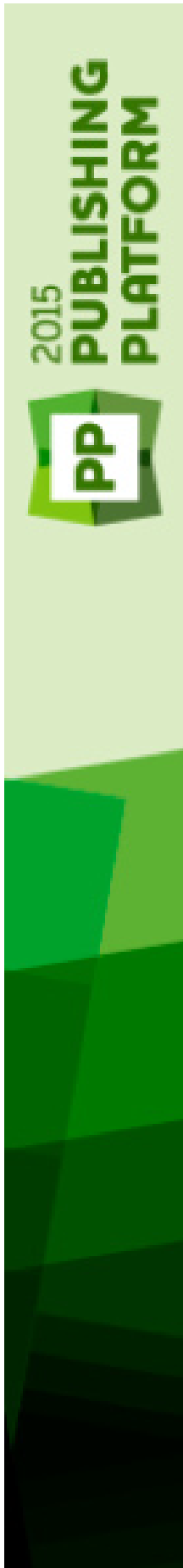


# Quark Publishing Platform 2015 September 2019 Update System Administration Guide



# Contents

<b>Introducing Quark Publishing Platform administration tasks.....</b>	<b>1</b>
<b>Deploying Quark Publishing Platform Server to external server containers.....</b>	<b>2</b>
<b>Setting up environment variables.....</b>	<b>2</b>
<b>Configuring the JVM.....</b>	<b>2</b>
<b>Deploying in external Tomcat .....</b>	<b>2</b>
Requirements and setup .....	3
Setup Tomcat running on Windows.....	3
Server startup and verification .....	5
Disabling the maximum Post size for Apache Tomcat.....	6
<b>Deploying Quark Publishing Platform Server in multi-server deployment .....</b>	<b>6</b>
Configuring cluster server instances or nodes.....	7
Configuring IIS as an HTTP load balancer .....	12
Configuring Apache Web Server as an HTTP load balancer .....	14
<b>Elasticsearch.....</b>	<b>16</b>
Download and install Elasticsearch .....	16
Configuring Platform server to leverage Elasticsearch .....	16
Configuring Elasticsearch for a multi-server environment .....	17
Configuring Elasticsearch for a single server environment.....	17
Run Elasticsearch .....	18
Providing an interface for Elasticsearch .....	18
Verify that Elasticsearch is running .....	19
<b>Lucene.....</b>	<b>19</b>
<b>Setting up the Amazon S3 bucket as a Platform server repository .....</b>	<b>20</b>
Create an S3 bucket in AWS.....	20
Create a policy .....	21
Create a role .....	22
Assign role to EC2 instance .....	22
Register Amazon S3 as a repository adapter in Platform.....	23
<b>Enabling Secure Sockets Layer (SSL) for Quark Publishing Platform Server .....</b>	<b>24</b>
<b>SSL support.....</b>	<b>24</b>
Enabling SSL in Quark Publishing Platform Server .....	24

Configuring the Platform server console to run with an SSL enabled server.....	25
Configuring Platform clients to log on to an SSL enabled Server .....	25
Verifying and using SSL.....	26
<b>Quark Publishing Platform Server — Manual configuration .....</b>	<b>27</b>
<b>Editing “ServerApp.properties” .....</b>	<b>27</b>
<b>Editing “PublishingPool.properties” .....</b>	<b>28</b>
<b>Configuring Quark Publishing Platform Renderer for Quark Publishing Platform usage .....</b>	<b>28</b>
<b>Editing “Qla.properties” .....</b>	<b>29</b>
<b>Extended configuration files.....</b>	<b>29</b>
<b>Configuring user display name in the User Activity panel in admin web client .....</b>	<b>30</b>
<b>Configuring Platform Server to be used for publishing with Quark XML Author - SharePoint Adapter .....</b>	<b>30</b>
<b>Configuring Platform Server to be used for publishing with Quark XML Author - FileNet Adapter .....</b>	<b>30</b>
<b>JVM memory allocation on Windows.....</b>	<b>31</b>
Using Quark Publishing Platform Server Console or Quark Publishing Platform Server Windows service .....	31
Using “Serverstartup.bat” .....	31
<b>Configuring Windows authentication or (SAML and waffle).....</b>	<b>31</b>
Steps to enable SSO in Platform Clients.....	32
Steps to configure Web Client for mixed mode authentication .....	33
<b>Configuring the SAML .....</b>	<b>33</b>
<b>Editing “log4j.xml” .....</b>	<b>39</b>
Adding user information to logged events.....	39
Configuring logging levels.....	39
Logging Messages.....	41
Monitoring database health.....	41
<b>Modifying search notification evaluation settings .....</b>	<b>42</b>
<b>Database properties .....</b>	<b>44</b>
<b>Transformation properties .....</b>	<b>45</b>
<b>Session timeout.....</b>	<b>45</b>
<b>Repository status updater .....</b>	<b>46</b>
<b>Moving Quark Publishing Platform Renderer .....</b>	<b>46</b>
<b>Integrating Quark Publishing Platform with LDAP Directory Servers: .....</b>	<b>47</b>
Managing LDAP profiles .....	47
Importing LDAP users and groups:.....	51
Synch behavior of LDAP users and groups .....	53
Connecting LDAP user passwords with Quark Publishing Platform Server .....	54
<b>Restricting Workspace Browser palettes .....</b>	<b>55</b>
<b>Changing query results settings .....</b>	<b>55</b>
<b>Setting up custom content type detection .....</b>	<b>55</b>
<b>Specifying a default PDF output style .....</b>	<b>56</b>

<b>Controlling delivery channel display settings in Web Client .....</b>	<b>56</b>
<b>Changing case-sensitivity for Quark Publishing Platform passwords .....</b>	<b>57</b>
<b>Managing filters and index service settings .....</b>	<b>57</b>
Index service settings.....	58
ASPOSE filter .....	59
APS filter .....	60
Configure processing of MS Office documents.....	60
Throttling mechanism for Parallel request by OfficeService and ChartingService..	62
QuarkXPress Server Filter .....	63
JAWS filter settings.....	64
XML Author filter settings .....	64
ImageMagick, Jaws, and DITA OT directories.....	65
<b>Full text indexing configuration.....</b>	<b>65</b>
<b>Charting Service.....</b>	<b>66</b>
<b>Integrating QLA with Quark Publishing Platform.....</b>	<b>66</b>
<b>Dynamic configuration .....</b>	<b>66</b>
<b>Enabling IPTC support .....</b>	<b>67</b>
<b>Failover setup .....</b>	<b>67</b>
<b>Encrypting a plain text password .....</b>	<b>71</b>
<b>Enabling forced log off during inactivity .....</b>	<b>72</b>
Configuring WebAdmin to enable forced log off.....	72
Configuring Workspace to enable forced log off .....	73
<b>Configuring messaging .....</b>	<b>74</b>
<b>Configuring clipboard service .....</b>	<b>74</b>
<b>Enable tracing and printing of JVM information .....</b>	<b>75</b>
<b>Configuring location of the publishing rescue folder.....</b>	<b>75</b>
 <b>Quark Publishing Platform Web Client — Manual configuration .....</b>	 <b>76</b>
<b>Configuration overview .....</b>	<b>76</b>
<b>Application level settings.....</b>	<b>76</b>
<b>Multi-Channel preview .....</b>	<b>78</b>
<b>Attributes for General pane.....</b>	<b>79</b>
<b>Role based Toolbar configuration .....</b>	<b>80</b>
<b>Restricting access to Web Client/Admin.....</b>	<b>82</b>
<b>Document save configuration .....</b>	<b>83</b>
Attribute mapping and revision settings.....	83
 <b>Quark Publishing Platform Clients — Manual configuration .....</b>	 <b>86</b>
<b>Creating and maintaining the log file (macOS only).....</b>	<b>86</b>
<b>Creating and maintaining the log file (Windows only) .....</b>	<b>86</b>
<b>Creating and maintaining the log file (Quark XML Author for Platform) .....</b>	<b>87</b>
<b>Suppressing the accessibility services warning - macOS .....</b>	<b>89</b>
<b>Displaying revision comments.....</b>	<b>89</b>
<b>Displaying first and last names .....</b>	<b>89</b>

<b>Changing the preview font and size (Windows only).....</b>	<b>90</b>
<b>Setting maximum number of assets to be fetched (Windows only) .....</b>	<b>90</b>
<b>Specify whether to use Chunked Encoding (Windows only).....</b>	<b>91</b>
<b>Specify support for lazy loading when searching (Windows only).....</b>	<b>91</b>
<b>Setting the lazy loading chunk size (Windows only) .....</b>	<b>91</b>
<b>Setting the service time out values for all remote service references</b>	
<b>(Windows only) .....</b>	<b>91</b>
<b>Setting the service timeout value for publishing service (Windows only) .....</b>	<b>92</b>
<b>Specify the font size of the copy tasting row (Windows only) .....</b>	<b>92</b>
<b>Specify the icon for a file extension (Windows only) .....</b>	<b>92</b>
<b>Controlling password retention (macOS only) .....</b>	<b>92</b>
<b>Using Mac clients with a proxy server .....</b>	<b>93</b>
<b>Using Windows clients with a proxy server .....</b>	<b>93</b>
<b>Mirroring the collection hierarchy on Check Out/Get .....</b>	<b>93</b>
Turning off collection mirroring: macOS .....	93
Turning off collection mirroring: Windows .....	94
<b>Configuring publishing channels .....</b>	<b>94</b>
Configuring publishing channels: macOS.....	94
Configuring publishing channels: Windows .....	95
<b>Configuring Delivery Channels.....</b>	<b>95</b>
Configuring delivery channels: macOS.....	95
Configuring delivery channels: Windows.....	96
<b>Setting preferences for Quark XML Author for Platform .....</b>	<b>96</b>
Setting the Check-out location .....	96
Setting the file deletion preference on Save and Close .....	97
Setting the quick search preference .....	97
Setting the display revision comments preference on Save and Close .....	97
<b>Configuring the Platform Adapters for Microsoft Office components for web sharing .....</b>	<b>98</b>
Configuring the Quark Publishing Platform Adapter for Microsoft Office - Word..	98
Configuring the Quark Publishing Platform Adapter for Microsoft Office - Excel ..	98
Configuring the Quark Publishing Platform Adapter for Microsoft Office - PowerPoint.....	99
Update published ClickOnce deployments .....	99
<b>Manual configuration for QuarkXPress and QuarkCopyDesk XTensions.....</b>	<b>101</b>
<b>Managing backups and file storage .....</b>	<b>103</b>
<b>Backing up Quark Publishing Platform Server .....</b>	<b>103</b>
Backing up your database .....	103
Backing up assets.....	103
Backing up Index files (full text search) .....	103
<b>Restoring Quark Publishing Platform Server .....</b>	<b>104</b>
Restoring Assets.....	104
Restoring Quark Publishing Platform Server database .....	104
Restoring Full Text Indexes .....	104

<b>Moving Quark Publishing Platform asset repository .....</b>	<b>105</b>
<b>Legal notices .....</b>	<b>106</b>

# 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 administering in the Quark Publishing Platform interface, see *A Guide to Quark Publishing Platform*. For information about installing Quark Publishing Platform software, see the *Getting Started with Quark Publishing Platform*.

# Deploying Quark Publishing Platform Server to external server containers

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

### 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 in Tomcat and IBM WebSphere as follows to accommodate Quark Publishing Platform Server:

- Set the minimum Java heap memory to 2048MB.
- Set the minimum PermGenSpace to 512MB.

### 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)" in your software package.

## Requirements and setup

Quark Publishing Platform Server requires JAVA 8 and Apache Tomcat 7.0.77 or above 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

➡ The existing Apache Tomcat installation folder in the steps below is `TOMCAT_HOME`

To deploy Quark Publishing Platform in an external Tomcat installation created by the Tomcat installer or deployed with binaries:

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 Tomcat is installed with the installer, in the `[TOMCAT_HOME]/qpp/publishing/AS-Busdoc.xslt`, `[TOMCAT_HOME]/qpp/publishing/BusDoc2QCD.xslt`, `[TOMCAT_HOME]/qpp/publishing/BusDoc2QXPS.xslt` and `[TOMCAT_HOME]/qpp/publishing/SmartDoc2QXPS.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"/>
```
4. If Tomcat is deployed with binaries, in the `[TOMCAT_HOME]/qpp/publishing/AS-Busdoc.xslt`, `[TOMCAT_HOME]/qpp/publishing/BusDoc2QCD.xslt`, `[TOMCAT_HOME]/qpp/BusDoc2QXPS.xslt` and `[TOMCAT_HOME]/qpp/publishing/SmartDoc2QXPS.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"/>
```
5. In the `[TOMCAT_HOME]/qpp/publishing/SmartDoc2QXPS.xslt` file, update the following paths:
 

```
<xsl:include
```

```
href="../../qpp/publishing/SmartDocTransformations.xslt"/>
<xsl:include
href="../../qpp/publishing/SmartDoc_xref_anchors.xslt"/>
<xsl:include
href="../../qpp/publishing/SmartDoc_StyleSheets.xslt"/>
```

6. 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**.
7. 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 Tomcat and the QuarkXPress Server should not be running on the same port if QuarkXPress Server and Tomcat are running on the same machine.
8. 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.
9. 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
```
  - For 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
```
10. 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"/>
```
  - For SQL Server:
 

```
<import
resource="classpath:com/quark/qpp/common/dao/rdbms/sqlserv
er/SqlServerDaoContext.xml"/>
```
11. Set the following global environment variables on the machine:

- **MAGICK\_HOME** : [Tomcat\_Home]\[Installation path of Platform Server\ImageMagick\bin.
  - **MAGICK\_FILTER\_MODULE\_PATH** : %MAGICK\_HOME%/modules/filters.
  - **MAGICK\_CODER\_MODULE\_PATH** : %MAGICK\_HOME%/modules/coders.
- 12.** 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
```
- 13.** 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
```
- 14.** *If 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
- 15.** *If Tomcat is deployed with binaries:*
- Open the [TOMCAT\_HOME]/bin/catalina.bat file and set the following parameters:
- JRE\_HOME=C:\Program Files\Java\jdk1.8.0\_131JAVA\_OPTS=-server -Xmx2048m -XX:MaxPermSize=128m
  - CATALINA\_OPTS=-Doracle.jdbc.J2EE13Compliant=true -Djava.security.krb5.conf=./qpp/conf/krb5.conf
- 16.** Launch the Tomcat monitor. In the **JAVA** tab, set the following parameters:
- Initial memory pool=1024 MB.
  - Maximum memory pool=2048 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.

### Disabling the maximum Post size for Apache Tomcat

The default maximum size of the Post parameter supported by tomcat is 2MB. If the Post parameter in the HTML of a smart document exceeds this 2MB limit, it causes a null pointer exception at the Server. The user is not disconnected but the HTTP request interceptor on the client side assumes that the unhandled exception is because of a server disconnection.

This issue can be fixed by setting a negative (or appropriate) value for the `maxPostSize` attribute. This negative value will disable the check for max Post size at the Tomcat layer.

1. Open the "server.xml" file located in the `{install_path}/conf` folder.
2. Assign a negative value to the `maxPostSize` attribute of the `<Connector>` tag:

