

Worklight® 4.2.1

# Worklight Administration Guide

January 2012

Document Version 4



© 2006-2012 Proprietary and Confidential of Worklight Inc.

The information contained herein is the propriety and confidential information of Worklight Inc. and is not to be disclosed or used in any way except as expressly authorized by written contract with Worklight.

# Table of Contents

<b>Document History</b> .....	<b>4</b>
<b>Introduction</b> .....	<b>5</b>
Purpose and Scope of This Document.....	5
Glossary .....	5
Product, Customization, Content .....	5
<b>Topology of a Worklight Instance</b> .....	<b>7</b>
Typical Topology.....	7
Other Topologies.....	8
<b>Installation and Configuration</b> .....	<b>9</b>
Step 1: Installing the Database.....	9
Creating the Worklight Database Schema for My SQL .....	9
Creating the Worklight Database Schema for Oracle .....	10
Step 2: Installing the Worklight Server.....	11
Step 3: Applying Environment-specific Customization.....	13
Step 4: Deploying Content.....	13
<b>Administration of a Worklight Server Cluster</b> .....	<b>17</b>
Configuring the Load Balancer .....	17
Worklight Server on Virtual Machines .....	17
Adding a Node to the Cluster .....	17
Removing a Node from the Cluster .....	17
Updating the Customization.....	18
Deploying or Re-deploying an Adapter .....	18
Deploying an App .....	18
Starting and Stopping the Worklight Server on Linux .....	18
Setting Worklight as a Linux Service.....	18
Starting and Stopping the Worklight Server on Windows.....	19
Setting Automatic Server Restart on Windows.....	19
Changing Credentials.....	20
Denying Access to Older App Versions.....	20
Displaying a Notification Message on App Startup .....	22
<b>Backup and Recovery</b> .....	<b>23</b>
<b>Monitoring</b> .....	<b>24</b>
Monitoring the Worklight Server .....	24
Tracking Logs .....	24
Routing Log Files to a Different Destination .....	24

Audit Log .....	25
Monitoring Backend Connectivity .....	25
Diagnosing Login & Authentication Issues .....	26
Is-Alive query .....	26
<b>Reports.....</b>	<b>28</b>
Activating Reports .....	29
Report Data Maintenance .....	31
<b>Push Notification .....</b>	<b>32</b>
The Push Notification Console .....	32
<b>Disaster Recovery Site .....</b>	<b>34</b>
Architecture.....	34
Data mirroring .....	34
Switching to Backup Site .....	35
Switching back to Master Site .....	35
<b>Support .....</b>	<b>36</b>
<b>Appendix A: Environment Setup and Pre-requisites .....</b>	<b>37</b>
Production Worklight Application Server.....	37
Production Worklight Database Server .....	37
Load balancer .....	38
<b>Appendix B: Worklight Properties .....</b>	<b>39</b>
Configuring the Worklight Server Location .....	39
Worklight Database Setup.....	40
Reports Database Setup .....	41
Protecting the Worklight Console .....	41
Push Notification Settings .....	41
Miscellaneous Settings.....	42
Storing Properties in Encrypted Format.....	42
Obsolete Properties .....	44
<b>Appendix C: JVM &amp; Tomcat Tuning.....</b>	<b>45</b>
<b>Appendix D: Internal Worklight Database Tables .....</b>	<b>46</b>
<b>Appendix E: HTTP Interface of Production Server .....</b>	<b>48</b>
Application API Requests.....	48
Web Application Resource Requests.....	51
Preview Application Resource Requests .....	52
<b>Appendix F: Upgrading a Production 4.0.3 Worklight Server to 4.2.1 .....</b>	<b>53</b>

# Document History

Version	Comment
2	Removed obsolete configuration properties
2	Updated version of Oracle and My SQL supported
3	Minor fixes
4	Moved the chapter Worklight Command-line Builder and Deployer to the <i>Worklight Developer's Reference Guide</i> Updated the instructions to developers for upgrading a Worklight 4.0 production environment. Minor fixes

# Introduction

## Purpose and Scope of This Document

The purpose of this document is to describe the tasks required to install and maintain Worklight production server.

The document does not discuss administration of Worklight Server prerequisite software, such as database or reverse proxy.

## Glossary

The following definitions apply throughout this document:

Term	Definition
Worklight Server	The primary run-time server responsible for serving app requests and publishing apps to users.
Reverse Proxy	A proxy server used in front of Web servers. All connections coming from the Internet addressed to one of the Web servers are routed through the reverse proxy server, which may either deal with the request itself or pass the request entirely or partially to the main web servers.
Web SSO	Web authentication infrastructure. Acts as a reverse proxy and protects the web front end applications of the organization. The Web SSO tier is responsible for the end-user authentication, and passes the end-user identity in a unified form (usually as an additional HTTP header) to all the web applications running in the organization internal network.  The Web SSO tier is also responsible for the Secured Sockets Layer (SSL) support, so that all the traffic between the client and Web SSO server is in https, and the traffic between Web SSO and web applications is plain http.
App	Worklight Client side software component, written in web technologies and sometimes augmented with native, device-specific code
Adapter	Worklight Server-side software component whose role is to connect the Worklight Server to various back-end (enterprise) systems

## Product, Customization, Content

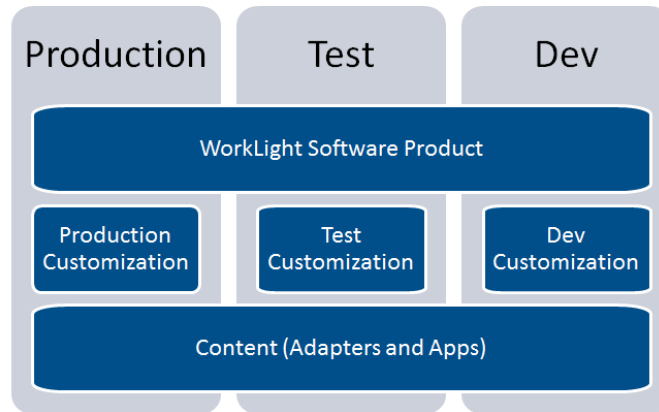
Worklight deployment technology uses 3 terms which are used throughout this document:

- **Product:** The Worklight Server software product as received from Worklight
- **Customization:** An enterprise deployment goes through lifecycle phases such as “test environment”, “staging environment” and finally “production environment”. A

Worklight *customization* is a set of configuration files which are environment specific. Example: Backend connectivity properties, Certificate configuration.

- **Content:** The customer-specific Worklight software components. Adapters and Apps. Typically identical between environments (production, test, development)

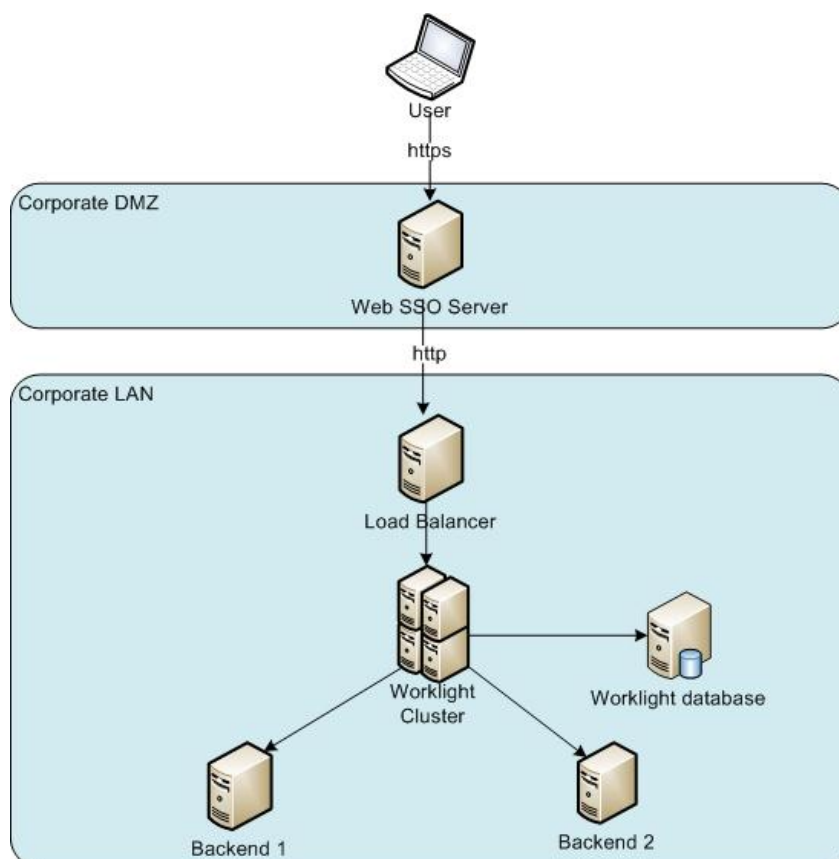
The diagram below explains these three terms graphically.



# Topology of a Worklight Instance

## Typical Topology

The administration tasks described in this document refer to a particular topology which is typical for organizations with an established extranet infrastructure. This topology is depicted in the following diagram.



Such topology is based on the following principles:

- The Worklight Server is installed in the organization LAN, connecting to various enterprise backend systems.
- The Worklight Server can be clustered for high availability and scalability.
- The Worklight Server uses a database. The database is used for caching data retrieved from backend systems, storing application usage data for reporting and analytics, and storing metadata required by the Server in run time. It is possible to cluster the database, using the tools and methods provided by the database vendor.
- The Worklight Server is installed behind a web authentication infrastructure (Web SSO) acting as a reverse proxy and providing SSL.

