WORKING WITH LOAD BALANCING AND QUEUEING FOR ADOBE INDESIGN CS5 SERVER
Contents

Introduction .................................................................................................................. 5
  Technical Overview ................................................................................................. 6
  Features .................................................................................................................... 6

Getting Started ........................................................................................................... 7
  Prerequisites ............................................................................................................. 7
  LBQ Setup ................................................................................................................ 7
  First Run ................................................................................................................... 8
  Troubleshooting ....................................................................................................... 8

Tools ............................................................................................................................. 9
  Web Client ................................................................................................................ 9
  JMeter ....................................................................................................................... 9
  Flex/Air Client ......................................................................................................... 9

Frequently Asked Questions ....................................................................................... 10
  Can I add InDesign Server instances to a running LBQ? ......................................... 10
  How do I set up multiple queues? ............................................................................. 10
  Is there a way to determine whether a job has failed? ............................................ 10
  Can ScriptArgs be attached to an EnqueueJob command? ...................................... 11
  How can I parameterize arguments to a script sent to IDS via LBQ? ......................... 11
  Can a client submit a new command before the prior command completes? .......... 11
  Can jobs be timed out if they take too long? ............................................................ 11
  Can the port used by LBQ be changed? ................................................................... 11
  Does LBQ have any built-in security features? ......................................................... 11
  How robust is LBQ if InDesign Server instances die? ............................................ 12
  Is there a WADL file for LBQ? ................................................................................ 12

Command Overview .................................................................................................... 12
  Command Structure ............................................................................................... 12
  Response Structure ............................................................................................... 12
  Command Categories ............................................................................................. 13
Introduction

This document contains information about Load Balancing and Queueing (LBQ), a new add-on component for Adobe® InDesign® CS5 Server. LBQ provides load balancing and job queueing across multiple instances of InDesign Server on one or more servers. LBQ is neither a plug-in nor a product; it is an optional add-on for InDesign Server that provides basic load-balancing functionality.

Because InDesign Server does not have built-in load-balancing functionality, Adobe developed LBQ to provide a simple option for InDesign Server users looking for a load-balancing solution. LBQ is optional. You do not need to use it to run InDesign Server successfully, and you are not required to provide load balancing for InDesign Server. It is one solution available for users who need load balancing.

The following network diagram shows how LBQ relates to InDesign Server.
Technical Overview

LBQ acts as an interface between one or more instances of InDesign Server and clients of InDesign Server. Clients initiate communication by submitting to LBQ an ExtendScript to run (a job request). These are stored and passed to the next available instance of InDesign Server. After InDesign Server completes the job, LBQ returns a response to the client that submitted the job.

LBQ interacts with clients using a RESTful API Web service. Unlike RPC-based Web services that use SOAP, RESTful Web Service calls (commands) are made entirely through a URL request. All parameters passed to the LBQ application are included in the query string of the URL. After a command is processed by an InDesign Server, LBQ returns a custom XML response.

Web Services requests submitted to LBQ are handled synchronously. This means that, when a Web services job request is sent by a client, an XML response is not returned to the client until that job request has been processed by InDesign Server. The synchronous nature of these communications may cause blocking issues for some clients. For more information, see “Frequently Asked Questions” on page 10.

LBQ interacts with InDesign Server using a CORBA interface. For communication to be successful, at least one share needs to be created that is accessible to both LBQ and InDesign Server. Which server the share is located on is not important. What matters is that both LBQ and InDesign server can read from and write to this share. These shares will contain the IOR files that InDesign Server generates on start-up. To learn more about this process, refer to Working With InDesign Server Java.

LBQ was built with the following technologies:

- Java
- CORBA
- OSGI
- Spring Framework
- Apache

Features

- Load balancing — LBQ queues job requests from clients and delegates them to be run on the next available instance of InDesign Server.

- Prioritized queuing — LBQ delegates job requests to various InDesign Server instances, based on the time the job was submitted. When each job has the same priority, the first job request submitted is the first job that is executed. If the client submits a job request with a higher priority than the jobs currently in the queue, that job request is delegated to an InDesign Server instance before other jobs previously submitted with lower priorities.

