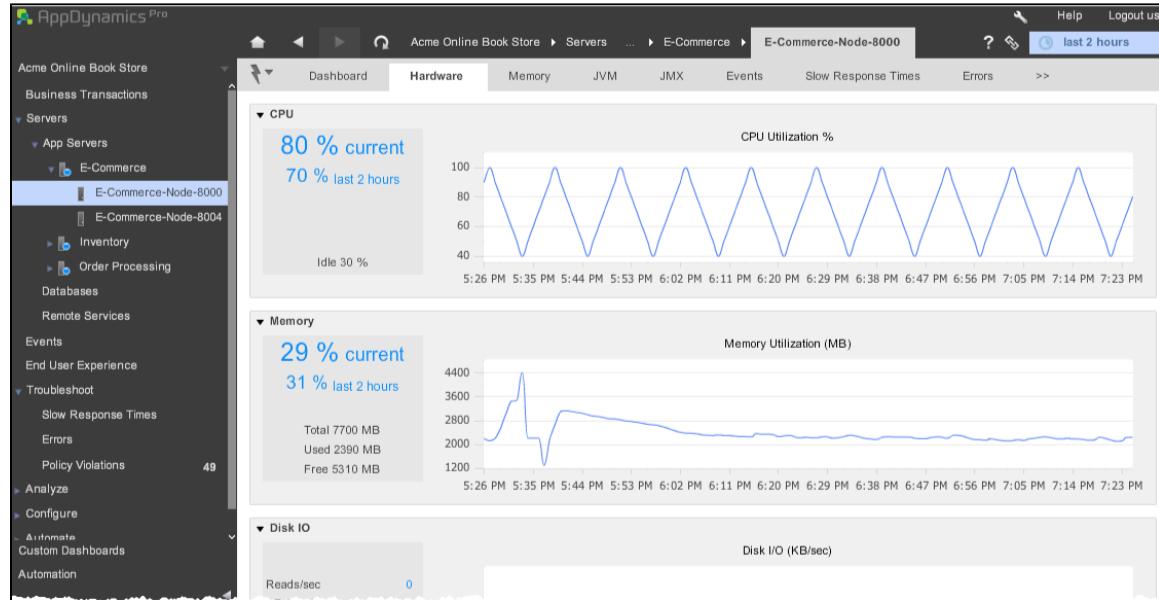
Click the **Hardware** subtab of **Nodes** tab of the tier dashboard.



Sort the nodes by CPU % to see if CPU usage is excessive.

Double-click one of those nodes to get its node dashboard. In its **Hardware** subtab you can see if routines are I/O bound or CPU bound by looking the corresponding graphs.

You can also see how much RAM is being used by application server heaps and if network I/O is heavy.



In most cases AppDynamics recommends that you restart/kill all app servers that are over 80% CPU.
In the **Hardware** subtab of **Nodes** tab of the tier dashboard, sort the nodes by memory usage to identify machines that are running out of RAM.

In most cases, memory usage 90% and above is considered dangerous and should probably be reduced by changing the application's cache settings. Or in the case of a memory leak, restart the application running on that machine.

Examine by Business Transaction

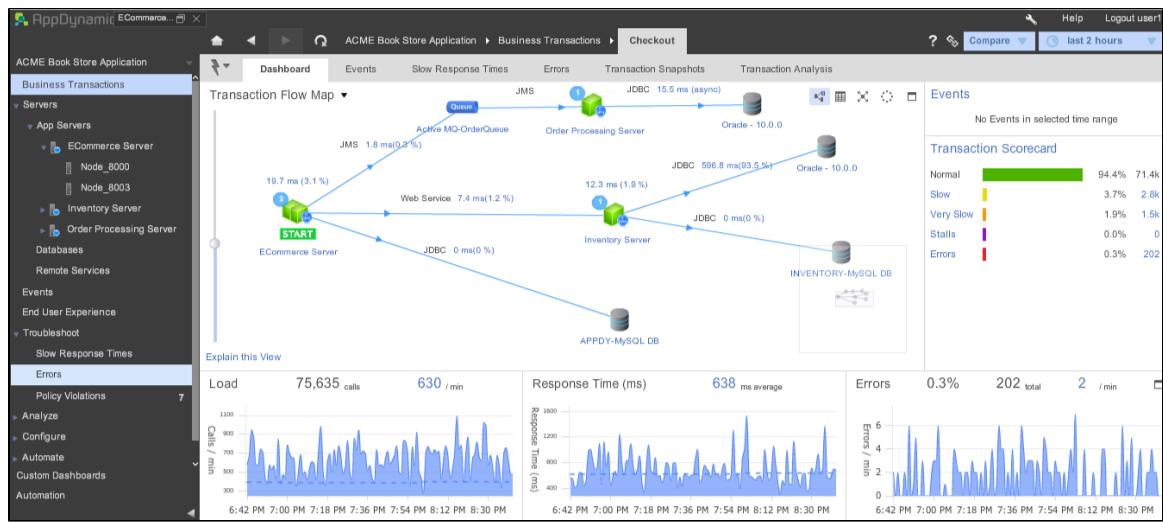
Business transactions divide up the load into logical operations that map to user requests, such as Login, Search, Checkout, etc. See [Business Transaction Monitoring](#), [Business Transactions List](#) and [Business Transaction Dashboard](#) for more information about business transactions.

Open the business transaction list, click **Filters On** if they are not visible, and enter the name of your tier in the Tier field.

Name	Health	End User Time (ms)	Page Render Time (ms)	Network Time (ms)	Server Time (ms)	Calls	Calls / min	Errors	Error %	Slow Trans	Very Slow	Stalled	Tier	Type
Fetch catalog	green	0	0	0	9	75,764	631	0	0	15	1	0	ECommerce Se...	Struts Action
Checkout	red	0	0	0	636	75,666	631	203	0.3	2,925	1,290	0	ECommerce Se...	Struts Action
Logout	green	526	447	9	5	75,666	631	0	0	7	0	0	ECommerce Se...	Struts Action
Login	green	0	0	0	4	75,664	631	0	0	3	0	0	ECommerce Se...	Struts Action
Homepage	green	0	0	0	2	75,664	631	0	0	5	0	0	ECommerce Se...	Servlet
Add to cart	yellow	0	0	83	75,664	631	0	0	13	0	99	0	ECommerce Se...	Struts Action

You can view the list of business transaction that originate in your module. You can them sort by load, performance, error counts, etc. You can see which transactions are deviating from normal and start your investigation with those.

Double-click a business transaction to see its dashboard. The business transaction dashboard shows the same flow and performance information as the tier dashboard, but only for one type of user request.



With this isolated view, you can investigate the services that are involved to fulfill only this transaction and where slowdowns start and find any critical errors that only exist for this transaction flow.

Sometimes it is important to have this isolated view, because in the tier dashboard you may see healthy flow between your module and many other services, but once you isolate a single business transaction, it may become apparent that one does not have such a healthy flow, that it was just masked by all the other transactions.

Learn More

- Logical Model
- Organizing Traffic as Business Transactions
- Business Transaction Monitoring
- Business Transactions List
- Business Transaction Dashboard
- Tier Dashboard
- Transaction Snapshots
- Call Graphs