# Other Topologies

The Worklight Server can be installed in additional topologies. In the most prominent one, the Worklight Server is installed in the DMZ and is not protected by a web authentication tier.

Administration of the Worklight Server in such topologies is not part of this document.

# Installation and Configuration

Installation of the Worklight Server in a production environment differs from the installation of a development environment. For the latter, refer to *Worklight 4.2.1 Development Environment Installation Guide* on the Worklight website at <http://www.worklight.com/>.

The Installation steps of the Worklight Server in production are:

1. Installing the Worklight database
2. Installing the Worklight Server software
3. Applying environment-specific customization
4. Deploying content

For information regarding hardware requirements and network configuration, please review Appendix A below.

## Step 1: Installing the Database

Worklight can use My SQL 5.5 or Oracle 11g as its database. Installation of the database system itself is not discussed in this document.

### *Creating the Worklight Database Schema for My SQL*

#### **To create the Worklight database schema for MySQL:**

1. Run a My SQL command-line client.
2. Enter the following command to create the Worklight database schema:

```
CREATE DATABASE Worklight CHARACTER SET utf8 COLLATE  
utf8_general_ci;
```

3. Enter the following command to allow the Worklight Server access to the database, replacing *Worklight-host* with the name of the host on which the Worklight Server will run:

```
GRANT ALL privileges ON Worklight.* TO `Worklight`@`Worklight-host`  
IDENTIFIED BY 'Worklight';
```

4. Enter the following command to allow the database administrator access to the database from the server hosting the database:

```
GRANT ALL privileges ON Worklight.* TO 'Worklight'@'localhost'  
    IDENTIFIED BY 'Worklight';  
Flush privileges;
```

Database tables will be created automatically when the Worklight Server first starts.

## Updating database properties in the `worklight.properties` file

Make sure to define the following properties in the `worklight.properties` file. If you change the database name, user name, or password, make sure these values are reflected in the appropriate properties.

```
DB_TYPE=MYSQL_DB_TYPE  
hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect  
wl.db.driver=com.mysql.jdbc.Driver  
wl.db.url=jdbc:mysql://<mysql-server-url>:3306/Worklight  
wl.db.username=Worklight  
wl.db.password=Worklight  
wl.db.validationQuery=select 1
```

## *Creating the Worklight Database Schema for Oracle*

### To create the Worklight database schema for Oracle:

1. Run an Oracle command-line client.
2. Enter the following commands to create the Worklight database user and schema, assign it a password, and assign it appropriate privileges:

```
CREATE USER worklight IDENTIFIED BY worklight;  
GRANT CONNECT TO worklight;  
GRANT RESOURCE TO worklight;  
GRANT CREATE VIEW TO worklight;
```

Database tables will be created automatically when the Worklight Server first starts

## Updating database properties in the `worklight.properties` file

Make sure to define the following properties in the `workLight.properties` file. If you change the database name, user name, or password, make sure these values are reflected in the appropriate properties.

```
DB_TYPE=ORACLE_DB_TYPE
hibernate.dialect=org.hibernate.dialect.Oracle10gDialect
wl.db.driver=oracle.jdbc.driver.OracleDriver
wl.db.url=jdbc:oracle:thin:@<oracle-server-url>:1521:<oracle-SID>
wl.db.username=worklight
wl.db.password=worklight
wl.db.validationQuery=select 1 from dual
```

## Step 2: Installing the Worklight Server

Installation of the Worklight Server is done in two steps: extract the binary distribution archive and applying the configuration. These steps should be *repeated* for each cluster node so that the content of the installation folder is the same on all the nodes.

### Opening the Binary Distribution Archive

Worklight provides the binary distribution as a compressed file in two formats:

- `Worklight-4.2.1-win64.zip` for Windows Server 2008
- `worklight-4.2.1-linux64.tar.gz` for Linux

The binary distribution archive includes:

- Platform-dependent JRE files (Java binaries)
- Tomcat distribution
- Worklight libraries
- Third-party libraries used by the Worklight Server
- Scripts for administration tasks

In both Windows and Linux, extract the distribution archive into a target installation folder of your choosing. This folder will now be referred to as the Worklight Root Directory.

Once extraction is completed, the target folder will have the following structure:

```
<Worklight Root Directory>\
  server\
    bin\ – Worklight startup, shutdown and support scripts
    conf\
      • Worklight Server configuration files, including
        worklight.properties and authenticationConfig.xml
        (These files are not part of the distribution. They are added in this
        folder when you install a customization.)
      • Other resources that define the behavior of the Worklight Server
        authenticator – Desktop authentication files
        certificates – Certificates for signing desktop widgets
        welcome – Welcome screen template
    lib\ – Including:
      • Third-party Java libraries, such as JDBC drivers and other back-end
        Java libraries
      • Authentication integration archives (login modules and
        authenticators)
    webapps\ – Worklight web applications
    log\ – Run-time and development log files. The directory is only
    created after the Server is started for the first time.
  tomcat\ The Apache Tomcat server software configured for running Worklight
  jre\ Worklight-required Java 6 runtime
  license\ License information of Worklight and of third-party software products
  used within Worklight
```

In Step 4, the various files will be copied into the above directories.

## Database Client

### MySQL

1. Download the MySQL JDBC driver Connector/J from <http://dev.mysql.com/downloads/connector/j/5.1.html>
2. Copy the jar to `lib\` directory above

### Oracle

1. Download the Oracle JDBC driver from: <http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html>
2. Copy the jar to `lib\` directory above

## Validating the Installation

To validate successful installation of the Worklight Server:

1. Start the Worklight Server by running the script `<Worklight Root Directory>\server\bin\wl-start`.
2. Verify that a message "Server started successfully" appears.

```
*****
***** Server started successfully *****
*****
```

## Step 3: Applying Environment-specific Customization

The customization of a Worklight Server is a folder containing configuration files, applications, libraries, and external web applications needed by the Worklight Server in run time.

- You should receive this folder from the Worklight developer in your organization.
- To apply the customization on the Worklight Server, run the `set-customization` script, which is located in the `<Worklight Root Directory>\server\bin` folder. This script accepts the customization folder path as input parameter and copies the resources to the appropriate folders under the Worklight Root Directory, overriding the existing resources (such as `worklight.properties`), if such exist.

It is recommended to maintain the customization files aside from the installation folder. This will allow independently updating the Worklight Server distribution and the configuration.

### Worklight Properties File

The Worklight properties file resides under `<Worklight Root Directory>\server\conf\worklight.properties`. This file is typically part of the customization. Still it is important to review it and make sure that it is tuned for production. See Appendix B for further details.

### Memory Tuning

It is recommended to tune Worklight to utilize available memory. Increasing memory is useful for handling large transaction load.

Memory tuning is done at the JVM level. See Appendix C for further details.

## Step 4: Deploying Content

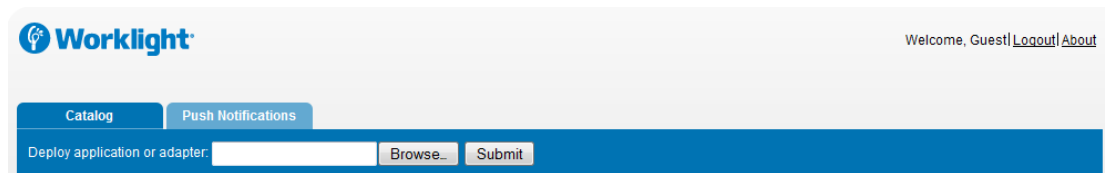
Customer-specific content includes **apps** that need to be served by the Worklight Server and their underlying integration **adapters**. Content can be deployed only after the customization has been set (Step 3 above). The developer creates Apps and Adapter by exporting them with the Worklight console.

1. Start the Server by running the script `<Worklight Root Directory>\server\bin\wl-start`.
2. Open a browser and enter the following URL:  
`http://<Worklight Server:8080>/console`

**Note:** 8080 is the default port.

If your Worklight Server is configured to require login, and you are not currently logged in, you will be prompted to log in.

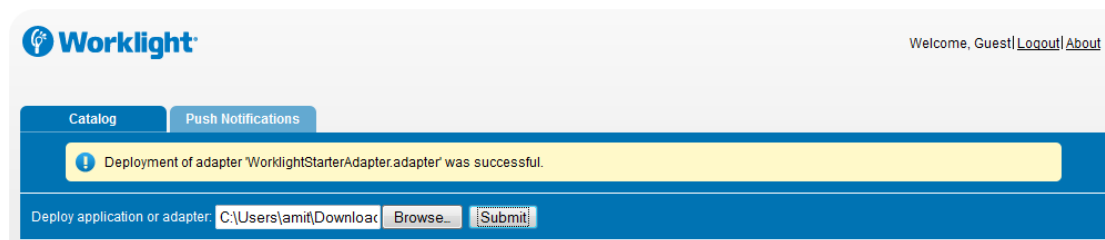
The Worklight Catalog page opens, and you can start performing adapter administration tasks.