- Status visibility — LBQ allows users to see the current status of all pending job requests.

- Logging — LBQ provides the ability to investigate the history of a job. This includes discovering whether a job succeeded or failed, the error message returned, the time the job ran, and which InDesign Server instance it ran on.

- Multiple queues — In the context of LBQ, a queue is a buffer that stores job requests pending processing. A queue maps to one or more InDesign Server instances, and jobs submitted to that queue are routed to one of its associated InDesign Server instances. LBQ supports the creation of
multiple queues. When multiple queues are created, the client can choose which queue to submit a job request to. This gives the client control over which InDesign Server instance a job is submitted to.

- **Editing queued jobs** — LBQ provides mechanisms for editing a queued job request before it is executed. This includes changing the job’s priority, the maximum number of retries allowed, and which queue or script to run. The only thing that cannot be changed is the job’s name.

### Getting Started

**Prerequisites**

- Install InDesign CS5 Server and include the Adobe JobQueue CS5 installation option. The LBQ is then installed as part of InDesign Server. Its installation folder is `<IDS>/JobQueue`. For example:

  **Windows:** `C:\Program Files\Adobe\Adobe InDesign CS5 Server\JobQueue`
  
  **Mac:** `/Application/Adobe InDesign CS5 Server/JobQueue`

- Set up a common folder that both InDesign Server and LBQ can read from and write to. This will be used to store IOR files used for communication with CORBA. If you run the InDesign Server instances and LBQ on the same machine, the folder can be a local folder. Examples:

  **Windows:** `C:\ServerTestFiles\LBQ\iors`

  **Mac:** `/ServerTestFiles/LBQ/iors`

  If you run the InDesign Server instances and LBQ on different machines, the folder can be a network share folder. Example:

  **Windows:** `\server-name\LBQ\iors`

  **Mac:** `/Volumes/LBQ/iors (Mount smb://server-name/LBQ/iors as /Volumes/LBQ/iors)`

- If the job requires that files be modified or created, it is recommended that a share folder for storing those files be created that both the client and InDesign Server can access. This is optional.

### LBQ Setup

Create the file `<IDS>/JobQueue/configuration/idsQueue.properties`, containing only this line:

iorFolder = `<the_ior_folder_you_created_above>`

An example with a local folder:

**Windows:** `iorFolder = C:\\ServerTestFiles\\LBQ\\iors`

**Mac:** `iorFolder = /ServerTestFiles/LBQ/iors`

An example with a network share folder:

**Windows:** `iorFolder = \\\server-name\\LBQ\\iors`

**Mac:** `iorFolder = /Volumes/LBQ/iors`

**NOTE:** This file is read by the LBQ server, which is a Java program. In Windows, for Java to read it correctly:
There must be no quotation marks and no spaces in the paths.

Backslashes must be escaped.

First Run

1. Execute InDesign Server with command arguments specifying that it listen for connections via CORBA.

   Sample startup scripts for starting InDesign CSS Server with multiple CORBA instances are included in the SDK. To use these scripts, modify their IORFolder locations to match existing pathnames from your local machine. Ensure that at least one of the IORFolder locations used in these scripts matches the iorFolder configuration used by LBQ and specified in the setup above. These scripts are in:

   `<IDS SDK>/samples/load-balancing-and-queuing-clients/
    indesignserver-startup-scripts/`

   Windows:  `startup-indesign-server-CORBA-4instances.bat`

   Mac:      `startup-indesign-server-CORBA-4instances.command`

   **NOTE:** In Windows batch files, you do not need to escape backslashes.

2. Start LBQ by executing the clover script:

   Windows:  `<IDS>/JobQueue/clover.cmd`

   Mac:      `<IDS>/JobQueue/clover.sh`

3. From the computer running LBQ, open a browser and navigate to:

   `http://localhost:8080/com.adobe.clover.application/api/GetVersion`

   If an XML representation of the LBQ version is returned, LBQ is running successfully.

