Quark Publishing Platform 10.5 System Administration Guide

Contents

Introducing Quark Publishing Platform administration tasks

Administering a Quark® Publishing Platform™ environment requires extensive hardware and software maintenance in addition to the controls in Quark Publishing Platform. This guide describes tasks administrators perform for security, system configuration adjustments, and integration with other systems. For information about administrating in the Quark Publishing Platform interface, see *A Guide to Quark Publishing Platform*. For information about installing Quark Publishing Platform software, see the *Quark Publishing Platform ReadMe*.

Deploying Quark Publishing Platform Server

You can deploy Quark Publishing Platform Server in an external Tomcat installation, in IBM WebSphere, and in Oracle WebLogic.

Setting up environment variables

To prepare for deployment, set the following environment variables.

➡ If you are deploying Quark Publishing Platform Server as a service, you must set these as global system environment variables.

- **MAGICK_HOME**: `[ImageMagick installation directory]\bin`
- **MAGICK_FILTER_MODULE_PATH**: `%MAGICK_HOME%/modules/filters`
- **MAGICK_CODER_MODULE_PATH**: `%MAGICK_HOME%/modules/coders`

Configuring the JVM

Regardless of which application server you are using, you must increase the memory available to the JVM as follows to accommodate Quark Publishing Platform Server:

- Set the minimum Java heap memory to 1024MB.
- Set the minimum PermGenSpace to 512MB.

Preparing the EAR file for deployment

A Platform EAR build consists of the following things:

- Platform Server Configuration files.
- Binaries of dependencies.
- A tool to generate the JEE EAR file.

The following folders are included in an EAR build:

- `conf`: Contains Platform Server configuration files.

- `database`: Contains database creation, update, and support files.
- `ImageMagick`: Contain ImageMagick application binaries.
- `Jaws`: Contains JAWS application binaries.
- `DITA-OT1.6`: Contains DITA OT binaries.
- `publishing`: Contains configuration and resource files for the Quark Publishing Platform framework.

The Platform Server application consists of following modules:

- `admin.war`: The Quark Publishing Platform administration Web application.
- `workspace.war`: The Quark Publishing Platform workspace Web application.
- `webservices.war`: The Quark Publishing Platform Server Web services module.
- `rest.war`: The Quark Publishing Platform server REST interface module.
- `messaging.war`: The HTTP-based messaging interface.
- `qxpsm.war`: QuarkXPress Server Manager.
- `qxpsmadmin.war`: The QuarkXPress Server Manager administration Web application.

To generate the Quark Publishing Platform Server JEE enterprise archive:

1. Copy the Quark Publishing Platform EAR build to the computer on which the application server is running.
2. Update the configuration files (including "Database.properties", "managerconfig.xml", "PluginContext.xml", and "QLA.properties.xml") as described in "*Quark Publishing Platform Server — Manual configuration* ."
3. Update the values for the JMX Agent **username** and **password** in "serverApp.properties". The values for **username** and **password** will be the same used to Log on to WebSphere's admin console.
4. Run `RunEARUpdater.bat`. A "For Deployment" folder is created, containing a deployable Quark Publishing Platform EAR file.

Deploying in IBM WebSphere

To deploy in IBM WebSphere (Supported version is 8.5):

1. Copy the "qpp" folder from "For Deployment" to the working directory of the application server instance where the EAR file will be deployed. For IBM WebSphere, this is typically the folder corresponding to the target profile.

➡ To deliver resources (Quark Publishing Platform or otherwise) to IBM FileNet, add the following to the end of the `[working directory]\properties\wsjaas.conf` file:

```
FileNetP8WSI {
     com.filenet.api.util.WSILoginModule required;
};
```

2. Launch the IBM WebSphere application server console by navigating to `http://localhost:port/ibm/console`.
3. Under **Applications**, click **New Application**.
4. Click **New Enterprise Application**.
5. Under **Path to the new application**, click **Choose File** and then select the .ear file in the "For Deployment" folder. (For more information, see "*Preparing the EAR file for deployment*.")
6. Under **Preparing for the application installation**, click **Next**.
7. Click **Next** for steps 1, 2, 3, and 4. Leave all settings at their default.
8. Click **Finish** for step 5.
9. Click **Save**.
10. Navigate to **Applications** > **Application Types** > **WebSphere enterprise applications** and click **qpp-server-10.1**.
11. Click **Class loading and update detection**.
12. Under **General properties** > **Class loader order**, click **Classes loaded with local class loader first (parent last)**.
13. Click **Save** to save all changes.
14. Navigate to **Applications** > **Application Types** > **WebSphere enterprise applications** and click **qpp-server-10.1** again.
15. Under **References**, click **Shared library references**.
16. Check the box next to **qpp-server-10.1**, then click **Reference shared libraries**.
17. On the next screen, click **New**.
18. Select any name. As classpath, enter the relative paths of the Platform publishing and *ext* folders.

    For example, to target this directory:

    ```
    C:\ProgramFiles\IBM\WebSphere\AppServer\profiles\AppSrv01\qpp\publishing
    ```

    ...you would enter `./qpp/publishing` and `./qpp/ext`
19. Click **Apply** or **OK** to save the settings.
20. Assign this library reference to the **qpp-server-10.1** application.
21. Start the **qpp-server-10.1** Web application.
22. To verify the installation, navigate to the following URLs:

    ```
    http://localhost:9080/admin
    http://localhost:9080/workspace
    ```

Deploying in Oracle WebLogic

To deploy Quark Publishing Platform Server in Oracle WebLogic (supported version is 12.1):

1. Start the WebLogic server.
2. Create a new WebLogic domain named `QPP_DOMAIN`.
3. To set the JVM parameter for specifying the location of the "krb5.conf" file, open the file `[WEB_LOGIC _DIR]\domains\base_domain\bin\setDomainEnv.cmd` and append the following to the end:

   ```
   set JAVA_OPTIONS=%JAVA_OPTIONS%
   -Djava.security.krb5.conf=./qpp/conf/krb5.conf
   ```

4. Copy the "qpp" folder from the "For Deployment" folder to the target WebLogic server domain directory (for example, `C:\oracle\Middleware\user_projects\domains\QPP_DOMAIN`).
5. Set the global system environment variable `CLASSPATH` on the WebLogic computer to the following value:

   ```
   [WEB_LOGIC_DIR]\domains\QPP_DOMAIN\qpp\publishing
   ```

   and

   ```
    [WEB_LOGIC_DIR]\domains\QPP_DOMAIN\qpp\ext
   ```

6. Restart WebLogic.
7. Open the Oracle Weblogic application server console using the following URL:

   ```
   http://localhost:port/console
   ```

8. Select the domain (`QPP_DOMAIN`), then click the **Web Applications** tab.
9. Check **Archived Real Path Enabled**.
10. Click **Save**.
11. Under **Domain Structure**, click **Deployments**.
12. Under **Deployments** on the right, click **Install** and supply the path of the .ear file in the "For Deployment" folder. (For more information, see "*Preparing the EAR file for deployment*.")
13. In the next screen, click **Next** with the default option (**Install this deployment as an application**) selected.
14. In the next screen, click **Next**.
15. Click **Finish**.
16. Click **Save**.
17. Start the Quark Publishing Platform Web application.
18. To verify the installation, navigate to the following URLs:

    ```
    http://localhost:7001/admin
    http://localhost:7001/workspace
    ```

Deploying in external Tomcat

Developed by the Apache Software Foundation, Apache Tomcat™ serves as the standard reference implementation for Java™ Servlet and JavaServer™ Pages technologies. *Tomcat*™ is a servlet container for managing Web applications.

When you install the standalone version of Quark Publishing Platform Server, the installation embeds an instance of Tomcat in the Quark Publishing Platform Server Java Virtual Machine (JVM™) to manage Quark Publishing Platform Web applications, such as Quark Publishing Platform Web Client.

However, if you are already running a Tomcat server for other Web applications, and you want the Quark Publishing Platform Web applications to use your existing Tomcat server, you can deploy Quark Publishing Platform Server in this instance of Tomcat. Deploying Quark Publishing Platform Server in your existing (that is, external) Tomcat server means you don't have to run a separate Quark Publishing Platform Server process on the server computer. If you want to deploy Quark Publishing Platform Server in your external Tomcat, you can use a separate build located in "Server (External Web Container)" on your DVD.

Requirements and setup

Quark Publishing Platform Server requires JVM 1.5.x or 1.6.x and Apache Tomcat 7.0.41 to be configured as a Web server. The computer on which Tomcat is running should be a 64-bit computer. You can add Quark Publishing Platform Server to an existing Tomcat installation.

Setup Tomcat running on Windows

To deploy Quark Publishing Platform in an external Tomcat installation created by the Tomcat installer or deployed with binaries: (The existing Apache Tomcat installation folder in the steps below is `TOMCAT_HOME`)

1. Copy the `[QPP_BUILD]/qpp` folder to the `[TOMCAT_HOME]` folder.
2. Copy the contents of the `[QPP_BUILD]/webapps` folder to the `[TOMCAT_HOME]/webapps` folder.
3. If a `[TOMCAT_HOME]/endorsed` folder does not exist, create this folder.
4. Copy the contents of the `[QPP_BUILD]/endorsed` folder to the `[TOMCAT_HOME]/endorsed` folder.
5. If the Tomcat is installed with the installer: In the `[TOMCAT_HOME]/qpp/publishing/AS-Busdoc.xslt`, `[TOMCAT_HOME]/qpp/publishing/BusDoc2QCD.xslt` and `[TOMCAT_HOME]/qpp/publishing/BusDoc2QXPS.xslt` files, update the following paths as given below:

```
<xsl:include href="./qpp/publishing/xref-dita-anchors.xslt"/>
<xsl:include href="./qpp/publishing/AS-StyleSheets.xslt"/>
<xsl:include href="./qpp/publishing/AS-Transformations.xslt"/>
<xsl:include href="./qpp/publishing/BusDocsWordTableStyles.xslt"/>
```

6. If the Tomcat is deployed with binaries: In the `[TOMCAT_HOME]/qpp/publishing/AS-Busdoc.xslt`, `[TOMCAT_HOME]/qpp/publishing/BusDoc2QCD.xslt` and `[TOMCAT_HOME]/qpp/BusDoc2QXPS.xslt` files, update the following paths:

```
<xsl:include href="../qpp/publishing/xref-dita-anchors.xslt"/>
<xsl:include href="../qpp/publishing/AS-StyleSheets.xslt"/>
<xsl:include href="../qpp/publishing/AS-Transformations.xslt"/>
<xsl:include href="../qpp/publishing/BusDocsWordTableStyles.xslt"/>
```

7 In the `[TOMCAT_HOME]/qpp/conf/ServerApp.properties` file:

- Enter the `webServer.port` value you configured for Tomcat (8080 for example).
- Set the `webServer.embeddedWebContainer` value to **false**

8 In the `[TOMCAT_HOME]/qpp/conf/ManagerConfig.xml` file:

- Enter the **IP address** or **hostname** of the QuarkXPress Server in the `name` element of the `connectioninfo` section.
- Enter the port for QuarkXPress Server in the `port` element .

➡ Keep in mind that your Tomcat and the QuarkXPress Server should not be running on the same port (i.e.8080) if QuarkXPress Server and Tomcat are running on the same machine.

9 In the `[TOMCAT_HOME]/qpp/conf/Qla.properties` file:

- set the **host name**, **port number**, and **serial number** of your instance of the QLA Server.

10 To configure the database, in the `[TOMCAT_HOME]/qpp/conf/Database.properties` file edit/add the following values:

- For Oracle:

```
qpp.jdbc.driverClassName = oracle.jdbc.driver.OracleDriver
qpp.jdbc.url = jdbc:oracle:thin:@<hostname><:portnumber><:oracle_sid>
qpp.jdbc.userName = QppOracleDB
qpp.jdbc.password = QppPassword
```

- Fro SQL Server:

```
qpp.jdbc.driverClassName = com.microsoft.sqlserver.jdbc.SQLServerDriver
qpp.jdbc.url
=jdbc:sqlserver://<your-host-name>\\<instanceName>;databaseName=qppdb
qpp.jdbc.userName = QppMSSQLDB
qpp.jdbc.password = QppPassword
```

11 Edit the `[TOMCAT_HOME]/qpp/conf/PluginsContext.xml` file by replacing the default entry of the HSQL Dao context with the required database Dao context:

- For Oracle:

```
<import
resource="classpath:com/quark/qpp/common/dao/rdbms/oracle/OracleDaoContext.xml"/>
```

- Fro SQL Server:

```
<import
resource="classpath:com/quark/qpp/common/dao/rdbms/sqlserver/SqlServerDaoContext.xml"/>
```

12 Set the following global environment variables on the machine:

- MAGICK_HOME : `[Tomcat_Home]\qpp\ImageMagick\bin`.
- MAGICK_FILTER_MODULE_PATH : `%MAGICK_HOME%/modules/filters`.
- MAGICK_CODER_MODULE_PATH : `%MAGICK_HOME%/modules/coders`.

13 Open the `[TOMCAT_HOME]/conf/catalina.properties` file and add the following line:

```
Shared.Loader
 :${catalina.home}/qpp/conf,${catalina.home}/qpp/lib/*.jar,
${catalina.home}/qpp/publishing, ${catalina.base}/qpp/ext
```

14 Open the `[TOMCAT_HOME]/conf/catalina.properties` file and search for `org.apache.catalina.startup.TldConfig.jarsToSkip=` and update:

```
org.apache.catalina.startup.TldConfig.jarsToSkip=a*.jar,b*.jar,c*.jar,d*.jar,e*.jar,f*.jar,g*.jar,
h*.jar,i*.jar,k*.jar,l*.jar,m*.jar,n*.jar,o*.jar,p*.jar,q*.jar,r*.jar,s*.jar,t*.jar,u*.jar,v*.jar,w*.jar,x*.
jar,y*.jar,z*.jar,ja*.jar,jc*.jar,jd*.jar,je*.jar,ji*.jar,jn*.jar
```

15 ***If the Tomcat is installed with the installer*** :

Launch the Tomcat monitor. In the **JAVA** tab, set `CATALINA_OPTS` under Java Options:

- If Platform Server is running with Oracle database: `Doracle.jdbc.J2EE13Compliant=true`
- For integration of LDAP (or give absolute path): `Djava.security.krb5.conf=./qpp/conf/krb5.conf`
- For Java PermGen memory space: `XX:MaxPermSize=256m`

16 ***If the Tomcat is deployed with binaries*** :

Open the `[TOMCAT_HOME]/bin/catalina.bat` file and set following params:

- `JRE_HOME=C:\Program Files\Java\jdk1.6.0_12`
- `JAVA_OPTS=-server -Xmx1536m -XX:MaxPermSize=128m`
- `CATALINA_OPTS=-Doracle.jdbc.J2EE13Compliant=true -Djava.security.krb5.conf=../qpp/conf/krb5.conf`

17 Launch the Tomcat monitor. In the **JAVA** tab, set the following parameters:

- `Initial memory pool=1024 MB`.
- `Maximum memory pool=1536 MB`.

Server startup and verification

After you install Quark Publishing Platform Server with Tomcat and specify the port for Quark Publishing Platform Server access, you can start Quark Publishing Platform Server and verify your configuration. Quark Publishing Platform Server and Tomcat are tied together. To start and stop Quark Publishing Platform Server, you need to start and stop Tomcat.

To confirm Quark Publishing Platform Web Client access, in you browser's address field enter: `http://[machineName]:[webServerPort]/workspace`

To confirm Quark Publishing Platform Web admin access, in you browser's address field enter: `http://[machineName]:[webServerPort]/admin`

QuarkCopyDesk, QuarkXPress, Quark Publishing Platform Client, Quark Publishing Platform Web Client, and Quark Publishing Platform Script Manager use the Tomcat port to log on to Quark Publishing Platform server.

Multi-server deployment

A Quark Publishing Platform multi-server cluster is a set of Quark Publishing Platform Server installations that has been configured to use a common database, shared asset repository, and message queue. A multi-server cluster offers the following advantages:

- It lets you serve a larger number of requests by adding hardware.
- It lets you add active load balancing.
- It provides redundant and reliable setup.
- Failure of any specific instance in a cluster does not result in complete service outage. Only a fraction of active sessions are affected by an instance failure, and subsequent requests can be routed to an available active Quark Publishing Platform Server instance.
- It is transparent to clients that communicate with the load-balancing HTTP server.

To load-balance requests, you must have an HTTP server. Any HTTP load balancer that supports the "sticky session" feature. The following HTTP servers have been tested as load-balancing servers:

- Microsoft IIS 7 with the latest version of the IIS Tomcat connector
- Apache 2.2 with the latest version of the JK Tomcat Connector for Apache 2.2
- Apache 2.2 with built in mod_proxy and mod_proxy_balancer DSO modules of the Apache 2.2.

➡ Only requests routed through a HTTP load balancer are load-balanced among different instances. Requests to specific instance are always served by that instance, irrespective of its existing load.

➡ RMI clients must connect directly to a specific server instance.

Configuring cluster server instances

The Quark Publishing Platform Server can be deployed as a standalone server or in external tomcat using the database on the initial computers. The first step in setting up cluster servers is to set up a Quark Publishing Platform Sever database for Oracle or MS SQL Server (HSQLDB not supported for multi-server deployment).

Platform Server deployed as a standalone server

On each Quark Publishing Platform Server computer:

1 In the "ActiveMQ.xml" file, add network connectors to integrate the server instance's message queue with one or more server instances in the cluster, like so (where `qpp-node-N` is the name of a computer where another instance of Quark Publishing Platform Server is deployed):

```
<networkConnectors>
  <networkConnector dynamicOnly="true" duplex="true" name="remote-nc"
  networkTTL="1000" uri="static:(tcp://qpp-node-N:61401)"/>
</networkConnectors>
```

2 Uncomment the transport connector(s) in the "ActiveMQ.xml" file to integrate the server instances' message queues with the IP address and name of the server instances in the cluster.

```
<<transportConnector
uri="tcp://localhost:${jms.openWirePort}?wireFormat.maxInactivityDuration=0"
 updateClusterClients="true" rebalanceClusterClients="true"
updateClusterClientsOnRemove="true" />
<transportConnector
uri="tcp://10.91.32.51:${jms.openWirePort}?wireFormat.maxInactivityDuration=0"/>
```

3 In the "ServerApp.properties" file, assign a unique value to `server.id.prefix`, identifying the server instance.

4 In the "server.xml" file, assign a unique value to the `jvmRoute` attribute of the `<Engine>` tag.

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="node1">
```

5 Edit the folder name "serviceArchivesMultiServer" (located at `QPP Server home/webapps/webServices/Web-INF`) to "services".

6 Repeat the above steps for all Quark Publishing Platform nodes.

Platform Server deployed in external Tomcat

On each Quark Publishing Platform Server computer:

1 Deploy a standalone instance of Quark Publishing Platform Server in external tomcat, using the database on the initial computer.

2 In the "ActiveMQ.xml" file, add network connectors to integrate the server instance's message queue with one or more server instances in the cluster, like so (where `qpp-node-N` is the name of a computer where another instance of Quark Publishing Platform Server is deployed):

```
<networkConnectors>
  <networkConnector dynamicOnly="true" duplex="true" name="remote-nc"
  networkTTL="1000" uri="static:(tcp://qpp-node-N:61401)"/>
</networkConnectors>
```

3 Uncomment the transport connector(s) in the "ActiveMQ.xml" file to integrate the server instances' message queues with the IP address and name of the server instances in the cluster.