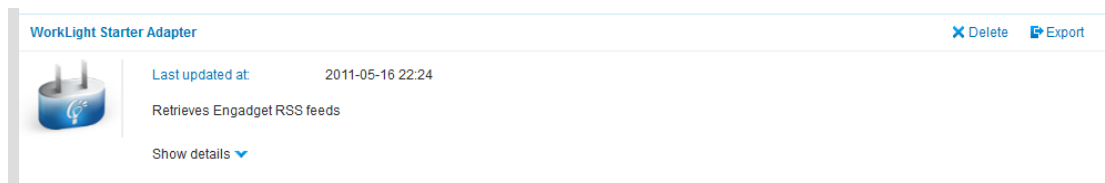
### To deploy an adapter:

3. Click **Browse**, then navigate to the `.adapter` file and select it.
4. Click **Submit**.

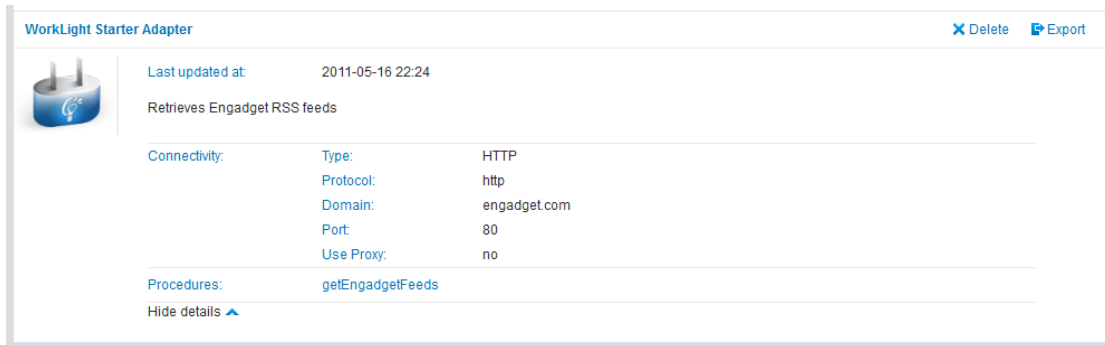
A message appears indicating whether the Deploy action has succeeded or failed.




As a result, the details of the deployed adapter are added to the Catalog:



5. Click **Show details** to view the adapter's connectivity details and the list of procedures it exposes:



WorkLight Starter Adapter [Delete](#) [Export](#)

 Last updated at: 2011-05-16 22:24  
Retrieves Engadget RSS feeds

Connectivity:	Type:	HTTP
	Protocol:	http
	Domain:	engadget.com
	Port:	80
	Use Proxy:	no

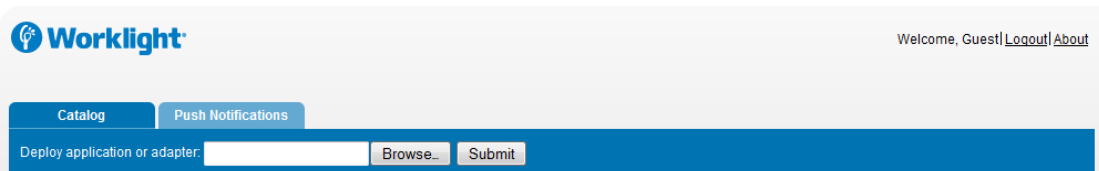
Procedures: [getEngadgetFeeds](#)

[Hide details](#) ▲

6. Repeat steps 3 – 5 for each adapter.

### To deploy an application:

7. In the Catalog page, click **Browse**, then navigate to the `.wlapp` file and select it.

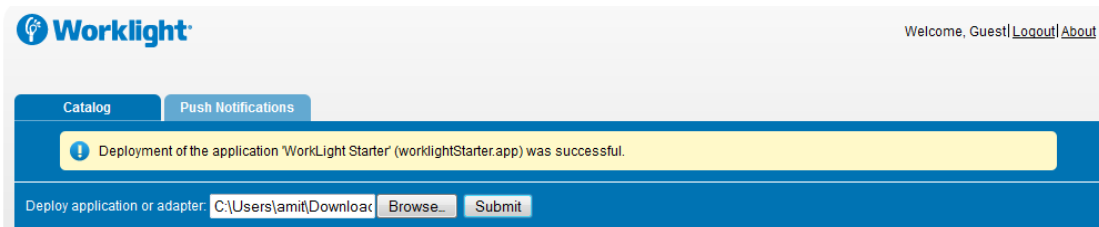


Worklight Welcome, Guest | [Logout](#) | [About](#)

Catalog Push Notifications

Deploy application or adapter:

8. Click **Submit**. A message appears indicating whether the Deploy action has succeeded or failed.



Worklight Welcome, Guest | [Logout](#) | [About](#)

Catalog Push Notifications

! Deployment of the application 'WorkLight Starter' (worklightStarter.app) was successful.

Deploy application or adapter:

As a result, the details of the deployed application are added to the Catalog:


Worklight® Welcome, Guest | [Logout](#) | [About](#)

**Catalog** | **Push Notifications**

Deploy application or adapter:











---

**WorkLight Starter** [Delete](#) [Export](#) [View Descriptor](#)



Retrieves RSS feed from engadget.com

Preview as:  Last updated at: 2011-05-16 15:46

	iPhone	Version 1.0	<span style="color: green;">●</span> Active	<input type="button" value="v"/>
	Android	Version 1.0	<span style="color: green;">●</span> Active	<input type="button" value="v"/>
	BlackBerry	Version 1.0	<span style="color: green;">●</span> Active	<input type="button" value="v"/>
	Windows Phone	Version 1.0	<span style="color: green;">●</span> Active	<input type="button" value="v"/>
	iPad	Version 1.0	<span style="color: green;">●</span> Active	<input type="button" value="v"/>
	Vista	Version 1.0		<a href="#">Install</a>
	Adobe Air	Version 1.0		<a href="#">Install</a>
	Dashboard	Version 1.0		<a href="#">Install</a>
	Facebook	Version 1.0		<a href="#">Publish</a>
	iGoogle	Version 1.0		<a href="#">Install</a>

© Copyright 2011 Worklight Inc. All rights reserved.

9. Repeat steps 7 – 8 for each App.

# Administration of a Worklight Server Cluster

The Worklight cluster consists of the load balancer, the cluster nodes, and the Worklight database shared by the cluster nodes.

All cluster nodes are identical, i.e. the content of the installation folder is the same in all nodes. Cluster nodes do not synchronize with each other in the runtime. All shared runtime data resides in the database, so that database changes made via one node are immediately available to all other nodes.

## Configuring the Load Balancer

Configuration of the load balancer depends on the vendor and is not discussed in this document, however it is common to define the range of the node IPs so they can be added or deleted dynamically.

The load balancer should be configured to support sticky session.

## Worklight Server on Virtual Machines

Worklight Servers can run on a VMware virtual machine. In such cases, one machine image is created and then duplicated deployed again and again.

## Adding a Node to the Cluster

To add a node, follow the instruction on creating a new Worklight Server above:

1. Add the node's IP to the load balancer or use an existing IP from a range which was pre-allocated to Worklight Servers
2. Install the Worklight Server
3. Apply the customization, as described in Step 3: Applying Environment-specific Customization, above
4. Start the Server
5. Deploy content

## Removing a Node from the Cluster

To remove a node from the cluster:

1. Stop the Server or shutdown the machine
2. Depending on how you configured the load balancer, you may have to manually remove the Server's IP address from the load balancer configuration

## Updating the Customization

1. Update the customization folder
2. Stop the Worklight Server
3. Apply the customization
4. Start the Worklight Server

Remember - This procedure should be repeated on each cluster node!

## Deploying or Re-deploying an Adapter

Follow the instructions above for deploying an adapter on a single Worklight Server. Perform this action on each of the Cluster nodes.

## Deploying an App

Using the Console, deploy the application on each node in the Worklight Server cluster.  
Important note: When re-deploying an App, it is important not to delete the old version of the app. Deleting the old app will require URL updates for all the app users.

## Starting and Stopping the Worklight Server on Linux

To start and stop the Worklight Server, use the Worklight LSB script under `<Worklight Root Directory>/bin`. The `worklight` script supports the following methods:

- `./worklight start` – starts the Worklight Server as a background service
- `./worklight stop` – stops the running background service
- `./worklight restart`
- `./worklight status` – displays the current status of the Worklight Server
- `./worklight run` – starts the Worklight Server in the foreground. This is useful in development scenarios to see the Server log output
- `./worklight debug` – starts the Worklight Server in the foreground and allows connecting to it via Java's JPDA debugging protocol (port 8787)

## Setting Worklight as a Linux Service

After installing the Worklight Server, run the following script using the `sudo` command to register Worklight as a service:

- `./wl-install <servername>`, where `servername` is the name of Worklight instance that you wish to create

To remove the Worklight service, run using `sudo`:

- `./wl-remove <servername>`, where `servername` is the name of Worklight instance that you wish to remove

The scripts automatically determine which Linux distribution you are running.

The scripts are not available for Mac OS X.

## Starting and Stopping the Worklight Server on Windows

To start and stop the Worklight Server, use the `wl-start` and `wl-stop` under the `<Worklight Root Directory>/bin` folder.

## Setting Automatic Server Restart on Windows

Worklight provides a Windows NT service that enables automatic server restart

The Worklight NT service is installed as part of the Worklight directory and resides under the `worklight\bin` folder. The service is made up of two files:

1. `WorkLightServerService.exe` – The NT service
2. `WorkLightServerService.exe.config` - A configuration file.

The service must be configured and registered with the NT services system before using it.