```
<Connector port="61400" redirectPort="61399"
maxHttpHeaderSize="8192" maxThreads="500" minSpareThreads="25"
maxSpareThreads="75" acceptCount="100"
connectionTimeout="30000" disableUploadTimeout="true"
maxPostSize="-1" enableLookups="false" URIEncoding="UTF-8"
server="Quark Publishing Platform Server"/>
```

3. Restart the Quark Publishing Platform Server.

### Deploying Quark Publishing Platform Server in 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 and 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.
  - HA PROXY on Redhat Linux 7.x
- ➡ Only requests routed through an HTTP load balancer are load-balanced among different instances. Requests to a 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 or nodes

The Quark Publishing Platform Server can be deployed as a standalone server or in an external Tomcat using the database on the initial computers, which are called nodes of the Platform multi-server cluster. The first step in setting up cluster servers is to set up a Quark Publishing Platform Server database for Oracle or MS SQL Server.

### Platform Server deployed as a standalone server to work as nodes

On each Quark Publishing Platform Server computer:

1. Open the "ActiveMQ.xml" file located in the `qpp/conf` folder.
2. In the "ActiveMQ.xml" file, uncomment and add the network connectors to integrate the server instance's message queue with one or more server instances in the cluster, where `qpp-node-N` is the name of a computer where another instance of Quark Publishing Platform Server is deployed.

There must be one connector for each node of the cluster except the node itself. In the following example, there is a cluster of three nodes.

Node 1 should contain the following entry:

```
<networkConnectors>
  <networkConnector
    dynamicOnly="true"
    duplex="false"
    name="node1-1"
    networkTTL="1"
    uri="static: (tcp://qpp-Node2:61401) "/>
  <networkConnector
    dynamicOnly="true"
    duplex="false"
    name="node1-2"
    networkTTL="1"
    uri="static: (tcp://qpp-node3:61401) "/>
</networkConnectors>
```

Node 2 should contain the following entry:

```
<networkConnectors>
  <networkConnector
    dynamicOnly="true"
    duplex="false"
    name="node2-1"
    networkTTL="1"
    uri="static: (tcp://qpp-Node1:61401) "/>
```

```
<networkConnector
  dynamicOnly="true"
  duplex="false"
  name="node2-2"
  networkTTL="1"
  uri="static: (tcp://qpp-node3:61401) "/>
</networkConnectors>
```

Node 3 should contain the following entry:

```
<networkConnectors>
  <networkConnector
    dynamicOnly="true"
    duplex="false"
    name="node3-1"
    networkTTL="1"
    uri="static: (tcp://qpp-Node1:61401) "/>
  <networkConnector
    dynamicOnly="true"
    duplex="false"
    name="node3-2"
    networkTTL="1"
    uri="static: (tcp://qpp-node2:61401) "/>
</networkConnectors>
```

➡ The name of the connectors (e.g. node1-1 and node1-2) should be unique throughout the cluster.

3. Open the “context.xml” file located in the `qpp/conf` folder.
4. Update the `<Context reloadable="false" crossContext="true" swallowOutput="true">` line to `<Context sessionCookiePath="/" reloadable="false" crossContext="true" swallowOutput="true">`.
5. Save the file.
6. In the “ActiveMQ.xml” file, uncomment the transport connector(s) to integrate the server instances’ message queues.

This should look like:

```
<transportConnector

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

uri="tcp://${server.machinename}:${jms.openWirePort}?wireFormat.maxInactivityDuration=0"/>
```

7. Open the “web.xml” file of the workspace and admin webapps, located in the `WEB-INF` folder for each webapp and uncomment the `SessionResetFilter` to handle sessions in case of node failure.

```
<filter>
<filter-name>SessionResetFilter</filter-name>
<filter-
class>com.quark.web.security.servlet.SessionResetFilter</filter-
class>
<init-param>
<param-name>jvmRoute</param-name>
<param-value>qpp1</param-value>
</init-param>
<init-param>
```

```
<param-name>sessionCookieName</param-name>
<param-value>JSESSIONID</param-value>
</init-param>
<init-param>
<param-name>sessionInitUrls</param-name>
<param-value>
/workspace/reconnectUser.qsp
/workspace/login.qsp
</param-value>
</init-param>
</filter>
<filter-mapping>
<filter-name>SessionResetFilter</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>
```

8. In the “ServerApp.properties” file located in the [QPP Server]/conf folder, assign a unique value to `server.id.prefix`, identifying the server instance.
9. In the “ServerApp.properties” file located in the [QPP Server]/conf folder, assign the IP of the machine to “server.machinename”, identifying the server instance.

10. In the “server.xml” file located in the [server-path]/conf folder, assign a unique value to the `jvmRoute` attribute of the <Engine> tag.

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

- ➡ The value of `jvmRoute` in the “web.xml” file (the file being edited in step 4) and in the “server.xml” file should be the same.

11. To allow the storage of session cookies at the application root, edit the “context.xml” file located in the [server-path]/conf folder, as follows:

```
<Context sessionCookiePath="/" reloadable="false"
crossContext="true" swallowOutput="true">
```

- ➡ This change is required for workflows involving SOAP based clients like Quark XML Author. For Web based applications like Quark Author Web Edition, the session cookie should be stored at the webapps path, so there is no change needed in the “context.xml” file.

12. Rename the “services” folder, located at  
`{TOMCAT_HOME}/webapps/webServices/WEB-INF`, to “services\_old”.
13. Rename the “serviceArchivesMultiServer” folder, located at  
`{TOMCAT_HOME}/webapps/webServices/WEB-INF`, to “services”.
14. In the “ServerStartup.bat” file located in the server installation folder, add the –  
`Dspring.profiles.active=multiserver` to run the server in cluster mode:

```
java -Xmx2048m -XX:MaxPermSize=256m
-classpath "%JavaClassPath%"
-Dspring.profiles.active=multiserver
-Doracle.jdbc.J2EE13Compliant=true
-Djava.security.krb5.conf=conf/krb5.conf
-Djava.endorsed.dirs="endorsed"
-
Djavax.xml.stream.XMLInputFactory=com.ctc.wstx.stax.WstxInputF
actory com.quark.qpp.Server PluginsContext.xml
```

15. In the “wrapper.conf” file located in the server installation folder, uncomment the following under the Java Additional Parameters section:

```
wrapper.java.additional.7=-Dspring.profiles.active=multiserver
```

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

16. Clear your browser cookies before you start your work.
17. Perform the steps to flush caches across all server instances as described in the following section, *“Configuring server caches flushing mechanism”*.

### Configuring server caches flushing mechanism

Platform Server provides two mechanisms to flush caches across all server instances in case of an update:

- Notification based - This mechanism clears the cache asynchronously. The Notification based cache flushing is accomplished by publishing of a message of type `CACHE` with property `FLUSH_INTERCEPTOR`. This is the default configured mechanism.
- REST request based - This mechanism clears the cache synchronously. The REST based mechanism makes an HTTP request to every peer node to trigger its cache flushing:

```
http://{peerNode}:61400/rest/service/xcache/flush/{interceptorName}. This mechanism is not the default configuration, but you may configure it if needed.
```

In order to configure REST based cache flushing mechanism, make the following changes:

1. Open the “CacheManager.xml” file found in  

```
{install_path}/Server/lib/qpp-server-common-{version}.jar
```
2. Comment the `asyncRemoteMessenger` and uncomment the `synchronousRemoteInvoker`. Inject this bean instead of `asyncRemoteMessenger` in `baseCacheFlushingInterceptor`.  

```
<bean id="synchronousRemoteInvoker"

class="com.quark.qpp.common.caching.SynchronousRemoteInvoker">
  <property name="remoteHosts" value="{server.peer.url}"/>
  <property name="httpContext"
value="{server.cache.remoteFlush.context}"/>
</bean>

<!-- <bean id="asyncRemoteMessenger"
class="com.quark.qpp.core.messaging.service.impl.AsyncRemoteMe
ssenger">
  <property name="messagePublisher" ref="messagePublisher"/>
</bean> -->
```
3. Find the `baseCacheFlushingInterceptor` and inject the `synchronousRemoteInvoker` bean in the `flushinginterceptor`:  

```
<bean id="baseCacheFlushingInterceptor"
class="com.quark.qpp.common.caching.BaseCacheFlushingAdvice"
abstract="true">
  <property name="remoteInvoker"
ref="synchronousRemoteInvoker"/>
</bean>
```
4. Save and close the file.



5. In the “ServerApp.properties” file located in the [QPP Server]/conf folder, uncomment the following:

```
# Uncomment following properties to configure QPP cluster with
synchronous cache update
# Set comma separate URIs of peer server instances which are
part of Cluster except the node itself
server.peer.url=http://{peerHostN 1}:61400,http://{peerHostN
2}:61400

# Http context which may be used to invoke cache flushing
server.cache.remoteFlush.context=rest/service/xcache/flush
```

6. For each node open the “rest-servlet.xml” file located in the [QPP Server]/webapps/rest/WEB-INF folder, uncomment the following:

```
<!-- Following code should be uncommented for synchronous REST
based cache flushing in clustered environment.
This enables "RemoteFlushController" which handles HTTP
requests from remote peers for flushing the cache. -->
<context:component-scan base-
package="com.quark.qpp.common.caching"/>
```

7. Restart the server.

## 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. Open the “ActiveMQ.xml” file located in the qpp/conf folder and add network connectors to integrate the server instance’s message queue with one or more server instances in the cluster, where qpp-node-N is the name of a computer where another instance of Quark Publishing Platform Server is deployed:

➡ Add these network connectors the same way you added them in Step 2 of the section: “*Platform Server deployed as a standalone server to work as nodes*”.

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:${jms.openWirePort}?wireFormat.maxInactiv
ityDuration=0"
updateClusterClients="true"
rebalanceClusterClients="true"
updateClusterClientsOnRemove="true"/>
<transportConnector
uri="tcp://${server.machinename}:${jms.openWirePort}?wireForma
t.maxInactivityDuration=0"/>
```

4. In the “ServerApp.properties” file located in the [QPP Server]/conf folder, assign a unique value to server.id.prefix, identifying the server instance.
5. In the “ServerApp.properties” file located in the [QPP Server]/conf folder, assign the IP of the machine to server.machinename attribute, identifying the server instance.
6. In the “server.xml” file located in the {TOMCAT\_HOME}/conf folder, assign a unique value to the jvmRoute attribute of the <Engine> tag.

```
<Engine name="Catalina" defaultHost="localhost"
```

```
jvmRoute="node1">
```

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

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

8. Rename the “serviceArchivesMultiServer” folder, located at `{TOMCAT_HOME}/webapps/webServices/WEB-INF`, to “services”.
9. Edit the “Catalina.bat” file located in the `{TOMCAT_HOME}/bin` folder. Set the following runtime parameter to run the server in multi-server mode:

```
Set CATALINA_OPTS=-Dspring.profiles.active=multiserver
```

- ➡ If apache tomcat is running as a “service” then add this parameter in the Apache tomcat monitor under the **Java** tab in the **Java option** text field: -  
`Dspring.profiles.active=multiserver`

10. Open the “web.xml” file of the workspace and admin webapps, located in the `{TOMCAT_HOME}/webapps/ WEB-INF` folder for each webapp and uncomment the `SessionResetFilter` to handle sessions in case of node failure as given below:

```
<filter>
  <filter-name>SessionResetFilter</filter-name>
  <filter-
class>com.quark.web.security.servlet.SessionResetFilter</filter-
r-class>
  <init-param>
    <param-name>jvmRoute</param-name>
    <param-value>qpp1</param-value>
  </init-param> <init-param>
    <param-name>sessionCookieName</param-name>
    <param-value>JSESSIONID</param-value>
  </init-param>
  <init-param>
    <param-name>sessionInitUrls</param-name>
    <param-value> /workspace/reconnectUser.qsp
/workspace/login.qsp </param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>SessionResetFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

- ➡ The value of the `<param-value>qpp1</param-value>` should be same as that in the “ server.xml” file of Step 6.

11. 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](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 Platform 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 Platform 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
/qxpsm/*=TomcatBalancer
/qxpsm=TomcatBalancer
/pluginwiris_engine/*=TomcatBalancer
/pluginwiris_engine=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 Website. Select the Website, 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 Website 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
JkMount /pluginwiris_engine/* TomcatBalancer
JkMount /pluginwiris_engine TomcatBalancer
JkMount /proofreader TomcatBalancer
JkMount /proofreader/* 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 IP 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 IP 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.

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`.

### Elasticsearch

A Multi-server deployment can leverage Elasticsearch to avoid regenerating the search index for each application node of the Platform multi-server cluster.

#### Download and install Elasticsearch

1. Download Elasticsearch from the following location:

[www.elastic.co/downloads/elasticsearch](http://www.elastic.co/downloads/elasticsearch)

2. Extract the “elasticsearch(version).zip” file to the file location where you want Elasticsearch installed. (For example: c:\elasticsearch)

➡ All Elasticsearch 5.x versions are supported.

To run Elasticsearch:

1. On Unix: Run `bin/elasticsearch`.

On Windows: Run `bin\elasticsearch.bat`.

2. Run `curl -X GET http://localhost:9200/`

#### Configuring Platform server to leverage Elasticsearch

A multi-server deployment can leverage Elasticsearch to avoid regenerating the search index for each application node of the Platform multi-server cluster.

Perform the following steps on each Quark Publishing Platform Server computer:

1. Open the “ESIndexingConfig.properties” file located in the `[QPP Server]/conf` folder.

2. Specify where Elasticsearch is running:

```
#Comma delimited list of host:tcp_port entries pointing to
Elasticsearch cluster nodes
es.cluster.nodes=platform2k12:9300
```

➡ The default port for Elasticsearch is 9300.

3. Specify the name of the index to be used for attributes:

```
es.attribute.index.name=qps-attribute-index
```

4. Specify the name of the index to be used for text:

```
es.text.index.name=qps-text-index
```

5. In the “ESAttributeIndexSettings.json” file located in the `[QPP Server]/conf` folder, specify the number of shards and replicas required for the index:

```
{ "index":
  { "number of shards":5,
    "number of replicas":1
  }
  ...
}
```

➡ Please see the following tech notes for recommendations on specifying the number of shards and replicas:

- <https://www.elastic.co/blog/how-many-shards-should-i-have-in-my-elasticsearch-cluster>

- <https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started-concepts.html>

## Configuring Elasticsearch for a multi-server environment

When deployed in a multi-server environment, where each server is running its own instance of Elasticsearch, the cluster is run in a Master-Child concept. One instance of Elasticsearch is considered the Master (based on which node was deployed first) and all other instances are considered Child instances. If the Master instance is terminated, one of the Child instances becomes the Master.

To configure each Elasticsearch instance as part of a cluster, perform the following steps for each Elasticsearch deployment:

1. Open the “elasticsearch.yml” file located in the `{Elasticsearch_install_path}/config` folder.
2. Specify the name of the cluster:  
`cluster.name: myESclustername`
3. Specify the name of this node:  
`node.name: nameofnode1`
4. Specify the network host name:  
`network.host: mynetworkhost`
5. Identify the other nodes (instances of Elasticsearch) in your cluster:  
`discovery.zen.ping.unicast.hosts: ["nameofnode2", "nameofnode3"]`
6. On each Quark Publishing Platform Server computer, in the “ESIndexingConfig.properties” file located in the `[QPP Server]/conf` folder, specify the name of the cluster:  
`#name of the Elasticsearch cluster to which this instance of Elasticsearch client will connect to.  
es.cluster.name=myESclustername`

## Configuring Elasticsearch for a single server environment

Elasticsearch is the default search engine for a multi-server profile while Lucene is the default for a single server profile.

To configure Elasticsearch as the default search engine in a single server configuration perform the following steps:

1. Open the “ServerStartup.bat” file located in the server installation folder, and update the following in bold:  
`java -server -Xmx1536m -XX:MaxPermSize=256m -classpath "%JavaClassPath%" -Doracle.jdbc.J2EE13Compliant=true -Djava.security.krb5.conf=conf/krb5.conf -Djava.endorsed.dirs="endorsed" Dspring.profiles.active=multiserver com.quark.qpp.Server PluginsContext.xml`
2. Update the “wrapper.conf” file located in the server installation folder, to add the multiserver profile:  
`wrapper.java.additional.12=-  
Dspring.profiles.active=multiserver`

- ➡ These changes do not go into effect until after restarting the server or Windows service.

### Run Elasticsearch

Running Elasticsearch requires at least Java 8 update 131 or later. The environment variable `JAVA_HOME` should be set appropriately.

Elasticsearch can be started using one of the following methods:

- Using a batch file:

To run Elasticsearch in the foreground as a new process, simply run the “elasticsearch.bat” located in the `/bin` folder.

- As a service:

To run Elasticsearch in the foreground as a new process, simply run the “elasticsearch.bat” located in the `/bin` folder.

To run Elasticsearch as a service, run the “service.bat” located in the `/bin` folder with the following commands:

- Install Elasticsearch as a service by executing the following command:  
`C:\elasticsearch-5.4.0\bin>service.bat install`
- After installation, start the Elasticsearch as a service by executing the following command: `C:\elasticsearch-5.4.0\bin>service.bat start`
- Stop the Elasticsearch service by executing the following command:  
`C:\elasticsearch-5.4.0\bin>service.bat stop`
- Remove the installed Elasticsearch service (and stop the service if started) by executing following command: `C:\elasticsearch-5.4.0\bin>service.bat remove`
- Start a GUI for managing the installed service by executing the following command: `C:\elasticsearch-5.4.0\bin>service.bat manager`

- ➡ For more information, see [www.elastic.co/guide/en/elasticsearch/reference/current/setup-service-win.html#setup-service-win](http://www.elastic.co/guide/en/elasticsearch/reference/current/setup-service-win.html#setup-service-win)

### Providing an interface for Elasticsearch

Elasticsearch does not provide a user interface, so Quark offers a 3rd party plugin to provide one.

Elasticsearch-head is a web-based front-end for an Elasticsearch cluster. (<https://github.com/mobz/elasticsearch-head#running-with-built-in-server>).

To deploy and configure the Elasticsearch plugin:

1. Configure the properties file to allow cross origin:

Add the following to the `\config\elasticsearch.yml` file:

- `http.cors.enabled: true`



- `http.cors.allow-origin: ""`

  2. Download the plugin from <https://github.com/mobz/elasticsearch-head>.
  3. Host the plugin as a Web App in any web container. This may be Platform Server or a separate external web container (e.g. Apache Tomcat).
  4. Once the Web App is up and running, navigate to the plugin UI.
  5. Using the plugin UI, connect to any running Elasticsearch instance, but first ensure that the Elasticsearch instance is running.

To utilize the plugin, perform the following steps for each Elasticsearch deployment:

1. Copy and paste the head plugin to the `{Elasticsearch_install_path}/plugins` folder.
2. To access this user interface:  
`{network host}:9300/_plugin/head/`

### Verify that Elasticsearch is running

1. Go to the Platform server installation folder.
2. Open the Platform server log file "QppServer.log" in the log folder.
3. Look for the following text:

```
"2018-09-24 17:12:12,646 INFO
[com.quark.qpp.searchservice.indexing.ElasticSearchAssetIndexe
r][main][User(Id:0, Name:System)] - Elasticsearch based
TextIndexer Registered.
```

### Lucene

Lucene is the default search engine for a single-server profile while Elasticsearch is the default search engine for a multi-server profile.

To configure Lucene as the default search engine in a multi-server configuration, perform the following steps:


1. Navigate to the `[QPP Server]/lib` folder.
2. Uncompress "qpp-server-textIndexing-elasticsearch-14.2.jar".
3. Open "com\quark\qpp\textindexing\es\impl\ESTextIndexerContext.xml".
4. In tag, change `profile="multiserver"` to `profile="!multiserver"`.
5. Compress the jar back to "qpp-server-textIndexing-elasticsearch-14.2.jar".
6. Uncompress "qpp-server-textIndexing-lucene-14.2.jar"
7. Open  
"com\quark\qpp\textindexing\adapter\impl\LuceneTextIndexerContext.xml"
8. In tag, change `profile="!multiserver"` to `profile="multiserver"`.
9. Compress the jar back to "qpp-server-textIndexing-lucene-14.2.jar".

### Setting up the Amazon S3 bucket as a Platform server repository

This section describes the steps necessary to use Amazon S3 as a repository adapter in Quark Publishing Platform.

#### Create an S3 bucket in AWS

To create an S3 bucket in AWS:

1. Log on to the AWS Console using a valid account and user.
2. On the **AWS Management Console**, under **Storage**, click **S3**.
3. Click **Create bucket**.
4. On the **Name and Region** pane, specify the bucket name and an appropriate region, and click **Next**.
5. On the **Configure Options** pane, check **Keep all version of an object in the same bucket**.  
 This is mandatory as Platform uses S3-provided versioning for asset version management. Also, be sure that you do NOT apply any Lifecycle rules on the bucket which will affect object versions.
6. On the **Set permissions** pane, block all public access to this bucket, and click **Next**.
7. On the **Review** pane, review your selections and click **Create bucket**.
8. Select the newly created bucket and click **Create Folder**.
9. Create an **Assets** folder inside the bucket.

Amazon S3 > qpp-assets-bucket

Overview Properties Permissions

Q Type a prefix and press Enter to search. Press ESC to clear.

Upload Create folder Download Actions Versions Hide Show

☐ Name

Assets

When you create a folder, S3 console creates an object with the above name appended by suffix "/" displayed as a folder in the S3 console. Choose the encryption setting for the object:

☒ None (Use bucket settings)

☐ AES-256  
Use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)

☐ AWS-KMS  
Use Server-Side Encryption with AWS KMS-Managed Keys (SSE-KMS)

Save Cancel

➡ *Optionally an appropriate server-side encryption can be used for data at rest.*

## Create a policy

Create a policy and assign the S3 bucket permissions to the EC2 instance where Platform Server is running.

The following S3 bucket permissions are required by Platform Server:

- s3:PutObject
- s3:GetObject
- s3:GetObjectVersion
- s3:DeleteObjectVersion

To create a policy and assign the permissions:

1. On the **AWS Management Console**, under **Security, Identity, & Compliance**, click **IAM**.
2. Select **Policies** from the **Dashboard** menu.
3. Click **Create policy**.
4. Select the **JSON** tab and create the policy using the following json:

```
{
  "version": "2012-10-17",
  "Statement": [
```

```
{
  "Effect": "Allow",
  "Action": [
    "s3:GetObject",
    "s3:PutObject",
    "s3:GetObjectVersion",
    "s3:DeleteObjectVersion",
  ],
  "Resource": [
    "arn:aws:s3:::qpp-assets-bucket/Assets/*"
  ]
}
```

5. On the **Review policy** pane, provide an appropriate name and description and click **Create policy**.

### Create a role

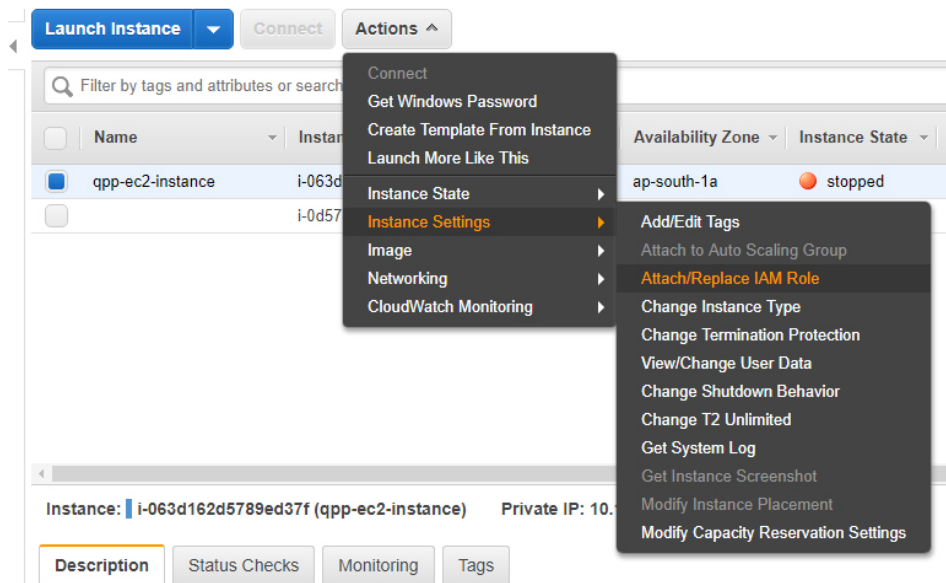
- ➡ If Platform is running an EC2 instance has a role already assigned to it, this step can be skipped by simply attaching the newly created policy to that role.

Create a role and assign your newly created policy to this role:

1. Select **Roles** from the **Dashboard** menu
2. Click **Create role**.
3. Click **AWS service** and select **EC2** as the service that will use this role.
4. Click **Next: Permissions**.
5. Under **Attach permissions policies**, check the box next to the policy you created in the previous step.
6. Click **Next: Tags**.
7. Click **Next: Review**.
8. On the **Review** pane, provide a **Role name**, review your selections and click **Create role**.

### Assign role to EC2 instance

If you created a role using the instructions in the previous sections, attach the role to the Platform EC2 instance:



### Register Amazon S3 as a repository adapter in Platform

1. Open Platform's web-based Admin client.
  2. Under **Administration > Storage**, create the new repository by providing the appropriate bucket name, region and folder.
- ➡ Choose an appropriate region from the list of all regions found here:
  - ➡ <https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.RegionsAndAvailabilityZones.html>

# Enabling Secure Sockets Layer (SSL) for Quark Publishing Platform Server

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

## SSL support

You can configure your application server containers and all Quark Publishing Platform clients to run in secure mode with 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 7.0.77 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 in Quark Publishing Platform Server

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 “server.xml” file located in the `{install_path}/conf` folder.
2. 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" />
```

3. Uncomment the following tag:

```
<Connector port="61399"
```

## ENABLING SECURE SOCKETS LAYER (SSL) FOR QUARK PUBLISHING PLATFORM SERVER

```
maxHttpHeaderSize="8192"MaxThreads="150" minSpareThreads="25"
maxSpareThreads="75"enableLookups="false"
disableUploadTimeout="true"acceptCount="100" scheme="https"
secure="true"clientAuth="false" sslProtocol="TLSv1.2" />
```

4. Replace 61399 with 61400 (or any port on which Tomcat will be listening for secure connections).
5. Save and close “server.xml.”
6. On Quark Publishing Platform Server machine, open the command prompt and execute the following command:

```
%JAVA_HOME%\bin\keytool -genkey -alias tomcat -keyalg RSA
```

7. Provide the details in the prompts. Password should be `changeit`.
  8. Restart the Quark Publishing Platform Server.
  9. Access the Admin webpage: `https://servername:61399/admin`
- ➡ This configuration creates one keystore with a private and public key pair. This is a self-sign certificate.
10. For more information on SSL configuration, including the use of a Certificate Authority, please see the Apache Tomcat SSL information here:

```
url:https://tomcat.apache.org/tomcat-7.0-doc/ssl-howto.html
```

- ➡ 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`.

### Configuring the Platform server console to run with an SSL enabled server

To run the server console with an HTTPS enabled server:

1. Open the “Quark.QPP.ServerConsole.exe” file in the `[QPP Server]/console` folder.
2. Set `useSSL="true"`.
3. In `<add key="serverHost" value="localhost"/>`, enter the server name contained in the SSL certificate.
4. Open the “ServerApp.properties” file in the `[QPP Server]/conf` folder.
5. Change the port number in `webServer.port=61400` to that of the SSL port, typically 443.
6. Restart the console.

### Configuring Platform clients to log on to an SSL enabled Server

macOS 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
```

for El-Capitan and Sierra MAC OS the command is:

```
echo | openssl s_client -connect[server name]:[Port] -  
servername[server name] > [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.
- ➡ Quark Publishing Platform clients running on Windows do not require a server-based SSL certificate.

### 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.
```



# 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 `jms.openWirePort` value to the port number opened for JMS communication via the OpenWire protocol. Java clients such as Quark Publishing Platform Script Manager connect to this port to listen to server notifications.
3. 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.
4. Set the `socketStreaming.port` value to the port number to be used for file transfer (upload/download).
5. 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.additionalnames=[non-default-ip1],[non-default-ip2]`.
6. Set the `server.additionalnames` value to specify the global IP address where the firewall is exposed.
7. Enter the `webServer.port` value you configured for Tomcat (8080, for example).
8. To allow users to log on to Quark Publishing Platform Server even when the directory server is not running, set `authentication.external.cacheTicket = true`.

9. To set the password case sensitivity, set the `server.password.case.sensitive` value to false if case sensitivity is not considered for comparing passwords. This option is not valid if using LDAP for external authentication.
10. To configure a session timeout, set the `session.maxIdle` value to specify the session timeout in seconds, and set the `session.eviction.thread.delay` value to specify the frequency of the session eviction thread in seconds.
11. The Repository status updater is the background thread that runs at a specified frequency to test whether the underlying repository is active or not and to update the status of the same in server database. To configure the repository status updater's sleep interval, set the `repository.status.updator.sleepInterval` value to specify the interval in seconds after which the repository status update thread should run.
12. If the `Realm verif. For Admin. Requests` is set in QuarkXPress Server, the username and password for QuarkXPress Server need to be set in this file. Set the following properties to configure the QuarkXPress Server:
  - Set the `qxps.username` value to specify the QuarkXPress Server username.
  - Set the `qxps.password` value to specify the QuarkXPress Server password.
  - Set the `qxps.locale` value to specify the location of QuarkXPress Server.

### 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 **beans**, **processes** and **publishing channels** should be defined in *.ext* files in the Server Installation *ext* folder, to separate custom extensions and maintain them after a software upgrade.

- 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 user display name in the User Activity panel in admin web client

You can configure Quark Publishing Platform to display the user name in the **User Activity** panel in one of three ways:

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

To change this setting, open the “WebAdminConfig.properties” file in the `[QPP Server]/webapps/admin/WEB-INF/classes` folder. Set the value of the `userNameFormatting` property to one of the following:

- 0 to display as `[username]`.
- 1 to display as `[username] ([first name] [last name])`.
- 2 to display as `[username] ([last name], [first name])`.

### 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.

### 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.

#### Using 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.

#### Using “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”.
3. Search for `java -server -Xmx2048m -XX:MaxPermSize=256m -classpath.2048m`. represents 2048MB 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 or (SAML and waffle)

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.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>
```

### 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" file and uncomment 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. 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" file and change the URL pattern mapping as shown below:

FROM:

```
<filter-mapping><filter-name>SecurityFilter</filter-name><url-pattern>*/</url-pattern></filter-mapping>
```

TO:

```
<filter-mapping><filter-name>SecurityFilter</filter-name><url-pattern>/login.qsp</url-pattern></filter-mapping>
```

4. Save the file.
5. Restart the server and access the corresponding web applications with the following URLs:
  - For access through native Platform user authentication use the URL :  
http://localhost:61400/workspace/Login.jsp
  - For access through Windows authentication use the URL:  
http://localhost:61400/workspace/

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

### Configuring the SAML

## Integration with SAML based Identity Providers

Platform provides integration with SAML 2.0 based Identity Providers for web based SSO so that users from external Identity Providers can be authenticated using SAML 2.0 based login mechanism provided by the underlying Identity Provider.