```
<<transportConnector
uri="tcp://localhost:${jms.openWirePort}?wireFormat.maxInactivityDuration=0"
 updateClusterClients="true" rebalanceClusterClients="true"
updateClusterClientsOnRemove="true" />
<transportConnector
uri="tcp://10.91.32.51:${jms.openWirePort}?wireFormat.maxInactivityDuration=0"/>
```

4 In the "ServerApp.properties" file, assign a unique value to `server.id.prefix`, identifying the server instance.

5 In the "server.xml" file, assign a unique value to the `jvmRoute` attribute of the `<Engine>` tag.

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="node1">
```

6 Edit the `tomcat_home/conf/context.xml` file as follows:

```
<Context sessionCookiePath="/">
```

7 Edit the folder name "serviceArchivesMultiServer" (located at `tomcat_home/webApps/webServices/WEB-INF`) to "services".

8 Repeat the above steps for all Quark Publishing Platform nodes.

Configuring IIS as an HTTP load balancer

To configure IIS 7 as an HTTP load balancer:

1 The first step is to deploy the Tomcat connector. (For more details on how to do this, see *http://tomcat.apache.org/connectors-doc/webserver_howto/iis.html*.) Create a new folder named "TomcatConnector," then put the "isapi_redirect.dll" file from the Tomcat Connector for IIS build.

2 Create a new file named "isapi_redirect.properties" in the same folder, and insert the following into the file:

```
extension_uri=/jakarta/isapi_redirect.dll
log_file=C:\TomcatConnector\Log\isapi.log
# Make sure the directory listed for log files in the
# "isapi_redirect.properties" file exists
log_level=info
worker_file=C:\TomcatConnector\worker.properties
worker_mount_file=C:\TomcatConnector\uriworker.properties
```

3 Create a new file named "workers.properties" in the same folder, and insert the following into the file, where `qpp-node1` and `qpp-node2` are the values of the `jvmRoute` attribute as specified in the "server.xml" file for each Quark Publishing Platform Server:

```
worker.list=TomcatBalancer
worker.TomcatBalancer.type=lb
worker.TomcatBalancer.balance_workers=qpp-node1,qpp-node2
worker.TomcatBalancer.sticky_session=True
worker.TomcatBalancer.sticky_session_force=True
worker.TomcatBalancer.method=Request
worker.TomcatBalancer.lock=Pessimistic

worker.qpp-node1.type=ajp13
worker.qpp-node1.host=qpp-node1

# This should be the same as the port in the AJP connector defined
# in the "server.xml" file for the Tomcat server in which Quark
# Publishing Platflrm is deployed
worker.qpp-node1.port=61398

worker.qpp-node1.lbfactor=3

worker.qpp-node2.type=ajp13
worker.qpp-node2.host=qpp-node2

# This should be the same as the port in the AJP connector defined
# in the "server.xml" file for the Tomcat server in which Quark
# Publishing Platflrm is deployed
worker.qpp-node2.port=61398

worker.qpp-node2.lbfactor=3
```

4 Create a new file named "uriworker.properties" in the same folder, and insert the following into the file:

```
/admin/*=TomcatBalancer
/admin=TomcatBalancer
/workspace/*=TomcatBalancer
/workspace=TomcatBalancer
/webservices/*=TomcatBalancer
/webservices=TomcatBalancer
/rest/*=TomcatBalancer
/rest=TomcatBalancer
/messaging/*=TomcatBalancer
/messaging=TomcatBalancer
/qxpsmadmin/*=TomcatBalancer
/qxpsmadmin=TomcatBalancer
```

```
/favicon.ico=TomcatBalancer
/*=TomcatBalancer
```

➡ This file contains the mappings of the URLs that need to be handled by IIS and forwarded to Quark Publishing Platform Servers.

5. Using the IIS management console, add a new virtual directory named "jakarta" to your IIS Web site, with the physical path of the directory where you placed the "isapi_redirect.dll" file.
6. Give the virtual directory execute permission. Select the virtual folder, double-click **Handler Mappings**, and then click **Edit Feature Permissions** in the **Actions** pane. Check **Execute** in the **Edit Feature Permissions** dialog box, and then click **OK**.
7. Add an ISAPI filter to the IIS Web site. Select the Web site, double-click **ISAPI Filters**, and then click **Add** in the **Actions** pane. In the **Add ISAPI Filter** dialog box, enter a name and the path of the "isapi_redirect.dll" file, and then click **OK**
8. Configure the ISAPI and CGI Restrictions feature. Navigate to the Server Home screen, double-click **ISAPI and CGI Restrictions**, and then click **Add** in the **Actions** pane. In the **Add ISAPI or CGI Restriction** dialog box, enter a name and the path of the "isapi_redirect.dll" file, check **Allow extension path to execute**, and then click **OK**.
9. Enable Windows authentication. Navigate to your Web site and double-click **Authentication**. Right-click **Windows Authentication** and choose **Enable**, disable the other authentication options, and then restart IIS.
10. To verify the installations, start Quark Publishing Platform Server on all nodes, then access each server using the computer name or IP address and port of the IIS server. The IIS server should serve and load-balance the request using the Tomcat connector. You can view the details in the log file identified in the "isapi_redirect.properties" file.

Configuring Apache Web Server as an HTTP load balancer

To configure Apache Web Server 2.2 with the latest version of the JK Tomcat Connector for Apache 2.2 as an HTTP load balancer:

1. The first step is to deploy the Tomcat connector. (For more details on how to do this, see *http://tomcat.apache.org/connectors-doc/miscellaneous/faq.html*.) Copy the "mod_jk.so" file from the Tomcat Connector download location to `{Apache_2.2_installation}/modules`.
2. Open the file `{Apache_2.2_installation}/conf/httpd.conf` in a text editor and insert the following content:

```
LoadModule jk_module modules/mod_jk.so

<IfModule jk_module>
  JkWorkersFile conf/workers.properties
     JkLogFile logs/mod_jk.log
 JkLogStampFormat "[%H:%M:%S] "
 JkRequestLogFormat "%T"
 JkLogLevel error
     JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories
  <Directory />
  AllowOverride All
  <Limit GET HEAD POST PUT DELETE OPTIONS>
      Order Allow,Deny
      Allow from all
  </Limit>
 </Directory>
 JkMount /workspace TomcatBalancer
 JkMount /workspace/* TomcatBalancer
 JkMount /webservices TomcatBalancer
 JkMount /webservices/* TomcatBalancer
 JkMount /admin TomcatBalancer
 JkMount /admin/* TomcatBalancer
 JkMount /rest TomcatBalancer
 JkMount /rest/* TomcatBalancer
 JkMount /messaging TomcatBalancer
 JkMount /messaging/* TomcatBalancer
 JkMount /qxpsm TomcatBalancer
 JkMount /qxpsm/* TomcatBalancer
 JkMount /qxpsmadmin TomcatBalancer
 JkMount /qxpsmadmin/* TomcatBalancer
 </IfModule>
```

3 Create a file named `{Apache_2.2_installation}/conf/workers.properties` and add the following content to it, where `qpp-node1` and `qpp-node2` are the values of the `jvmRoute` attribute as specified in the "server.xml" file for each Quark Publishing Platform Server:

```
worker.list=TomcatBalancer

worker.TomcatBalancer.type=lb
worker.TomcatBalancer.balance_workers=qpp-node1, qpp-node2
worker.TomcatBalancer.sticky_session=True

# This should be commented out, if it is not, you cannot connect to the
Apache Web server which
# acts as the load balancer.
worker.TomcatBalancer.sticky_session_force=True

worker.TomcatBalancer.method=Request
worker.TomcatBalancer.lock=Pessimistic

worker.qpp-node1.type=ajp13

# This should be the IPP address of the first QPP node
worker.qpp-node1.host=Server 1

# 8009 is for external tomcat server
# 61398 is for embedded tomcat with QPP
worker.qpp-node1.port=61398

worker.qpp-node1.lbfactor=3

worker.qpp-node2.type=ajp13

# This should be the IPP address of the first QPP node
worker.qpp-node2.host=Server 2

# 8009 is for external tomcat server
# 61398 is for embedded tomcat with QPP
worker.qpp-node2.port=61398

worker.qpp-node2.lbfactor=3
```

4 Restart Apache 2.2 httpd.

5 To verify the installations, start Quark Publishing Platform Server on all nodes, then access each server using the computer name or IP address and port of the Apache server. The Apache server should serve and load-balance the request using the Tomcat connector. You can view the details in the log files at `{Apache_2.2_Installation}/logs`.

Restricting access to Web Client/Admin

To restrict the webapp so that the workspace and admin pages can only be accessed from within your own network, use a Tomcat Valve to restrict access based on an IP address range.

1 Open the "conf" folder in you Quark Publishing Platform Server folder.
2 Open "server.xml" in a text-editing application.
3 Edit the context elements corresponding to the admin and workspace applications to engage a `RemoteAddrValve`. The valve has to be configured to allow only intranet traffic. The final config would look something like the following:

```
 <Context path="/qxpsm" docBase="qxpsm">
  <Manager pathname="" />
</Context>
<Context path="/workspace"docBase="workspace">
  <Manager pathname="" />
  <Valve className="org.apache.catalina.valves.RemoteAddrValve" allow="
10\.91\..*\..* "/>
</Context>
<Context path="/webservices"docBase="webservices">
  <Manager pathname="" />
</Context>
<Context path="/messaging" docBase="messaging">
  <Manager pathname="" />
</Context>
<Context path="/admin"docBase="admin">
  <Manager pathname="" />
  <Valve className="org.apache.catalina.valves.RemoteAddrValve" allow="
10\.91\..*\..*  " />
</Context>
<Context path="/rest"docBase="rest">
  <Manager pathname="" />
</Context>
```

➡ The value of the `allow` attribute is a regex for matching IP addresses. Here the intranet IP's are assumed to be in 10.91.x.x range. You need to update this value based on your network IP ranges.

Enabling SSL for Quark Publishing Platform Clients

You can configure Quark Publishing Platform with different security options. In addition to your own network security specifications, you can specify Secure Sockets Layer (SSL) protocol for your Quark Publishing Platform client applications.

Secure Sockets Layer (SSL) support

You can configure Tomcat (and therefore all Quark Publishing Platform clients) to run in secure mode with Secure Sockets Layer (SSL) technology. This section explains the configuration process.

➡ It is also possible to run Quark Publishing Platform without embedding Tomcat in JVM. See "*Deploying in external Tomcat*" for information about setting up Quark Publishing Platform without embedding Tomcat.

To manage Web applications in the Quark Publishing Platform environment, Quark Publishing Platform Server embeds an instance of Apache Tomcat 7.29 in its JVM. The four Web applications in Quark Publishing Platform include Quark Publishing Platform Web Client, Quark Publishing Platform Console, Quark Publishing Platform Renderer Manager, and Quark Publishing Platform Web Services.

When you enable SSL, it applies to all Quark Publishing Platform client applications deployed in Quark Publishing Platform Server.

Enabling SSL

The instructions below address two scenarios. The "server.xml" file you edit contains XML tags for both scenarios, which you need to enable or disable by "commenting" and "uncommenting" specific tags.

To enable SSL for secure HTTP for all Quark Publishing Platform Web applications:

1 Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "server.xml" in a text-editing application.

3 Comment the following tag:

```
<Connector port="61400" maxHttpHeaderSize="8192"maxThreads="150"
minSpareThreads="25" maxSpareThreads="75"enableLookups="false"
redirectPort="61399" acceptCount="100"connectionTimeout="20000"
disableUploadTimeout="true" />
```

4 Uncomment the following tag:

```
<Connector port="61399" maxHttpHeaderSize="8192"MaxThreads="150"
minSpareThreads="25" maxSpareThreads="75"enableLookups="false"
disableUploadTimeout="true"acceptCount="100" scheme="https"
secure="true"clientAuth="false" sslProtocol="TLS" />
```

5 Replace `61399` with `61400` (or any port on which Tomcat will be listening for secure connections).

6 Save and close "server.xml."

7 Mac OS X QuarkXPress and QuarkCopyDesk users must fetch a SSL certificate from Quark Publishing Platform Server before they can log on. To do so, each user should launch Terminal and run the following command, substituting the IP address of the Quark Publishing Platform Server computer for `[server name]`:

```
echo | openssl s_client -connect [server name]:443 > [server name].pem
```

This command retrieves a copy of the server certificate named "[server name].pem". Put this file in the `~/Library/Application Support/Quark/QPP/Certificates` folder (or, if you have customized the plist file at `~/Library/Application Support/Quark/QPP/[QPP Framework Version]`, put the file there).

If the "Quark Publishing Platform" and "Certificates" folders do not already exist at the above locations, create them there manually.

Windows users do not require a server SSL certificate.

This change means Quark Publishing Platform client applications can use HTTPS. For example, the URL for a Quark Publishing Platform Web Client user would be as follows: `https://[server name]:61399/workspace`.

Enabling HTTP and HTTPS

To enable HTTP and HTTPS:

1 Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "server.xml" in a text-editing application.

3 Uncomment the following tag:

```
<Connector port="61399" maxHttpHeaderSize="8192"MaxThreads="150"
minSpareThreads="25" maxSpareThreads="75"enableLookups="false"
disableUploadTimeout="true"acceptCount="100" scheme="https"
secure="true"clientAuth="false" sslProtocol="TLS" />
```

4 Save and close "server.xml."

With this change, Quark Publishing Platform Web application users can access Quark Publishing Platform Server with HTTPS or HTTP. For example, a Quark Publishing Platform Web Client user could use either of the following URLs: `http://<machine-name>:61400/workspace` or `https://<machine-name>:61399/workspace`.

Verifying and using SSL

To verify and use SSL:

1. Start the Quark Publishing Platform Server.
2. Test Quark Publishing Platform Web Client access by entering the following: `https://[machine IP/name]:61399/workspace`.

Keystores and SSL certificates

A *certificate* is a file on a Web server that is used in encryption and confirmation between two endpoints to establish a secure connection. A *keystore* is essentially a database of digital certificates on the Web server.

You can obtain an SSL certificate from a trusted Certificate Authority (CA). Import the certificate into the keystore used by the Quark Publishing Platform Server's JVM.

For more information about the importance of keystores, use the following URL: *http://tomcat.apache.org/tomcat-6.0-doc/ssl-howto.html*.

Quark Publishing Platform Server — Manual configuration

You can change the default configuration after you install Quark Publishing Platform Server. In addition to setting parameters with JConsole while Quark Publishing Platform Server is running, you can adjust settings in different .xml files and .properties files. You can also adjust memory allocation for your JVM configuration.

Editing "ServerApp.properties"

To edit "ServerApp.properties":

1. Open the "ServerApp.properties" file in the `[QPP Server]/conf` folder.
2. Set the `rmi.port` value to the number of the port on which the RMI registry listens. Java-based RMI clients such as ScriptManager connect through this port.
3. Set the `rmi.servicePort` value to the port number used by the RMI server where Quark Publishing Platform service objects are registered.
4. Set the `namingservice.port` value to the port number used for listening to object resolution requests by the CORBA naming service.
5. Set the `serverORB.port` value to the port number used by the ORB where Quark Publishing Platform service objects are activated.
6. Set the `jms.openWirePort` value to the port number opened for JMS communication via the OpenWire protocol. Java clients such as Quark Publishing Platform Script Manfager connect to this port to listen to server notifications.
7. Set the `webServer.port` value to the port number on which Tomcat listens for HTTP connections. Quark Publishing Platform Web Client and SOAP clients connect through this port. The value should be set to that specified for the HTTP connector in the Tomcat "server.xml" file.
8. Set the `socketStreaming.port` value to the port number to be used for file transfer (upload/download).
9. To bind Quark Publishing Platform Server to a particular IP address, set the `server.machinename` value to that IP address and set `server.bindtoip=true`. Or, if you have multiple network cards and you want to bind Quark Publishing Platform to all of their IP addresses, set `server.machinename=localhost`,

`server.bindtoip=false`, and `server.addtionalnames=[non-default-ip1],[non-default-ip2]`.

10 Set the `server.additionalnames` value to specify the global IP address where the firewall is exposed.

11 Enter the `webServer.port` value you configured for Tomcat (`8080`, for example).

12 To allow users to log on to Quark Publishing Platform Server even when the directory server is not running, set `authentication.external.cacheTicket = true`.

Editing "PublishingPool.properties"

To edit "PublishingPool.properties":

1. Open the "PublishingPool.properties" file in the Server Installation `conf` folder.
2. Set the `publishingThread.pool.maxActive` value to specify the maximum number of background publishing threads that can run simultaneously.
3. Set the `publishingThread.pool.maxIdle` value to specify the maximum number of idle threads in the pool.
4. Set the `publishingThread.pool.minIdle` value to specify the minimum number of idle threads in the pool.
5. Set the `publishingThread.pool.maxWait` value to specify the time in milliseconds that the publishing request should wait while borrowing a thread from the pool.
6. Set the `publishingThread.pool.minEvictableIdleTimeMillis` value to specify the time in milliseconds for a thread to be in the pool before it can be evicted.
7. Set the `publishingThread.pool.timeBetweenEvictionRunsMillis` value to specify the time in milliseconds after which the evictor thread should run to remove idle threads.

Configuring Quark Publishing Platform Renderer for Quark Publishing Platform usage

To configure Quark Publishing Platform Renderer for Quark Publishing Platform usage:

1. Open the QXPSM Admin Client by navigating to the following URL in a Web browser: `http://[QPP server name]:[port]/qxpsmadmin`.
2. In the **Manage Servers** pane, click **Add Servers** and add the Quark Publishing Platform instance of Quark Publishing Platform Renderer. (For more information, see *A Guide to Quark Publishing Platform Renderer*.)

Editing "Qla.properties"

To edit "Qla.properties":

1. Open the "Qla.properties" file in the `[QPP Server]/conf` folder.

2. Enter the IP address or hostname of the QLA Server in the `QlaServer.machinename=` field.
3. Enter the port number of the QLA Server in the `QlaServer.port=` field.
4. If you have a backup QLA server, enter the IP address (or hostname) and port number in the `Backup.QlaServer.machinename=` and `Backup.QlaServer.port=` fields.
5. Enter the Quark Publishing Platform serial number in the `Qla.SerialNumber=` field.