### NT Service configuration

Use an editor to modify the file `WorkLightServerService.exe.config` and configure the following

Key	Description	Default Value
<code>ServiceName</code>	The name that will appear in the NT services window	Worklight Server
<code>DependentService</code>	NT Services that must be started prior to starting the Worklight Server	Set to RDBMS service name if needed (for MySQL if on the same server, for Oracle – use NT service name for Oracle client)

### Registering the NT Service

- Identify the location of the .Net framework directory. Typically this would be under `C:\Windows\Microsoft.NET\Framework64\v2.0.xxx`.
- Identify the utility `InstallUtil.exe` which installs services.
- Using a command line prompt, change directory to `server\bin` in command prompt.

- Run the following command from the command prompt:

```
C:\Windows\Microsoft.NET\Framework64\v2.0.50727\InstallUtil.exe  
WorkLightServerService.exe
```

## Running as an NT Service

Under Windows NT services (Start > Control Panel > Administrative Tools > Services) you will now find the Worklight service. By default, the service is configured as automatic, meaning that it will load on Window start.

## Changing Credentials

Worklight Server configuration will typically save the following credentials:

- Username/password to the Worklight database
- Username/password to other custom databases
- Username/password to certificates which allow the stamping of apps

All credentials are stored in the `worklight.properties` file. See Appendix B for information on individual properties stored in this configuration file.

Passwords can be encrypted. For further information see Appendix B under the section: *Storing properties in encrypted format*

## Denying Access to Older App Versions

When upgrading an app to a newer version, it is sometimes useful to force users to upgrade. Denying access to an older version due to phase-out policy or due to security issues encountered in the older version.

Using the console you can deny access to a specific version, of a specific app on a specific mobile operating system as well as provide a custom message to the user.

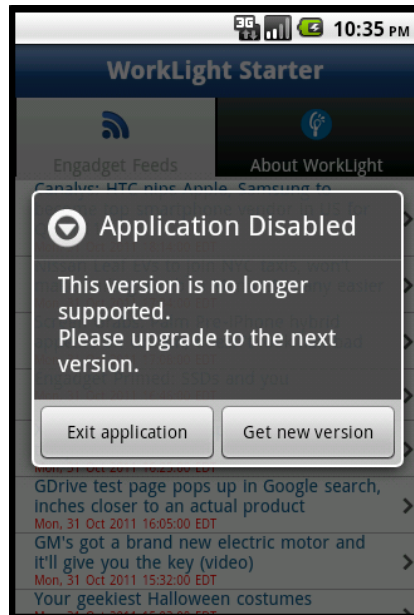
Under the right app, change the status of the application version to be disabled from Active to Disabled, and add a custom message:

The screenshot shows the 'WorkLight Starter' interface. At the top, there are tabs for 'Catalog', 'Push Notifications', 'Reports', and 'Active Users'. Below the tabs is a deployment bar with the text 'Deploy application or adapter: Choose File No file chosen' and a 'Submit' button. The main content area is titled 'WorkLight Starter' and features a lightbulb icon. Below the icon, it says 'Retrieves RSS feed from engadget.com' and 'Last updated at: 2011-10-31 22:13'. A list of application versions is shown for different platforms:

Platform	Version	Status	Notification Text	URL to app store or market
iPhone	Version 1.1	Active		
Android	Version 1.0	Disabled	This version is no longer supported. Please upgrade to the next version.	http://market.android.com/myapp
BlackBerry	Version 1.0	Active		
Windows Phone	Version 1.0	Active		
iPad	Version 1.0	Active		
Vista	Version 1.0	Install		

You can also specify a URL for the new version of the app (usually in the applicable app store or market).

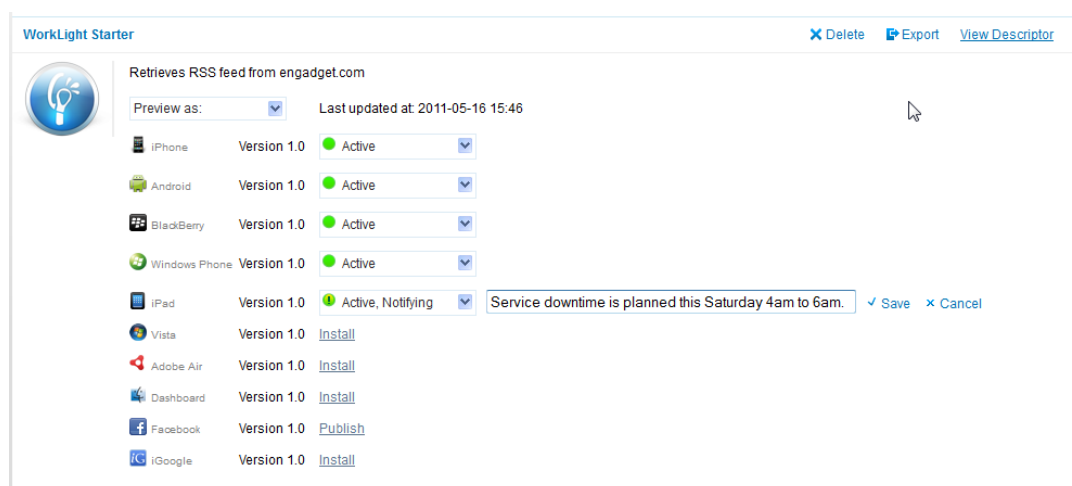
The user will receive the message and will be forced to close the app, or upgrade to the latest version.



## Displaying a Notification Message on App Startup

Similarly, you can also set a notification message that will be displayed for the user when the application starts, but will not cause the application to exit. This is useful when you want to notify application users with temporary messages, such a planned service downtime.

Under the right app, change the status of the application version from Active to Active, Notifying, and add a custom message:



# Backup and Recovery

The customization and the content (adapters and applications) should be backed up outside the Worklight instance, for example in a source control system.

The Worklight database should be backed up as is.

# Monitoring

The underlying structure of Worklight logging mechanism is a popular open source product called Log4J. Log4J has a large variety of logging options. Log4J is configured via an XML file which resides on `<Worklight Root Directory>\server\lib\log4j.xml`

## Monitoring the Worklight Server

Worklight reports errors, warnings and informational messages into a log file. Administrators should track this log file, flag out warning and errors and handle them.

The log file name is `server.log` and is located under: `<Worklight Root Directory>\server\log\server`

**Note:** Under the log folder you will find additional log files. These are all subset of the `server.log` file and should be ignored.

## Tracking Logs

Worklight suggests to use industry standard log file monitoring tools (such as Splunk) to monitor logs and highlight Errors and Warnings for treatment.

## Routing Log Files to a Different Destination

In Log4J terminology, the log output destination is called an “appender”. Administrators can configure log4J to use other appenders and route the Worklight log to a different destination.

**Example:** route log files to the Windows NT event viewer:

- Download log4j software from the web
- Extract NTEventLogAppender.amd64.dll from the zip file and rename it to NTEventLogAppender.dll; copy the changed file to C:\Windows\System32.
- Edit log4j.xml residing in `<Worklight Root Directory>\server\lib`

```
<!-- Default appenders section -->
<appender name="EVENTVIEWER"
  class="org.apache.log4j.nt.NTEventLogAppender">
  <param name="source" value="WorkLight"/>
  <param name="Threshold" value="INFO" />
  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%d %-5r %-5p [%c]
      (%t:%x) %m%n" />
  </layout>
```

```
</appender>

<!-- Default Categories, under root (saves on both file and event
viewer) -->
<root>
  <appender-ref ref="CONSOLE" />
  <appender-ref ref="FILE" />
  <appender-ref ref="EVENTVIEWER" />
</root>
```

## Audit Log

Worklight adapters can be configured to write audit log information. This configuration is part of the adapter development process.

Audit log files are created on each Worklight node and are located under: `<Worklight Root Directory>\server\log\audit\audit.log`.

## Monitoring Backend Connectivity

Worklight adapters are designed to communicate with backend systems. It is sometimes useful to monitor the raw data which is transferred between Worklight and backend systems. You can configure log4j to emit this data to a log file using the following configuration:

- Backup log4j.xml which resides under `<Worklight Root Directory>\server\lib`
- Edit log4j.xml
- Locate the tag `<appender name="FILE" ... >` remove the "Threshold" parameter or set it to a more detailed level than "INFO"
- Locate the tag `<appender name="CONSOLE" ... >` remove the "Threshold" parameter or set it to a more detailed level than "INFO"
- Locate the line `<category name="com.srndpt.adapters">`. Change "priority" from "INFO" to "DEBUG".
- Locate the line `<category name="com.worklight.integration">`. Change "priority" from "INFO" to "DEBUG".
- Save the file and restart the Server

**Warning:** This configuration can generate significant amounts of data and is recommended only in a testing environment or for a very short time under a production environment

# Diagnosing Login & Authentication Issues

It is possible to diagnose login and authentication issues by modifying the logging sensitivity as follows:

- Backup log4j.xml which resides under `<Worklight Root Directory>\server\lib`
- Edit log4j.xml
- Locate the line `<category name="com.worklight.auth">`. Change "priority" from "INFO" to "DEBUG".
- Save the file and restart the Server

**Warning:** This configuration can generate significant amounts of data and is recommended only in a testing environment or for a very short time under a production environment

## Is-Alive query

Worklight provides query URLs that help determine Server vitality. Such queries are typically used from a load balancer or from a monitoring app (e.g. Patrol).