4. Run the Web client in:

   `<IDS SDK>/samples/load-balancing-and-queuing-clients/client-web`

5. Choose the EnqueueJob command and set the `scriptId` to:

   `<IDS SDK>/samples/load-balancing-and-queuing-clients/
    sample-extendscripts/delay.jsx`

   After 20 seconds, the browser should display a response indicating that command execution was successful. In addition, the console window for the instance of InDesign Server that executed the job request should display the `app.consoleout` statements from the ExtendScript.

   **NOTE:** The `scriptId` argument must be URL-encoded when submitted to LBQ.

Troubleshooting

- If LBQ responds to a version query but never shows a response when a job request is submitted, ensure that the path to the IOR folder specified in `<IDS>/JobQueue/configuration/idsQueue.properties is correct.`

- If InDesign Server instances do not show output when the `delay.jsx` file is run, ensure that the path to the IOR folder specified in `<IDS>/JobQueue/configuration/idsQueue.properties is correct.`
If jobs do not disappear from a queue, ensure that the path to the IOR folder specified for that queue is correct.

If LBQ encounters errors when trying to start Apache Tomcat, ensure that there are no conflicts with an existing Apache Tomcat installation.

If the IOR files do not appear in the IORFolder(s) specified when InDesign Server is started, verify that the folder exists and that the account used to execute the scripts has write permissions to the IORFolder(s).

Tools

Web Client

Location: `<IDS SDK>/samples/load-balancing-and-queueing-clients/client-web`

This is a simple Web client that uses the browser's XMLHttpRequest object to send a command to LBQ and observe the response. It is intended as a tool to assist developers in constructing the proper command URI quickly and observing the response. Because this client uses the Web browser, it is limited by the number of connections per server permitted by the browser. For this reason, it does not work well with a large number of concurrent requests.

JMeter

JMeter is an open source, Java-based, desktop application that is part of the Apache Jakarta project. It can send each request from a separate thread, which makes it useful for load testing.

Flex/Air Client

Location: `<IDS SDK>/samples/load-balancing-and-queueing-clients/client-air`

This is a complete client that allows you to do the following:

- Submit multiple InDesign Server job requests to LBQ.
- Start, stop, and reinitialize queues.
- Observe the status of each queue and job in near real-time.
- Cancel and edit queued jobs.
- View the history of commands submitted by this client to LBQ.
- Submit jobs that require parameters to be passed to them.

To build and run this Flex/Air client, you need:

- Adobe Flex 3.5 SDK
- Adobe Flex 3.5 Data Visualization Components

Both are available for free download from:

http://www.adobe.com/cfusion/entitlement/index.cfm?e=flex3sdk
If you use Flex Builder 3, its default SDK is lower than 3.5, so you need to install Adobe Flex 3.5 SDK by unzipping it to `<Flex Builder 3>/sdks/3.5`, and then configure Flex Builder to use it.

Whether you use Flex 3.5 SDK or Flex Builder 3, you need to install Flex 3.5 Data Visualization Components into the Flex 3.5 SDK by unzipping its package onto the SDK folder. This extracts the following into the SDK installation:

- `datavisualization.swc` into the `frameworks\libs` folder
- `datavisualization.RSL` into the `frameworks\rsls` folder
- `datavisualization_rb.swc` into the appropriate `frameworks\locale\<locale>` folder

Frequently Asked Questions

Can I add InDesign Server instances to a running LBQ?

This is supported. Reinitialize the default queue by running the following; LBQ will rescan the folder specified in the `idsQueue.properties` for newly added or deleted IOR files:

```
http://localhost:8080/com.adobe.clover.application/api/idsqueue/ReinitializeQueue
```

You can also reinitialize named queues by using additional parameters; for example:

```
http://localhost:8080/com.adobe.clover.application/api/idsqueue/ReinitializeQueue?directoryName=iors&queueName=bob
```

How do I set up multiple queues?