## Configuring Identity Provider with Platform

### *Specifying Identity Provider metadata*

Open file <Server>/webapps/auth/WEB-INF/saml.xml and locate the following metadata bean:

```
<bean id="metadata"
class="org.springframework.security.saml.metadata.CachingMetadata
Manager">
```

Identity Provider metadata can be provided in either of the following ways:

- **As HTTP URL of the Identity Provider metadata**

Uncomment ExtendedMetadataDelegate bean which takes in HTTPMetadataProvider as the constructor argument and specify the url of the IdP metadata as shown below:

```
<bean
class="org.springframework.security.saml.metadata.ExtendedMetadat
aDelegate">
    <constructor-arg>
        <bean
class="org.opensaml.saml2.metadata.provider.HTTPMetadataProvider"
>
            <constructor-arg>
                <value type="java.lang.String">http://foo.com/idp-
metadata.xml</value>
            </constructor-arg>
        </bean>
    </constructor-arg>
</bean>
...
...
</bean>
```

- **As local XML file containing Identity Provider metadata**

Paste the Identity Provider metadata xml file in folder  
<Server>\webapps\auth\WEB-INF\classes

Uncomment ExtendedMetadataDelegate bean which takes in FilesystemMetadataProvider as the constructor argument and specify the name of the metadata xml file as shown below:



```

<bean
class="org.springframework.security.saml.metadata.ExtendedMetadat
aDelegate">

    <constructor-arg>

        <bean
class="org.opensaml.saml2.metadata.provider.FilesystemMetadataPro
vider">

            <constructor-arg>

                <value type="java.io.File">classpath:idp-
metadata.xml</value>

            </constructor-arg>

        </bean>

    ...

    ...

</bean>

```

### ***Specifying SP entity Id of the metadata***

Specify entity Id of the service provider in entityId property of the metadataGeneratorFilter bean as shown below.

```

<bean id="metadataGeneratorFilter"
class="org.springframework.security.saml.metadata.MetadataGenerat
orFilter">

    <property name="entityId" value="https://platform/auth"/>

    ...

    ...

</bean>

```

### ***Specifying entityBaseURL***

Specify the “entityBaseURL” matching with the front-end URL of the platform’s ‘auth’ webapp. This should be specified in bean metadataGeneratorFilter as shown below:

```

<property name="entityBaseURL"
value="https://platform.server.com/auth"/>

```

This is the endpoint which receives authentication assertions from the identity providers.

**Load balancer configuration**

If there are multiple platform server nodes with a load balancer acting as front end ,the following configuration needs to be added.

Update contextProvider bean, implementation from SAMLContextProviderImpl, to SAMLContextProviderLB as shown below:

Comment out the following bean implementation

```
<bean id="contextProvider"
class="org.springframework.security.saml.context.SAMLContextProviderImpl" />
```

Uncomment following bean implementation by specifying load balancer properties:

```
<bean id="contextProvider"
class="org.springframework.security.saml.context.SAMLContextProviderLB">

    <property name="scheme" value="https"/>

    <property name="serverName" value="myserver.com"/>

    <property name="serverPort" value="443"/>

    <property name="includeServerPortInRequestURL"
value="false"/>

    <property name="contextPath" value="/auth"/>

</bean>
```

Additionally, ensure to specify the entityBaseUrl matching the front-end URL of load balancer in the bean

“org.springframework.security.saml.metadata.MetadataGenerator” as also explained above.

**Mapping username of the authenticated user**

After the user is authenticated by the underlying Identity Provider, SAML data sent in the response is used to map the username of the user in Platform. The SAML assertion provided by the by Identity Provider contains NameId and AttributeStatements. Implementation of bean “samlUserDetailsService” determines whether the username should be mapped from the NameId or one of the AttributeStatements.

For mapping username from the NameId, following implementation should be used:

```
<bean id="samlUserDetailsService"
class="com.quark.qpp.auth.saml.SamlNameIdBasedUsernameMapping">
```

```

        <property name="stripDomainName" value="false" />
    </bean>

```

Whereas, for mapping username from one of the AttributeStatements, following implementation should be used by providing the name of the corresponding attribute:

```

<bean id="samlUserDetailsService"
class="com.quark.qpp.auth.saml.SamlAttributeBasedUsernameMapping"
>

    <property name="stripDomainName" value="false" />

    <property name="attributeName"
value="user.email.id"></property>

</bean>

```

### Key management

SAML messages exchanges involve usage of cryptography for signing and encryption of data. Platform contains a default JKS key store *samlKeystore.jks*, declared in bean *keyManager*, with a sample private certificate usable for test purposes as shown below in file *saml.xml*.

```

<bean id="keyManager"
class="org.springframework.security.saml.key.JKSKeyManager">

<!-- key store file -->

    <constructor-arg value="classpath:samlKeystore.jks" />

<!--Password for keystore -->

    <constructor-arg type="java.lang.String" value="nalle123"
/>

    <constructor-arg>

<map>

    <!-- private keys with alias-password value pairs -
->

        <entry key="apollo" value="nalle123" />

</map>

</constructor-arg>

<!-- Alias of the default certificate -->

    <constructor-arg type="java.lang.String" value="apollo" />

</bean>

```

The keystore file `samlKeystore.jks` contains private key with alias “apollo” which can be used for initial testing, but for security it should be replaced with your own key by using either of the following mechanisms:

- **Self-signed key**

If your Identity Provider does not require keys signed by a specific certification authority, you can generate your own self-signed key using the Java utility *keytool* as shown below:

```
keytool -genkeypair -alias mykey -keypass changeit -keystore samlKeystore.jks
```

The keystore will now contain additional `PrivateKeyEntry` with alias *mykey* which can now be used in the *keyManager* as shown below:

```
<bean id="keyManager"
class="org.springframework.security.saml.key.JKSKeyManager">

    <constructor-arg value="classpath:samlKeystore.jks" />

    <constructor-arg type="java.lang.String" value="nalle123"
/>

    <constructor-arg>
        <map>
            <entry key="mykey" value="changeit" />
        </map>
    </constructor-arg>

    <constructor-arg type="java.lang.String" value="mykey" />
</bean>
```

- **Certification Authority signed key**

Keys signed by certification authorities are typically provided in `.p12/.pfx` format which can be converted and imported to Java keystore using the following command (substituting the passwords and `.pfx` name):

```
keytool -importkeystore -srckeystore ca-signed.pfx -srcstoretype pkcs12 -srcstorepass
your_source_store_password -destkeystore samlKeystore.jks -deststoretype JKS -
deststorepass your_destination_store_password
```

### **Enabling and using Single Sign-On in web clients**

- **Enabling single sign-on in workspace web-client:**

Update file `<Server>webapps/workspace/WEB-INF/web.xml` by enabling single sign-on as shown below:

```
<context-param>

    <param-name>enableSingleSignOn</param-name>

    <param-value>true</param-value>

</context-param>
```

- Enabling single sign-on in admin web-client

Update file <Server>webapps/admin/WEB-INF/web.xml by enabling single sign-on as shown below:

```
<context-param>

    <param-name>enableSingleSignOn</param-name>

    <param-value>true</param-value>

</context-param>
```

### Single sign-on UI in web-clients

After enabling it in the web.xml files, Single Sign-on button will appear on the logon screen of Web clients. For Identity Provider authentication, Single Sign-on button can be used without entering the username and password. Following is the screenshot of how Single sign-on button appears.

## Editing “log4j.xml”

### Adding user information to logged events

You can configure the log4j.xml file to populate each logged event with the user ID and user name:

1. Open the “log4j.xml” file in the {QPP Server}/conf folder.
2. Set the `conversionPattern` parameter:

```
<appender name="QpsServerAllFileAppender"
class="org.apache.log4j.RollingFileAppender">
  <param name="file" value="log/QppServer.log"/>
  <param name="maxFileSize" value="10MB"/>
  <param name="maxBackupIndex" value="10"/>
  <layout class="org.apache.log4j.EnhancedPatternLayout">
    <param name="conversionPattern" value="%d %p [%c] [%t]

[User (Id:%properties{qpp.user.id},Name:%properties{qpp.user.name})] - % m%n"/>
  </layout>
</appender>
```

### 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 “log4j.xml” file in the `{QPP Server}/conf` folder.
2. 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:  