Queries can be done for the Server as a whole, for a specific adapter, specific app or combination of. Example queries:

Query	What it does
<code>http://server-url:8080/ws/rest/vitality</code>	Check the Server as a whole
<code>http://server-url:8080/ws/rest/vitality?app=MyApp</code>	Check the Server and the <i>MyApp</i> application
<code>http://server-url:8080/ws/rest/vitality?app=MyApp&amp;adapter=MyAdapter</code>	Checks the Server, the <i>MyApp</i> application, and the <i>MyAdapter</i> adapter.

Queries return XML with a list of <ALERT> tags. One for each test

### Example Query and Response

`http://server-url:8080/ws/rest/vitality?app=MyApp`

A successful response will return an <ALERT> for each of the two tests and will look like this:

```
<ROOT>
  <ALERT>
    <DATE> 2011-05-17T15:31:35.583+0300 </DATE>
    <EVENTID>0</EVENTID>
    <SUBJECT>SRV</SUBJECT>
    <TYPE>I</TYPE>
```

```

    <COMPUTER>worklight.acme.com</COMPUTER>
    <DESCRIPTION>Server is running</DESCRIPTION>
  </ALERT>
  <ALERT>
    <DATE> 2011-05-17T15:31:35.640+0300 </DATE>
    <EVENTID>0</EVENTID>
    <SUBJECT>APPL</SUBJECT>
    <TYPE>I</TYPE>
    <COMPUTER>worklight.acme.com</COMPUTER>
    <DESCRIPTION>Application 'MyApp' is deployed</DESCRIPTION>
  </ALERT>
</ROOT>

```

### Return values:

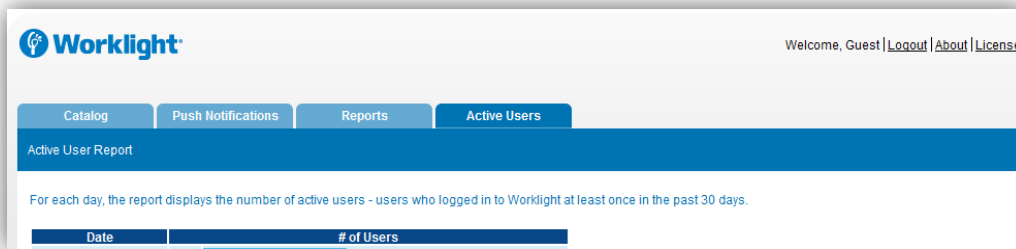
Return attribute	Possible values
DATE	Date value in JavaScript format
EVENTID	0 for the running Server, deployed adapter, or deployed application. 1 for not deployed adapter 2 for not deployed application 3 for malfunctioning Server
SUBJECT	SRV for Worklight Server ADPT for adapter APPL for application
TYPE	I – valid, E – error
COMPUTER	Reporting computer name
DESCRIPTION	Status description in plain text

The returning XML contains additional attributes. These attributes are constant, remain undocumented and should not be used

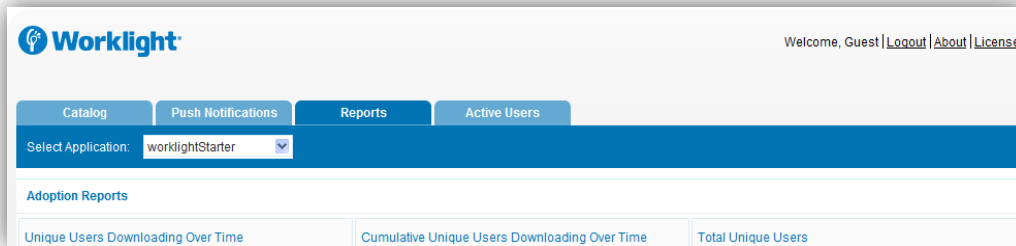
# Reports

Worklight provides three reporting mechanisms:

1. **Active Users report**, useful when the Worklight licensing model is based on active users.



2. **Adoption and usage reports**. Built-in reports on application adoption and usage.



The reports are described in the following table:

Name	Description	Type	Parameters
Unique users Downloading over Time	Number of new unique users who downloaded/installed an application (and did not downloaded/installed in the past) each day	Multiple bar or line charts, line per application environment and for the total	<ul style="list-style-type: none"> <li>• Start date</li> <li>• End date</li> <li>• App ID</li> <li>• Default: Last week (today – 1 to today – 8)</li> </ul>
Cumulative Unique Users Downloading over Time	Number of cumulative unique users who downloaded/installed an application (and did not downloaded/installed in the past) each day.	Multiple bar or line charts, line per application environment and for the total	<ul style="list-style-type: none"> <li>• Start date</li> <li>• End date</li> <li>• App ID</li> <li>• Default:</li> </ul>

Name	Description	Type	Parameters
			Last week
Total Unique Users	Total number of unique users who ever downloaded an application.	Pie chart, with sectors representing application environments	<ul style="list-style-type: none"> <li>App ID</li> <li>Date, default: yesterday</li> </ul>
Total Daily Visits	Total number of daily visits per application environment within the specified period. A daily visit is an access to the application by a specific user on a specific day. A user accessing the application more than once on a specific day counts as a single visit.	Pie chart, with sectors representing application environments	<ul style="list-style-type: none"> <li>Start date</li> <li>End date</li> <li>App ID</li> <li>default: last week</li> </ul>
Daily Visits over Time	Daily number of visits, total and per application environment.	Multiple bar or line charts, line per application environment and for the total	<ul style="list-style-type: none"> <li>Start date</li> <li>End date</li> <li>App ID</li> <li>default: last week</li> </ul>
Seven-day Activity Distribution	Activity distribution throughout the day, counting number of hourly visits, total and per application environment. An hourly visit is the application being open for a specific user in a specific environment on a specific hour.	Multiple bar or line charts, line per application environment and for the total	<ul style="list-style-type: none"> <li>start date (default: last week)</li> <li>App ID</li> </ul>

3. **Raw Data for external consumption.** In addition, Worklight emits raw data which allows the organization's OLAP system to extract the required information and present it via corporate reporting mechanisms.  
Important note: Worklight appends information to this data feed. The assumption is that deletion logic is handled by the customer. Raw data feeds can grow very fast.

## Activating Reports

### Active Users reports

Active user reports are always on.

## Out-of-the-box reports

**Step 1:** Enable reports in the `worklight.properties` file. Set `reports.produceReports` to `true` as described in Appendix B.

**Step 2:** Worklight reports are built using the Worklight technology itself. Thus, reports require a special report Adapter and report App. Using the Worklight console, deploy the Adapter and App which can be found under `<Worklight Root Directory>\resources\reports`

**Step 3:** Consider moving the reports tables to a different database. See Appendix B for further details

**Step 4:** Validate that reports are working by browsing to the second Console tab called *Reports*. Note that the graphs themselves will only appear after 24 hours when the reports activity daemon has aggregated initial activity data.

Important upgrade note: If you are upgrading from an older version of Worklight, the reports database will require an upgrade before working. To receive the script, please contact Worklight customer support.

## Raw data

- **Step 1:** Enable raw data output in the `worklight.properties` file. Set `reports.exportRawData` to `true` as described in Appendix B.
- **Step 2:** Consider moving the raw data table to a different database. See Appendix B, section “reports database setup” for further details
- **Step 3:** Validate that reports are working by viewing data in the table `APP_ACTIVITY_REPORT` in the database

The raw data table contains the following information

Column	Description
ACTIVITY_TIMESTAMP	Time of entry (in UTC)
GADGET_NAME	Worklight App name
GADGET_VERSION	App version
ACTIVITY	See activity list below
ENVIRONMENT	Application environment name (iPhone, Android, etc.)
SOURCE	User identifier
ADAPTER	Worklight Adapter name
PROC	Worklight Procedure name
USERAGENT	User agent from HTTP header of client device
SESSION_ID	A unique identifier for the user’s session on

Column	Description
	the Server
IP_ADDRESS	IP address of the client

Possible activities:

Activity	Description
Init (*)	App initialization
Login	Successful authentication via the app
Adoption New (*)	New user first time ever
Adoption (*)	Known user, but logs in via a new application environment
Query	Procedure call to an adapter
Logout (*)	User logout
SetUserPrefs (*)	Client API call to store user preferences
LogActivity	Client API call to intentionally log information into this database table.

(\*) Currently not available via native Worklight API

## Report Data Maintenance

### Out of the box reports

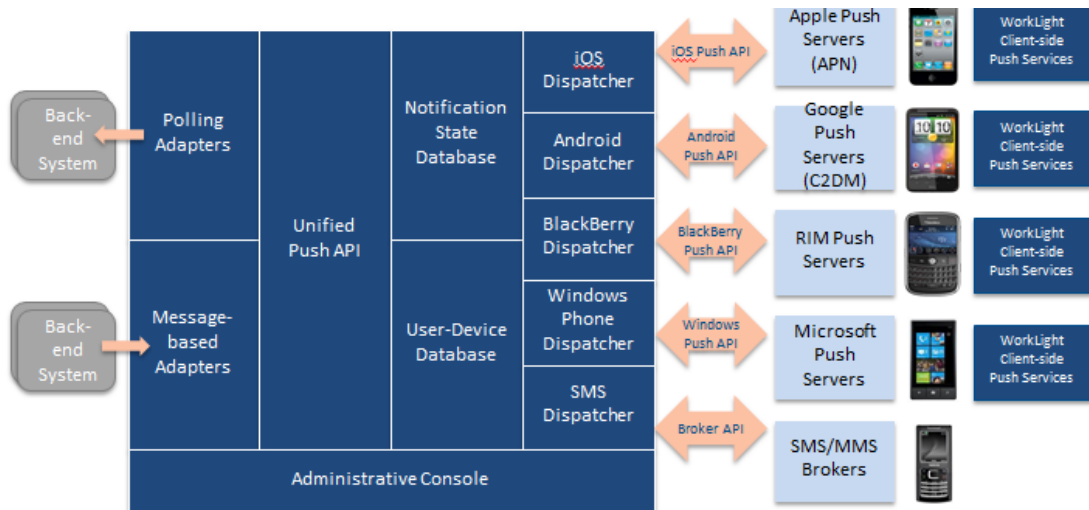
Out of the box reports are based on raw data that is stored in 7 tables called GADGET\_STAT\_[#]. These tables are maintained by Worklight and are automatically purged on a daily basis

### Raw Data

Worklight appends information to this data feed. Worklight will not delete or update existing records as the assumption is that data ownership moves to the customer organization and the data is typically consumed by a BI system such as Cognos, Business Objects and the like.