➡ The QLA Server Console and QLA Client applications display your Quark Publishing Platform serial number.

6. Save and close "Qla.properties."

Extended configuration files

Configuration files are split into *base* and *ext* for ease of deployment.

User specific custom **bean's**, **processes** and **publishingchannels** should be defined in *.ext* files in the Server Installation `ext` folder.

- ChannelConfig-ext.xml
- content-mimetype-mappings-ext.xml
- custom-xml-types-ext.xml
- IndexingChannels-ext.xml
- PluginsContext-ext.xml
- ProcessConfig-ext.xml
- PublishingConfig-ext.xml

Configuring Platform Server to be used for publishing with Quark XML Author - Sharepoint Adapter

To configure Quark Publishing Platform Server to be used for publishing with Quark XML Author - Sharepoint Adapter:

1. Open the "sharepoint.properties" file in the `[QPP Server]/publishing` folder.
2. Set the `sharepoint.username` value to be the login name of a user with access to Sharepoint sites.
3. Set the `sharepoint.userpassword` value to be the login password of the user specified above.
4. Set the `sharepoint.userdomain` value to be the domain of the user specified above.
5. Set the `sharepoint.sitecollection` value to be the URL of a Sharepoint Site Collection which contains documents required during publishing.
6. Restart the Server.

Configuring Platform Server to be used for publishing with Quark XML Author - FileNet Adapter

To configure Quark Publishing Platform Server to be used for publishing with Quark XML Author - FileNet Adapter:

1. Open the "contentengine.properties" file in the `[QPP Server]/publishing` folder.
2. Set the `filenet.stanza` value to be the stanza for filenet content engine connection. You need to edit the parameter only when the filenet server is configured to use a stanza other than FileNet.
3. Set the `filenet.username` value to be the FileNet username.
4. Set the `filenet.userpassword` value to be the value to be the FileNet password of the user specified above.
5. Set the `filenet.connectionuri` value to be the Connection URI of the FileNet Content Engine WebService.
6. Restart the Server.
7.

JVM memory allocation on Windows

On Windows, you can specify JVM memory allocation in different locations, depending on how you start Quark Publishing Platform Server. On 64-bit operating systems, you can allocate more memory for the Quark Publishing Platform Java process. In either case, you should not allocate more than 50 percent of available memory.

If you use Quark Publishing Platform Server Console or Quark Publishing Platform Server Windows service

1. Stop Quark Publishing Platform Server.
2. If you start Quark Publishing Platform Server with Quark Publishing Platform Server Console or Quark Publishing Platform Server Windows service, open the "wrapper.conf" file.
3. Search for the `wrapper.java.maxmemory` property.
4. Adjust the value. On a 64-bit operating system, you can make the value larger.
5. Save your changes and restart Quark Publishing Platform Server.

If you use "Serverstartup.bat"

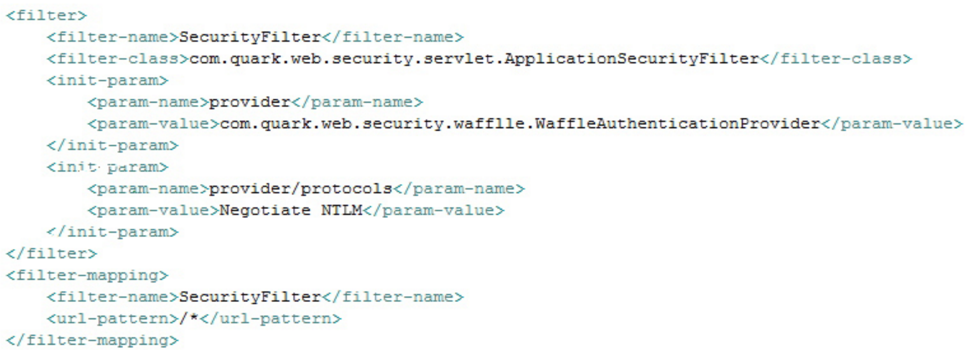
1. Stop Quark Publishing Platform Server.
2. If you start Quark Publishing Platform Server with the "ServerStartup.bat" file in the Quark Publishing Platform Server installation folder, open "ServerStartup.bat" in a text-editing application.
3. Search for `java -server -Xmx1536m -XX:MaxPermSize=256m -classpath`. `1536m` represents 1536MB of RAM allocated to Quark Publishing Platform Server.

4. On a 64-bit operating system, you can make the value larger.
5. Save your changes and restart Quark Publishing Platform Server.

Configuring Windows authentication

Windows users can log on to Quark Publishing Platform transparently with their Windows user credentials, without ever having to see a login dialog box. Quark Publishing Platform supports all Windows authentication schemes, including NTLM-v1/NTLM-v2 and Negotiate/Kerberos. The Platform also supports mixed-mode, so Windows authentication and Platform authentication can coexist.

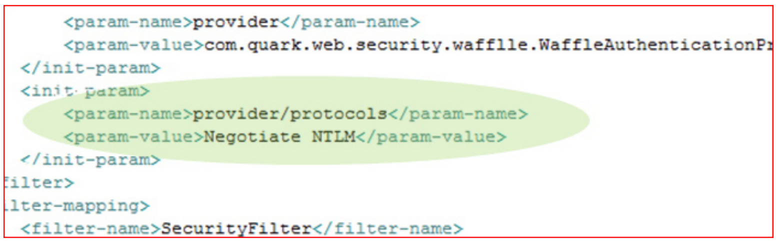
You configure Windows authentication with a pluggable HTTP servlet filter. You can easily enable and disable authentication at deployment time by adding a security filter to each Web application's "web.xml" file.



```
<filter>
    <filter-name>SecurityFilter</filter-name>
    <filter-class>com.quark.web.security.servlet.ApplicationSecurityFilter</filter-class>
    <init-param>
        <param-name>provider</param-name>
        <param-value>com.quark.web.security.waffle.WaffleAuthenticationProvider</param-value>
    </init-param>
    <init-param>
        <param-name>provider/protocols</param-name>
        <param-value>Negotiate NTLM</param-value>
    </init-param>
</filter>
<filter-mapping>
    <filter-name>SecurityFilter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

Each web application of Quark Publishing Platform can be configured to use different Windows authentication schemes as indicated below.

Web application	NTLM only	Negotiate/Kerberos only	NTLM and Kerberos
Workspace	NTLM	Negotiate	NTLM Negotiate
Admin	NTLM	Negotiate	NTLM Negotiate
Web services	NTLM	Kerberos	NTLM Negotiate
QXPSM	N/A	N/A	N/A
Messaging	N/A	N/A	N/A
REST	N/A	N/A	N/A



```
        <param-name>provider</param-name>
        <param-value>com.quark.web.security.waffle.WaffleAuthenticationPr
    </init-param>
    <init-param>
        <param-name>provider/protocols</param-name>
        <param-value>Negotiate NTLM</param-value>
    </init-param>
ilter>
lter-mapping>
    <filter-name>SecurityFilter</filter-name>
```

Steps to enable SSO in Platform Clients

For the Admin web app:

1. Go to `C:\Program Files\Quark\Quark Publishing Platform\Server\webapps\admin\WEB-INF`
2. Open the Web.xml and search for the following snippet:

```
</p><p>
  <filter></p><p>
    <filter-name>SecurityFilter</filter-name></p><p>

<filter-class>com.quark.web.security.servlet.ApplicationSecurityFilter</filter-class></p><p>

    <init-param></p><p>
      <param-name>provider</param-name></p><p>

<param-value>com.quark.web.security.waffle.WaffleAuthenticationProvider</param-value></p><p>

    </init-param></p><p>
    <init-param></p><p>
      <param-name>provider/protocols</param-name></p><p>
      <param-value>NTLM</param-value></p><p>
    </init-param></p><p>
  </filter></p><p>
  <filter-mapping></p><p>
    <filter-name>SecurityFilter</filter-name></p><p>
    <url-pattern>/*</url-pattern></p><p>
  </filter-mapping>
</p><p>
```

3. Uncomment the above snippet and save the file.

For Workspace and Desktop clients go to the respective web app, open the web.xml file and uncomment the same snippet.

- For desktop clients: `C:\Program Files\Quark\Quark Publishing Platform\Server\webapps\webservices\WEB-INF`
- For workspace: `C:\Program Files\Quark\Quark Publishing Platform\Server\webapps\workspace\WEB-INF`

➡ The Platform server should be running as a service with an account as Local System.

Steps to configure Web Client for mixed mode authentication

1. Go to `C:\Program Files\Quark\Quark Publishing Platform\Server\webapps\workspace`. Make a copy of PreLogin.jsp and name it local.jsp.
2. Go to `C:\Program Files\Quark\Quark Publishing Platform\Server\webapps\workspace\WEB-INF`
3. Open the Web.xml and search for the following snippet:

```
< filter >
   < filter-name > SecurityFilter </ filter-name >
   < filter-class > com.quark.web.security.servlet.ApplicationSecurityFilter
 </ filter-class >
   < init-param >
     < param-name > provider </ param-name >
     < param-value >
com.quark.web.security.waffle.WaffleAuthenticationProvider </ param-value
 >
   </ init-param >
   < init-param >
     < param-name > provider/protocols </ param-name >
     < param-value > NTLM </ param-value >
   </ init-param >
   <init-param>
     <param-name>exclude-url-patterns</param-name>
     <param-value>/local.jsp,/Login.jsp</param-value>
   </init-param>
</ filter >
< filter-mapping >
  < filter-name > SecurityFilter </ filter-name >
  < url-pattern > /* </ url-pattern >
</ filter-mapping >
```

4 Uncomment the above snippet and save the file. The bolded lines from the above snippet need to be added.

5 Restart the server and access the corresponding web applications with following URLs:

- For access through native Platform user authentication use the URL : `http://localhost:61400/workspace/local.jsp`
- For access through Windows authentication use the URL: `http://localhost:61400/workspace/`

➡ The steps above can be repeated for Wed Admin or other web apps as well.

Configuring logging levels

You can edit the "log4j.xml" file to adjust logging levels, and you can use JConsole to change logging levels after starting Quark Publishing Platform Server. You can also set different logging levels for exceptions.

Changing logging levels in "log4j.xml"

You can change the logging levels for Quark Publishing Platform Web Client and Quark Publishing Platform Server. Options include `ERROR`, `INFO`, `WARN`, `DEBUG`, `SQLTRACE`, and `TRACE`.

- `ERROR` = includes messages that indicate disrupted and failed requests.
- `INFO` = includes messages that indicate the state of services.
- `WARN` = includes non-critical service error messages
- `DEBUG` = includes messages that indicate server resource usage.
- `SQL_TRACE` = includes messages according to activity related to SQL requests.
- `TRACE` = includes messages according to activity related to requests.

Refer to Java documentation for more information about logging levels.

To change logging levels:

1 Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "log4j.xml" in a text-editing application.

3 To define the logging level for Quark Publishing Platform Web Client activity, scroll to `<logger name=com.quark.qpp.web.webeditor`. The structure is as follows:

```
<logger name="com.quark.qpp.web.webeditor" additivity="false"> <level
value="INFO" /> <appender-ref ref="WebHubAsyncAppender" /> </logger>
```

4. To define the logging level for Quark Publishing Platform Server activity, scroll to `<logger name=com.quark.qpp`. The structure is as follows:

   ```
   <logger name="com.quark.qpp"> <level value="INFO" /> </logger>
   ```

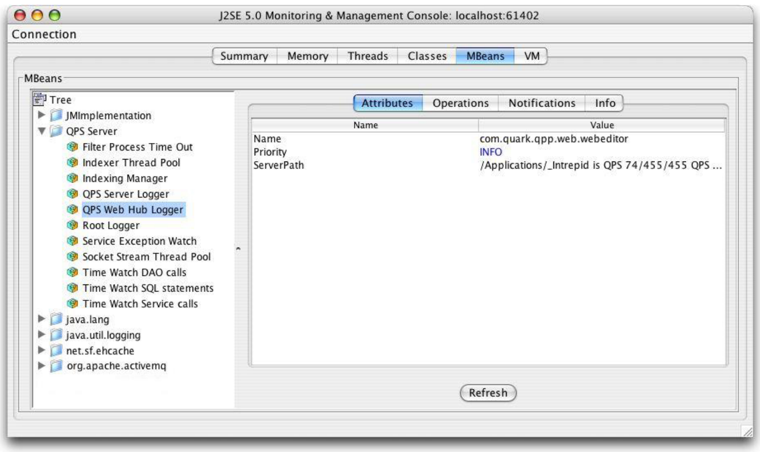
5. To define the logging level for other activity, scroll to the "<root>." The structure is as follows:

   ```
   <root> <priority value="ERROR" /> <appender-ref ref="QppServerAsyncAppender"
    /> </root>
   ```

6. Save and close "log4j.xml."

Changing logging levels after starting Quark Publishing Platform Server

1. With Quark Publishing Platform Server running, display the **Platform Server Console**.
2. Click **JConsole** to display a window for monitoring different aspects of Quark Publishing Platform Server performance.
3. Click the **MBeans** tab. Separate Quark Publishing Platform Server functions display in tree format on the left side of the **MBeans** tab.
4. Open **Platform Server** > **Logging** in the tree.
5. Click the **Operations** tab.



Name	Value
Name	com.quark.qpp.web.webeditor
Priority	INFO
ServerPath	/Applications/_Intrepid is QPS 74/455/455 QPS ...

Use JConsole to adjust logging priority levels

6. To edit the log level of Quark Publishing Platform Server or the Quark Publising Platform Web Client, set the logging level in the text box and click the corresponding button.

The changes you make in JConsole take effect immediately, but when you restart Quark Publishing Platform Server, the settings in "log4j.xml" are applied.

Changing logging for exceptions

You can set logging for known and unknown exceptions by editing two values in the "ServerApp.properties" file.

1. Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "ServerApp.properties" in a text-editing application.

3 If you don't want to log Quark Publishing Platform exceptions in the Quark Publishing Platform Server log, set `server.logqppserviceexception` to `false`.

4 If you want to avoid logging unknown exceptions, set `server.logthrowable` to `false`.

5 Save and close "ServerApp.properties."

Modifying search notification evaluation settings

Quark Publishing Platform Server notifies all open **Search Results** palettes if the assets displayed in the palettes have been modified. You can edit several parameters to influence the strategy and resources used to evaluate conditions and deliver these query notifications. The optimal value for most parameters depends on the database and hardware you choose for your Quark Publishing Platform Server.

➡ Only an experienced administrator should change the settings described below. Please consult Quark Enterprise Support for assistance.

1 Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "Query.properties" in a text-editing application.

3 You can optimize performance with an HSQL database by directing Quark Publishing Platform Server to store asset metadata in a temporary table before delivering query notifications. To optimize search performance with an HSQL database, scroll to the `query.notification.useTempTable=` parameter and set the value to `true`.

➡ If you do not use an HSQL database, set the value to `false`. By default, the Quark Publishing Platform Server Installer sets this parameter correctly for your chosen Quark Publishing Platform Database option.

4 Quark Publishing Platform Server uses two pools of threads to evaluate query notifications — the "Generic Notification Evaluator Thread Pool Configuration" and the "Simple Notification Evaluator Thread Pool Configuration." The "generic" thread pool evaluates query notifications using a database. The "simple" thread pool uses a simpler strategy to evaluate query notification without involving a database. Adjust the following properties in the "Generic Notification Evaluator Thread Pool Configuration" area:

- To specify the maximum number of concurrent threads that operate in the background to evaluate notifications that require database access, adjust the `query.notification.generic.pool.maxActive` value. Increase this value to improve performance when using hardware with several processors and extensive system memory.
- To specify the maximum number of idle threads in the pool, adjust the `query.notification.generic.pool.maxIdle` property.

- To specify the minimum number of idle threads in the pool, adjust the `query.notification.generic.pool.minIdle` property.
- To specify the minimum time that a thread can be idle before it can be removed from the pool, set the time in milliseconds for the `query.notification.generic.pool.minEvictableIdleTimeMillis` property.
- To specify the number of milliseconds after which a background evictor thread should run to clean up idle threads, adjust the `query.notification.generic.pool.timeBetweenEvictionRunsMillis` property.

5. Query notifications for certain asset changes can be evaluated efficiently using a simpler strategy that does not require database access. Adjust the following properties in the "Simple Notification Evaluator Thread Pool Configuration" area:

- To specify the maximum number of concurrent threads that operate in the background, adjust the `query.notification.simple.pool.maxActive` value. Increase this value to improve performance when using hardware with several processors and extensive system memory.
- To specify the maximum number of idle threads in the pool, adjust the `query.notification.simple.pool.maxIdle` property.
- To specify the minimum number of idle threads in the pool, adjust the `query.notification.simple.pool.minIdle` property.
- To specify the minimum time that a thread can be idle before it can be removed from the pool, set the time in milliseconds for the `query.notification.simple.pool.minEvictableIdleTimeMillis` property.
- To specify the number of milliseconds after which a background evictor thread should run to cleanup idle threads, adjust the `query.notification.simple.pool.timeBetweenEvictionRunsMillis` property.

6. To configure JMS topology for query notification, adjust the `query.notification.topicPerSession` property. To emulate the behavior of versions of Quark Publishing Platform prior to 8.1, set this value to `false`.
7. Do not edit the values of other properties in the "Query.properties" configuration file.
8. Save and close "Query.properties."
9. Restart Quark Publishing Platform Server for these settings to take effect.

Database properties

You can manually specify the database connection URL, database user name, database user password, and database connection pool size.

To change database properties:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "Database.properties" in a text-editing application.
3. Scroll to the `Database related configuration` section.

4 To specify the driver class name, replace the `qpp.jdbc.driverClassName` value. For example, for an HSQL database, you would use `qpp.jdbc.driverClassName=org.hsqldb.jdbcDriver`.

5 To specify the database connection URL, replace the path after `qpp.jdbc.url`.

6 To specify the database user name, replace the `qpp.jdbc.userName` value.

7 To specify the database user password, replace the `qpp.jdbc.password` value.

8 To specify the database connection pool size, change the `qpp.jdbc.maxActive` value.

9 To specify the maximum and minimum numbers of connections that can be idle, change the `qpp.jdbc.maxIdle` and `qpp.jdbc.minIdle` values.

10 To specify the minimum time in milliseconds that a connection must be in the pool before it can be evicted, change the `qpp.jdbc.minEvictableIdleTimeMillis` value.

11 To specify the minimum time in milliseconds after which the evictor thread should run to remove an idle connection, change the `qpp.jdbc.timeBetweenEvictionRunsMillis` value.

12 To adjust the pooling of prepared statements, change the `qpp.jdbc.poolPreparedStatements` value.

13 To specify the maximum number of connections to be made in the no-autocommit pool , change the `qpp.jdbc.maxActive_noAutoCommit` value.

14 Save and close "Database.properties."

➡ In there is a firewall configured between the Platform Server and the Database server that can terminate TCP sessions when they are idle for a long time, the following configuration changes should be done in the "Database.properties" file .

- Set the `qpp.jdbc.minIdle` value to *0*
- Set the `qpp.jdbc.minEvictableIdleTimeMillis` value to *60000*
- Set the `qpp.jdbc.timeBetweenEvictionRunsMillis` value to *60000*

Platform server needs to be restarted for these settings to take effect.

Transformation properties

You can use transformation properties to modify the output of various predefined transformations. Changes to the "Transformation.properties" file alter the external commands that are invoked for transforming assets. Commands in this file can contain predefined macros such as `<temp>`, `<source>`, `<extension>` and `<output>`.

To change transformation properties:

1 Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "Transformation.properties" in a text-editing application.

3. To specify a pattern that does not occur as a value (or part of a value) for any other property, specify a value for `transformer.spaceEncodingPattern`. This pattern is used for escaping spaces in commands and other property values.
4. To specify which file types ImageMagick is allowed to handle, enter a comma-separated, case-insensitive list of file extensions in `imTransformer.extensions`.
5. To specify the temporary file location to be used by ImageMagickTransformer, specify a path for `imTransformer.tempDir`.
6. To specify the ImageMagick command to be executed for generating output with ImageMagickTransformer, enter a value for `imTransformer.jpg.imCommand`.
7. To specify which file types the JawsTransformer is allowed to handle, enter a comma-separated, case-insensitive list of file extensions in `jawsTransformer.extensions`.
8. To specify the temporary file location to be used by JawsTransformer, specify a path for `jawsTransformer.tempDir`.
9. To specify the Jaws command to be executed for generating output with JawsTransformer, enter a value for `jawsTransformer.jpg.command`.
10. Save and close "Transformation.properties."

Session timeout

You can specify how long individual sessions can remain idle before they are timed-out, as well as the frequency of testing for idle sessions.

To specify timeout parameters:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "ServerApp.properties" in a text-editing application.
3. Scroll to the `session.maxIdle=` entry.
4. Enter the number of seconds a session can be idle before timing out.
5. To specify the frequency for Quark Publishing Platform Server to perform a background check for idle sessions, enter a number (measured in seconds) in the `session.eviction.thread.delay` property.
6. Save and close "ServerApp.properties."

➡ Web Client timeout settings can be changed by adjusting the entry `ajaxTimeout` in the file "Workspace-Config.xm" (this entry is in milliseconds) . This file is at *<Quark Publishing Platform Server Install folder>webapps/workspace/WEB-INF/classes*

Repository status updater

You can specify a background thread that runs frequently to confirm the Quark Publishing Platform File Server status.

To specify the repository status update interval:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "ServerApp.properties" in a text-editing application.
3. Scroll to the `repository.status.updator.sleepInterval=` entry.
4. Enter a value to specify the number of seconds after which the repository status updater thread will run.
5. Save and close "ServerApp.properties."

Moving Quark Publishing Platform Renderer

If you need to move your Quark Publishing Platform Renderer to a different computer, you don't have to re-install Quark Publishing Platform Server. Instead, you can edit the "ManagerConfig.xml" file, as follows:

1. Open the "conf" folder in the Quark Publishing Platform Server folder.
2. Open the "ManagerConfig.xml" file.
3. Scroll to the `<connectioninfo>` section at the bottom of "ManagerConfig.xml."
4. Change the `name` entry value to the IP address or hostname of the new Quark Publishing Platform Renderer.
5. Change the `port` entry value to the port number you specified for the new Quark Publishing Platform Renderer.
6. Save the "ManagerConfig.xml" file and launch Quark Publishing Platform Server.
7. To verify the change, search through the "QppServer.log" file for the line `Successfully registered with QXPS`.

➡ Alternatively, you can open the Quark Publishing Platform Renderer Manager client with the URL `http://[server]:[port]/qxpsmadmin` and make these changes there.

➡ Unless you check **Realm verif. For Admin. Requests** and enter a user name and password in the **HTTP** tab of the QuarkXPress **Server Configuration** dialog box, you can leave the `<user>` and `<password>` entries blank. This also applies to the username and password fields in the "ServerApp.properties" file (`qxps.username` and `qxps.password`).

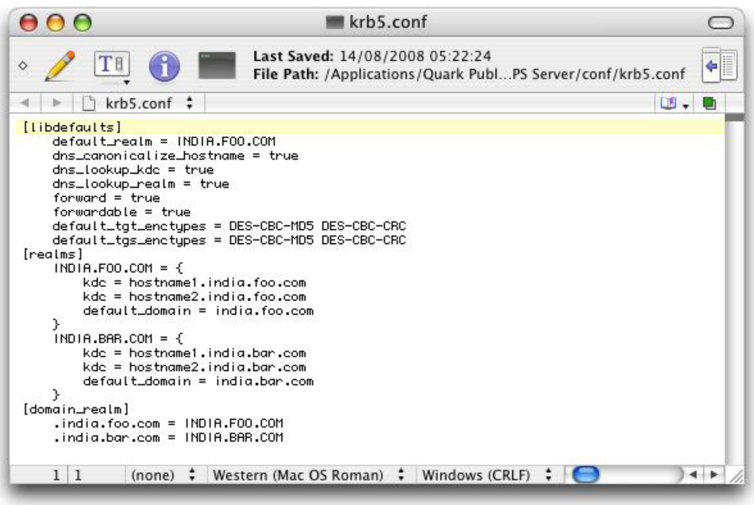
Integrating Quark Publishing Platform with LDAP

For information about integrating Quark Publishing Platform with LDAP, see "Managing User Lists with LDAP" in *A Guide to Quark Publishing Platform*.

Using Kerberos authentication

To use Kerberos authentication:

1 Open "krb5.conf" in a text-editing application. The "krb5.conf" file contains the Kerberos configuration information, which includes the locations of the Key Distribution Center (KDC) and admin servers for the Kerberos realms. Kerberos is a secure method for authenticating requests for a service over a computer network.


krb5.conf
Last Saved: 14/08/2008 05:22:24
File Path: /Applications/Quark Publ...PS Server/conf/krb5.conf
krb5.conf
1 1 (none) Western (Mac OS Roman) Windows (CRLF)