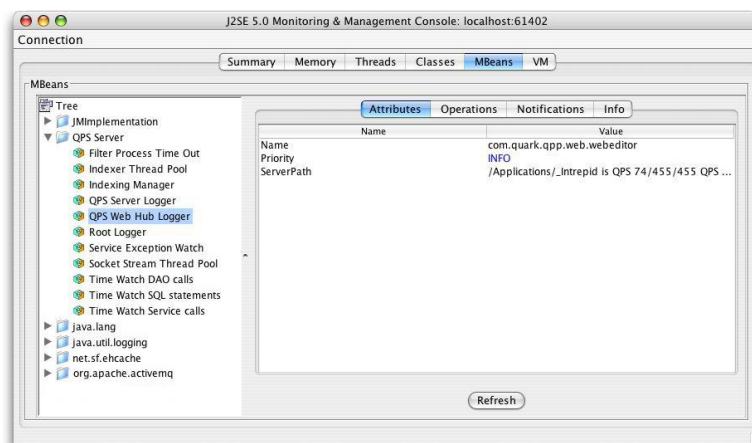
```
<logger name="com.quark.qpp.web.webeditor" additivity="false">
  <level value="INFO" />
  <appender-ref ref="WebHubAsyncAppender" />
</logger>
```
3. To define the logging level for Quark Publishing Platform Server activity, scroll to `<logger name=com.quark.qpp`. The structure is as follows:  

```
<logger name="com.quark.qpp">
  <level value="INFO" />
</logger>
```
4. 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>
```
5. 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.



Use JConsole to adjust logging priority levels

6. To edit the log level of Quark Publishing Platform Server or the Quark Publishing 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.”

### Logging Messages

Use the following appenders in log4j.xml to log various events to a separate log file named `MessagePublisher.log`:

- MessagePublisherAppender
- MessagePublisherAsyncAppender
- MessagePublisherFileAppender

### Monitoring database health

User the following appenders in log4j.xml to log information helpful in monitoring database health in the “DatabaseMonitoring.log” file:

- DatabaseMonitoringAppender
- DatabaseMonitoringAsyncAppender
- DatabaseMonitoringFileAppender

### 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. 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.
4. 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.
5. 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`.
  6. The default display is used when executing a search for which a user specific display does not exist ( for example, when browsing a collection for the first time). To specify the default query display settings, adjust the following properties in the “Default Query display Settings” area:

To specify the default query display settings for asset queries, adjust the following properties:

- To specify the columns to be displayed in asset queries, adjust the `query.display.default.assets.columns` value to contain a list of the columns you want displayed.
- To specify the display mode for asset queries (PLAIN, THUMBNAİL or SNIPPET), adjust the `query.display.default.assets.displaymode` property.
- To specify the explore mode for asset queries (PLAIN, PROJECT, PROJECT AND PAGE, RELATION, RELATED ASSETS, COLLECTIONS), adjust the `query.display.default.assets.exploremode` property.
- To specify the desired grouping of the assets, adjust the `query.display.default.assets.grouping` property.
- To specify the desired sorting of the assets, adjust the `query.display.default.assets.sorting` property.

To specify the default query display settings for collection queries, adjust the following properties:

- To specify the columns to be displayed in collection queries, adjust the `query.display.default.collections.columns` property to contain a list of the columns you want displayed.

- To specify the display mode for collection queries (PLAIN or THUMBNAIL), adjust the `query.display.default.collections.displaymode` property.
  - To specify the desired sorting of the assets in collection queries, adjust the `query.display.default.collections.sorting` property.
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. In the “Database.properties” file located in the `[QPP Server]/conf` folder.
  2. Scroll to the `Database related configuration` section.
  3. 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`.
  4. To specify the database connection URL, replace the path after `qpp.jdbc.url`.
  5. To specify the database user name, replace the `qpp.jdbc.userName` value.
  6. To specify the database user password, replace the `qpp.jdbc.password` value.
  7. To specify the database connection pool size, change the `qpp.jdbc.maxActive` value.
  8. To specify the maximum and minimum numbers of connections that can be idle, change the `qpp.jdbc.maxIdle` and `qpp.jdbc.minIdle` values.
  9. 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.
  10. 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.
  11. To adjust the pooling of prepared statements, change the `qpp.jdbc.poolPreparedStatements` value.
  12. To specify the maximum number of connections to be made in the no-autocommit pool, change the `qpp.jdbc.maxActive_noAutoCommit` value.
  13. Save and close “Database.properties.”
- ➡ If 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 “Transformation.properties” file located in the `[QPP Server]/conf` folder.
2. 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.
3. To specify which file types ImageMagick is allowed to handle, enter a comma-separated, case-insensitive list of file extensions in `imTransformer.extensions`.
4. To specify the temporary file location to be used by ImageMagickTransformer, specify a path for `imTransformer.tempDir`.
5. To specify the ImageMagick command to be executed for generating output with ImageMagickTransformer, enter a value for `imTransformer.jpg.imCommand`.
6. To specify which file types the JawsTransformer is allowed to handle, enter a comma-separated, case-insensitive list of file extensions in `jawsTransformer.extensions`.
7. To specify the temporary file location to be used by JawsTransformer, specify a path for `jawsTransformer.tempDir`.
8. To specify the Jaws command to be executed for generating output with JawsTransformer, enter a value for `jawsTransformer.jpg.command`.
9. 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 “ServerApp.properties” file located in the `[QPP Server]/conf` folder.

2. Scroll to the `session.maxIdle=` entry.
3. Enter the number of seconds a session can be idle before timing out.
4. 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.
5. Save and close “ServerApp.properties.”

➡ Web Client timeout settings can be changed by adjusting the `ajaxTimeout` entry in the “Workspace-Config.xml” file (this entry is in milliseconds). This file is located at `{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 “ManagerConfig.xml” file located in the `[QPP Server]/conf` folder.
2. Scroll to the `<connectioninfo>` section at the bottom of “ManagerConfig.xml.”
3. Change the `name` entry value to the IP address or hostname of the new Quark Publishing Platform Renderer.
4. Change the `port` entry value to the port number you specified for the new Quark Publishing Platform Renderer.
5. Save the “ManagerConfig.xml” file and launch Quark Publishing Platform Server.
6. 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]/qxpsadmin` 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 Directory Servers:

Many system administrators use directory services to manage users on an enterprise network, such as Lightweight Directory Application Protocol (LDAP). Quark Publishing Platform administrators are not required to use LDAP, but if they rely on LDAP for other systems, then Quark Publishing Platform can be configured so that Quark Publishing Platform users can log on to Quark Publishing Platform Server with the same domain username and password they use for other systems on their enterprise network.

- ➔ Quark Publishing Platform Server works with LDAP v3 supported by this implementation.

## Managing LDAP profiles

Before you can import LDAP users/groups into Quark Publishing Platform, you must create an LDAP profile to bind to your LDAP Server. An LDAP profile lets you supply the LDAP-Server credentials along with other details that are necessary to allow import of LDAP users.

To create an LDAP profile:

1. Click **User Profiles**.
2. Click **Manage LDAP Profiles**. The **Manage LDAP Profiles** dialog box displays.

Server Name	Port	Scheme
india-dc1.india.quark.com	389	LDAP

Manage LDAP Profiles dialog box

3. Under the **LDAP Profiles** list pane, click the + button to create a new LDAP profile.
4. Enter a name for the LDAP profile.
5. Choose the LDAP server's authentication method from the **Authentication Type** drop-down menu. Valid values include **Kerberos**, **Digest-MD5**, and **Simple**.
  - ➔ Most of the LDAP servers running on Windows like Microsoft Active Directory LDAP Server use Kerberos authentication, but LDAP servers running on Linux/Solaris like Open Ldap Server or IBM Directory Server use Simple Authentication.
6. Enter the realm name in the **Realm** field.
  - ➔ Each profile must have a unique realm name.
7. Enter the user name and password of a user who has read access to the Directory Server (that is, the designated name in the Directory Server). This user authentication will be used to retrieve the list of users from the Directory Server.
8. To add an LDAP server to the profile, click the + button under the **Servers** list.
9. Enter the Server Name, Port and Scheme for the new server.
10. Click **Apply**.
 

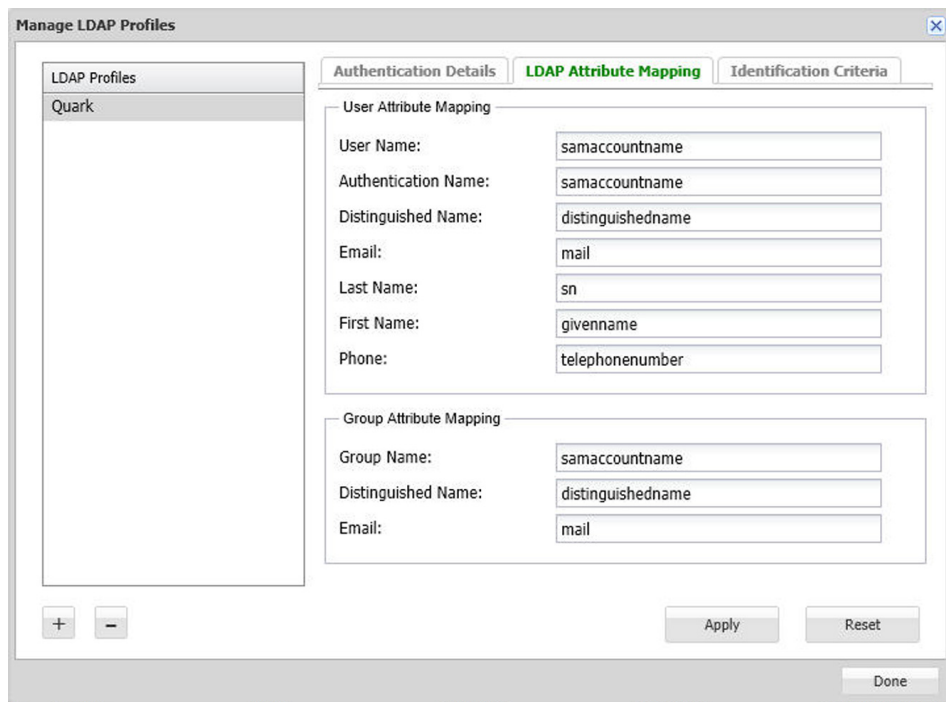
The **Identification Criteria** tab displays requesting that you fill in the value for **Base DN** (A base dn is the point from where a server will search for users). For more information, see "[Identification criteria](#)".
11. Click **Done**.

### Ldap Attribute Mapping:

Before you can import LDAP users into Quark Publishing Platform, you must map the LDAP attributes to Quark Publishing Platform user's attributes.

*To map LDAP user attributes:*

1. Click **User Profiles**.
2. Click **Manage LDAP Profiles**. The **Manage LDAP Profiles** dialog box displays. Select the **LDAP Attribute Mapping** tab.



LDAP Attribute Mapping tab of the Manage LDAP Profiles dialog box

- ➔ LDAP attributes need to be mapped to the corresponding attributes of Users and Groups in Quark Publishing Platform. For each field in the tab, the corresponding LDAP attribute name should be specified.

The following 3 attributes are **mandatory** and should be mapped with your existing LDAP Server attributes as explained below:

1. **User Name** : This is the name of the attribute that contains the user name that you see in Quark Publishing Platform Server after import and is used to log on to Platform.

- ➔ For example, in Microsoft windows active directory, the attribute name is **samaccountname**. In Other LDAP servers like Open LDAP Server , IBM Directory Server this value is **uid**.

2. **Authentication Name**: This is the name of the attribute that contains the user name used for authentication in your LDAP server.

3. **Distinguished Name**: This is the name of the attribute that contains the distinguished value or unique identifier of the user in the LDAP server.

- ➔ For example, in Microsoft Windows Active Directory the name of the attribute is **distinguishedname** and it holds the value of user as : **cn=Joginder Singh,ou=users,dc=example,dc=com**. In some other LDAP servers like IBM Directory Server and Open Ldap server on linux/solaris , the attribute used is :uid or uid.

*To map LDAP Group Attributes:*

1. **Group Name**: This is the name of the attribute that contains the name of the group in your LDAP server.

2. **Distinguished Name** : This is the name of the attribute that contains the complete list of distinguished names of the groups in your LDAP server.  
 ➔ For example, in Microsoft Windows Active Directory the name of the attribute is **distinguishedname** and it hold the value of group as :  
**CN=Managers,ou=groups,dc=example,dc=com**. In some other LDAP servers like IBM Directory Server and Open Ldap server on linux/solaris , the attribute used is **CN**
3. **Email** : This is the name of the attribute that contains the value of email in your LDAP server.

## Identification criteria:

Use the identification criteria tab to define the attributes and filters that enable you to define the search criteria and provide more efficient and effective searches. The identification criteria allows you to describe the base dn for any active directory in your organization. A base dn is the point from where a server will search for users.

To describe the LDAP identification criteria:

1. Click **User Profiles**.
2. Click **Manage LDAP Profiles**. The **Manage LDAP Profiles** dialog box displays. Select the **Identification Criteria** tab.

LDAP Attribute Mapping tab of the Manage LDAP Profiles dialog box

3. Select the LDAP profile whose identification criteria you want to set.
4. Fill out the fields:
  - Enter the base string in the **Base DN** field. This string represents the search base used to search for users in the LDAP Server computer's hierarchy. The **Base DN** field is mandatory.



- The **User Identification Criteria** field allows you to filter for specific users in your active directory. The default string represents the search base used to search for all users in the Active Directory. This field allows you to search for categories like *Person*, eliminating machines from showing up in your searches. The **User Identification Criteria** field is optional.
- The **Group Identification Criteria** field allows you to search for specific groups in your active directory. The default string represents the search base used to search for all groups in the Active Directory. The **Group Identification Criteria** field is optional.
- The **Group Member Attribute Name** field denotes the common attribute of the set of users in a group. The **Group Member Attribute Name** field is optional.
- The **Member Identifying Attribute Name** field is used to denote membership in a group. The **Member Identifying Attribute Name** field is optional.
- The **Filters to get all Users of Group** field allows you to specify the criteria needed to search for all the users of a particular group. The **Filters to get all Users of Group** field is optional.
- The **Filters to get all Sub-Groups of Group** field allows you to specify the criteria needed to search for all the sub-groups of a particular group. The **Filters to get all Sub-Groups of Group** field is optional.
- The **Filters to get all Groups of User** field allows you to specify the criteria needed to search for all the groups of a particular user. The **Filters to get all Groups of User** field is optional.

5. Click **Apply**.

### Importing LDAP users and groups:

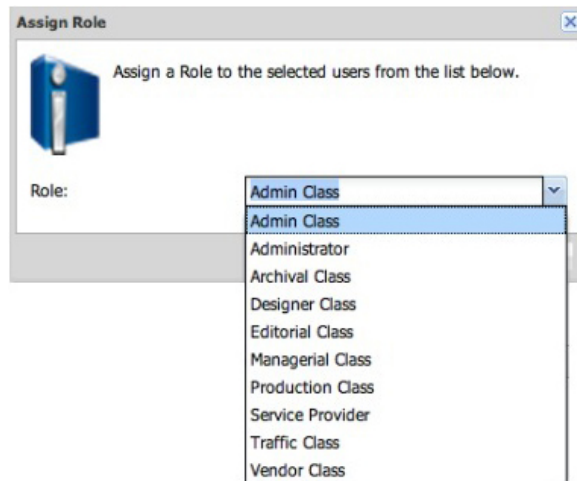
To import users and groups from your LDAP service:

1. Click **User Profiles**.
2. Click **Import Users and Groups From LDAP**. The **Import Users and Groups from LDAP** dialog box displays.

Use the Import Users and Groups From LDAP dialog box to add users or groups to your Quark Publishing Platform list.

3. The search controls at the top of the dialog box let you specify which users or groups you want to be able to import. To create a search:
  - Choose the name of the appropriate LDAP profile from the **Profile Name** drop-down menu.
  - Choose to search **Users** or **Groups**.
  - Enter the search criteria in the **Search Criteria** field. For example, to import all LDAP users that meet the criteria `samaccountname=*`, enter `samaccountname=*`
  - Enter the base string in the **Base String** field. In this string, `ou` is an abbreviation for “organizational unit” and `dc` is short for “domain component.” This string represents the search base used to search for users in the Active Directory server computer’s hierarchy.
  - Click **Save** to save the search and provide a name for the search in the dialog box that displays.
4. Choose the search to execute. The matching users or groups are listed on the left.
5. Select one or more names from the **LDAP Users/Groups** list on the left and click the right arrow to add the selected users or groups to the **Quark Publishing Platform** list of users or groups on the right.

The **Assign Role** dialog box displays.



Use the Assign Role dialog box to assign a role to a Quark Publishing Platform user or group.

6. Choose a role from the **Role** drop-down menu in the **Assign Role** dialog box and then click **OK**.

➔ You can change the search criteria, execute more searches, and continue to add users/groups until you click **OK** in the **Import Users from LDAP** dialog box.

7. Click **OK**. The users or groups in the list on the right are imported.

- ➔ If you duplicate a Quark Publishing Platform user or group that you imported from your LDAP service, the duplicated user or group is not connected to your LDAP service (that is, it's like creating a new user or group).
- ➔ You cannot have two Quark Publishing Platform users or groups with the same name. If you attempt to add a user or group name that is the same as an existing name, you will not be allowed to add the duplicate name.
- ➔ When you import a group, all users in that group will be imported. If the group you imported contains sub-groups, the sub-group and all users within that sub-group will also be imported. When you assign a Role to a group, all users in that group will be assigned that role. If you import a group from LDAP, you cannot add users to that group using Quark Publishing Platform, you must add users to these groups using active directory.
- ➔ A Quark Publishing Platform administrator must have the “**Manage LDAP Users**” privilege for the **Import Users from LDAP** functionality to be available.
- ➔ If you combine users into groups, you can route an asset to a group, and any user in that group can check out the asset and work on it. To create and delete groups, click **Groups**.

### Synch behavior of LDAP users and groups

To configure the LDAP synchronization time delay:

1. Open the “ServerApp.properties” file located in the [QPP Server]/conf folder.
2. Set the value for `ldap.synchronization.thread.delay=` to `true`.

### 3. Save and close “ServerApp.properties.”

Each synch with the active directory, either manual or automated, synchronizes all users and groups imported via LDAP from the active directory server to Platform Server.

### Synch of existing groups

If a group exists in the active directory, then the following actions occur during the synch process:

- Group metadata is updated (e.g. email addresses...).
- Subgroups are updated.
- Metadata for existing subgroups is updated.
- Missing users of existing subgroups are imported.
- Missing subgroups along with users are imported.
- Missing users are imported.

### Synch of deleted groups

When an imported group is deleted from the active directory server, then this group will be ignored during the synch process and will remain in the same state without any metadata update.

### Synch of existing users

If a user exists in the active directory, then user information like first name, last name, email address, and phone number will be updated.

### Synch of deleted users

If an imported user is deleted from the active directory server, then this user will be ignored during the synch process and will remain in the same state without any metadata update.

## 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 “ServerApp.properties” file located in the [QPP Server]/conf folder.
2. 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).
3. To indicate that it is required to evaluate the “CollectionInfo#isAccessibleChildrenAvailable()” flag at runtime for every collection in context, the value of `collections.evaluateAccessibleChildrenFlag=` should be `true` (default).
4. Once an element is fetched from the LDAP server, it will be cached in the Platform Server and provided from the cache for a specified number of seconds, and fetched again after that time. To specify time to live (in seconds) for LDAP cache elements, set the value for `ldap.cache.timetolive=`.
5. To specify the LDAP synchronization time delay (in seconds), set the value for `ldap.synchronization.thread.delay=` to `true`. If LDAP synchronization is not required, set the value to -1.
6. Save and close “ServerApp.properties.”

### Restricting Workspace Browser palettes

To restrict the number of **Workspace Browser** windows a user can have open at one time:

1. Open the “Query.properties” file located in the [QPP Server]/conf folder.
2. Specify a value for the `query.maxWatchedQueryCountPerSession` parameter.
3. Save and close “Query.properties”.

### Changing query results settings

- To limit the number of results that will be returned for an *Assignments* based search, specify a positive non-zero value for the `query.maxAssignmentFetchCount` parameter.
- To limit the number of results that will be returned for a *User* based search, specify a positive non-zero value for the `query.maxRowFetchCount` 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, RTE, 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-mimetypes-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 “ServerApp.properties” file located in the [QPP Server]/conf folder.
2. If you want to specify case-sensitivity for user passwords, set `server.password.case.sensitive` to `true`. Otherwise, enter `false`.
3. 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:

- The “AsposeFilterServiceConfig” file provides previews and thumbnails of Word, RTF, EXCEL, POWER POINT and plain text files.
- The “QxpsFilterServiceConfig.properties” file provides previews and thumbnails of QuarkXPress projects and QuarkCopyDesk articles.
- The “IMFilterServiceConfig.properties” file provides previews and thumbnails of most image files.
- The “JawsFilterServiceConfig.properties” file provides previews and thumbnails of certain image files — PDF files, EPS files, and Adobe® Illustrator® files.
- The “XADocFilterServiceConfig.properties” file 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 “Indexing.properties” file located in the `[QPP Server]/conf` folder.
  2. To specify the maximum number of concurrent indexing threads that operate in the background, adjust the `indexingThread.pool.maxActive=` value.
  3. 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.
  4. To specify the maximum number of idle threads in the pool, adjust the `indexingThread.pool.maxIdle=` value.
  5. To specify the minimum number of idle threads in the pool, adjust the `indexingThread.pool.minIdle=` value.
  6. To specify the number of milliseconds a thread can be in the pool before it is evicted, adjust the `indexingThread.pool.minEvictableIdleTimeMillis=` value.
  7. 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.
8. To index assets not previously indexed when Quark Publishing Platform Server starts, enter `true` for the `indexing.indexPendingAssetsOnStartup=` value.
  9. Save and close “Indexing.properties.”



## ASPOSE filter

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

1. Open the “AsposeFilterServiceConfig.properties” file located in the [QPP Server]/conf folder.
2. `asposefilter.previewType=png`. Supported preview types are **png** and **jpg**.
3. `asposefilter.generatePreview=true`.
4. `asposefilter.generateThumbnail=true`.
5. `asposefilter.generateTableImages=false`.
6. `asposefilter.generateChartImages=false`.
7. `asposefilter.thumbnailHeight= entry`.
8. `asposefilter.thumbnailWidth= entry`.
9. `asposefilter.previewDpi= entry`.
10. `asposefilter.reditionPreviewDpi= entry`.
11. 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).
12. To specify the maximum number of pages for which preview should be generated, enter a value in the `asposefilter.previewPages= entry`.
13. To set the preview page layout margin settings for RTF, XML, HTML, HTM and TXT type files enter values in the following:
  - `asposefilter.text.topMargin= entry`.
  - `asposefilter.text.bottomMargin= entry`.
  - `asposefilter.text.rightMargin= entry`.
  - `asposefilter.text.leftMargin= entry`.
14. Save and close “AsposeFilterServiceConfig.properties.”

## Configure ASPOSE filter for PDF preview

The Platform Server provides two filter implementations for the indexing of PDF Documents:

- Jaws Filter
- Aspose Filter

By default the JAWS Filter is configured for PDF document indexing. In order to configure the ASPOSE Filter for PDF document indexing, do the following:

1. Open the “AsposeFilterServiceConfig.properties” file located in the [QPP Server]/conf folder.
2. Look for and uncomment the following:
 

```
<!-- <supported-file-type>
```

```
<mac-os-type/>
<file-extension>pdf</file-extension>
<mime-type/>
<creator-code/>
</supported-file-type>
<supported-file-type>
  <mac-os-type/>
  <file-extension/>
  <mime-type>application/pdf</mime-type>
  <creator-code/>
</supported-file-type>
<supported-file-type>
  <mac-os-type/>
  <file-extension/>
  <mime-type>application/x-pdf</mime-type>
  <creator-code/>
</supported-file-type> -->
```

3. Open “JawsFilterServiceConfig.properties.” in a text editing application.
4. Look for and comment out the following supported file type:

```
<supported-file-type>
  <mac-os-type/>
  <file-extension>pdf</file-extension>
  <mime-type/>
  <creator-code/>
</supported-file-type>
```

5. Save and close “AsposeFilterServiceConfig.properties.”
6. Save and close “JawsFilterServiceConfig.properties.”

## APS filter

Adjust the APS filter settings:

1. Open the “ApsFilterServiceConfig.properties” file located in the [QPP Server]/conf folder.
2. `apsfilter.generatePreview=true.`
3. `apsfilter.generateThumbnail=true.`
4. `apsfilter.generateAttributes=true.`
5. `apsfilter.generateText=true.`
6. To specify the time the process should wait before exiting, if indexing is not done, enter a value in the `apsfilter.processTimeOut=` entry (in milliseconds).
7. To specify the maximum number of pages for which preview should be generated, enter a value in the `apsfilter.previewPages=` entry.
8. Save and close “ApsFilterServiceConfig.properties.”

## Configure processing of MS Office documents

Adjust the MS Office filter settings:

1. Open the “OfficeServiceConfig.properties” file in the [QPP Server]/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.
7. To set the option to remove or hide the hidden columns/rows from the output formats HTML, XHTML, SMARTTABLE and CALS, set the following properties:
  - `office.defaultHiddenColumnDisplayType=` entry. The allowed values for this option are `hide` or `remove`.
  - `office.defaultHiddenRowDisplayType=` entry. The allowed values for this option are `hide` or `remove`.
8. Set the `office.defaultHorizontalResolutionoffice.ppt.document.defaultImageFormat` property to set the default image format for PowerPoint document output.
9. Set the `office.defaultHorizontalResolutionoffice.ppt.document.image.defaultHorizontalScalingFactor` property to set the default horizontal scaling factor for PowerPoint document output. This property is applicable for JPEG, GIF, PNG and BMP image formats.
10. Set the `office.defaultHorizontalResolutionoffice.ppt.document.image.defaultVerticalScalingFactor` property to set the default vertical scaling factor for PowerPoint document output. This property is applicable for JPEG, GIF, PNG and BMP image formats.
11. Set the `office.defaultHorizontalResolutionoffice.ppt.component.defaultImageFormat` property to set the default image format for PowerPoint document's slide output.
12. Set the `office.defaultHorizontalResolutionoffice.ppt.component.image.defaultHorizontalScalingFactor` property to set the default horizontal scaling factor for PowerPoint document's slide. This property is applicable for JPEG, GIF, PNG and BMP image formats.
13. Set the `office.defaultHorizontalResolutionoffice.ppt.component.image.defaultVerticalScalingFactor` property to set the default vertical scaling factor for PowerPoint document's slide. This property is applicable for JPEG, GIF, PNG and BMP image formats..

14. Set the `office.defaultHorizontalResolutionoffice.visio.document.defaultImageFormat` property to set the default image format for Visio document output.
15. Set the `office.defaultHorizontalResolutionoffice.visio.document.image.defaultResolution` property to set the default resolution of the image for Visio document output. This property is applicable for JPEG, GIF and PNG image formats.
16. Set the `office.defaultHorizontalResolutionoffice.visio.component.defaultImageFormat` property to set the default image format for Visio document's page output.
17. Set the `office.defaultHorizontalResolutionoffice.visio.component.image.defaultResolution` property to set the default resolution of the image for Visio document's page. This property is applicable for JPEG, GIF and PNG image formats.

### Throttling mechanism for Parallel request by OfficeService and ChartingService

To configure the OfficeService and ChartingService to do parallel requests change the two properties files :

- "ChartingPool.properties"
- "OfficePool.properties"

#### Configuring "ChartingPool.properties"

1. Open the "ChartingPool.properties" file located in the `[QPP Server]/conf` folder
2. Set the `chartingThread.pool.maxActive` property to specify the maximum number of threads that can be active at any given moment.
3. Set the `chartingThread.pool.whenExhaustedAction` property to specify what should be done if the pool size has already been reached.
  - Set to `0` to return immediately throwing a `NoSuchElement` exception.
  - Set to `1` to block for some time depending on value of `maxWait` for an available thread.
  - Set to `2` to increase the size of the pool and return a new thread instance.
4. Set the `chartingThread.pool.maxWait` property to specify the amount of time in milli-seconds for which to wait for an available thread. Set to `-1` to wait indefinitely.
5. Set the `chartingThread.pool.maxIdle` property to specify the maximum no of idle threads in the pool.

6. Set the `chartingThread.pool.minIdle` property to specify the minimum no of idle threads in the pool.
7. Set the `chartingThread.pool.minEvictableIdleTimeMillis` property to specify the minimum time in milli-seconds for a thread to be in the pool before it can be evicted.
8. Set the `chartingThread.pool.timeBetweenEvictionRunsMillis` property to specify the time in milli-seconds after which the evictor thread should run to remove idle thread. Set to `-1` to indicate Never.

### Configuring "OfficePool.properties"

1. Open the "OfficePool.properties" file located in the `[QPP Server]/conf` folder
2. Set the `officeThread.pool.maxActive` property to specify the maximum number of threads that can be active at any given moment.
3. Set the `officeThread.pool.whenExhaustedAction` property to specify what should be done if the pool size has already been reached.
  - Set to `0` to return immediately throwing a `NoSuchElementException`.
  - Set to `1` to block for some time depending on value of `maxWait` for an available thread.
  - Set to `2` to increase the size of the pool and return a new thread instance.
4. Set the `officeThread.pool.maxWait` property to specify the amount of time in milli-seconds for which to wait for an available thread. Set to `-1` to wait indefinitely.
5. Set the `officeThread.pool.maxIdle` property to specify the maximum no of idle threads in the pool.
6. Set the `officeThread.pool.minIdle` property to specify the minimum no of idle threads in the pool.
7. Set the `officeThread.pool.minEvictableIdleTimeMillis` property to specify the minimum time in milli-seconds for a thread to be in the pool before it can be evicted.
8. Set the `officeThread.pool.timeBetweenEvictionRunsMillis` property to specify the time in milli-seconds after which the evictor thread should run to remove idle thread. Set to `-1` to indicate Never.

### QuarkXPress Server Filter

To adjust QuarkXPress filter settings:

1. Open the "QxpsFilterServiceConfig.properties" file located in the `[QPP Server]/conf` folder.
2. Set the `qxpsfilter.useSpreadForArticles` value to `true` to specify that previews for articles are generated for each spread.
3. Set the `qxpsfilter.useSpreadForProjects` value to `true` to specify project previews with spreads.

4. 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`.
5. 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`.
6. 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.
7. 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 `.5`, `2`, `3`, and `6` are 50 percent, 200 percent, 300 percent, and 600 percent, respectively.
8. 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.
9. 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 “JawsFilterServiceConfig.properties” file located in the `[QPP Server]/conf` folder.
2. 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.
3. Save and close “JawsFilterServiceConfig.properties.”

### XML Author filter settings

The XML Author filter is part of the configuration for XML files. XADocFilter is used to generate Previews and Thumbnails of XML files. To specify settings for the XML Author filter:

1. Open the “XADocFilterServiceConfig.properties” file located in the `[QPP Server]/conf` folder.
2. To specify the amount of time a process will wait for indexing (with XADocFilter) to complete, adjust the `xaDocfilter.processTimeOut=` value.

3. To specify the preview height and width, adjust the `xaDocfilter.previewHeight= value` and the `xaDocfilter.previewWidth= value`.
4. To specify the maximum number of pages for which a preview should be generated, adjust the `xaDocfilter.previewPages= value`.
5. Save and close “XADocFilterServiceConfig.properties.”

### ImageMagick, Jaws, and DITA OT directories

By default, ImageMagick, Jaws, and 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 “ServerApp.properties” file located in the `[QPP Server]/conf` folder.
2. Scroll to the `IMAGE_MAGICK_HOME=` entry.
3. Enter the path to your existing ImageMagick bin folder.
4. Scroll to the `JAWS_HOME=` entry.
5. Enter the path to your existing Jaws bin folder.
6. Scroll to the `DITA_HOME=` entry.
7. Enter the path to your existing DITA Open Toolkit folder.
8. Save and close “ServerApp.properties.”

### Full text indexing configuration

To configure FTSIndex filter settings:

1. Open the “LuceneTextIndexingConfig.properties” file located in the `[QPP Server]/conf` folder.
  2. 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.
  3. 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.
4. To specify the number of changes after which an index compaction should be performed, set a value for `lucene.maxModificationWithoutOptimisation`.

5. Save and close “LuceneTextIndexingConfig.properties.”

### Charting Service

To configure the properties 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:
  - `charting.defaultImageFormat=png`
  - `charting.defaultWidth=600`
  - `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 “Qla.properties” file located in the `[QPP Server]/conf` folder.
  2. Enter the current IP address or hostname of the computer where you installed QLA in the `QlaServer.machinename=` parameter.
  3. Enter the port number in the `QLAServer.port=` parameter.
  4. 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.
  5. 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.
6. Save and close “Qla.properties.”

### Dynamic configuration

The Configuration service is used to manage publishing configuration:

- Adding/updating new publishing channels
- Adding/updating new publishing process
- Adding/updating new MIME types for content type



The Configuration service is responsible for the following:

- Managing the configuration files.
- Capturing updates.
- Reinitializing corresponding bean definitions, avoiding restarts.
- Using the reinitialize API to avoid restarts even after changing the configuration files manually on the server .

In a multi-server environment, each node in a cluster must have access to a single .ext folder and the shared location must be appended to the classpath system environment variable.

### Creating a custom XML mime-type

```
https://http://localhost:61400/rest/service/config/xmlmimetypes?
op=create&mimetype=application/xml;
format= researchreport&parentmimetype=application/xml;
format= smartcontent&xpath =/*[local-name()='section']/@type='
researchreport
```

### Mapping the XML mime-type to a content type in Platform

```
https://http://localhost:61400/rest/service/config/xmlmimetypes?
op=mapcontenttype&contenttype=Research
Report&mimetypes=application/xml;
format=researchreport
```

### Creating Publishing Channels

```
https://http://localhost:61400/rest/service/config/publishingchan
nels?
op=create&publishingchannellist=<publishingChannelList><publishin
gChannelInfoId="researchReportPdf"
```

### Defining Publishing Channels for Asset indexing

```
https://http://localhost:61400/rest/service/config/publishingchan
nels?
contenttype=Research Report&loginname=Admin&loginpassword=Admin
```

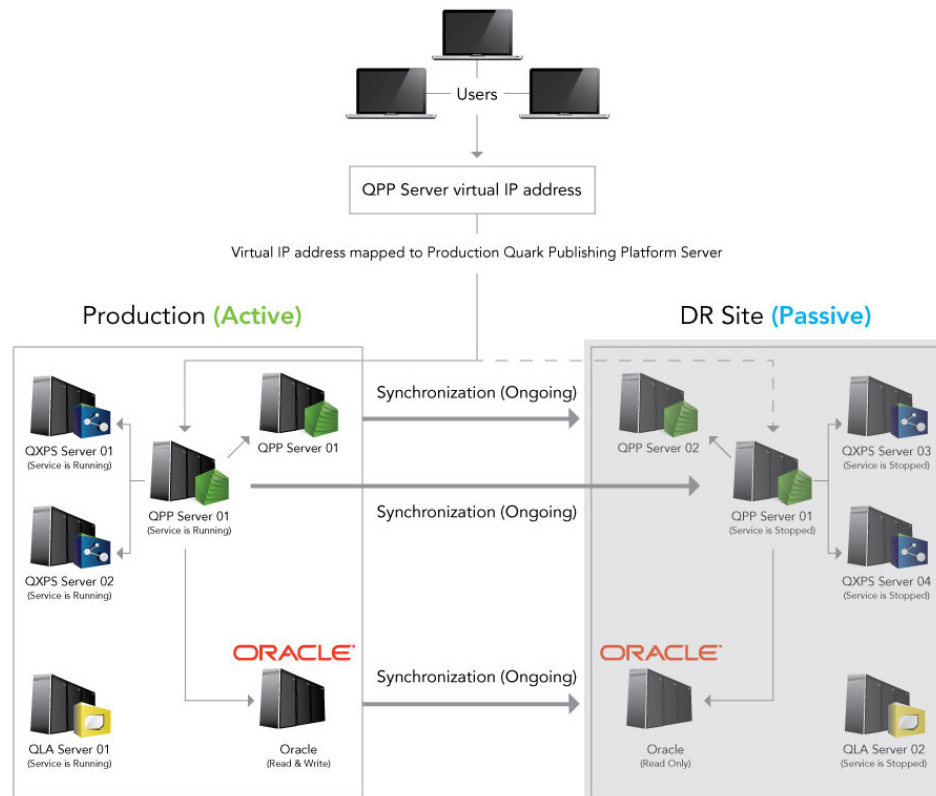
## Enabling IPTC support

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`.

## Failover setup

This topic 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.



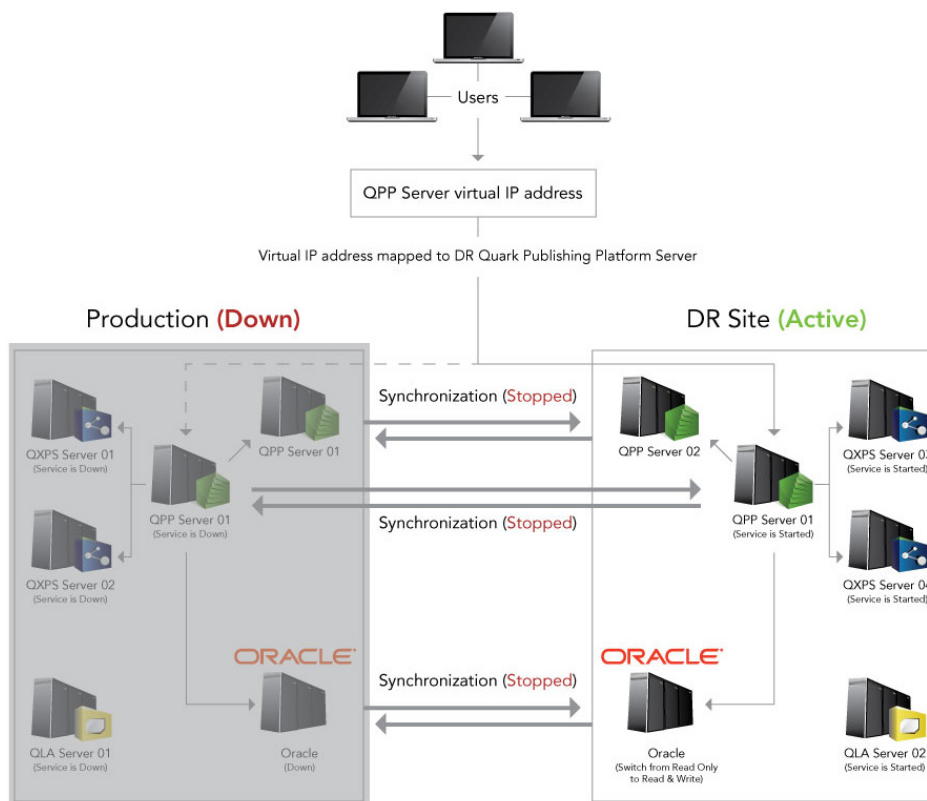
### Failover setup: Normal operation

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

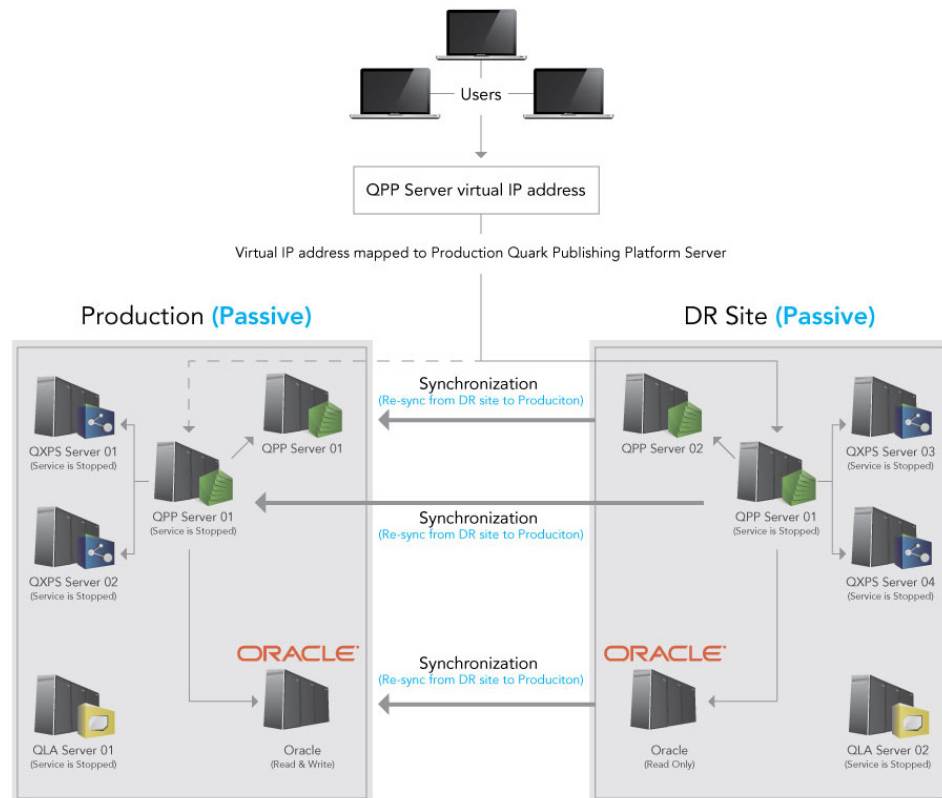
➡ Synchronization of the Quark Publishing Platform File Server folder(s) and Quark Publishing Platform Oracle database must be done at the same time and at the same intervals to maintain data integrity.

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 address.



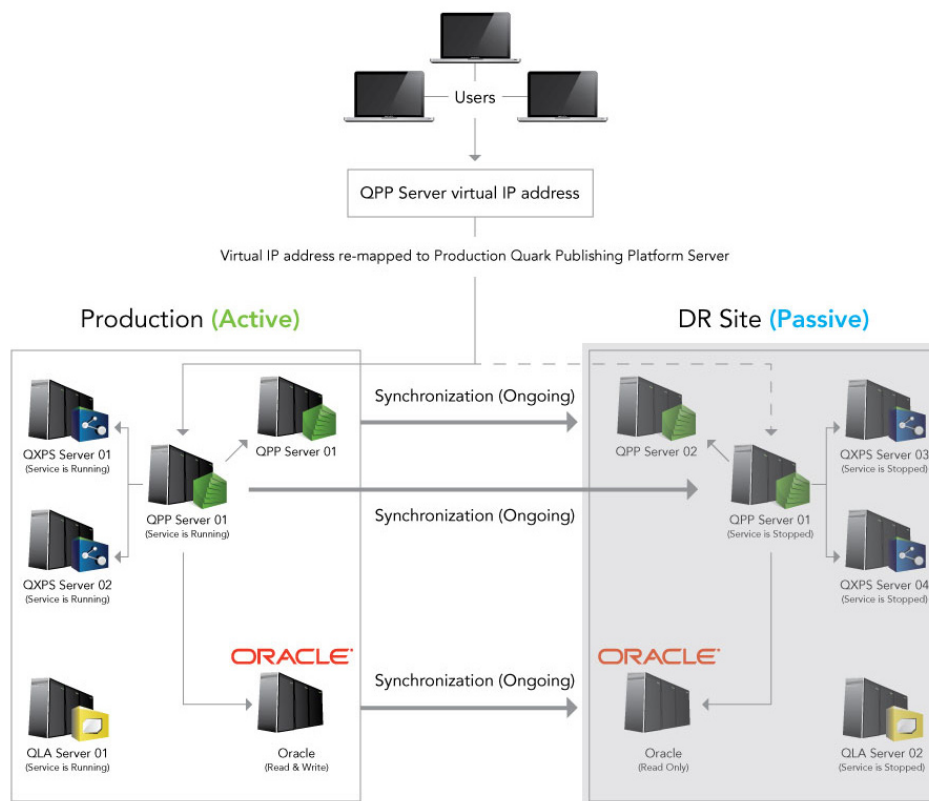
Failover setup: Production server goes down

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.



### 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.



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-13.2.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:

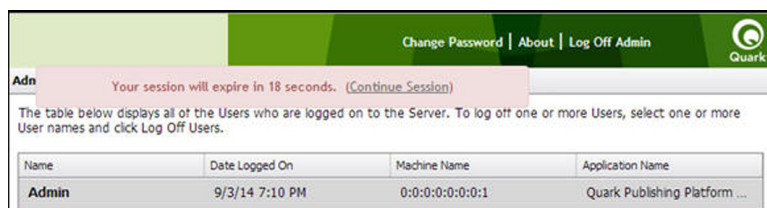
- `session-idle-time`: Specifies the time (in seconds) of inactivity after which a user will be forced to log off of WebAdmin.
- `exclude-url-patterns`: Specifies URL patterns which are not considered user activity.
- `pre-expiry-mag-time`: Specifies the time (in seconds) before pending inactivity bases 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.jsp/workspace/assetHeaderUpdate.jsp</p
aram-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**.



Forced log off countdown

## Configuring messaging

To configure Quark Publishing Platform Server to display **Asset Workflow Changed** messages when any listed attribute is changed on the related asset:

1. Open the “AssetWorkflowChangedMessageAttributes.xml” file located in the `[install_path]/Server/conf/` folder.
2. Set the attributes IDs or names to trigger the **Asset Workflow Changed** message when any listed attribute is changed on the related asset.

