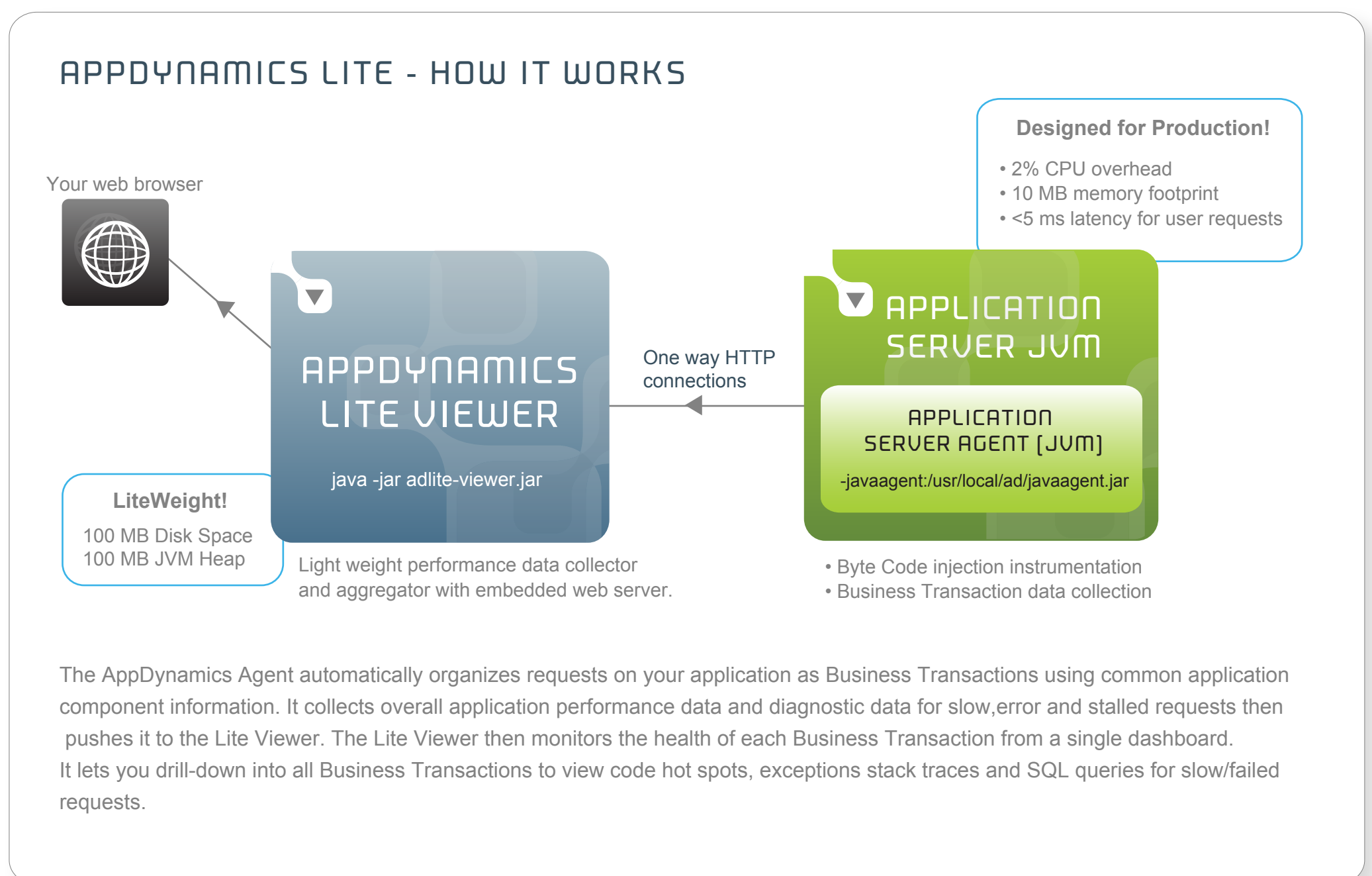# AppDynamics

WELCOME TO THE APPDYNAMICS LITE EXPERIENCE!