Multiple queues are set up by configuring one or more InDesign Servers to write IOR files to multiple shares in their startup scripts. Refer to *Working with InDesign Server in Java* for more information. Each of those shares can be configured to be a unique queue in LBQ. The default queue is configured with the `idsQueue.properties` and additional queues are configured with the `startQueue` command. To set up multiple queues, it is important to understand that there is a one-to-many relationship between IOR file shares and IOR files and a one-to-one relationship between IOR file shares and LBQ queues. As a result, each queue maps to one or more InDesign Server instances but no instance of InDesign server maps to more than one queue.

Is there a way to determine whether a job has failed?

There are two ways to do this:

- Examine the command response returned by each command after it has executed. The Flex/Air LBQClient demonstrates how to use the command response to construct a running log of all commands submitted during the current session.

- Examine the running log file in `<IDS>/JobQueue/configuration/idsQueue.log`. This log file contains a complete history of all commands submitted by all clients, with parameters and results. In addition, it contains LBQ-specific events, which may be used to troubleshoot any errors encountered by the application.
Note: To reduce the volume of messages sent to the idsQueue.log file, change the "Threshold" level in the file <IDS>/JobQueue/foundation/com.adobe.clover.config.log4j/log4j.properties. Options are "info", "warn", "error", and, most verbose, "debug".

Can ScriptArgs be attached to an EnqueueJob command?

The EnqueueJob command does not provide a mechanism for including script arguments. A workaround for this is to either write the values into the script or generate a script with the parameters written into it that calls the intended script using the app.doScript command. This approach is demonstrated in the AIR client included in the InDesign CS5 Server SDK.

How can I parameterize arguments to a script sent to IDS via LBQ?

A dynamically created script can be passed directly as part of the URL. For example:


Can a client submit a new command before the prior command completes?

LBQ uses synchronous connections to communicate with clients. For example, when a client sends an EnqueueJob command to LBQ, a response is returned after the command is executed. While the client is waiting for a response, it cannot submit a subsequent request over the same HTTP connection until the previous request is completed. There are three workarounds for this:

▶ Use a multithreaded platform like Java to develop the LBQ client. If each request is submitted from a unique thread, the client is not blocked from submitting additional requests while waiting for previous requests to complete.

▶ Establish multiple unique aliases for the LBQ server, which the client can use to send requests. This is not an ideal solution, but it works when the client is written for a platform that is not multithreaded and allows only a limited number of concurrent connections to the same host (for example, a Web browser).

▶ If developing a client in Flex, do not use the HTTPService object to communicate with LBQ. Instead, we recommend that you use as3httpclientlib, which can make multiple requests to LBQ without blocking. The Flex/Air client demonstrates how this is done.

Can jobs be timed out if they take too long?

No, because stopping a specific InDesign Server instance is not supported.

Can the port used by LBQ be changed?

By default, LBQ uses port 8080. This configuration is set in the Apache configuration files and may be changed without affecting other areas of LBQ.
Does LBQ have any built-in security features?

No, this version of LBQ does not have any security features beyond those provided through Apache.

How robust is LBQ if InDesign Server instances die?

There is no periodic “ping.” But, if an instance dies, the next request sent to that instance will fail. LBQ will then wait briefly, rescan that instance’s IOR file, and try again. This handles the case where an instance crashes and is relaunched by the InDesign Server Service, overwriting the old IOR. However, this does not handle the case where an instance of InDesign Server hangs.

**NOTE:** Refer to *Introduction to Adobe InDesign CS5 Server* for more information about the InDesign Server Service.

Is there a WADL file for LBQ?

No, a WADL file is not provided for LBQ.

Command Overview

Command Structure

All interaction between the client and LBQ is done through a series of commands that share a consistent structure. Each command is executed with one URI request. The command is specified in the path of the URI, and the parameters are provided via a query string. All command parameters need to be URL encoded to ensure successful communication.

The LBQ location is http://localhost:8080/.