# Push Notification

Worklight's unified push notification mechanism enables sending mobile notifications to the mobile phones. Notifications are sent via the vendor infrastructure. For example, iPhone notifications are sent from the Worklight Server to specialized Apple servers and from there to the relevant phones.



Push notification currently works for iOS and Android.

There are two types of configuration properties related to the push notifications framework:

- Setting of length of the internal message queue, which temporarily stores notifications before being pushed to APNS and C2DM. This is done by setting the property `push.internalQueue.maxLength`.
- Setting the optional proxy through which notifications should be sent to APNS and C2DM. This is done by setting the properties `push.c2dm.proxy.*` and `push.apns.proxy.*`.

All configuration properties are described in detail under [Appendix B, Push Notification Settings](#).

## The Push Notification Console

The Push Notifications tab in the Worklight Console provides you a quick view of the various entities in the push notification chain.

Catalog **Push Notifications**

**Event Sources**

Event Source 1 Active ■

Adapter 1

831,792 subscribers

▶ Disable

Event Source 2 Disabled ■

Adapter 2

831,792 subscribers

▶ Enable


**Push Services**

Apple Active ■

▶ Disable    ▶ Get error report

Google Error ■

Waiting to complete authentication



Enter the text above

**Notifications to Apps**

Application 1 Notifications Disabled ■

▶ Enable

Application 2 Notifications Active ■

▶ Disable

Application 3 Notifications Active ■

▶ Disable

© Copyright 2011 Worklight Inc. All rights reserved.

The left column displays the list of data sources configured in your Worklight Server, including the number of users subscribed to notifications from each source.

The middle column displays the push notification services used via the Worklight Server. The Console displays the activity status for each service and allows performing manual operations necessary to fix online problem, such as responding to the captcha for the Google Notification Service.

The right column displays deployed applications, which can receive push notifications.

# Disaster Recovery Site

Within the site itself, Worklight provides a redundancy at every level: compensating load balancers, multiple Worklight Servers which scale linearly and Database redundancy via Oracle RAC. Still, some customers prefer to provide another level of redundancy via a disaster recovery **site**.

A disaster recovery site, is a second, physically separate IT center on which a copy of the IT systems exists and springs into operation if the original site is down.

Worklight supports the creation of a separate disaster recovery site and in fact such a site exists for some of our customers.

The key administrative factors for such a site are:

- Architecture
- Data mirroring from master to backup site.
- Switching to back up site on disaster.
- Switching back to master site

**Important note:** The success of a recovery site is in the details. In order for such a site to work, one must develop a strict written procedure and test it on a regular basis.

## Architecture

The Architecture of the backup site is a copy of the original site. Special care must be taken to:

- Provide access to all corporate backend systems.
- Create a switch which transfers incoming requests from master to backup site.

It is important to note that Worklight relies on one single database, so a live-live configuration of master and back-up sites is not encouraged. Still, it could be done in some cases depending on business use. Please contact Worklight support for further information

## Data mirroring

For the backup site to work, data on the master site must be mirrored to the backup set on a regular basis:

Component	Description	Mirror frequency.
Worklight Database	All tables must be mirrored with the exception of report tables and cache tables.  Note that this leaves only tables which are relatively small in size	Highly dependent on implementation and can range from few minutes to 24 hours. For further details please contact Worklight support

Component	Description	Mirror frequency.
Worklight Software, customization and content	Any change in Worklight software, customization or content must also be installed on the mirror servers	As it happens

## Switching to Backup Site

The effects of switch are:

1. All clients will lose context and disconnect. In case of an authenticated App this would mean the user will be prompted for re-login.
2. Report information will be lost (unless previously mirrored)
3. Cache is lost. If Cache was implemented for various queries – an additional server fetch will be required to fill cache.

## Switching back to Master Site

Before switching back to master site, the database must be mirrored-back to the master site.

# Support

- Under the Worklight website at <http://www.worklight.com/> you will find the entire Worklight documentation set, Worklight training material and online forums where you can post questions.
- you may send your questions to [support@worklight.com](mailto:support@worklight.com)
- As a customer, you have access to Worklight account manager through which you can contact a Worklight support representative directly.

In case you would like Worklight to troubleshoot your setup, please archive the following folders and send them to [support@worklight.com](mailto:support@worklight.com)

#	Folder name	Location	Remark
1.	Log	<Worklight Root Directory>\server\log	Including sub folders.
2.	Conf	<Worklight Root Directory>\server\conf	Including sub folders.
3.	Data	<Worklight Root Directory>\server\data	Contains the deployed adapters. Note: may contain private data.
4.	widget-resources	<Worklight Root Directory>\server\widget-resources	Contains the deployed applications. Note: may contain private data.

# Appendix A: Environment Setup and Pre-requisites

The following section outlines the required hardware and software for the different Worklight components including system requirements and network requests used by Worklight and should be enabled by network and security administrators

## Production Worklight Application Server

Component	Requirement
Computer and Processor	Any modern quad core Intel machine
Memory	4GB
Hard disk	50GB of free disk space
Operating System	One of: <ul style="list-style-type: none"><li>• Red Hat Enterprise Linux (RHEL) Version 5.5 64 bit – English US Locale</li><li>• Windows 2008 R2 Server 64 bit edition – English (United States) locale.</li></ul>
Network Requirements	<ul style="list-style-type: none"><li>• Access to backend systems – directly or via proxy</li><li>• Access to authentication infrastructure</li><li>• Access to the Worklight database server</li><li>• Access to mail server if required</li><li>• Access to Facebook API (api.facebook.com) if required – directly or via proxy</li><li>• By default, Worklight Server is configured at port 8080. See Appendix B on how to change this default setting.</li></ul>
Additional third-party Software Requirements	<ul style="list-style-type: none"><li>• If My SQL is used, My SQL Connector/J 5.1</li><li>• If Oracle is used, Oracle 11g client</li></ul>

### Notes

Multiple Worklight Servers can be configured in order to achieve scaling and redundancy

## Production Worklight Database Server

Component	Requirement
Database	One of:

	<ul style="list-style-type: none"> <li>• MySQL 5.5 with latest patches</li> <li>• Oracle 11g Database server, Standard or Enterprise Editions</li> </ul>
Hardware	Any machine that is supported by the above database server
Network	<p>By default, MySQL 5 is configured at port 3306.</p> <ul style="list-style-type: none"> <li>• See MySQL Documentation on how to change this default setting.</li> <li>• Once changed, the port must also be changed in the worklight properties file. See Appendix B on how to change this default.</li> </ul>

### Notes

Clustering technologies such as Oracle RAC can be used to achieve redundancy.

## Load balancer

Any commercial Load balancer, software or hardware which can support the following:

- Sticky session
- Reverse proxy capabilities
- Optional: SSL Acceleration

## Appendix B: Worklight Properties

Worklight contains many configuration options which are important for production environment. These options are stored in the `worklight.properties` file, which is located in the `<Worklight Root Directory>\conf` directory.

### Configuring the Worklight Server Location

These configuration settings are normally done by the Worklight Installer. However, you can change it after the Server has been installed if necessary.

#### To configure the Worklight Server location:

- Set the values of the following properties in the `worklight.properties` file:

Property Key	Property Value
<code>publicWorkLightHostname</code>	<p>The IP address or hostname of the computer running Worklight. If the Worklight Server is behind a reverse proxy, the value should be the IP address or hostname of the reverse proxy.</p> <p>This property must be identical for nodes within the same cluster.</p> <p>Default: IP of current machine.</p>
<code>publicWorkLightPort</code>	<p>The port for accessing the Worklight Server. If the Worklight Server is behind a reverse proxy, the value should be the port for accessing the reverse proxy.</p> <p>This property must be identical for nodes within the same cluster.</p> <p>Default: Same as <code>local.httpPort</code> below.</p>
<code>publicWorkLightProtocol</code>	<p>The protocol for accessing the Worklight Server. The valid values are <code>http</code> and <code>https</code>. If the Worklight Server is behind a reverse proxy, the value should be the protocol for accessing the reverse proxy.</p> <p>This property must be identical for nodes within the same cluster.</p> <p>Default: <code>HTTP</code>.</p>
<code>local.bindAddress</code>	<p>This IP address on which Tomcat listens for http requests. The default is the hostname on which the Server is installed.</p> <p>Change this value when it is required to listen only on the IP address of a reverse proxy which forwards the requests to the Server.</p> <p>Default: <code>0.0.0.0</code></p>

Property Key	Property Value
<code>local.httpPort</code>	<p>The port on which Tomcat listens to http requests.</p> <p>Default: 8080.</p> <p>Note: The default is for Worklight to run over Tomcat. Tomcat uses an additional port (8005). To run multiple instances of Worklight on the same machine, you must change this port for one of the servers. This is done by changing the port in <code>&lt;Worklight Root Directory&gt;\tomcat\conf\server.xml</code></p>

## Worklight Database Setup

Worklight uses defaults to access the Worklight database. The following keys allow you to change settings.