```
[libdefaults]
    default_realm = INDIA.FOO.COM
    dns_canonicalize_hostname = true
    dns_lookup_kdc = true
    dns_lookup_realm = true
    forward = true
    forwardable = true
    default_tgt_enctypes = DES-CBC-MD5 DES-CBC-CRC
    default_tgs_enctypes = DES-CBC-MD5 DES-CBC-CRC
[realms]
    INDIA.FOO.COM = {
        kdc = hostname1.india.foo.com
        kdc = hostname2.india.foo.com
        default_domain = india.foo.com
    }
    INDIA.BAR.COM = {
        kdc = hostname1.india.bar.com
        kdc = hostname2.india.bar.com
        default_domain = india.bar.com
    }
[domain_realm]
    .india.foo.com = INDIA.FOO.COM
    .india.bar.com = INDIA.BAR.COM
```

Specify realm settings in the "krb5.conf" file

2 Edit "krb5.conf" to specify all realm settings. This includes setting the realm names, default realm, realm-domain mappings, and KDCs for the realms.

3 Save and close "krb5.conf."

➡ The convention is to enter realm names in uppercase letters and domain names in lowercase letters.

Using Simple authentication

If you are using a directory server other than Active Directory on Windows, then in most cases you will use the Simple authentication scheme. To use Simple authentication:

1 In the LDAP Attribute mapping set the Authentication Name= :uid: `User Name=uid`.

2 For Simple authentication, the `Realm Name` is optional.

Connecting LDAP user passwords with Quark Publishing Platform Server

If you configure Quark Publishing Platform according to the instructions below and the connection between Quark Publishing Platform Server and Directory Server breaks, users can still log on to Quark Publishing Platform Server with their Directory Server credentials.

When a user logs on to Quark Publishing Platform Server successfully for the first time, encrypted user credentials are stored in Quark Publishing Platform. These credentials can then be used for subsequent log-on operations when there is no connection between Quark Publishing Platform Server and Directory Server.

To specify that the LDAP user password can be retrieved, encrypted, and stored in the Quark Publishing Platform Server database:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "ServerApp.properties" in a text-editing application.
3. To allow users to log on to Quark Publishing Platform Server when the connection between Quark Publishing Platform Server and Directory Server is broken, change the value for `authentication.external.cacheTicket=` to `true`. The default setting for `authentication.external.cacheTicket=` is `false` (that is, disabled).
4. Save and close "ServerApp.properties."

Restricting Workspace Browser palettes

To restrict the number of **Workspace Browser** windows which can be opened at a time by a user, open the file `\Server\conf\Query.properties` and specify a value for the `query.maxWatchedQueryCountPerSession` parameter.

Setting up custom content type detection

By default, Quark Publishing Platform is configured to recognize Quark formats, common picture formats, common content formats (Microsoft Office, plain text, RTF, PDF, HTML), XML, XHTML, DITA documents, and BusDoc documents. However, you can configure Quark Publishing to automatically attempt additional, custom content types. To do so:

1. To recognize a particular content type by examining a file's MIME type, make sure that MIME type is listed in the "tika-mimetypes.xml" file. If the MIME type is not included in this file, you can add it to the "custom-mimetypes" file. You can identify file types by file extension, by magic bytes, or (for XML files) by the root element. For example:

   ```
   <mime-type type="application/dita+xml; format=busdoc">
     <sub-class-of type="application/dita+xml;format=topic"/>
     <_comment>Business Documents</_comment>
   </mime-type>
   ```

   You can assign different MIME types to structurally similar but semantically different XML files. You can differentiate XML file types based on processing instructions applied inside that XML. Add a MIME type entry to the "custom-xml-types-ext.xml" file present in "ext" folder of the Platform server, to do so. For example:

   ```
   <mime-type type="application/dita+xml; format=busdoc"
   sub-class-of="application/dita+xml; format=topic">
     <pi name="Xpress">productLine="busdoc"</pi>
   </mime-type>
   ```

   For more information, see *http://tika.apache.org*.

2. Configure a Quark Publishing Platform content type based on the identified MIME type. To do so, add a mapping entry to the "content-mimetype-mappings-ext.xml"

file present in "ext" folder of the Platform server. Assets with the specified custom MIME type will be checked in with the specified custom content type. For example:

```
<content-type name="Business Document">
  <mime-type>application/dita+xml; format=busdoc</mime-type>
</content-type>
```

3. Add a corresponding mapping entry to the "Indexing Channels-ext.xml" file present in "ext" folder of the Platform server. For example:

```
<mapping contenttype="Ratings Document" channel="ratingsPreviewChannel">
  <parameter name="channelParam1">param value</parameter>
  <parameter name="channelParam2">param value</parameter>
</mapping>
```

Specifying a default PDF output style

Quark Publishing Platform Web Client users can retrieve copies of QuarkXPress projects or QuarkCopyDesk articles as PDFs. The settings for these PDFs are defined exclusively in the default PDF output style for QuarkXPress Server. To specify the default PDF output style, update the default Job Jackets file as described in "Job Jackets dialog box" in *A Guide to QuarkXPress Server*. The default PDF style can also be mentioned in the Publishing Channel used to take the PDF. The default channel for taking PDF from QuarkXPress projects is "qxpPdf"

Controlling delivery channel display settings in Web Client

Delivery channels are not visible by default. To make them available, set the property `displaydeliverychannels` to `true` in the file `QPP_HOME\webapps\workspace\WEB-INF\classes\Workspace-Config.xml`:

```
<property name="enableDeliveryChannels" value="true" />
```

Changing case-sensitivity for Quark Publishing Platform passwords

To specify case-sensitivity for Quark Publishing Platform user passwords:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "ServerApp.properties" in a text-editing application.
3. If you want to specify case-sensitivity for user passwords, set `server.password.case.sensitive` to `true`. Otherwise, enter `false`.
4. Save and close "ServerApp.properties."

➡ The default setting for `server.password.case.sensitive` is `true`. Quark Publishing Platform administrators should inform users if case-sensitive passwords are required.

Managing filters and index service settings

Quark Publishing Platform Server uses four filters for generating previews, and you can change the values in the ".properties" files for each filter. These files include:

- "AsposeFilterServiceConfig" (provides previews and thumbnails of Word, RTF,EXCEL,POWER POINT and plain text files)
- "QxpsFilterServiceConfig.properties" (provides previews and thumbnails of QuarkXPress projects and QuarkCopyDesk articles)
- "IMFilterServiceConfig.properties" (provides previews and thumbnails of most image files)
- "JawsFilterServiceConfig.properties" (provides previews and thumbnails of certain image files — PDF files, EPS files, and Adobe® Illustrator® files)
- "XADocFilterServiceConfig.properties" (provides previews and thumbnails of XML files)

  The following commands are common to all four filters:

- `<FILTER_NAME>filter.generateAttributes`: Set this to `true` to generate filter-specific attributes.
- `<FILTER_NAME>filter.generatePreview`: Set this to `true` to generate previews.
- `<FILTER_NAME>filter.generateThumbnail`: Set this to `true` to generate thumbnails.
- `<FILTER_NAME>filter.generateText`: Set this to `true` to extract text and enable full-text search.
- `<FILTER_NAME>filter.previewWidth`: Set this to specify the width of previews (measured in pixels).
- `<FILTER_NAME>filter.previewHeight`: Set this to specify the height of previews (measured in pixels).
- `<FILTER_NAME>filter.thumbnailHeight`: Set this to specify the height of thumbnails (measured in pixels).
- `<FILTER_NAME>filter.thumbnailWidth`: Set this to specify the width of thumbnails (measured in pixels).
- `<FILTER_NAME>filter.previewPages`: Set this to specify the number of preview pages to generate.
- `<FILTER_NAME>filter.processTimeOut`: Set this to specify the number of milliseconds after which an incomplete index preview process will terminate.

Index service settings

You can specify parameters for `asset indexing`. As a separate background process that runs inside Quark Publishing Platform Server, indexing can consume significant system resources. By editing the "Indexing.properties" file as described below, you can manage index processes (called "indexing thread pools") by adjusting the number of concurrent thread pools and the time intervals between thread pools.

To manage timing and threads for asset indexing:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "Indexing.properties" in a text-editing application.

3. To specify the maximum number of concurrent indexing threads that operate in the background, adjust the `indexingThread.pool.maxActive=` value.
4. To specify the amount of time (in milliseconds) to wait for an available thread from the pool, adjust the `indexingThread.pool.maxWait=` value. (A value of `-1` indicates waiting indefinitely.)
5. To specify the maximum number of idle threads in the pool, adjust the `indexingThread.pool.maxIdle=` value.
6. To specify the minimum number of idle threads in the pool, adjust the `indexingThread.pool.minIdle=` value.
7. To specify the number of milliseconds a thread can be in the pool before it is evicted, adjust the `indexingThread.pool.minEvictableIdleTimeMillis=` value.
8. To specify the number of milliseconds after which the evictor thread should run to evict idle threads, adjust the `indexingThread.pool.timeBetweenEvictionRunsMillis=` value. The evictor thread runs in the background.

When the "evictor" thread is run, it releases all idle threads from the pool beyond `indexingThread.pool.maxIdle` value. However, the number of idle threads specified as `indexingThread.pool.minIdle` value are maintained in the pool.

9. To index assets not previously indexed when Quark Publishing Platform Server starts, enter `true` for the `indexing.indexPendingAssetsOnStartup=` value.
10. Save and close "Indexing.properties."

ASPOSE Filter

The ASPOSE filter handles text files, Microsoft Word, Excell and PowerPoint documents, RTF files and plain text files. Adjust the ASPOSE filter settings:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "ASPOSEFilterServiceConfig.properties" in a text-editing application.
3. `apsfilter.generatePreview=true`.
4. `apsfilter.generateThumbnail=true`.
5. `apsfilter.generateAttributes=true`.
6. To specify the time the process should wait before exiting, if indexing is not done, enter a value in the `asposefilter.processTimeOut=` entry (in milliseconds).
7. To specify the maximum number of pages for which preview should be generated, enter a value in the `asposefilter.previewPages=` entry.
8. Save and close "ASPOSEFilterServiceConfig.properties."

Configure processing of MS Office documents

Adjust the MS Office filter settings:

1. Open the "OfficeServiceConfig.properties" file in the Server Installation `conf` folder.

2 Set the `office.defaultImageFormat` property to specify the default output format type. Default format is PNG.

3 Set the `office.defaultHorizontalResolution` property to specify the default horizontal resolution of previews.

4 Set the `office.defaultVerticalResolution` property to specify the default vertical resolution of previews.

5 Set the `office.onePagePerSheet` property to **true** and all content of the sheet will output to only one page.

6 Set the `office.showHiddenWorksheets` property to control the visibility of hidden worksheets.

QuarkXPress Server Filter

To adjust QuarkXPress filter settings:

1 Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "QxpsFilterServiceConfig.properties" in a text-editing application.

3 Set the `qxpsfilter.useSpreadForArticles=` value to `true` to specify that previews for articles are generated for each spread.

4 Set the `qxpsfilter.useSpreadForProjects=` value to `true` to specify project previews with spreads.

5 By default, XML deconstruction is enabled. The default setting for the `qxpsfilter.generateDeconstructedXML=` entry is `true.` To disable XML deconstruction, set the entry to `false`.

6 By default, Full Text Search (FTS) is enabled. The default setting for the `qxpsfilter.generateText=` entry is `true`. To disable FTS, set the entry to `false`.

➡ A `true` value for `qxpsfilter.generateDeconstructedXML` also enables FTS for QuarkXPress projects and QuarkCopyDesk articles. Text generation is dependent on XML generation. If you set `generateDeconstructedXML` to `false`, Quark recommends also setting `qxpsfilter.generateText` to `false`.

7 To specify the scale of preview images, adjust the `qxpsfilter.previewScale=` value. The default value of `1` indicates previews at 100 percent. The values of `2`, `3`, and `6` are 200 percent, 300 percent, and 600 percent, respectively.

8 To specify the scale of thumbnail images, adjust the `qxpsfilter.thumbnailScale=` value. The default value of `1` indicates previews at 100 percent. The values of `2`, `3`, and `6` are 200 percent, 300 percent, and 600 percent, respectively.

9 To specify image quality for JPEG previews, adjust the `qxpsfilter.jpegQuality=` value. Enter `1` for the highest quality, `2` for high quality, `3` for medium quality, and `4` for lowest quality.

10 Save and close "QxpsFilterServiceConfig.properties."

JAWS filter settings

The JAWS filter is part of the ImageMagick® Filter Service Configurations for EPS, PDF, and Adobe® Illustrator® files. To specify settings for the JAWS filter:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "JawsFilterServiceConfig.properties" in a text-editing application.
3. To specify the resolution for thumbnails and previews (measured in dots-per-inch), adjust the `jawsfilter.resolution=` value.

➡ The `jawsfilter.resolution=` value is the fallback resolution used to scale the preview image to the specified size when the proper resolution cannot be calculated.

4. Save and close "JawsFilterServiceConfig.properties."

XML Author filter settings

The XML Author filter is part of the configuration for XML files. To specify settings for the XML Author filter:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "XADocFilterServiceConfig.properties" in a text-editing application.
3. To specify the amount of time a process will wait for indexing (with XADocFilter) to complete, adjust the `jawsfilter.resolution=` value.
4. Save and close "XADocFilterServiceConfig.properties."

ImageMagick, Jaws, and DITA OT directories

By default, ImageMagick, Jaws, the DITA Open Toolkit are included with Quark Publishing Platform. However, if you have existing installations of ImageMagick or Jaws, perform the following tasks:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "ServerApp.properties" in a text-editing application.
3. Scroll to the `IMAGE_MAGICK_HOME=` entry.
4. Enter the path to your existing ImageMagick bin folder.
5. Scroll to the `JAWS_HOME=` entry.
6. Enter the path to your existing Jaws bin folder.
7. Scroll to the `DITA_HOME=` entry.
8. Enter the path to your existing DITA Open Toolkit folder.
9. Save and close "ServerApp.properties."

Full text indexing configuration

To configure FTSIndex filter settings:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "LuceneTextIndexingConfig.properties" in a text-editing application.
3. By default, the storage location for index files is in the Quark Publishing Platform installation folder. However, you can change the location by changing the `lucene.index.dir=` parameter to indicate the folder in which you want to store index files.
4. Specify the language (class) used most frequently for text indexing and query term analysis by changing the `lucene.analyzerClass=` parameter. The default setting — `StandardAnalyzer` — recognizes English semantics for Full Text Search. But if the majority of the text in your workflow is not English, you can specify another language to search more accurately according to that language's semantics. The languages are listed in the "LuceneTextIndexingConfig.properties" file.

➡ "LuceneTextIndexingConfig.properties" contains information for each parameter, as well as links to the Apache documentation.

5. To specify the number of changes after which an index compaction shoud be performed, set a value for `lucene.maxModificationWithoutOptimisation`.
6. Save and close "LuceneTextIndexingConfig.properties."

Charting Service

To configure the propeties for the Charting service modify the below properties:

1. Enter the default output format (PDF/HTML/IMAGE/JSON) in the `charting.defaultOutputFormat` parameter.
2. Enter the default values for the image output properties in the following parameters:
    1. `charting.defaultImageFormat=png`
    2. `charting.defaultWidth=600`
    3. `charting.defaultScale=`
3. Enter the path to the default HTML template: `charting.defaultHTMLTemplateURI=classpath\:DefaultHTMLTemplate.html`

Integrating QLA with Quark Publishing Platform

To reconfigure Quark® License Administrator (QLA) primary and backup server settings with Quark Publishing Platform:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "Qla.properties" in a text-editing application.
3. Enter the current IP address or hostname of the computer where you installed QLA in the `QlaServer.machinename=` parameter.
4. Enter the port number in the `QLAServer.port=` parameter.

5. If you have a backup QLA server, enter the correct IP address or hostname and port values in the `Backup.QlaServer.machinename=` and `Backup.QlaServer.port=` parameters.
6. In the `Qla.SerialNumber=` parameter, enter the QLA serial number for Quark Publishing Platform Server. The QLA Server Console and QLA Client applications display your serial number.

➡ The serial number is updated in "Qla.properties" based on the validation code that you provided when you installed or updated Quark Publishing Platform Server.

7. Save and close "Qla.properties."

Enabling IPTC support

IPTC values in a picture file can be recognized and included in the picture's attribute list. To enable or disable IPTC functionality, open the file `\Server\conf\IMFilterServiceConfig.properties` and set the `imfilter.generateIPTCAttributes` property to `true` or `false`. You can also specify preview/thumbnail resolution by setting a value for `imfilter.resolution`.

RMI and CORBA clients only

By default, Quark Publishing Platform clients require only port 61400. However, if you are using or developing your own RMI and CORBA clients, you might want to read the topics below.

Changing ports used by Quark Publishing Platform Server

By default, Quark Publishing Platform clients require only port 61400. However, if you are using or developing your own RMI and CORBA clients, you can adjust settings for the server and file streaming ports in your Quark Publishing Platform environment. By default, Quark Publishing Platform Server provides services on network ports 61400 to 61407, but you can configure all these ports.

Changing the default ports

By default, Quark Publishing Platform clients require only port 61400. However, if you are using or developing your own RMI and CORBA clients, change the ports as follows:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "ServerApp.properties" in a text-editing application.
3. To specify the port for running Java-based RMI clients (such as Quark Publishing Platform Script Manfager), enter a value in the `rmi.port=` property. Remote Method Invocation (RMI) is a standard remote procedure call that allows Java objects on your network to run remotely.

4. To specify the port to which service are bound, enter a value in the `rmi.serviceport=` property.
5. To specify the port for managing IIOP requests, enter a value in the `namingservice.port=` property. Internet Inter-Orb Protocol (IIOP) is a message protocol for handling objects on a TCP/IP network. QuarkXPress, QuarkCopyDesk, and Quark Publishing Platform Client connect through this port.
6. To specify the port on which CORBA objects are exported, enter a value in the `serverORB.port=` property.
7. To specify the port for all Quark Publishing Platform client application communication, enter a value in the `jms.openWirePort=` property. In a Quark Publishing Platform environment, Java Messaging Service (JMS) uses OpenWire® protocol.
8. To indicate which port the Tomcat server is running on, enter a value in the `webServer.port=` property. Quark Publishing Platform Web Client connects through this port.
9. Save and close "ServerApp.properties."

Specifying the Tomcat server port

After changing default ports, you must also specify the port for running the Tomcat server in a separate file, as follows:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "server.xml" in a text-editing application.
3. Change the Web server port.
4. Save and close "server.xml."

Changing the file streaming port

To change the file streaming port:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "SocketStreaming.properties" in a text-editing application.
3. To specify the port for file uploads and downloads, enter a value in the `socketStreaming.port=` property.
4. Save and close "SocketStreaming.properties."

Multiple network cards

By default, Quark Publishing Platform clients require only port 61400. However, if you are using or developing your own RMI and CORBA clients, and the computer running Quark Publishing Platform Server has more than one network interface card, you can bind to a particular card for Quark Publishing Platform Server or to any and all IP addresses on the computer.

Binding a specific IP address

To bind to a specific IP address:

1 Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "ServerApp.properties" in a text-editing application.

3 Scroll to the `server.machinename=` entry. Enter the specific IP address of the network card to which you want to bind.

4 Scroll to the `server.bindtoip=` entry. Enter `true` for Quark Publishing Platform Server to only bind to the IP address and name identified by the "server.machinename=" entry.

5 Save and close "ServerApp.properties."

Editing "server.xml"

You must also edit "server.xml," as follows:

1 Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "server.xml" in a text-editing application.

3 In the "connector" tag, add the IP address. For example, change `<Connector port=61400` to `<Connector address = "<IP Address>" port="61400"`.

4 Save and close "server.xml."

Binding all IP addresses to a single computer

To bind to all IP addresses on a single computer:

1 Open the "conf" folder in your Quark Publishing Platform Server folder.

2 Open "ServerApp.properties" in a text-editing application.

3 Scroll to the `server.machinename=` entry. Enter `localhost`.

4 Scroll to the `server.bindtoip=` entry. Enter `false` for Quark Publishing Platform Server to bind to any and all IP addresses on the computer.

5 Scroll to the `server.additionalnames=` entry. Enter the IP address of the network card to which you want to bind. If you have multiple IP addresses, you can separate entries with a comma (for example, `server.additionalnames= 10.91.43.266,10.X.Y.Z`). If your computer has only one network card, leave this field blank.

➡ This list includes non-default IPs only. Do not specify the default IP of the computer here because Quark Publishing Platform Server automatically uses the default IP address. Adding a default IP here is not recommended.

6 Save and close "ServerApp.properties."

Firewalls with NAT

If you are using or developing RMI and CORBA clients and you want to provide access to Quark Publishing Platform Server over the Internet through a firewall running Network Address Translation (NAT) services, you must specify the public IP address to which the Quark Publishing Platform Server's private IP address is mapped.

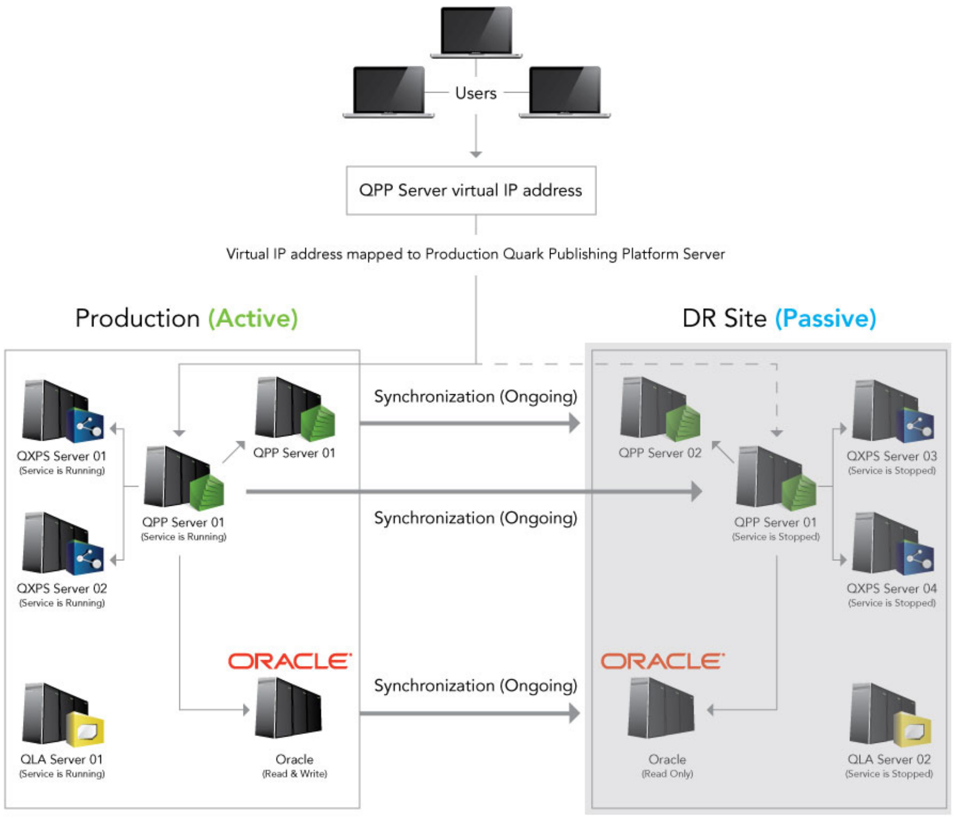
To specify the public IP address:

1. Open the "conf" folder in your Quark Publishing Platform Server folder.
2. Open "ServerApp.properties" in a text-editing application.
3. Scroll to the `server.additionalnames=` entry.
4. Enter the public IP address to which the Quark Publishing Platform Server's private IP address is mapped.
5. Save and close "ServerApp.properties."

Failover setup

This topics describes one way you can configure a Quark Publishing Platform installation for failover.

The key to a setup like this is to use a virtual IP address for the Quark Publishing Platform Server computer.


Users
QPP Server virtual IP address
Virtual IP address mapped to Production Quark Publishing Platform Server
Production (Active)
DR Site (Passive)
QXPS Server 01 (Service is Running)
QXPS Server 02 (Service is Running)
QPP Server 01 (Service is Running)
QPP Server 01
QLA Server 01 (Service is Running)
ORACLE
Oracle (Read & Write)
Synchronization (Ongoing)
Synchronization (Ongoing)
Synchronization (Ongoing)
QPP Server 02
QPP Server 01 (Service is Stopped)
QXPS Server 03 (Service is Stopped)
QXPS Server 04 (Service is Stopped)
ORACLE
Oracle (Read Only)
QLA Server 02 (Service is Stopped)

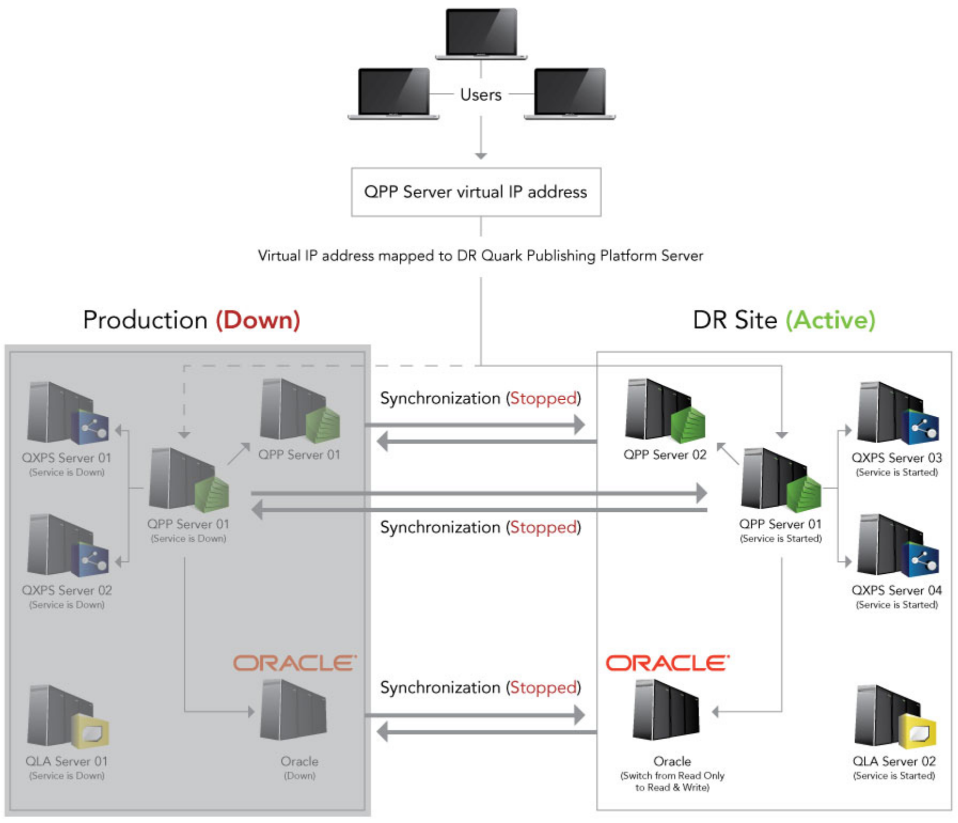
Failover setup: Normal operation

During normal operations, the Production site is up and running and the virtual IP address is mapped to that server. The DR (disaster recovery) site is in dormant mode, with all Quark Services stopped. The following things are being synchronized from the Production servers to the DR server:

- Quark Publishing Platform repository folder(s)
- Quark Publishing Platform database
- `QPP Server/conf` folder
- `QPP Server/Index` folder

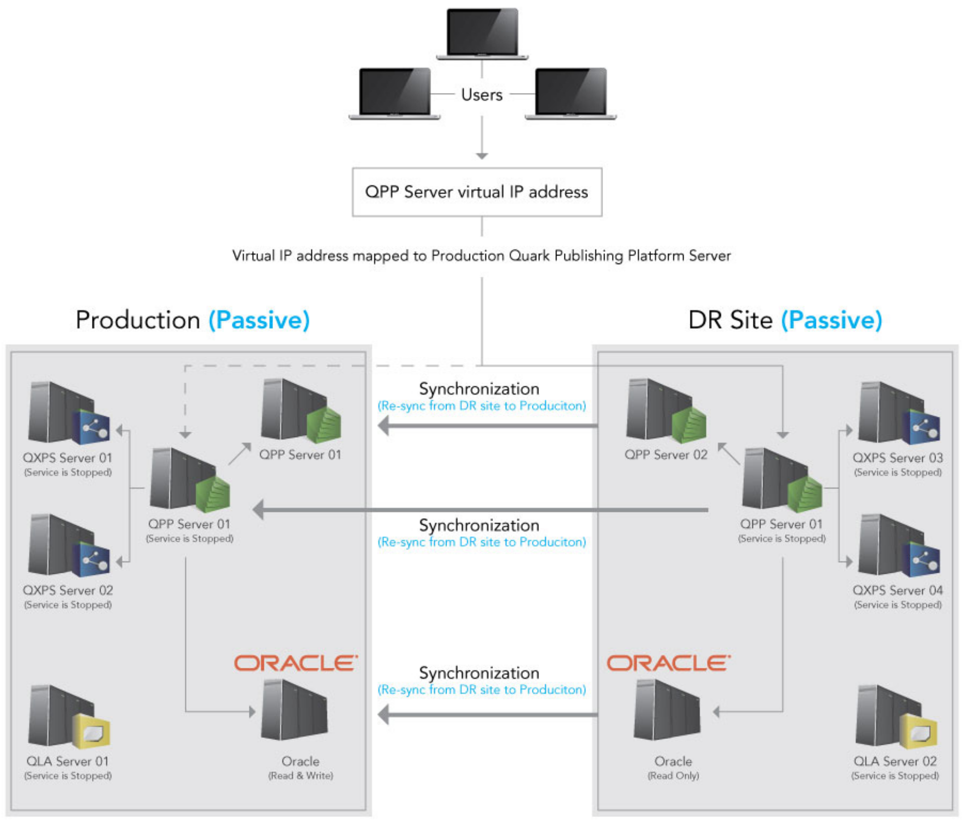
➡ Synchronization of the Quark Publishing Platform File Server folder(s) and Quark Publishing Platform Oracle database must be done at the same time and at the same intervals to maintain data integrity.

If the Production site goes down, synchronization stops and the virtual IP address is re-mapped to the DR site. End users should not notice the change because they are still using the same virtual IP adddress.


Users
QPP Server virtual IP address
Virtual IP address mapped to DR Quark Publishing Platform Server
Production (Down)
DR Site (Active)
QXPS Server 01 (Service is Down)
QPP Server 01
QPP Server 01 (Service is Down)
QXPS Server 02 (Service is Down)
ORACLE
QLA Server 01 (Service is Down)
Oracle (Down)
Synchronization (Stopped)
Synchronization (Stopped)
Synchronization (Stopped)
QPP Server 02
QPP Server 01 (Service is Started)
QXPS Server 03 (Service is Started)
QXPS Server 04 (Service is Started)
ORACLE
Oracle (Switch from Read Only to Read & Write)
QLA Server 02 (Service is Started)

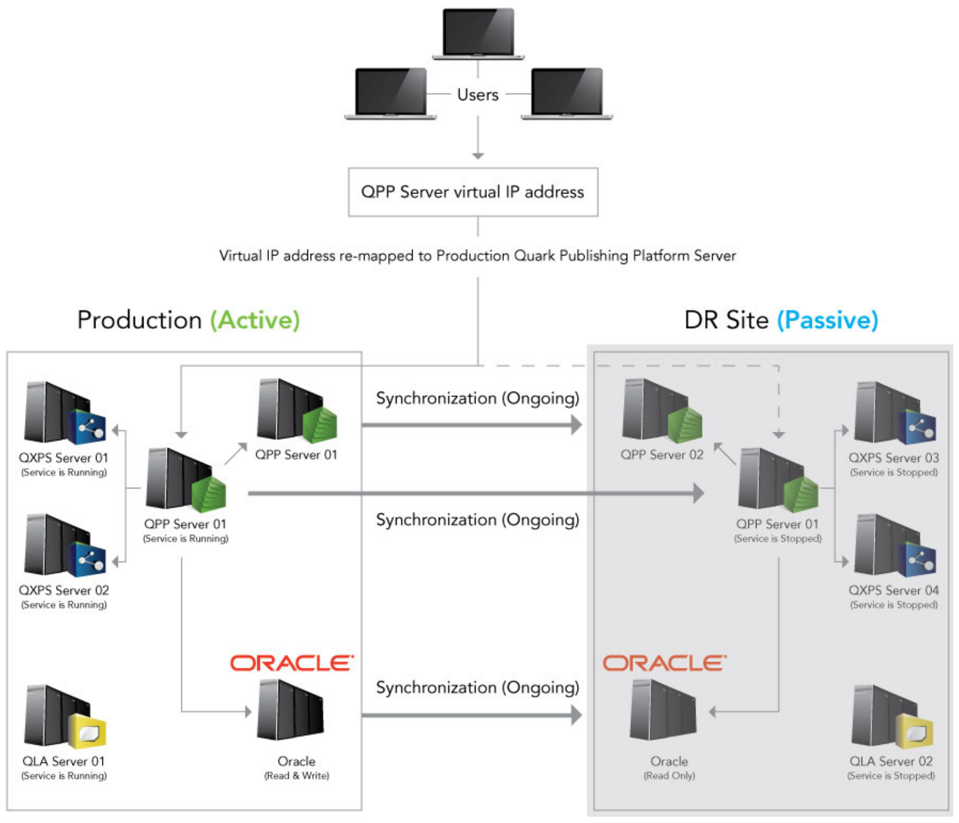
Failover setup: Production server goes down

At this point, an administrator should perform an orderly shutdown of the DR site, then restart synchronization from the DR site to the Production site.


Users
QPP Server virtual IP address
Virtual IP address mapped to Production Quark Publishing Platform Server
Production (Passive)
DR Site (Passive)
Synchronization
(Re-sync from DR site to Produciton)
Synchronization
(Re-sync from DR site to Produciton)
Synchronization
(Re-sync from DR site to Produciton)
QXPS Server 01 (Service is Stopped)
QPP Server 01
QPP Server 01 (Service is Stopped)
QXPS Server 02 (Service is Stopped)
ORACLE
Oracle (Read & Write)
QLA Server 01 (Service is Stopped)
QPP Server 02
QPP Server 01 (Service is Stopped)
QXPS Server 03 (Service is Stopped)
QXPS Server 04 (Service is Stopped)
ORACLE
Oracle (Read Only)
QLA Server 02 (Service is Stopped)

Failover setup: Synchronizing from DR site to Production site

When synchronization is complete between the DR site and the Production site, remap the virtual IP address to the Production site, reverse the synchronization so that changes are pushed from the Production server to the DP site, and then launch all of the Quark Publishing Platform components.


Users
QPP Server virtual IP address
Virtual IP address re-mapped to Production Quark Publishing Platform Server
Production (Active)
DR Site (Passive)
QXPS Server 01 (Service is Running)
QPP Server 01
QPP Server 01 (Service is Running)
QXPS Server 02 (Service is Running)
ORACLE
Oracle (Read & Write)
QLA Server 01 (Service is Running)
Synchronization (Ongoing)
Synchronization (Ongoing)
Synchronization (Ongoing)
QPP Server 02
QPP Server 01 (Service is Stopped)
QXPS Server 03 (Service is Stopped)
QXPS Server 04 (Service is Stopped)
ORACLE
Oracle (Read Only)
QLA Server 02 (Service is Stopped)

Failover setup: Normal operation restored

Encrypting a plain text password

In Quark Publishing Platform 10.1 and later, you can encrypt all of your passwords that are in plain text. For example, if you are using an SQL Server or Oracle as a database in the Platform Server then your database user password is stored in plain text in the "database.properties" file. You can encrypt this password.

To encrypt plain text passwords:

1. Open the command prompt and navigate to the Platform server installation folder.
2. Launch the "encrypt.bat" file with your password (`encrypt.bat password`) to retrieve its encrypted value.
3. Copy the generated encrypted value of the password and use this encrypted value in any of properties file (for example, "database.properties").
4. Open the "lib" folder in your Quark Publishing Platform Server folder.
5. Open "qpp-server-app-10.1.jar" and navigate to `\com\quark\qpp\app` inside the jar file.

6 Open "ServerStartupContext.xml " in a text-editing application and under the `securePlaceholderConfig` node, add an entry for each of the properties file that you used the encrypted value in. Remove this entry from the `placeholderConfig` node if it exists.

7 Save the file, update the jar and restart the server.

You can use any of the algorithms mentioned in the "ServerStartupContext.xml" file and you can modify the key used to encrypt the password (default is QUARK).

Enabling forced log off during inactivity

Configuring WebAdmin to enable forced log off

Forced Log-Off can be enabled by engaging a predefined Servlet Filter in the "Web.xml" file in the `[QPP Server]/webapps/admin` folder:

- `session-idle-time`: Specifies the time (in seconds) of inactivity after which a user will be forced to log off of WebAdmin.
- `exclude-url-patterns`: Specifies URL patterns which are not considered user activity.
- `pre-expiry-mag-time`: Specifies the time (in seconds) before pending inactivity bases log off, a warning or countdown message will be shown.