Sample Command

<table>
<thead>
<tr>
<th>Command</th>
<th>com.adobe.clover.application/api/idsqueue/EnqueueJob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Value</td>
</tr>
<tr>
<td>scriptName</td>
<td>C%3A%5CLBQTest.jsx</td>
</tr>
<tr>
<td>jobName</td>
<td>job1</td>
</tr>
<tr>
<td>priority</td>
<td>5</td>
</tr>
<tr>
<td>queueName</td>
<td>Default</td>
</tr>
<tr>
<td>maximumRetryCount</td>
<td>3</td>
</tr>
</tbody>
</table>
Response Structure

Each command receives a unique XML response. The root element of the response is the name of the command submitted. The child elements differ depending on the command. The response contains the command parameters that were supplied, along with additional information from LBQ.

Sample Response

```xml
<?xml version="1.0" encoding="UTF-8"?>
<jobStatus>
  <scriptResult>Success!</scriptResult>
  <errorCode>null</errorCode>
  <errorMessage>null</errorMessage>
  <ior>IOR:000000000000002249444c3a636f6d2f616466f62652f6964732f4170706c696361746966f6e3a312e30000000000000000000f3135332e3332372e31383500c8000000013ff6964735f706f61fe9b33984a0100ee840031000000002000000000000008000000415454400000000010000001c00000000000100000000000010501000100101090000000100010109</ior>
  <iorFile>\windows-machine\LBQ\ior\ior1.txt</iorFile>
  <executionTime>22140</executionTime>
  <startTime>Fri Aug 28 12:45:37 PDT 2009</startTime>
  <endTime>Fri Aug 28 12:45:59 PDT 2009</endTime>
  <waitTime>172</waitTime>
  <priority>0.5</priority>
  <scriptIdName>C:\LBQTest.jsx</scriptIdName>
  <jobName>job1</jobName>
  <queueName>Default</queueName>
  <jobNumber>1</jobNumber>
  <retryCount>0</retryCount>
  <maximumRetryCount>3</maximumRetryCount>
  <submitTime>Fri Aug 28 12:45:36 PDT 2009</submitTime>
</jobStatus>
```

Command Categories

The remainder of this document contains details for each command, ordered alphabetically.

There are three categories of commands:

<table>
<thead>
<tr>
<th>Category</th>
<th>Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job management</td>
<td>EnqueueJob</td>
</tr>
<tr>
<td></td>
<td>ModifyJob</td>
</tr>
<tr>
<td></td>
<td>CancelJob</td>
</tr>
</tbody>
</table>
### cancelJob

**Command**  
com.adobe.clover.application/api/idsqueue/CancelJob

**Action**  
Cancels an existing job request that is in a queue, waiting to be processed.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Required?</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jobName</td>
<td>✔</td>
<td>None</td>
<td>The name of an existing job in a queue.</td>
</tr>
<tr>
<td>queueName</td>
<td></td>
<td>“Default”</td>
<td>The name of the queue that contains this job. If not specified, the name “Default” is assumed.</td>
</tr>
</tbody>
</table>

#### Response

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scriptResult</td>
<td>A string indicating either that the command was successful or why it failed.</td>
</tr>
<tr>
<td>jobName</td>
<td>The name of the job provided in the command request. If not specified, the name “Default” is assumed.</td>
</tr>
<tr>
<td>queueName</td>
<td>The name of the queue provided in the command request.</td>
</tr>
</tbody>
</table>

### EnqueueJob

**Command**  
com.adobe.clover.application/api/idsqueue/EnqueueJob

**Action**  
Adds a job request to LBQ for processing by InDesign Server.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Required?</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scriptName</td>
<td>✔</td>
<td>None</td>
<td>Provides a path with which InDesign Server can locate the script.</td>
</tr>
<tr>
<td>jobName</td>
<td></td>
<td>None</td>
<td>A unique name for the job. Other commands use this name to access this job. If not specified, the name “Default” is assumed.</td>
</tr>
<tr>
<td>priority</td>
<td></td>
<td>.5</td>
<td>The priority for this job relative to other jobs in the queue. Can be between 0 and 1; higher numbers are processed sooner.</td>
</tr>
<tr>
<td>queueName</td>
<td></td>
<td>“Default”</td>
<td>The name of an existing queue in which to place this job.</td>
</tr>
<tr>
<td>maximumRetryCount</td>
<td></td>
<td>3</td>
<td>The number of times this job is retried if it fails.</td>
</tr>
</tbody>
</table>