```
<!-- List of attribute ids or names to be considered for
"ASSET WORKFLOW CHANGED MESSAGE"
The "ASSET WORKFLOW CHANGED MESSAGE" will be published if
there is any changes in the attributes
of the mentioned asset/>
<util:list id="assetWorkflowAttributes">
  <value>name</value>
  <value>Status</value>
  <value>Workflow</value>
  <value>Routed to</value>
</util:list>
</beans>
```

## Configuring clipboard service

Use the “ClipboardConfig.xml file” to configure the clipboard service. This file contains format conversion detail.

- **ClipboardFormat**: Represents a clipboard data format with the format name and its mime-type.
- **ClipboardFormats**: Represents all possible clipboard data formats.
- **ChannelMapping**: Represent a binding of the publishing channel with the given clipboards data format. The given channel will be used to generate the clipboard data format specified in the attribute `clipboardformat`.
- **ChannelMappings**: A default set of all publishing channel to clipboard data formats mappings.

```
<ClipboardConfig>
  <ClipboardFormats>
    <ClipboardFormat>
      <name>svg</name>
      <mimeType>image/svg+xml</mimeType>
    </ClipboardFormat>
    ...
  </ClipboardFormats>
  <ChannelMappings>
    <ChannelMapping id="fetchExcelTable"
contentType="Microsoft Excel Table" clipboardformat="svg">
      <param name="OUTPUT_FORMAT">image</param>
      <param
name="OUTPUT_FORMAT_PROPERTIES">imageformat=svg</param>
    </ChannelMapping>
    ...
  </ChannelMappings>
</ClipboardConfig>
```



## Enable tracing and printing of JVM information

se the following debugging options in the “wrapper.conf” file located in the server installation folder, to enable tracing and printing of the JVM information:

- `wrapper.java.additional.7=-XX:+HeapDumpOnOutOfMemoryError` :  
Dumps heap to a file when `java.lang.OutOfMemoryError` is thrown.
- `wrapper.java.additional.8=-XX:HeapDumpPath=.\\log\\` : Sets the path to directory or filename for heap dump.
- `wrapper.java.additional.9=-XX:ErrorFile=.\\log\\hs_err_pid%p.log` :  
If an error occurs or JVM crashes, save the error data to this file.

## Configuring location of the publishing rescue folder

Configure the path used to store the QuarkXPress Server publishing rescue folder, where publishing data is to be saved in the case of any publishing failure. If a path is not specified, a default rescue folder named “PublishingRescueFolder”, will be created in the current working directory.

There are two ways to configure the path:

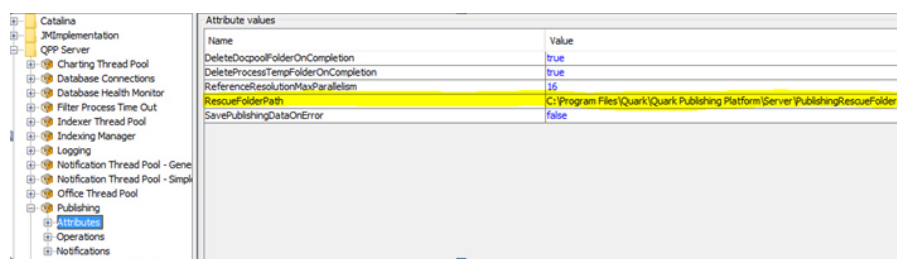
1. Edit the “PublishingUtilContext.xml” file (`[install-path]/Server/publishing`) to configure the path.

➡ This method requires a server restart.

```
<propertyname="rescueFolderPath" value="C:\Program
Files\Quark\Quark Publishing
Platform\Server\MyCustomPublishingRescueFolder"/>
```

2. Use JConsole as shown below:

➡ No server restart is required with this method.



# 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 log on 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 in a 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;s
endEmail;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:

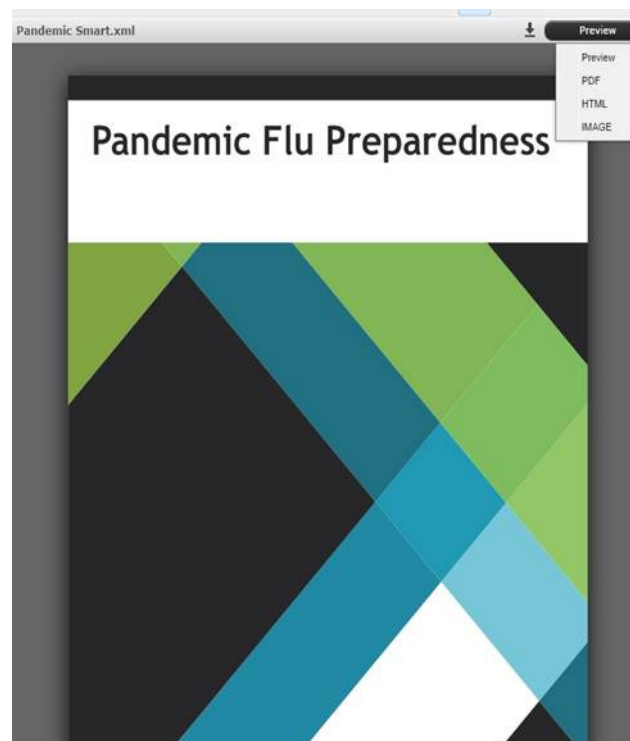
- `displayName`: Specifies the channel name displayed in the user interface (optional). If not given, then `outputFormat` is used.
- `Id`: Specifies the Publishing Channel Id as defined in the Platform Server.
- `ContentType`: Specifies the content type of the asset .
- `ApplyToChildContentTypes`: Specifies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

```



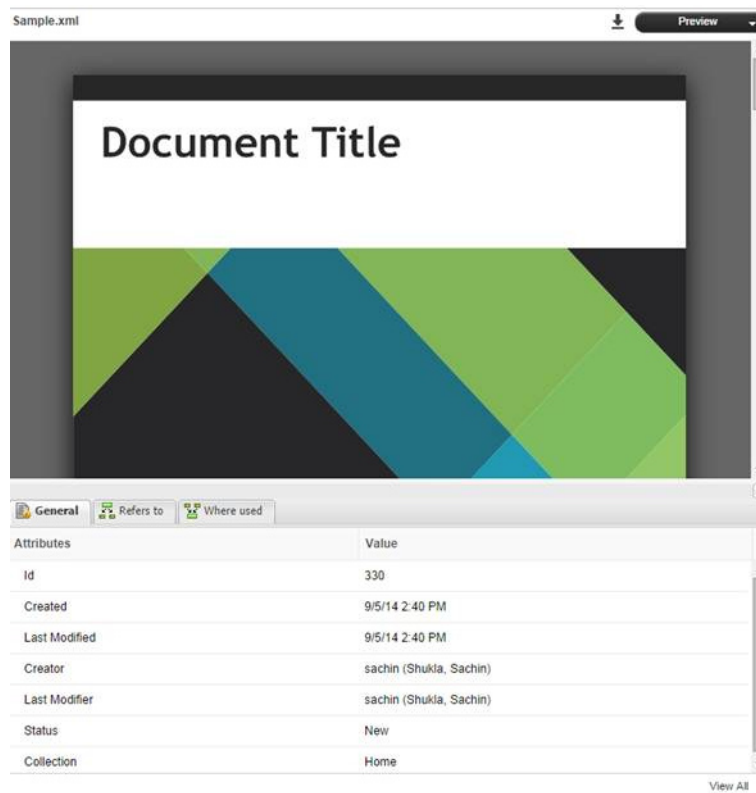
## Attributes for General pane

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>

```



## 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.

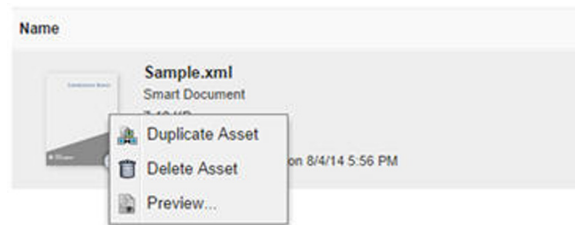
- **Role name:** Specify the name of role whose configuration is to be defined.
- **Item ID:** Specify the id of the item whose appearance needs to be configured.
- **showInToolbar:** To hide the menu item in the toolbar, set this attribute to false.
- **showInContext:** To hide the menu item in the context menu, set this attribute to false.
- **showInNewMenu:** To hide the menu item 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"/>
  </Role>
</ToolbarConfig>
```

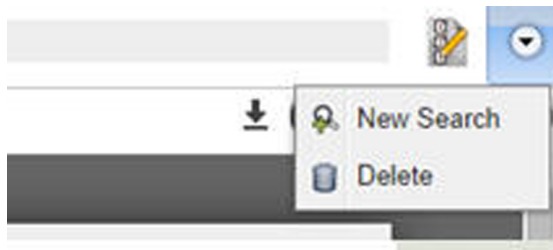
```
<Item id ="duplicate-asset" showInToolbar="false"/>
<Item id ="delete-asset"/>
<Item id ="show-edit-attributes" showInContextMenu="false"/>
<Item id ="show-Asset_preview" showInToolbar="false"/>
.....
</Role>
</ToolbarConfig>
```

The Context menu for the Guest role will look like:



Context menu for Guest role

Toolbar items for the Guest role will look like:



Toolbar items for Guest role

Here is the list of item IDs:

- `checkin`: Check in the Asset
- `show-check-out`: Check out the Asset
- `cancel-checkout`: Cancel Asset Check out
- `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 only applicable for menu items displayed in the toolbar and the context menus. 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 only applicable for menu items in the **New Asset** menu dropdown, 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. For more information, see *Quark Author Web Edition Configuration Guide*.

### 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 “server.xml” file located in the `{install_path}/conf` folder.
2. 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 IPs are assumed to be in 10.91.x.x range. You need to update this value based on your network IP ranges.