WHAT IS IT?

AppDynamics Lite lets you monitor user requests in real time. It organizes them into Business Transactions and calculates their health based on failed requests (slow, error, and stalls). It provides always-on automatic diagnostic visibility into the slow/failed ones and lets you drill even deeper to confirm root cause when needed.

APPDYNAMICS LITE - HOW IT WORKS

Your web browser

Designed for Production!
• 2% CPU overhead
• 10 MB memory footprint
• <5 ms latency for user requests

APPDYNAMICS LITE VIEWER

java -jar adlite-viewer.jar

One way HTTP connections

APPLICATION SERVER JVM

APPLICATION SERVER AGENT (JVM)
-javaagent:/usr/local/ad/javaagent.jar

LiteWeight!
100 MB Disk Space
100 MB JVM Heap

Light weight performance data collector and aggregator with embedded web server.

• Byte Code injection instrumentation
• Business Transaction data collection

The AppDynamics Agent automatically organizes requests on your application as Business Transactions using common application component information. It collects overall application performance data and diagnostic data for slow,error and stalled requests then pushes it to the Lite Viewer. The Lite Viewer then monitors the health of each Business Transaction from a single dashboard. It lets you drill-down into all Business Transactions to view code hot spots, exceptions stack traces and SQL queries for slow/failed requests.

HOW DO I DO THIS?

All you need to get started is the AppDynamicsLite.zip file (available at http://www.appdynamics.com/free).
The file contains two archives: 1) LiteViewerBeta.zip and 2) AppServerAgentLiteBeta.zip.

Here's how to use them:

1 EXTRACT LITE VIEWER AND START

A. EXTRACT
LiteViewerBeta.zip

B. FROM THE VIEWER DIRECTORY RUN
java -jar adlite-viewer.jar

Note: Runs on port 8990 by default.
Choose a different port by using
the run time option "-Dadlite.port".

2 INSTALL AGENT ON JVM

Byte Code Instrumentation based Java Agent
(http://java.sun.com/j2se/1.5.0/docs/api/java/
lang/instrument/package-summary.html)

A. EXTRACT
AppServerAgent.zip

Uses the same filesystem
as the app server.

B. EDIT APP SERVER STARTUP SCRIPT

add -javaagent: <path to javaagent.jar>
to the JVM arguments.

If you modified the default Viewer port,
add this system property to the agent:
-Dappdynamics.viewer.port=<viewer_port>.

3 RESTART APP SERVER AND ACCESS APPLICATION

A. RESTART APP SERVER

B. ACCESS the Lite Viewer

Default login information:
Username = admin
Password = admin
(unless you set up auth using
-Dadlite.user and -Dadlite.password)

When you start AppDynamics Lite it
auto-discovers your Business Transactions
and the backends they access (ie: JDBC,
JMS, Web Services).

4 ORGANIZE REQUESTS AS BUSINESS TRANSACTIONS & START MONITORING

- Rename auto-discovered transactions
- Exclude unimportant ones
- Look at up to 30 transactions at a time

Monitor overall Business Transaction health based on their slow/failed requests, drill-down into the ones that were slow or had errors

Hot Spots
com.acme.FooClass.fooMethod
com.acme.FooClass1.barMethod
com.acme.FooClass2.
fooBarMethod

Hot Spots to diagnose slow requests

Errors
java.lang.NullPointer at
com.acme.FooClass.fooMethod
com.acme.FooClass1.barMethod
com.acme.FooClass2.fooBarMethod

Exception stack traces for error requests

Stalls
java.net.PlainSocketImpl
connectToAddress(Ljava/net/InetAddress...)
java.net.PlainSocketImpl.connect...
org.axis.WebService.invoke...
com.acme.Order.ProcessOrder

Real-time stall thread dumps for stalled requests

Learn more
Interact with other users at **http://LiteForums.AppDynamics.com**
Read documentation at **http://LiteDocs.AppDynamics.com**