```
<filter>
  <filter-name>SessionTimeoutCookieFilter</filter-name>

<filter-class>com.quark.web.activity.servlet.SessionTimeoutCookieFilter</filter-class>

  <init-param>
        <param-name>session-idle-time</param-name>
        <param-value>60</param-value>
  </init-param>
  <init-param>
        <param-name>exclude-url-patterns</param-name>
        <param-value>/admin/keepAlive.qsp</param-value>
  </init-param>
  <init-param>
        <param-name>pre-expiry-mag-time</param-name>
        <param-value>40</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>SessionTimeoutCookieFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

Forced Log-Off can also be enabled in the "Admin-Home.jsp" file in `QPP Home/webapps/admin/jsp` folder:

```
<!-- Uncomment for "Forced Logoff on Inactivity" -->
  <script type="text/javascript" src="resources/js/SessionTimer.js></script>

  <link type="text/css" href="resources/js/SessionTimer.css
re;="stylesheet></link>

function logoutDueToInactivity ()  {
  window.location.href = "logout.qsp";
};
<%-- Initialise inactivity monitor --%>
SessionTimer.init(logoutDueToInactivity);
```

Configuring Workspace to enable forced log off

Forced Log-Off can be enabled by engaging a predefined Servlet Filter in the "Web.xml" file in the `[QPP Server]/webapps/workspace` folder:

- `session-idle-time`: Specifies the time (in seconds) of inactivity after which a user will be forced to log off of WebAdmin.
- `exclude-url-patterns`: Specifies URL patterns which are not considered user activity.
- `pre-expiry-mag-time`: Specifies the time (in seconds) before pending inactivity bases log off, a warning or countdown message will be shown.

```
<filter>
  <filter-name>SessionTimeoutCookieFilter</filter-name>

<filter-class>com.quark.web.activity.servlet.SessionTimeoutCookieFilter</filter-class>

  <init-param>
        <param-name>session-idle-time</param-name>
        <param-value>300</param-value>
  </init-param>
  <init-param>
        <param-name>exclude-url-patterns</param-name>

<param-value>/workspace/keepAlive.qsp/workspace/assetHeaderUpdate.qsp</param-value>

  </init-param>
  <init-param>
        <param-name>pre-expiry-mag-time</param-name>
        <param-value>60</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>SessionTimeoutCookieFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

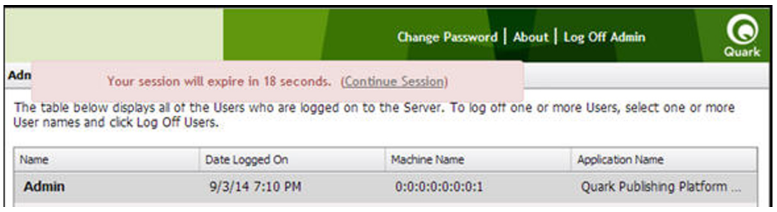
Forced Log-Off can also be enabled in the "User-Home.jsp" file in `QPP Home/webapps/workspace/jsp` folder:

```
<!-- Uncomment for "Forced Logoff on Inactivity" -->
  <script type="text/javascript" src="resources/js/SessionTimer.js></script>

  <link type="text/css" href="resources/js/SessionTimer.css
rel="stylesheet></link>

<%-- Initialise inactivity monitor --%>
SessionTimer.init(UserHomeUI.logout);
```

The countdown message will be shown with remaining seconds. The user can prevent the forced log off by clicking **Continue Session**.



Your session will expire in 18 seconds. (Continue Session)

The table below displays all of the Users who are logged on to the Server. To log off one or more Users, select one or more User names and click Log Off Users.

Name	Date Logged On	Machine Name	Application Name
Admin	9/3/14 7:10 PM	0:0:0:0:0:0:0:1	Quark Publishing Platform ...

Forced log off countdown

Quark Publishing Platform Web Client: Manual configuration

The following sections describe how to configure the advanced settings of Quark Publishing Platform Web Client.

Configuration overview

The workspace configuration is organized into two categories:

- Application level settings
- Document level settings

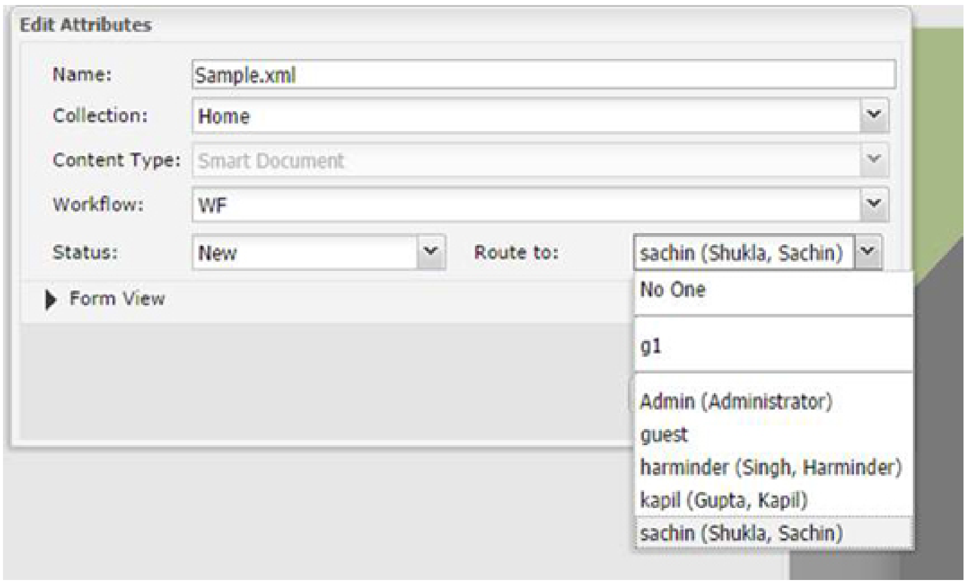
These are set in the "Workspace-Config.xml" configuration file in the `[QPP Server]/webapps /workspace/WEB-INF/classes` folder.

Application level settings

The attribute elements of `ApplicationSettings` defines various application level settings:

- `userNameFormatting`: Specifies the format of the **user name** to be displayed in all dialogs:

1. `0` to display login name <username>
2. `1` to display <First Name> <Last Name>
3. `2` to display <Last Name>, <First Name>



- `layoutPreview`: Specifies the look and feel of the preview window for assets belonging to the QuarkXPress content type:

1 `True` to display the preview in a separate browser window
2 `False` to display the preview ina javascript window instead of a browser.

- `viewrevision_expandAll`: Specifies whether all revision comments need to be expanded in the view revisions dialog.
- `supported_picture_extension`: Specifies the possible file types allowed when importing a picture/text in the article/project editors.
- `picture_editing_extension`: Specifies the possible file formats allowed when editing pictures in the article/project editors.
- `defaultPreviewScale`: Specifies the default zoom preview setting to be used when editing articles/projects and viewing assets (live preview). Valid values for this parameter range from 0.1 to 5.
- `ajaxTimeout`: Specifies the time period for all background requests originating from browser after which the requests will be marked as timed out. The value is specified in milliseconds .
- `showFormView`: Specifies whether to show the form view panel in the **Check In** dialog. Set to `true` to show the form view pane.
- `topBannerJspPath`: Specifies the path to the .jsp file to be included to show the top banner in the assignment page.
- `logoFilePath`: Specifies the path to the logo image shown in the top banner in the assignment page .
- `enabledPublishingChannels`: Defines the list of publishing channels that need to be honored when selecting an asset.
- `enableDeliveryChannels`: Defines whether delivery channels needs to be enabled or not.

- `enabledDeliveryChannels`:Defines the list of delivery channels that need to be honored when selecting an asset. This would get enabled when the `enableDeliveryChannels` option is set to `true`.
- `allowPublishedRenditionDownload`: Defines whether to allow the download output displayed for the various previews using the publishing defined for the selected asset .

```
<ApplicationSettings>
  <Add key="viewrevision_expandAll" value="false"/>
  <Add key="supported_picture_extension" value="bmp;jpg;jpeg;tif;tiff;gif;"/>

  <Add key="picture_editing_extension" value="jpg;jpeg;tif;tiff;eps"/>
  <Add key="userNameFormatting" value="0"/>
  <Add key="layoutPreview" value="true"/>
  <Add key="defaultPreviewScale" value="0.8"/>
  <Add key="ajaxTimeout" value="300000"/>
  <Add key="showFormView" value="false"/>
  <Add key="topBannerJspPath" value="Header.jsp"/>
  <Add key="logoFilePath" value="images/login/topbanner-innerpage-left.png"/>

  <Add key="enabledPublishingChannels"
value="qxpPdf;qxpEpub;qxpAppStudio;qxpAppStudioPackage;busDocPdf;busDocHtml;busDocQxp/>

  <Add key="enableDeliveryChannels" value="false"/>
  <Add key="enabledDeliveryChannels"
value="checkInToSharepoint;checkInToFileNet;checkInToDocumentum;sendEmail;sendToFTPServer"/>

  <Add key="allowPublishedRenditionDownload" value="true"/>
</ApplicationSettings>
```

Multi-Channel preview

For each content type you will specify the publishing channels that should be available to the user in the **Preview** tab of the assignment page. Publishing channels for preview are configured using the `<PreviewSettings>` element:

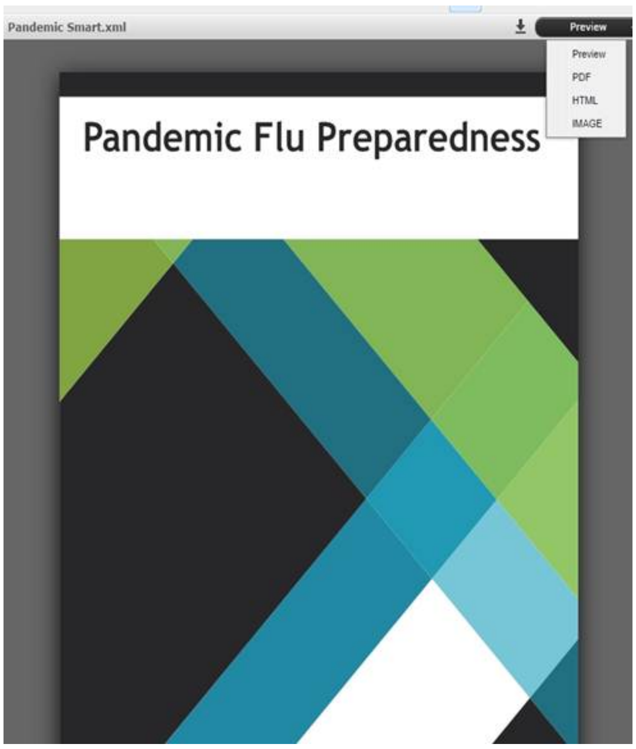
- `displayName`: Specifies the channel name displayed in the user interface. (optional) If not given, then `outputFormat` is used
- `Id`: Specifies the Publishing Channel Id as defined in the Platform Server
- `ContentType`: Specifies the content type of the asset .
- `ApplyToChildContentTypes`: Specifies whether to include the child content types .
- `outputFormat` : Supported values:

1. IMAGE_ARCHIVE: An image archive for the published output, which will be rendered inside a web page
2. HTML_ARCHIVE: An HTML archive for the published output, which will be rendered as is, pointing to the file name present in the HTML Archive
3. PDF_ARCHIVE: The published output PDF, which will be rendered as is

- `downloadChannel`: (optional) Used in case a different channel needs to be invoked for download of selected channel preview

```
<PreviewSettings>
  <ChannelMappings>
    <ChannelMapping contentType ="Business Document" applyToChildContentTypes
 ="true">
      <Channels>
        <Channel id="busDocPdf" outputFormat="PDF_ARCHIVE"
```

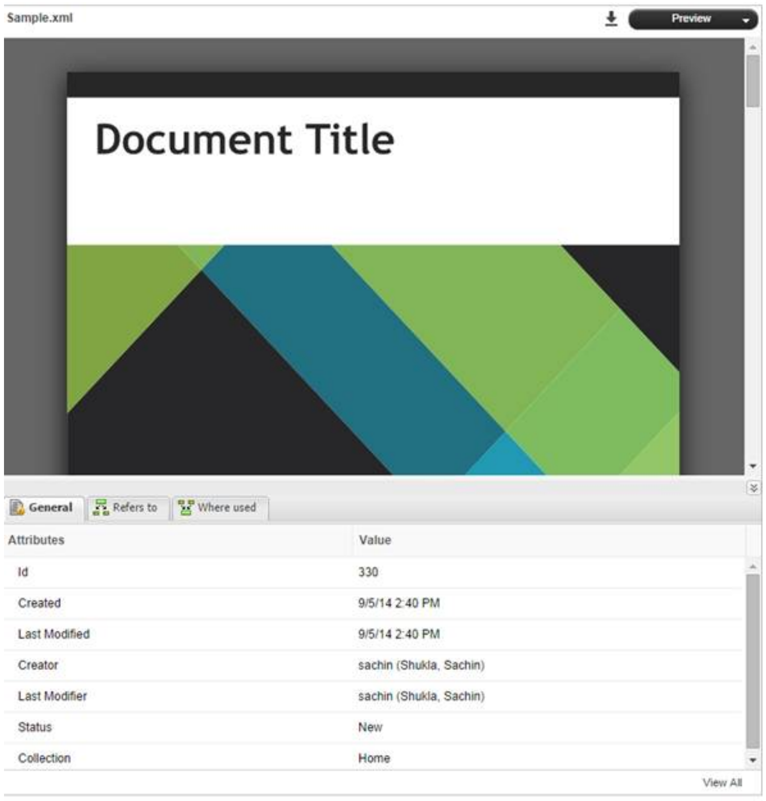
```
displayName="PDF"/>
        <Channel id="busDocHtml" outputFormat="HTML_ARCHIVE"
displayName="HTML"/>
        <Channel id="busDocJpeg" outputFormat="IMAGE_ARCHIVE"
displayName="IMAGE"/>
      </Channels>
    </ChannelMapping>
    <ChannelMapping contentType="Smart Content"
applyToChildContentTypes="true">
      <Channels>
        <Channel id="smartDocPdf" outputFormat="PDF_ARCHIVE"
displayName="PDF"/>
        <Channel id="smartDocHtml" outputFormat="HTML_ARCHIVE"
displayName="HTML"/>
        <Channel id="smartDocJpeg" outputFormat="IMAGE_ARCHIVE"
displayName="IMAGE"/>
      </Channels>
    </ChannelMapping>
  </ChannelMappings>
</PreviewSettings>
```



Attrbutes for General pane

The attributes for the **General** tab are configured using the `<PreviewAttributes>` element. The values specify the list of attributes to be displayed in the **General** tab for the selected asset.