Property Key	Property Value
<code>DB_TYPE</code>	<p><b>MySQL:</b> <code>MYSQL_DB_TYPE</code></p> <p><b>Oracle:</b> <code>ORACLE_DB_TYPE</code></p>
<code>wl.db.username</code>	<p>Worklight database user name.</p> <p>Default: <code>Worklight</code></p>
<code>wl.db.password</code>	<p>Worklight database password.</p> <p>Default: <code>Worklight</code></p>
<code>wl.db.driver</code>	<p><b>MySQL:</b> <code>com.mysql.jdbc.Driver</code></p> <p><b>Oracle:</b> <code>oracle.jdbc.OracleDriver</code></p>
<code>wl.db.url</code>	<p>JNDI URI to database. Examples:</p> <ul style="list-style-type: none"> <li><code>jdbc:oracle:thin:@heidi:1521:orcl</code></li> <li><code>jdbc:mysql://localhost:3306/Worklight42</code></li> </ul>
<code>hibernate.dialect</code>	<p><b>MySQL:</b> <code>org.hibernate.dialect.MySQL5InnoDBDialect</code></p> <p><b>Oracle:</b> <code>org.hibernate.dialect.Oracle10gDialect</code></p>
<code>reports.produceReports</code>	<p>Emit report data. This flag relates to the Worklight built-in reports</p> <p>Default is false.</p>
<code>reports.exportRawData</code>	<p>Emit report data. This flag relates to a different report mechanism which only exports raw data for usage by external reporting/BI systems</p> <p>Default is false.</p>

# Reports Database Setup

By default all Worklight reports mechanisms use a single reports database. The reports database is set to be the same database as the Worklight database. This default can be changed using the parameters below. For further information see the Reports chapter in this guide.

Property Key	Property Value
<code>wl.db.reports-url</code>	JDBC path to Worklight Reports database Default: refers to Worklight database
<code>wl.db.reports-username</code>	Worklight Reports database user name. Default: refers to Worklight database
<code>wl.db.reports-password</code>	Worklight Reports database password Default: refers to Worklight database

# Protecting the Worklight Console

You can protect the Worklight Console by defining user credentials required to access it. These settings can also be encrypted as described in [Storing Properties in Encrypted Format](#).

Property Key	Property Value
<code>console.username</code>	Name of the user that can access the Console
<code>console.password</code>	User's passwords

In addition to defining these two properties, you should also configure the `authenticationConfig.xml` file, located under `<Worklight Root Directory>\server\conf`, as described in the *Worklight 4.2.1 Developer's Reference Guide* document.

# Push Notification Settings

Property Key	Property Value
<code>push.internalQueue.maxLength</code>	Maximum length of the internal queue of messages waiting to be pushed to each push service
<b>C2DM Proxy Settings</b>	
<code>push.c2dm.proxy.enabled</code>	Whether access to Google C2DM needs to be performed via proxy. Default is <code>false</code> .

Property Key	Property Value
<code>push.c2dm.proxy.protocol</code>	May be either <code>http</code> or <code>https</code>
<code>push.c2dm.proxy.host</code>	C2DM proxy host
<code>push.c2dm.proxy.port</code>	C2DM proxy port. Use <code>-1</code> for the default port.
<code>push.c2dm.proxy.user</code>	Proxy username, if the proxy requires authentication. Empty username means no authentication.
<code>push.c2dm.proxy.password</code>	Proxy password, if the proxy requires authentication.
<b>APNS proxy settings</b>	
<code>push.apns.proxy.enabled=false</code>	Whether access to APNS needs to be performed via proxy. Default is <code>false</code> .
<code>push.apns.proxy.type</code>	Must be <code>SOCKS</code>
<code>push.apns.proxy.host</code>	APNS proxy host
<code>push.apns.proxy.port</code>	APNS proxy port

## Miscellaneous Settings

Property Key	Property Value
<code>serverSessionTimeout</code>	Client inactivity timeout, after which the Worklight session is invalidated. Default is 30 minutes
<code>bitly.username</code>	User name for accessing the bit.ly API for creating a shortened URL for mobile web apps through the Worklight Console
<code>bitly.apikey</code>	The bit.ly API Key

## Storing Properties in Encrypted Format

Some properties are too sensitive to be written in clear text within the properties file. To encrypt properties, please go through the process below.

- Properties in `worklight.properties` can be kept either in open or in encrypted form.
- The encrypted property is determined by the suffix `".enc"` on its name, for example:  
`console.password.enc=TYakEHRba3rIU7pNjxtDxoAdqijKIEt7cy4mCr0iaEj0rY08ODK00yqR`

- The Worklight configuration is accessed for a property. If the property is not found, but the same encrypted property (with .enc suffix) is defined, Worklight automatically decrypts the value and returns it to the caller.

## Storing the master key

All of the encrypted values use the same secret key which is stored in the special variable called **worklight\_enc\_password**. This variable is defined as an Operating system environment variable

- Windows: set an environment variable under the user running the Worklight Server. When running under a Windows NT service, the password should be defined as a service property via the registry editor. See <http://support.microsoft.com/kb/197178> for more information
- Linux: set the environment variable

## Encryption

To encrypt Worklight properties, use the `encrypt.bat` utility under the `server\bin` folder. This utility accepts a file which contains the properties that should be encrypted and the encryption password. The utility outputs the encrypted values to the same file (so that sensitive data is deleted).

Example:

The input file for the encryption is called `secret.properties` and contains the following data:

```
worklight_enc_password=abc123
certificate.password=certificatepwd123
wl.db.password=edf545
```

After calling the utility the file `secret.properties` contains the following data:

```
#Copy the contents of this file to the worklight.properties file.
#Keep the password value in the secure system property
    worklight_enc_password

#Sun Sep 05 14:27:28 IDT 2010
certificate.password=0UMedjRFHF8tVjTT1MtrUDA5AB6ncgrjQ00f9nN+z1L0r
wl.db.password=aslj34rjk3IOU8989JKJK787hhhuhkjhk67hkjhk33+333j
```

# Obsolete Properties

The following properties are no longer required:

Category	Properties
Proxy settings	<code>proxy.enabled</code> , <code>proxy.nonProxyHosts</code> , <code>proxy.host</code> , <code>proxy.port</code> , <code>proxy.username</code> , <code>proxy.password</code> , <code>https.proxy.host</code> , <code>https.proxy.port</code>
Public resource server settings	<code>publicResourceServer.deployDestination</code> , <code>publicResourceServer.host</code> , <code>publicResourceServer.port</code> , <code>publicResourceServer.filesRootDir</code>
Environments	<code>environment.igoogle</code> , <code>environment.netvibes</code> , <code>environment.iphone</code> , <code>environment.vista</code> , <code>environment.dashboard</code> , <code>environment.embedded</code> , <code>environment.facebook</code> , <code>environment.air</code> , <code>environment.android</code> , <code>environment.blackberry</code>
Certificate settings	<code>certificate.certificatesDirPath</code> , <code>certificate.keyStoreFilePath</code> , <code>certificate.keyAlias</code> , <code>certificate.keyStorePassword</code> , <code>certificate.keyAliasPassword</code> , <code>certificate.PFXFilePath</code> , <code>certificate.password</code> , <code>certificate.DERFilePath</code> , <code>certificate.P7BFilePath</code> , <code>vista.linux.osslsigncodeFilepath</code>
Push notification settings	<code>push.apns.certificatePassword</code> , <code>push.c2dm.senderID</code> , <code>push.c2dm.senderPassword</code>
Miscellaneous settings	<code>devmode</code> , <code>guid</code> , <code>wlclientTimeout</code> , <code>backend.request.timeout</code>

# Appendix C: JVM & Tomcat Tuning

The Worklight Server is based on Apache Tomcat and the Java Virtual Machine. For optimal Worklight performance, it is important to tune these underlying software components.

## JVM heap

We recommend providing the JVM with as much memory possible within the machine's RAM limits, taking into consideration other processes running on that machine. As a typical example a machine with 4GB RAM, we can define a 2GB heap.

Linux: Edit the file `worklight` residing in the `bin` directory. Add the following line at the beginning of the file after the `WORKLIGHT_HOME` definition.

```
export CATALINA_OPTS="-Xms2048m -Xmx2048m $CATALINA_OPTS"
```

Windows: Edit the file `wl-start.bat` residing in the `bin` directory. Add the following line before the line calling "catalina.bat":

```
set CATALINA_OPTS=-Xms2048m -Xmx2048m %CATALINA_OPTS%
```

## Tomcat

By default, Worklight configures Tomcat to handle 50 requests concurrently. This is enough to process about 200 requests per second per Server, assuming that a single request will take less than 250ms.

By default, Worklight configures Tomcat for a database pool of 50 connections, which corresponds to the number of Tomcat threads above.