### Response

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scriptResult</td>
<td>A string indicating either that the command was successful or why it failed.</td>
</tr>
<tr>
<td>errorCode</td>
<td>If an error occurred, an error number; otherwise, null.</td>
</tr>
<tr>
<td>errorMessage</td>
<td>If an error occurred, an error message; otherwise, null.</td>
</tr>
<tr>
<td>ior</td>
<td>A string containing information needed for a client to connect to a CORBA object.</td>
</tr>
<tr>
<td>iorFile</td>
<td>The file used by LBQ to communicate with a specific instance of InDesign Server.</td>
</tr>
<tr>
<td>executionTime</td>
<td>The time to execute the request, in milliseconds.</td>
</tr>
<tr>
<td>startTime</td>
<td>The time at which the job request was started. (See “Sample Response” on page 13.)</td>
</tr>
<tr>
<td>endTime</td>
<td>The time at which the job request finished.</td>
</tr>
<tr>
<td>waitTime</td>
<td>The elapsed time between submitTime and startTime, in milliseconds.</td>
</tr>
<tr>
<td>priority</td>
<td>The priority for this job as specified in the command request.</td>
</tr>
<tr>
<td>scriptName</td>
<td>The name of the ExtendScript executed.</td>
</tr>
<tr>
<td>jobName</td>
<td>The name of the job provided in the command request.</td>
</tr>
<tr>
<td>queueName</td>
<td>The name of the queue containing the job as provided in the command request.</td>
</tr>
<tr>
<td>jobNumber</td>
<td>A number for the job generated by the server.</td>
</tr>
<tr>
<td>retryCount</td>
<td>The number of times the job was tried.</td>
</tr>
</tbody>
</table>
GetVersion

Command    com.adobe.clover.application/api/GetVersion
Action      Retrieves the LBQ version.
Parameters  None
Response    Contains the LBQ version.

IDSStatus

Command    com.adobe.clover.application/api/idsqueue/IDSStatus
Action      Retrieves the status of all jobs in all queues managed by LBQ. If this command is used too often, it affects the performance of LBQ. This command should not be used in regular polling and should be replaced with a QueueStatus or JobStatus command where possible.
Parameters  None
Response    Name Description
queue/queueName The name of a queue. Information for each queue is returned in this structure, one queue structure for each queue:
               <queue>
               <queueName>queuename1</queueName>
               <queuedJobs>names of jobs in this queue</queuedJobs>
               </queue>
queue/queuedJobs All jobs in a specific queue (as shown for queueName). For details about each job, see JobStatus.
iorFile      The file used by LBQ to communicate with a specific instance of InDesign Server.

JobStatus

Command    com.adobe.clover.application/api/idsqueue/JobStatus
Action      Retrieves the status of a specific job. This command affects LBQ performance less than IDSStatus and QueueStatus.
### ModifyJob

**Command**
com.adobe.clover.application/api/idsqueue/ModifyJob

**Action**
Modifies a job request that exists in a queue, waiting to be processed.

---

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Required?</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jobName</td>
<td>✓</td>
<td>None</td>
<td>The name of the job for which status is requested.</td>
</tr>
<tr>
<td>queueName</td>
<td></td>
<td>„Default“</td>
<td>The name of the queue containing the job.</td>
</tr>
</tbody>
</table>

**Response**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queueName</td>
<td>The name of the queue.</td>
</tr>
<tr>
<td>jobName</td>
<td>The name of the job.</td>
</tr>
<tr>
<td>scriptResult</td>
<td>A string indicating either that the command was successful or why it failed.</td>
</tr>
</tbody>
</table>

---

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Required?</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scriptName</td>
<td></td>
<td>Current value</td>
<td>Provides a path for InDesign Server to the script to be executed.</td>
</tr>
<tr>
<td>jobName</td>
<td>✓</td>
<td>None</td>
<td>The name of the job to modify.</td>
</tr>
<tr>
<td>priority</td>
<td></td>
<td>Current value</td>
<td>The priority for this job relative to other jobs in the queue. Can be between 0 and 1; higher numbers are processed sooner.</td>
</tr>
<tr>
<td>queueName</td>
<td></td>
<td>Current value</td>
<td>The name of the queue containing this job.</td>
</tr>
<tr>
<td>maximumRetryCount</td>
<td></td>
<td>Current value</td>
<td>The number of times the job is retried if it fails.</td>
</tr>
</tbody>
</table>