```
<PreviewAttributes>
  <PreviewAttribute>Id</PreviewAttribute>
  <PreviewAttribute>Created</PreviewAttribute>
  <PreviewAttribute>Last modified</PreviewAttribute>
  <PreviewAttribute>Creator</PreviewAttribute>
  <PreviewAttribute>Last modifier</PreviewAttribute>
  <PreviewAttribute>Status</PreviewAttribute>
  <PreviewAttribute>Collection</PreviewAttribute>
  ........
</PreviewAttributes>
```



Attributes	Value
Id	330
Created	9/5/14 2:40 PM
Last Modified	9/5/14 2:40 PM
Creator	sachin (Shukla, Sachin)
Last Modifier	sachin (Shukla, Sachin)
Status	New
Collection	Home

Role based Toolbar configuration

This section describes how to define the toolbar items for a desired role. Role based configuration for toolbar items are configured using the `<ToolbarConfig>` element.

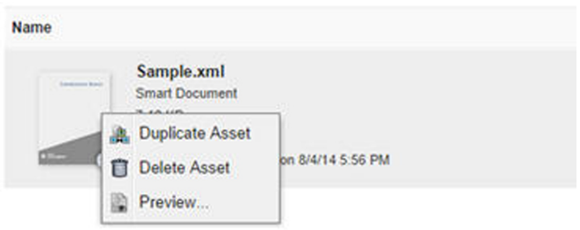
- `Role name`: Specify the name of role whose configuration is to be defined.
- `Item Id`: Specify the id of the item whose appearance needs to be configured
- `showInToolbar`: If the menu item is not to be shown in the toolbar, set this attribute to false
- `showInContext`: If the menu item is not to be shown in the context menu, set this attribute to false .
- `showInNewMenu` :If the menu item is not to be shown in the New Asset menu, set this attribute to false
- `showInTemplateMenu`: If the menu item is not to be shown in the New Asset from Template, set this attribute to false

The following is the XML structure of the toolbar configuration:

```
<Role name ="[ROLE NAME]">
  <Item id ="[ITEM ID]"
    showInToolbar ="[true/false (defaults to true)]"
    showInContextMenu ="[true/false (defaults to true)]"
    showInNewMenu ="[true/false (defaults to true)]"
    showInTemplateMenu ="[true/false (defaults to true)]"/>
</Role>
```

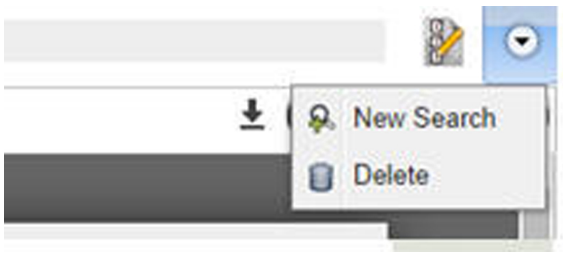
```
<ToolbarConfig>
  <Role name ="Guest">
    <Item id ="new_qcd_menu_item"/>
    <Item id ="new-search-btn"/>
    <Item id ="duplicate-asset" showInToolbar="false"/>
    <Item id ="delete-asset"/>
    <Item id ="show-edit-attributes" showInContextMenu="false"/>
    <Item id ="show-Asset_preview" showInToolbar="false"/>
    ........
  </Role>
</ToolbarConfig>
```

The Context menu for the Guest role will look like:



Context menu for Guest role

Toolbar items fo the Guest role will look like:



Toolbar items for Guest role

Following is the list of item ids:

- `checkin` : Checkin Asset
- `show-check-out` : Checkout Asset
- `cancel-checkout` : Cancel Asset Checkout
- `get-asset` : Get Asset
- `new-search-` btn : New Search (Not available as Context Menu Item)
- `show-Asset-Preview` : Preview Asset
- `show-edit-attributes`: Edit Asset Attributes
- `attachment-info` : Show Attachment Information
- `view-revisions` : View Asset Revisions
- `duplicate-asset` : Duplicate Asset
- `delete-asset` : Delete Asset
- `reindex-asset` : Reindex Asset (Not shown in Toolbar by default)
- `publish-menu-btn` : Publish an asset

- `deliver-menu-btn` : Deliver an asset
- `restore` : Restore assets
- `archive-asset` : Archive assets
- `open-collection` : Open the collection to which the asset belongs
- `refresh-datadoc` : Refresh a datadoc
- `unlink-datadoc` : Unlink a datadoc
- `open-readonly` : Open asset as read-only
- `new_qcd_menu_item` : Create new QuarkCopyDesk Article
- `new_qxp_menu_item` : Create new QuarkXpressProject from template

➡ The `showInToolbar` and `showInContextMenu` attributes are applicable for menu items meant to be displayed in the toolbar and the context menus only. For buttons configured in the *ui-extension.xml* file, `showInToolbar` is not applicable, as the configuration is meant for adding a toolbar button itself. Additionally the button can be added to context menu, hence only `showInContextMenu` is applicable for *ui-extension.xml* buttons. The `showInNewMenu` and `showInTemplateMenu` attributes are applicable for menu items in the **New Asset** menu dropdown only, and are not applicable while setting up role based configuration for *ui-extension.xml* menus in this section.

➡ The rest of the configurable elements specified in "Workspace-config.xml" are for integration with Quark Author Web Edition. A separate configuration guide exists for this purpose.

Quark Publishing Platform Clients — Manual configuration

You can change a variety of configuration options for Quark Publishing Platform clients and Platform XTensions after installation. See the following topics for more information.

Creating and maintaining the log file (Mac OS X only)

The setting to create and maintain the log file is mentioned in the `com.quark.qpp.client.Quark.QPP.Client.config.plist` file. This file can be found here: `~/Library/Preferences\Quark\QPP\{version}`.

To change the location of the log file, define the new log location using the `LogFileName` attribute. The default log location is `~/Library/Logs/Quark Publishing Platform Client.Log`

To change the log file size, define the new size using the `LogFileSize` attribute.

Users can enable and disable the logging of exceptions using the `LogException` attribute.

The supported values are:

- (NO) Disables the exception logging
- (YES) Enables the exception logging

Creating and maintaining the log file (Windows only)

The setting to create and maintain the log file is mentioned in the `Quark.QPP.Client.config` file. This file can be found where the application is installed. Users can enable and disable logging using the `QPSLogIsEnabled` attribute.

The supported values are:

- 0 (NO)
- 1 (YES)

To change the location of the log file, define the new log location using the `QPSLogFile` attribute. The default log location:

- for Quark Publishing Platform client is `AppData\Quark Inc\Quark Publishing Platform\{version}\Logs`
- for Platform Client related operations in QuarkXPress is `AppData\Roaming\Quark Software Inc\QuarkXPress {version}\{version}\Logs`

To change the size of the log file size, define the new size using the `QPSLogFileSize` attribute. The size is defined in `Bytes`. The default value of the Out of the Box config is 5242880 bytes.

Use the `QPSLogFileMode` attribute to specify how the log file is written.

Allowed values are:

- 1 (CreateNew). Specifies that the operating system should create a new file. If the file already exists, a `System.IO.IOException` is thrown.
- 2 (Create). Specifies that the operating system should create a new file. If the file already exists, it will be overwritten.
- 3 (Open). Specifies that the operating system should open an existing file. If the files does not exist, a `System.IO.FileNotFoundException` is thrown..
- 4 (OpenOrCreate). Specifies that the operating system should open a file if it exists, otherwise, a new file should be created..
- 5 (Truncate). Specifies that the operating system should open an existing file. Once opened, the files should be truncated so tht its size is zero bytes.
- 6 (Append). Opens the file if it exists and seeks to the end of the file, or creates a new file.

➡ FileMode - OpenOrCreate (4) allows one to open a file with the following access : FileAccess.Read, FileAccess.Write, FileAccess.ReadWrite, FileAccess.Append. Attempting to read from a file opened with FileMode - Truncate (5) will throw an exception.

The `QPSLogFileShare` provides the option to restrict or share the log file created with other processes. Default value is 1("Read")

Allowed values are:

- 0 (None). Declines sharing of the current file.
- 1 (Read). The default value. Allows opening of the file for reading.
- 2 (Write). Allows opening of the file for writing.
- 3 (ReadWrite). Allows opening of the file for reading or writing .
- 4 (Delete). Allows subsequent deleting of a file.
- 16 (Inheritable). Makes the file handle inheritable by child processes.

To specify the maximum number of backup log files us the `QPSLogFileBackupNumber` attribute.

Creating and maintaining the log file (Quark XML Author for Platform)

The following configuration in the `Quark.CMSAdapters.config` file allows the system integrator or user to set the log file path: `<!-- Defines log file path.--> <add key="LogFilePath" value="%APPDATA%\Quark\XML Author\Logs\CMS Adapter Log.txt"/>`

The setting to create and maintain the log file is mentioned in the `Quark.QPP.Client.config` file. This file can be found in the application folder. Users can enable and disable logging using the `QPSLogIsEnabled` attribute.

The supported values are:

- 0 (NO)
- 1 (YES)

To change the location of the log file, define the new log location using the `LogFilePath` attribute. The log location is `AppData\Quark\XML Author\Logs\CMS Adapter Log.txt`

To change the size of the log file size, define the new size using the `QPSLogFileSize` attribute. The size is defined in `Bytes`. The default value of the Out of the Box config is 5242880 bytes.

Use the `QPSLogFileMode` attribute to specify how the log file is written.

Allowed values are:

- 1 (CreateNew). Specifies that the operating system should create a new file. If the file already exists, a `System.IO.IOException` is thrown.
- 2 (Create). Specifies that the operating system should create a new file. If the file already exists, it will be overwritten.
- 3 (Open). Specifies that the operating system should open an existing file. If the files does not exist, a `System.IO.FileNotFoundException` is thrown..
- 4 (OpenOrCreate). Specifies that the operating system should open a file if it exists, otherwise, a new file should be created..
- 5 (Truncate). Specifies that the operating system should open an existing file. Once opened, the files should be truncated so tht its size is zero bytes.
- 6 (Append). Opens the file if it exists and seeks to the end of the file, or creates a new file.

➡ FileMode - OpenOrCreate (4) allows one to open a file with the following access : FileAccess.Read, FileAccess.Write, FileAccess.ReadWrite, FileAccess.Append. Attempting to read from a file opened with FileMode - Truncate (5) will throw an exception.

The `QPSLogFileShare` provides the option to restrict or share the log file created with other processes. Default value is 1("Read")

Allowed values are:

- 0 (None). Declines sharing of the current file.

- 1 (Read). The default value. Allows opening of the file for reading.
- 2 (Write). Allows opening of the file for writing.
- 3 (ReadWrite). Allows opening of the file for reading or writing .
- 4 (Delete). Allows subsequent deleting of a file.
- 16 (Inheritable). Makes the file handle inheritable by child processes.

Suppressing the accessibility services warning - Mac OS X

By default, an alert indicating that "Accessibility services are not enabled" may display at launch in Quark Publishing Platform Client for Mac OS X. To prevent this warning from displaying:

1. Control+click the Quark Publishing Platform Client application icon and choose **Show Package Contents**. A new window displays.
2. Open the "Info.plist" file in a text editor.
3. Locate the following lines:

   ```
   <key>QPPDisableAccessibilityWarning</key>
   <string>0</string>
   ```

4. Change the zero in the `<string>` element to a 1.

Displaying revision comments

By default, when you display the **View Revisions** dialog box, you must expand each revision to see its revision comments. To make revision comments display automatically on Mac OS X:

1. Navigate to `~/Library/Preferences/Quark/QPP/[QPP Framework Version]`.
2. Open the file named "com.quark.qpp.client.[Application Name].config.plist" in a text editor.
3. Locate the following lines:

   ```
   <key>ExpandAllRevisionComments</key>
   <false/>
   ```

4. Change the `<false/>` element to `<true/>`.

On Windows, open the "[application name].exe.config" file, locate the following line, and change the zero to a 1.

```
<add key ="ExpandAllRevisionComments" value="0"/>
```

Displaying first and last names

You can configure Quark Publishing Platform to show user names in one of three ways:

- `[username]`
- `[username] ([first name] [last name])`
- `[username] ([last name], [first name])`

To change this setting:

- For **Quark Publishing Platform Client for Mac OS X**, open the file `~/Library/Preferences/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist`, locate the following text, and set the `<string>` value to 0 for `[username]`, 1 for `[username] ([first name] [last name])`, or 2 for `[username] ([last name], [first name])`:

```
<key>UserNameFormattingStyle</key>
<string>0</string>
```

- For **Quark Publishing Platform Client for Windows**, open the "Quark Publishing Platform Client.exe.config" file (in the Quark Publishing Platform Client application folder), locate the following text, and set the `value` attribute to `DEFAULT` for `[username]`, `FIRSTNAME_LASTNAME` for `[username] ([first name] [last name])`, or `LASTNAME_FIRSTNAME` for `[username] ([last name], [first name])`.

```
 <add key="UserNameFormattingStyle" value="DEFAULT"/>
```

- For **QuarkXPress for Mac OS X**, open the file `~/Library/Preferences/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist`, locate the following text, and set the `<string>` value to 0 for `[username]`, 1 for `[username] ([first name] [last name])`, or 2 for `[username] ([last name], [first name])`:

```
<key>UserNameFormattingStyle</key>
<string>0</string>
```

- For **QuarkCopyDesk for Mac OS X**, open the file `~/Library/Preferences/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist`, locate the following text, and set the `<string>` value to 0 for `[username]`, 1 for `[username] ([first name] [last name])`, or 2 for `[username] ([last name], [first name])`:

```
<key>UserNameFormattingStyle</key>
<string>0</string>
```

- For **QuarkXPress and QuarkCopyDesk for Windows**, open the "Quark.QPP.client.config" file (in the application folder), locate the following text, and set the `value` attribute to `DEFAULT` for `[username]`, `FIRSTNAME_LASTNAME` for `[username] ([first name] [last name])`, or `LASTNAME_FIRSTNAME` for `[username] ([last name], [first name])`.

```
 <add key="UserNameFormattingStyle" value="DEFAULT"/>
```

Changing the preview font and size (Windows only)

To change the font and size used for attribute values to a font other than the default search palette font and size, add the following key to the `<appSettings>` section of the “Quark Publishing Platform Client.exe.config” file, with the desired font and size in the “value” attribute:

```
<add key="FontName_Text Preview" value="Arial, 18"/>
```

Setting maximum number of assets to be fetched (Windows only)

To change the maximum number of assets to be fetched per collection when the user performs a `Get Asset` command or a `Get Collection` command, add the following key to the `<appSettings>` section of the “Quark Publishing Platform Client.exe.config” file, with the desired number in the “value” attribute:

```
<add key="MaximumAssetFetchPerCollection" value="50"/>
```

➡ The default value is `50`.

Specify whether to use Chunked Encoding (Windows only)

To specify whether or not to use **Chunked Encoding** when transferring files on HTTP, add the following key to the `<appSettings>` section of the “Quark Publishing Platform Client.exe.config” file. Supported values are `0` and `1`.