For finer grained configuration, please contact Worklight customer support.

# Appendix D: Internal Worklight Database Tables

The following table below provides a list of common Worklight database tables and their usage.

Note that the database is accessed by the Worklight Server only, should not be written into and is likely to change from one Worklight release to another

Name	Description
APP_VERSION_ACCESS_DATA	Exceptions on App availability used for the “Deny App” feature
AUTH_ASSOCIATED_IDENTITY	Association between the user IDs in different realms
AUTH_PROTECTED_RESOURCE	URL patterns and realms for deployed applications
BACKEND_SUBSCRIPTIONS	Definitions of the cached data: procedures + parameter values
CLUSTER_SYNC	Cluster synchronization time
G_ADOPTION_NEW_USERS_ONLY_D	Statistics for new users
G_USER_ACTIVITY_D_PER_USER	Statistics for user activity
G_USER_ACTIVITY_H	Hourly statistics per users
APP_ACTIVITY_REPORT	Statistics for application activity
GADGET_APPLICATIONS	Deployed gadgets per application environment
GADGET_INSTANCE	Gadgets instance ID (new app download from the Server)
GADGET_REPORT_INSTANCE	The gadget instance that is used in the reports console (6 in total are used)
GADGET_STAT_1	Statistics for # day of the week
GADGET_STAT_2	Statistics for # day of the week
GADGET_STAT_3	Statistics for # day of the week
GADGET_STAT_4	Statistics for # day of the week
GADGET_STAT_5	Statistics for # day of the week
GADGET_STAT_6	Statistics for # day of the week
GADGET_STAT_7	Statistics for # day of the week

Name	Description
GADGET_USER	A mapping of user-ids and the gadget instances they use
GADGET_USER_PREF	User preferences according to unique user identifier
GADGETS	Deployed gadgets
NOTIFICATION_APPLICATION	Push notification table
NOTIFICATION_DEVICE	Push notification table
NOTIFICATION_MEDIATOR	Push notification table
NOTIFICATION_USER	Push notification table
PROPERTIES	N/A
REPORTS_JOBS_LOG	Statistics job logs
USAGE_DATA	Usage reports.

# Appendix E: HTTP Interface of Production Server

## Application API Requests

### Request Structure

```
{Protocol}://{Worklight Server}/apps/services/api/{Application ID}/{Application Environment}/{Application Instance ID}/{Action}
```

### Request Headers

Header Name	Data Type	Description	Valid Values
x-wl-app-version	String	Version of the application	
X-WL-Cookie	String	Protection mechanism for XSS attacks.	

### Request Elements

Element	Data Type	Description	Valid Values
Protocol	String		HTTP
Worklight Server	String	Hostname or IP address (and possibly port) identifying the Worklight Server	
Application ID	String	Unique Identifier of the application within the Worklight Server. Every application deployed on the Worklight Server must have a unique identifier	Up to 256 alphanumeric and underscore characters
Application Environment	String	Name of the environment the application is running on	preview, igoogole, vista, dashboard, iphone, embedded, facebook, air
Application	positive	Unique identifier of a	1, 2, ...

Element	Data Type	Description	Valid Values
Instance ID	Integer	application instance. Each application downloaded receives a different id.	
Action	String	Requested action	Details below

## Actions

Action	HTTP Request	Parameters
init	POST	x, isAjaxRequest, _ – refer to <a href="#">Common Parameters</a> below
heartbeat	POST	x, isAjaxRequest, _ – refer to <a href="#">Common Parameters</a> below
setprefs	POST	x, isAjaxRequest, _ – refer to <a href="#">Common Parameters</a> below userprefs – a JSON block
logactivity	POST	x, isAjaxRequest, _ – refer to <a href="#">Common Parameters</a> below activity – string
query	POST	x, isAjaxRequest, _ – refer to <a href="#">Common Parameters</a> below filterList – JSON block parameterList – JSON block sorterList – JSON block <b>Note:</b> When the action is <code>query</code> , the request URL has the following structure: <code>.../query/{Adapter Name}/{Procedure Name}</code> where <code>Adapter Name</code> and <code>Procedure Name</code> are strings
logout	POST	x, isAjaxRequest, _ – refer to <a href="#">Common Parameters</a> below
login	GET	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below realm – string
authentication	GET	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below action – string

Action	HTTP Request	Parameters
		parent – string
updates	GET	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below skin – current skin name (string) checksum – the checksum of the current skin (string) skinLoaderChecksum – the checksum of the skin selection code (string)
getup	POST	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below
deleteup	POST	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below userprefkey – the user preference to delete
getuserinfo	POST	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below
getgadgetprefs	POST	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below
notifications	POST	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below subscribe – json string containing subscribe options unsubscribe – when specified, designates an unsubscribe action updateToken – the update notification token (string) adapter – the name of the notification adapter (string) eventSource – the name of the notification event source (string) alias – notification subscription alias (string)
fbcallback	GET or POST	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below popup – string
composite	POST	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below requests – a JSON string containing information

Action	HTTP Request	Parameters
		on other actions to invoke. This action is used to combine several actions in a single HTTP request.
appversionaccess	GET	x, isAjaxRequest – refer to <a href="#">Common Parameters</a> below

## Common Parameters

Parameter	Values	Comments
x	A random decimal number	Included with all GET and POST requests to prevent caching by the Vista Sidebar HTTP engine.
isAjaxRequest	true	Included with every GET and POST request only from Adobe AIR application
–	None	Included with every POST request only from Webkit-based browsers and application frameworks: Safari, Chrome, and Adobe AIR

# Web Application Resource Requests

## Request Structure

```
{Protocol}://{Worklight Server}/apps/services/www/{Application ID}/{Application Environment}/{Application Resource Path}
```

## Request Elements

See [Application API Request Elements](#) for details about the following request elements: Protocol, Worklight Server, Application ID, and Application Environment.

Element	Data Type	Description	Valid Values
Application Resource Path	String	HTML, image, JavaScript, CSS and any other application resource	Example values: img/bg.png, myWidget.html, js/myWidget.js

# Preview Application Resource Requests

## Request Structure

```
{Protocol}://{Worklight Server}/apps/services/preview/{Application ID}/{Application Environment}/{Application Version}/{Application Resource Path}
```

## Request Elements

See [Application API Request Elements](#) for details about the following request elements: Protocol, Worklight Server, Application ID, and Application Environment.

Element	Data Type	Description	Valid Values
Application Resource Path	String	HTML, image, JavaScript, CSS and any other application resource	Example values: img/bg.png, myWidget.html, js/myWidget.js

# Appendix F: Upgrading a Production 4.0.3 Worklight Server to 4.2.1

## Introduction

Upgrading the production Worklight Server from version 4.0.3 to version 4.2.1 consists of the following steps:

- Upgrading your customization and content (adapters and apps) to 4.2.1 in your development environment
- Creating a backup of your database
- Installing the 4.2.1 Worklight Server
- Upgrading the database and configuring the Server to use the database
- Setting your 4.2.1-ready customization and content on the new 4.2.1 Server
- If upgrading from 4.0.3, completing the database upgrade

## Developer

1. Install the Worklight 4.2.1 development environment and upgrade your applications' source code and adapters to 4.2.1
2. Create a Worklight project with the production apps and adapters. Set the appropriate customization.
3. Create and deploy a `.wlappp` file for each version and each environment that exists on the old Worklight 4.0 Server. For each version in Worklight 4.0 do the following:
  - a. Change the descriptor file such that the version number of all supported environments will be equal to the application version
  - b. Build your applications. The builder creates application deployable files for your applications in all applicable environments. The application deployable files are called `*-all.wlappp` and are placed in the `bin` folder under each project.
4. Deploy your adapters and export them to receive `*.adapter` files.
5. Provide the updated apps and adapters to the administrator.

## Administrator

1. Create a copy of your production Worklight customization.
2. In the new copy, update the `worklight.properties` file:
  - a. Remove the `devmode` property. Add instead the `production=true` property.
  - b. Optionally remove obsolete properties, as defined in [Obsolete Properties](#).
3. Backup your production Worklight database.
4. Create a new folder and install a Worklight 4.2.1 Server into this folder. We'll refer to this folder as `[worklight-4.2.1-home-folder]`.

5. Shut down the Worklight 4.0.3 Server.
6. Start the database upgrade process, by running the following pre-deploy upgrade script:
  - For My SQL, run: `mysql [worklight-database-name] -u root -p < [worklight-4.2.1-home-folder]/server/bin/upgrade/upgrade-mysql-pre-deploy.sql`
  - For Oracle, run the following SQL script in your preferred database console: `[worklight-4.2.1-home-folder]/server/bin/upgrade/upgrade-oracle-pre-deploy.sql`
7. Set the customization you just updated.
8. Start the Worklight 4.2.1 Server.
9. Deploy the adapter and application files to the Worklight Server. You can do that from the Worklight Console or using the command-line deployer, described in this document.

If you are using an Oracle database, you finished the upgrade procedure.

Perform the following steps only if you are using a My SQL database and are upgrading from Worklight 4.0.3:

10. Shut down the Worklight 4.2.1 Server.
11. Run the post-deploy upgrade script by using the following command:  
`mysql [worklight-database-name] -u root -p < [worklight-4.2.1-home-folder]/server/bin/upgrade/upgrade-mysql-post-deploy.sql`
12. Start the Worklight 4.2.1 Server.

You finished the upgrade procedure.