## Document save configuration

### Attribute mapping and revision settings

The `AttributeMapping` element for each content type present in the `workspace-config.xml` file allows you to:

- Specify the document attributes that should be mapped to Platform attributes.
- Specify save configuration when a document is saved to the Platform Server.

```

<AttributeMapping>
  <ComponentTypes>
    <ComponentType name="Smart Section">
      <RevisionSettings>
        <RevisionSetting operation="SaveDocumentRevision">
          <SaveSilently>false</SaveSilently>
          <SaveAsMinorVersion>false</SaveAsMinorVersion>
        </RevisionSetting>
        <RevisionSetting operation="SaveDocument">
          <SaveSilently>false</SaveSilently>
          <SaveAsMinorVersion>false</SaveAsMinorVersion>
        </RevisionSetting>
        <RevisionSetting operation="ExportComponent">
          .....
        </RevisionSetting>
        <RevisionSetting operation="SaveComponent">
          .....
        </RevisionSetting>
      </RevisionSettings>
      <Attributes>
        <Attribute name="Text preview"
xpath="/smart:section/smart:title"
indexingOption="ALL_VERSIONS"/>
        <Attribute name="Global ID" xpath="/smart:section/@id"
indexingOption="ALL_VERSIONS"/>
        <Attribute name="File extension" value="xml"
indexingOption="ALL_VERSIONS"/>
        .....
      </Attributes>
    </ComponentType>
  </ComponentTypes>
</AttributeMapping>

```

## Revision settings

The `RevisionSettings` element specifies the document versioning scheme (major or minor) and configuration with regard to the **Save** dialog visibility while saving a document.

- `Operation`: supported values are:
  1. `SaveDocument`: Saving a document to the server. Settings value referred to when a user clicks the **Save and Close** button in the Editor.
  2. `SaveDocumentRevision`: Saving a document revision to the server. Settings value referred to when a user clicks the **Save** button in the Editor.
  3. `ExportComponent`: Exporting a section of this component type from document of another component type and saving it as a document to the server. Settings value referred to when a user executes a **Create Component** operation in the Editor.
  4. `SaveComponent`: Saving a component as a new document version to the server. Settings value referred to when a component document is checked out inline in the main document and then checked in.
- `SaveSilently`: Specifies whether the document is saved without showing the **Save** dialog or not.
- `SaveAsMinorVersion`: Specifies whether the document is saved as a minor version or not.

## Attribute mapping

The `Attributes` element specifies the mapping of data from the document to the Platform Server attributes. The values of the Platform attributes can be specified either as static text or an XPath to the content in a document.

- `name`: Specifies the name of the Platform Server attribute.
- `value`: Specifies the static attribute value.
- `xpath`: Specifies the XPath to be used for setting the Platform attribute value.

➡ You can either set XPath or a static value.

- `indexingOption`: Supported values are:
  1. `INITIAL_VERSION`: Set this value to trigger indexing only for the first version of the document. This is the default value of the `indexingOption`.
  2. `ALL_VERSIONS`: Set this value to trigger indexing for every revision
- `inheritValueFromTemplate`: Specifies whether the attribute value is inherited from the server template. The default value is `false`.

### Attribute mapping configuration

If you want to configure some attributes so that attribute values are fetched from the document for the “Smart Document” content type:

- Update the `Attributes` key for content type ‘Smart Document’:

```

<Attributes>
  <Attribute name="Text preview"
    xpath="/smart:section/smart:title"
    indexingOption="ALL_VERSIONS"/>
  <Attribute name="Global ID" xpath="/smart:section/@id"
    indexingOption="ALL_VERSIONS"/>
  <Attribute name="Title" xpath="/smart:section/smart:title"
    indexingOption="ALL_VERSIONS"/>
  <Attribute name="File extension" value="xml"
    indexingOption="ALL_VERSIONS"/>
  <Attribute name="TextAttr"

    xpath="/smart:section/smart:body//smart:meta/smart:attribute[@
name='disease']/smart:value"
    indexingOption="ALL_VERSIONS"/>
  <Attribute name="DateAttr" value="2014-08-14"
    indexingOption="ALL_VERSIONS"/>
  <Attribute name="NumberAttr" value="15"
    indexingOption="ALL_VERSIONS"/>
  <Attribute name="BooleanAttr"

    xpath="/smart:section/smart:body//smart:meta/smart:attribute[@
name='transmission']/smart:value"
    indexingOption="ALL_VERSIONS"/>
  <Attribute name="DateTimeAttr"

    xpath="/smart:section/smart:body//smart:meta/smart:attribute[@
name='host']/smart:value"
    indexingOption="ALL_VERSIONS"/>
</Attributes>

```

When a document of type “Smart Document” is checked in, the value of attributes like `Text preview` and `Title` are fetched from the document based on the XPath and get saved along with the document.

# 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 (macOS 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 file 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 that the file 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.

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 use 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 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 file 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 file should be truncated so that the file 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.

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 - macOS

By default, an alert indicating that “Accessibility services are not enabled” may display at launch in Quark Publishing Platform Client for macOS. 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 macOS:

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 macOS**, 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 macOS**, 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 macOS**, 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 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 the 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 (macOS 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 macOS**, 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 macOS**, 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 macOS 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 `~/Library/Application Support/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist` file, locate the following text, and change `<false/>` to `<true/>`:  

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

```
<key>UseProxy</key>
<false/>
```
- For **QuarkCopyDesk for macOS**, open the `~/Library/Application Support/Quark/QPP/[QPP Framework Version]/com.quark.qpp.client.{Application}.config.plist` file, 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: macOS

To turn off collection mirroring on macOS:

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: macOS

To configure publishing channels on macOS:

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>
<string>smartDocHtml5Publication</string>
<string>busDocHtml5Publication</string>
<string>visioDocPdf</string>
<string>fetchVisioPage</string>
<string>fetchPowerPointSlide</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,
smartDocHtml5Publication,
busDocHtml5Publication, visioDocPdf, fetchVisioPage,
fetchPowerPointSlide"/>
```

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: macOS

To configure delivery channels on macOS:

1. For all macOS 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

- ➡ The following settings are also applicable for the Platform Adapter for Microsoft Office.

## Setting the Check-out location

To set the Check-out 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 Check-out location Go to **File > Preferences > General** and browse to the location you want to set as the Check-out location.

This preference can be reset in this way:

- 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 using the **Advanced > Reset to Default** feature.

## 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 the following way:

- 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 using the **Advanced > Reset to Default** feature.

## Setting the quick search preference

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 quick search 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 the following way:

- 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 using the **Advanced > Reset to Default** feature.

## 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 the following way:

- 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 using the `Advanced > Reset to Default` feature.

### Configuring the Platform Adapters for Microsoft Office components for web sharing

- ➔ The web sharing configuration is only recommended for standalone server installations and not for external web container deployments or EAR based deployments. If web sharing configuration is required under those scenarios you will need to work with Quark Technical Support.

This chapter gives instruction on installing the following Microsoft Office components:

- Quark Publishing Platform Adapter for Microsoft Office - Word
- Quark Publishing Platform Adapter for Microsoft Office - Excel
- Quark Publishing Platform Adapter for Microsoft Office - PowerPoint

The deployment location will be in the “Quark Publishing Platform Server” webapps folder on the Platform server machine.

The following steps need to be completed before installing any of the components:

1. Copy and extract the archive “Platform<version and build>\_Adapter\_for\_Office.zip”.
2. From the extracted archive, copy/overwrite all files from the folder “webapps\ROOT” to the “Quark Publishing Platform Server\webapps\ROOT” folder.
3. Copy the contents of the folder “webapps\clientinstallers” to the “Quark Publishing Platform Server\webapps\clientinstallers” folder.

Then perform the steps in the following sections to install the individual components.

### Configuring the Quark Publishing Platform Adapter for Microsoft Office - Word

To install the Platform Adapter for Microsoft Word:

1. Extract the contents of the archive “Word Adapter.zip” file from the `webapps\clientinstallers` folder on the Platform server machine to the `webapps\clientinstallers\Word Adapter` folder.
2. Open the “ConfigureAdapter.bat” file located in the `clientinstallers/Word Adapter` folder.
3. Update the `HostName/IP` and the `Port`, and then run the Batch file.

### Configuring the Quark Publishing Platform Adapter for Microsoft Office



**- Excel**

To install the Platform Adapter for Microsoft Excel:

1. Extract the contents of the archive “Excel Adapter.zip” file from the `webapps\clientinstallers` folder on the Platform server machine to the `webapps\clientinstallers\Excel Adapter` folder.
2. Open the “ConfigureAdapter.bat” file located in the `clientinstallers/Excel Adapter` folder.
3. Update the `HostName/IP` and the `Port`, and then run the Batch file.

**Configuring the Quark Publishing Platform Adapter for Microsoft Office  
- PowerPoint**

To install the Platform Adapter for Microsoft Excel:

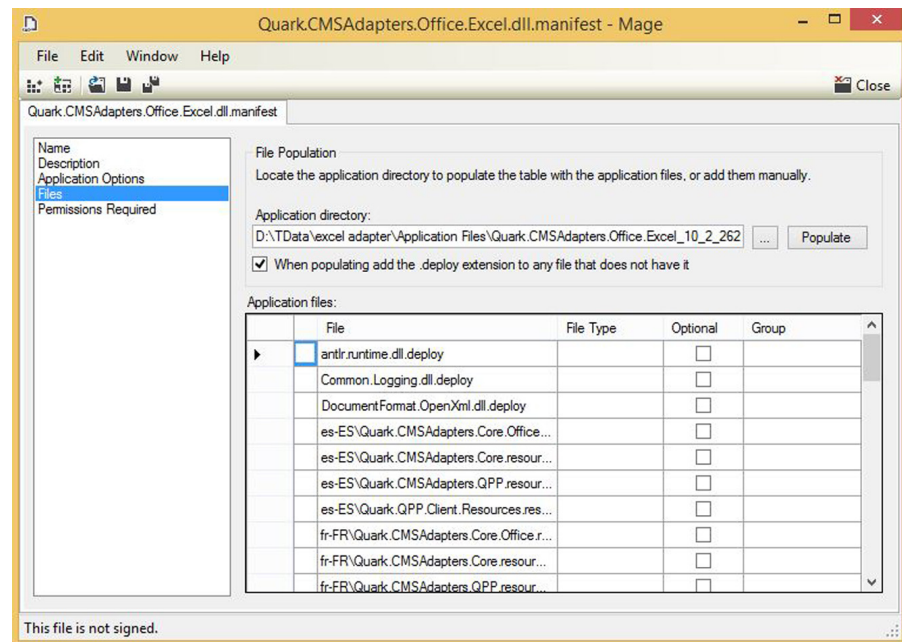
1. Extract the contents of the archive “PowerPoint Adapter.zip” file from the `webapps\clientinstallers` folder on the Platform server machine to the `webapps\clientinstallers\PowerPoint Adapter` folder.
2. Open the “ConfigureAdapter.bat” file located in the `clientinstallers/PowerPoint Adapter` folder.
3. Update the `HostName/IP` and the `Port`, and then run the Batch file.

**Update published ClickOnce deployments**

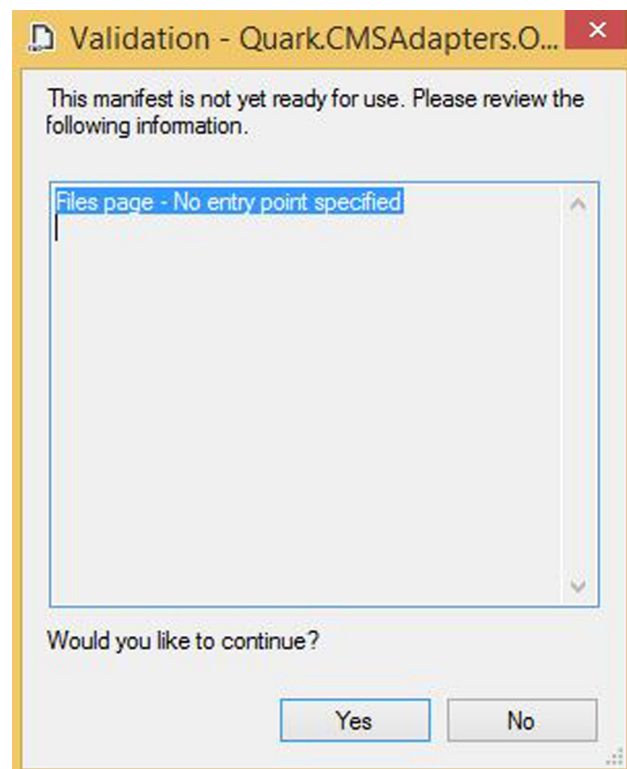
To update a published ClickOnce deployment package, use the Manifest Generation and Editing Tool (`mage.exe`), which is installed with Microsoft Visual Studio.

**Update the package and the application manifest**

1. Go to `Application Files\Quark.CMSAdapters.Office.<component>_%HighestVersion%` and delete the “Quark.CMSAdapters.Office.<component>.vsto” file, where component is Word, Excel or PowerPoint.
2. Make updates to the configuration files as required.
3. Run the Mage tool. Using the Visual Studio Command Prompt from Visual Studio Tools, type `mage.exe`.
4. in Mage, open the application manifest “Quark.CMSAdapters.Office.<component>.dll.manifest” in the `\Application Files\App_%.HighestVersion%\` folder.



5. Select **Files** and check the **When populating add the .deploy extension to any file that does not have it** option. Then click on the **Populate** button, and save the file by either clicking **Save** or **File > Save**.



6. Click **Yes** in the **Validation** dialog.



7. Provide the certificate file location path and its password (if any) by selecting the **Sign with certificate file** option. Click OK.

## Update VSTO

1. To update the “Quark.CMSAdapters.Office.<component>.vsto” present in the package folder use the following command:

```
mage -Update
"<path>\Quark.CMSAdapters.Office.<component>.vsto" -
AppManifest
"<path>\Quark.CMSAdapters.Office.<component>.dll.manifest" -
CertFile "<certificate Path>\<certificate>.pfx" -Password
"<password>"
```

2. Copy the “Quark.CMSAdapters.Office.Word.vsto” file from the package folder to  
\Application Files\App\_<HighestVersion>\.

## Manual configuration for QuarkXPress and QuarkCopyDesk XTensions

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

- ➡ In Windows, the QuarkXPress preference folder is located:  
C:\Users\<username>\AppData\Local\Quark\QuarkXPress 2016 and the  
QuarkCopyDesk preference folder is located: C:\Users\<user  
name>\AppData\Local\Quark\QuarkCopyDesk 2016
- ➡ In macOS, the QuarkXPress preference folder is located:  
/Volume/users/username/Library/Preferences/Quark/QuarkXPress 2016.  
The QuarkCopyDesk preference folder is located:

`/Volume/users/username/Library/Preferences/Quark/QuarkCopyDesk  
2016.`

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 “QPPCopyDeskXT.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.

## 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.

### Restoring Full Text Indexes

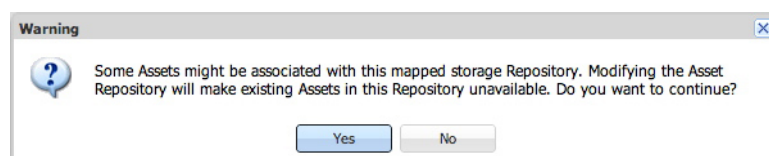
Restore the “index” folder in the location specified in the “LuceneTextIndexingConfig.properties” file.

## 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.

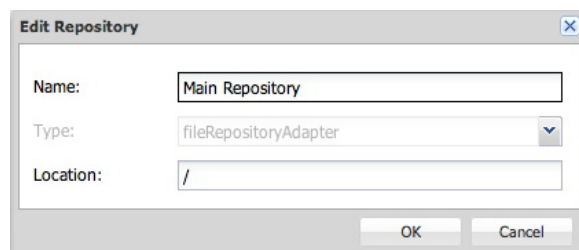
To update the asset repository path:

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.



This warning displays when you edit an asset repository

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



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

©2022 Quark Software Inc. and its licensors. All rights reserved.

Quark, the Quark logo, and Quark Publishing Platform are trademarks or registered trademarks of Quark Software Inc. and its affiliates in the U.S. and/or other countries. All other marks are the property of their respective owners.