```
<add key="UseChunkedEncoding" value="0"/>
```

➡ The default value is `0`.

Specify support for lazy loading when searching (Windows only)

To specify support for the lazy loading strategy for searching, add the following key to the `<appSettings>` section of the “Quark Publishing Platform Client.exe.config” file, with the desired strategy in the “value” attribute:

```
<add key="LazyLoadingMode" value="LAZY_LOADING_SCROLLBAR"/>
```

➡ The supported values are:

- `NO_LAZYLOADING`: All assets at once configured by the chunk size.
- `LAZY_LOADING_HYPERLINK`: The results specified are retrieved based on the chunk size, when the results specified by the chunk size are retrieved, a hyperlink would appear to enable fetching the next chunk of results.
- `LAZY_LOADING_SCROLLBAR`: The results specified are retrieved based on the chunk size, the scroll bar would enable fetching the next chunk of results.

➡ The default value is `LAZY_LOADING_SCROLLBAR`.

Setting the lazy loading chunk size (Windows only)

To change the chunk size of the lazy loading search results to be fetched, add the following key to the `<appSettings>` section of the “Quark Publishing Platform Client.exe.config” file, with the desired size in the “value” attribute:

```
<add key="LazyLoadingChunkSize" value="50"/>
```

➡ The default value is `50`.

Setting the service time out values for all remote service references (Windows only)

To change service timeout value for all remote service references, add the following key to the `<appSettings>` section of the “Quark Publishing Platform Client.exe.config” file, with the desired time (in seconds) in the “value” attribute:

```
<add key="ServiceTimeoutInSeconds" value="0"/>
```

➡ The default value is `0`. Use the default value specified in the Platform Server configuration.

Setting the service timeout value for publishing service (Windows only)

To change service timeout value for the publishing service, add the following key to the `<appSettings>` section of the “Quark Publishing Platform Client.exe.config” file, with the desired time (in seconds) in the “value” attribute:

```
<add key="PublishingServiceTimeoutInSeconds" value="600"/>
```

➡ The default value is `600`. A value of `0` denotes that the settings specified by `ServiceTimeoutInSeconds` will be in effect.

Specify the font size of the copy tasting row (Windows only)

To change the font size used for the copy tasting row, add the following key to the `<appSettings>` section of the “Quark Publishing Platform Client.exe.config” file, with the desired font size in the “value” attribute:

```
<add key="CopyTastingRowFontSize" value="25"/>
```

Specify the icon for a file extension (Windows only)

To associate an icon with a specified file extension, add the following key to the `<appSettings>` section of the “Quark Publishing Platform Client.exe.config” file, with the path to the desired icon in the “value” attribute:

```
<add key="icon_<file extension>" value="Path of icon"/>
```

➡ The key specifies the "file extension" and the value specifies the file path..

Controlling password retention (Mac OS X only)

You can configure Quark Publishing Platform clients to remember the user name but not the user password between logins. To configure this option:

- For **Quark Publishing Platform Client**, open the file `~/Library/Preferences/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist`, locate the following text, and change `<false/>` to `<true/>`:
```
<key>RememberPassword</key>
<false/>
```
- For **QuarkXPress for Mac OS X**, open the file `~/Library/Preferences/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist`, locate the following text, and change `<false/>` to `<true/>`:
```
<key>RememberPassword</key>
<false/>
```
- For **QuarkCopyDesk for Mac OS X**, open the file `~/Library/Preferences/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist`, locate the following text, and change `<false/>` to `<true/>`:
```
<key>RememberPassword</key>
<false/>
```

Using Mac clients with a proxy server

If you want Mac OS X clients outside of the firewall to be able to connect to Quark Publishing Platform Server through a proxy server, do the following:

- For **Quark Publishing Platform Client**, open the file `~/Library/Application Support/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist`, locate the following text, and change `<false/>` to `<true/>`:
```
<key>UseProxy</key>
<false/>
```
- For **QuarkXPress for Mac OS X**, open the file `~/Library/Application Support/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist`, locate the following text, and change `<false/>` to `<true/>`:
```
<key>UseProxy</key>
<false/>
```
- For **QuarkCopyDesk for Mac OS X**, open the file `~/Library/Application Support/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist`, locate the following text, and change `<false/>` to `<true/>`:
```
<key>UseProxy</key>
<false/>
```

Using Windows clients with a proxy server

To use a Windows client with a proxy server:

1. Open the "[application name].exe.config" file and locate the following line:

   ```
   <!-- <add key="ProxyAddress" value="http://<proxyname>:<portnumber>"/> -->
   ```

2. Uncomment the line and insert the appropriate proxy details.
3. Save and close the file.

Mirroring the collection hierarchy on Check Out/Get

By default, Quark Publishing Platform clients mirror the collection hierarchy on their local drive when they check out or get an asset. However, you can change this option.

Turning off collection mirroring: Mac OS X

To turn off collection mirroring on Mac OS X:

1. For Quark Publishing Platform Client, open the file `~/Library/Preferences/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.Quark Publishing Platform Client.config.plist`, locate the following text, and change `<true/>` to `<false/>`:

   ```
   <key>MirrorCollectionHierarchy</key>
   <true/>
   ```

2. For QuarkXPress, open the file `~/Library/Preferences/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.QuarkXPress.config.plist`, locate the following text, and change `<true/>` to `<false/>`:

   ```
   <key>MirrorCollectionHierarchy</key>
   <true/>
   ```

3. For QuarkCopyDesk, open the file `~/Library/Preferences/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.QuarkCopyDesk.plist`, locate the following text, and change `<true/>` to `<false/>`:

   ```
   <key>MirrorCollectionHierarchy</key>
   <true/>
   ```

Turning off collection mirroring: Windows

To turn off collection mirroring on Windows:

1. For Quark Publishing Platform Client, open the "Quark Publishing Platform Client.exe.config" file, locate the following key, and set `value` to `0`.

   ```
   <add key="MirrorCollectionHierarchy" value="1"/>
   ```

2. For QuarkXPress, open the "Quark.QPP.Client.config" file, locate the following key, and set `value` to `0`.

   ```
   <add key="MirrorCollectionHierarchy" value="1"/>
   ```

3 For QuarkCopyDesk, open the "Quark.QPP.Client.config" file, locate the following key, and set `value` to `0`.

```
<add key="MirrorCollectionHierarchy" value="1"/>
```

Configuring publishing channels

You can control which publishing channels are supported by Quark Publishing Platform.

Configuring publishing channels: Mac OS X

To configure publishing channels on Mac OS X:

1 For all Mac OS clients, open the following file:

```
~/Library/Preferences/Quark/QPP/[QPP
Framework  Version]/com.quark.qpp.publishing.preferences.v7.plist
```

2 Locate the following key, which specifies the list of publishing channels that are honored by the client, and add or remove channels.

```
<key>EnabledPublishingChannels</key>
<array>
  <string>qxpPdf</string>
  <string>qxpEpub</string>
  <string>qxpAppStudio</string>
  <string>qxpAppStudioPackage</string>
  <string>busDocPdf</string>
  <string>busDocHtml</string>
  <string>busDocQxp</string>
  <string>busDocAppStudio</string>
  <string>busDocAppStudioPackage</string>
  <string>ditaDocPdf</string>
  <string>ditaDocWordRTF</string>
  <string>ditaDocHtml</string>
  <string>collectBusdocForOutput</string>
  <string>collectDitaForOutput</string>
</array>
```

3 Add or remove the desired publishing channels.

Configuring publishing channels: Windows

To configure publishing channels on Windows:

1 For Quark Publishing Platform Client, open the "Quark Publishing Platform Client.exe.config" file and locate the following key, which specifies the list of publishing channels that are honored by the client.

```
<add key="EnabledPublishingChannels"
value="qxpPdf,qxpEpub,qxpAppStudio,qxpAppStudioPackage,
busDocPdf, busDocHtml,busDocQxp,busDocAppStudio,
busDocAppStudioPackage, ditaDocPdf,ditaDocWordRTF,ditaDocHtml,
qxpAppStudio, collectBusdocForOutput, collectDitaForOutput"/>
```

2 Add or remove the desired publishing channels.

3 For QuarkXPress and QuarkCopyDesk, open the "Quark.QPP.Client.config" file and make the same changes.

Configuring Delivery Channels

You can control which delivery channels are supported by Quark Publishing Platform.

Configuring delivery channels: Mac OS X

To configure delivery channels on Mac OS X:

1 For all Mac OS X clients, open the following file:

```
~/Library/Preferences/Quark/QPP/[QPP
Framework Version]/com.quark.qpp.publishing.preferences.v5.plist
```

2 Locate the following lines:

```
<key>EnableDeliveryChannels</key>
<false/>
```

3 Change the `<false/>` element to `<true/>`

4 Locate the following key, which specifies the list of delivery channels that are honored by the client:

```
<key>EnabledDeliveryChannels</key>
<array>
 <string>checkInToSharepoint</string>
 <string>checkInToFileNet</string>
 <string>sendEmail</string>
 <string>checkInToDocumentum</string>
 <string>sendToFileSystem</string>
 <string>sendToFTPServer</string>
</array>
```

5 Add or remove the desired delivery channels.

Configuring delivery channels: Windows

To configure publishing channels on Windows:

1 For all Windows clients, open the following file:

```
~application name.exe.config
```

2 Locate the following lines:

```
<add key ="EnableDeliveryChannels" value="0">
```

3 Change the `0` to a `1`

4 Locate the following line, which specifies the list of delivery channels that are honored by the client:

```
<add key="EnabledDeliveryChannels"
value="checkInToSharepoint,
       checkInToFileNet,
       sendEmail,
       checkInToDocumentum,
       sendToFTPServer,
       sendToFileSystem"/>
```

5 Add or remove the desired delivery channels.

Setting preferences for Quark XML Author for Platform

Setting the Checkout location

To set the Checkout location preference, use the `CheckOutLocation` key under the `<appSettings>` section of the "Quark.CMS.Adapters.config" file. This sets the initial preference on first launch of the application.

To change the Checkout location Go to **File** > **Preferences** > **General** and browse to the location you want to set as the Checkout location.

This preference can be reset in one of the following two ways:

- Reset to Default : User can change the preferences by changing the `CheckOutLocation` key under the `appSettings` section of the "Quark.CMS.Adapters.config" file and and using the `Advanced > Reset to Default` feature.
- Change Configuration: An Administrator updates the files and then shares the updated "Quark.CMS.Adapters.config" file at any location and asks the user to use the `Advanced > Change Configuration` feature to load the new preferences.

Setting the file deletion preference on Save and Close

To set the File Deletion preference, use the `FileDeletionOption` key under the `<appSettings>` section of the "Quark.CMS.Adapters.config" file. This sets the initial preference on first launch of the application.

To change the file deletion preference Go to **File** > **Preferences** > **General** and select one of the following values:

1. Delete without Warning
2. Never Allow Deletion
3. Ask Before Deletion

This preference can be reset in one of the following two ways:

- Reset to Default : User can change the preferences by changing the `FileDeletionOption` key under the `appSettings` section of the "Quark.CMS.Adapters.config" file and and using the `Advanced > Reset to Default` feature.
- Change Configuration: An Administrator updates the files and then shares the updated "Quark.CMS.Adapters.config" file at any location and asks the user to use the `Advanced > Change Configuration` feature to load the new preferences.

Setting the quick search preference on Save and Close

To set the Quick Search preference, use the `QuickSearchOption` key under the `<appSettings>` section of the "Quark.CMS.Adapters.config" file. This sets the initial preference on first launch of the application.

To change the file deletion preference, go to **File** > **Preferences** > **Search** and select one of the following values:

1. Name

2. Content
3. Name and Content

This preference can be reset in one of the following two ways:

- Reset to Default : User can change the preferences by changing the `QuickSearchOption` key under the `appSettings` section of the "Quark.CMS.Adapters.config" file and and using the `Advanced > Reset to Default` feature.
- Change Configuration: An Administrator updates the files and then shares the updated "Quark.CMS.Adapters.config" file at any location and asks the user to use the `Advanced > Change Configuration` feature to load the new preferences.

Setting the display revision comments preference on Save and Close

To set the Display Revision Comments preference, use the `RevisionCommentsDisplayOption` key under the `<appSettings>` section of the "Quark.CMS.Adapters.config" file. This sets the initial preference on first launch of the application.

To change the file deletion preference Go to **File** > **Preferences** > **Search** and select one of the following values:

1. Always
2. Never
3. New Assignments Only

This preference can be reset in one of the following two ways:

- Reset to Default : User can change the preferences by changing the `RevisionCommentsDisplayOption` key under the `appSettings` section of the "Quark.CMS.Adapters.config" file and and using the `Advanced > Reset to Default` feature.
- Change Configuration: An Administrator updates the files and then shares the updated "Quark.CMS.Adapters.config" file at any location and asks the user to use the `Advanced > Change Configuration` feature to load the new preferences.

Manual configuration for QuarkXPress and QuarkCopyDesk XTensions

The QuarkXPress and QuarkCopyDesk XTensions have xml based preferences stored in the `QPPXPressXT.xml` and `QPPXPressXT.xml` files in the application preference folder.

➡ In Windows, the QuarkXPress preference folder is located: `C:\Users\<username>\AppData\Local\Quark\QuarkXPress 10` and the QuarkCopyDesk preference folder is located: `C:\Users\<user name>\AppData\Local\Quark\QuarkCopyDesk 10`

➡ In Mac OS X, the QuarkXPress preference folder is located: `/Volume/users/username/Library/Preferences/Quark/QuarkXPress 10` and the QuarkCopyDesk preference folder is located: `/Volume/users/username/Library/Preferences/Quark/QuarkCopydesk 10`

Most of the preferences are set using the user interface, but setting the location of the Platform menu items is done manually.

To set the location of the Platform menu items:

- Set the `location` attribute in the `PlatformMenuItemsLocation` node, under the `<Basic>` section of the “QPPXPressXT.xml” and "QPPXPressXT.xml" files. Set to one of the following values:

1. `0` File Menu (Default Value)
2. `1` Platform Menu
3. `2` to show in both file and Platform Menu

Managing backups and file storage

You choose the backup software and determine the backup intervals for your Quark Publishing Platform database, your Quark Publishing Platform assets, and essential files, such as your Quark® Job Jackets® files and script files. Quark recommends synchronizing backups for your database, assets, and essential files to avoid inconsistency if you have to restore a backup. Quark also recommends periodic testing to verify that your backups can be restored successfully, if necessary.

If you move your asset repository, follow the instructions in "*Moving Quark Publishing Platform asset repository*."

Backing up Quark Publishing Platform Server

Quark recommends stopping Quark Publishing Platform Server before performing a backup, but stopping Quark Publishing Platform Server is not required. Back up your database, your assets, and essential files (such as your FTS index files) on a separate storage device. Although you can back up your entire "Quark Publishing Platform Server" folder, the most important folders include the following:

- "conf" folder (contains system configuration files, including files edited manually after installation)
- "index" folder (contains the Full Text Search index files)

Backing up your database

The database contains all metadata for Quark Publishing Platform assets.

If you use a Microsoft® SQL database or an Oracle database, use the backup tools and instructions provided with MS-SQL or Oracle.

If you use an embedded HSQL database for your Quark Publishing Platform Server, the database information is stored exclusively in the "database" folder in the "Quark Publishing Platform Server" folder, and you must back up the "database" folder to protect metadata and preserve your workflow configuration. If the "database" folder cannot be recovered, you must identify Quark Publishing Platform assets manually for recovery.

Backing up assets

You specify the software and intervals for backing up your Quark Publishing Platform asset repository.

➡ Asset names are encrypted in the Quark Publishing Platform asset repository.

Backing up Index files (full text search)

Quark Publishing Platform Server indexes all files checked into the database so you can perform a search within the text content of Quark Publishing Platform assets. Quark Publishing Platform Server stores index information for full-text search operations in the "index" folder. The "index" folder is at the root level of your Quark Publishing Platform Server folder and is the default location for the files required for full-text indexes. See "*Full text indexing configuration*" to learn how to change the full-text index storage location by editing "LuceneTextIndexingConfig.properties." If you change the location, back up the new location.

Restoring Quark Publishing Platform Server

If you do not have to restore your Quark Publishing Platform asset repositories, the storage paths to the asset repositories will still be valid after you restore your Quark Publishing Platform database. If you must restore your Quark Publishing Platform asset repositories as well as your database, specify the updated storage location according to the instructions in "*Moving Quark Publishing Platform asset repository*."

For example, if the hard drive fails on the computer running Quark Publishing Platform Server, your latest backup should be stored in a separate location. Re-install Quark Publishing Platform Server according to the instructions in the *Quark Publishing Platform ReadMe*. After you re-install Quark Publishing Platform Server, make sure it is not running before you restore your database, assets, and other files.

Restoring Assets

Try to use the same path you used for the old asset repository. For example, if you replace the hard drive on the computer running Quark Publishing Platform Server, you can copy the asset repository backup to the same location. Even if you need a new computer, try to have the same path (for example, `C:\QPP_Assets`).

If you can search for Quark Publishing Platform assets and you need to see the encrypted file names, view the "File Path" attribute when your search is complete. The master file and all asset revisions are stored in the same storage location. The encrypted asset names follow the same pattern (for example, `34.1.1.1.JPG`). The first number in the encrypted name denotes the asset ID. The second number identifies the version number, the third number identifies the rendition type for previews, and the fourth number identifies the first page of the preview.

➡ If you have to restore hardware, launch Quark Publishing Platform Client and re-establish the link with one or more repositories defined in the **Repository** tab of the **Administration: Storage** screen. Do not delete the storage repository and try to create another one.

Restoring Quark Publishing Platform Server database

If you use a Microsoft SQL database or an Oracle database, use the restore tools and instructions provided with MS-SQL or Oracle. When you re-install Quark Publishing

Platform Server, you can enter the correct MS-SQL or Oracle information during the installation process.

If you use an embedded HSQL database for your Quark Publishing Platform Server, the database information is stored in the "database" folder in the "Quark Publishing Platform Server" folder. After you re-install, the "database" folder will contain default files.

To restore your HSQL database settings:

1. Open your backup "database" folder.
2. Copy the following files to the root level of your newly installed "database" folder in the Quark Publishing Platform Server folder: "qppdb.log," "qppdb.properties," and "qppdb.script."
3. Replace the files, when prompted.
4. Start Quark Publishing Platform Server.

➡ Maintain the hierarchy of folders and files in the "database" folder.

Restoring Full Text Indexes

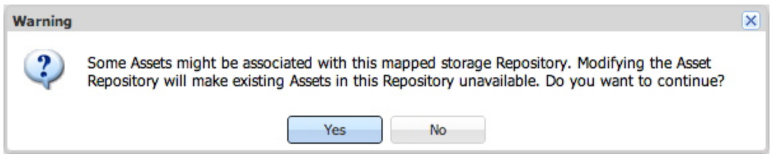
Restore the "index" folder in the location specified in the "LuceneTextIndexingConfig.properties" file.

Moving Quark Publishing Platform asset repository

If you move your Quark Publishing Platform asset repository, you can update the asset repository path with Quark Publishing Platform Client. See "Configuring storage options" in *A Guide to Quark Publishing Platform* for more information about specifying asset storage.

To update the asset repository path:

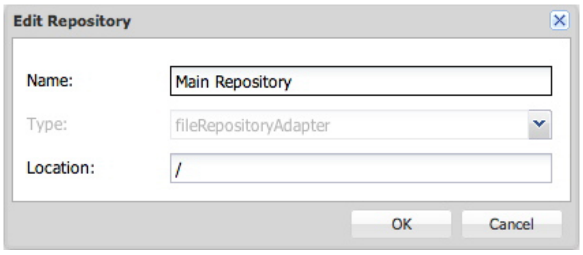
1. In a Web browser, navigate to `http://[server]:[port]/admin` and log on with administration privileges.
2. Display the **Administration** screen and click **Storage**. The **Administration: Storage** screen displays.
3. Display the **Repository** tab, which includes one or more entries in the **Repository Name** column.
4. Select a repository, and choose **Edit** > **Edit Repository**. A warning message displays.



Some Assets might be associated with this mapped storage Repository. Modifying the Asset Repository will make existing Assets in this Repository unavailable. Do you want to continue?

This warning displays when you edit an asset repository

5 Click **Yes**. The **Edit Repository** dialog box displays.



Name: Main Repository
Type: fileRepositoryAdapter
Location: /

The **Edit Repository** dialog box

6 Enter a new location in the **Location** field.

7 Repeat steps 4–6 for every repository listed in the **Repository** tab.

Legal notices

Index