**Response**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scriptResult</td>
<td>A string indicating either that the command was successful or why it failed.</td>
</tr>
<tr>
<td>jobName</td>
<td>The name of the job provided.</td>
</tr>
<tr>
<td>queueName</td>
<td>The name of the queue containing the job.</td>
</tr>
<tr>
<td>jobNumber</td>
<td>A number for the job generated by the server.</td>
</tr>
<tr>
<td>submitTime</td>
<td>The time at which the job was submitted into the queue.</td>
</tr>
<tr>
<td>priority</td>
<td>The priority specified for this job.</td>
</tr>
</tbody>
</table>
QueueStatus

**Command**  
com.adobe.clover.application/api/idsqueue/QueueStatus

**Action**  
Retrieves the status of all jobs in the specified queue. This command is preferred over IDSStatus for performance reasons, but it should be replaced with JobStatus when possible.

**Parameter**

<table>
<thead>
<tr>
<th>Name</th>
<th>Required?</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queueName</td>
<td></td>
<td>“Default”</td>
<td>The name of the queue for which status is requested.</td>
</tr>
</tbody>
</table>

**Response**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queueName</td>
<td>The name of the queue.</td>
</tr>
<tr>
<td>queuedJobs</td>
<td>The names of all jobs in the queue. For details about each job, see JobStatus.</td>
</tr>
</tbody>
</table>
| instance/iorFile | The IOR files associated with each instance, in this general structure:  
 | <instance>  
 |  <iorFile>file info</iorFile>  
 |  <iorFile>file info</iorFile>  
 | ...  
 | </instance> |

ReinitializeQueue

**Command**  
com.adobe.clover.application/api/idsqueue/ReinitializeQueue

**Action**  
Clears all jobs out of a queue.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Required?</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queueName</td>
<td>✔</td>
<td>None</td>
<td>The name of the queue to reinitialize.</td>
</tr>
<tr>
<td>directoryName</td>
<td>✔</td>
<td>None</td>
<td>The directory containing the IOR files to be associated with this queue.</td>
</tr>
</tbody>
</table>
StartQueue

Command: com.adobe.clover.application/api/idsqueue/StartQueue

Action: Starts a queue into which jobs can be placed or resumes a stopped queue. Starting a new queue is optional; you can use the existing Default queue if only one queue is needed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Required?</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queueName</td>
<td>✔</td>
<td>None</td>
<td>The name for the new queue or the queue to restart.</td>
</tr>
<tr>
<td>directoryName</td>
<td>✔</td>
<td>None</td>
<td>The directory containing the IOR files to be associated with this queue.</td>
</tr>
</tbody>
</table>

Response

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queueName</td>
<td>The name of the queue.</td>
</tr>
<tr>
<td>iorFolder</td>
<td>The name of the directory containing IOR files that will be used to pass jobs for this queue to InDesign Server.</td>
</tr>
<tr>
<td>scriptResult</td>
<td>A string indicating either that the command was successful or why it failed.</td>
</tr>
</tbody>
</table>

StopQueue

Command: com.adobe.clover.application/api/idsqueue/StopQueue

Action: Stops (pauses) a queue that is currently running and holds all queued jobs. Use StartQueue to resume processing jobs in the queue.

Parameter

<table>
<thead>
<tr>
<th>Name</th>
<th>Required?</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queueName</td>
<td>✔</td>
<td>None</td>
<td>The name of the queue to stop.</td>
</tr>
</tbody>
</table>

Response

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queueName</td>
<td>The name of the queue.</td>
</tr>
<tr>
<td>scriptResult</td>
<td>A string indicating either that the command was successful or why it failed.</td>
</tr>
</tbody>
</table>