

# 3DSMax Job Submission

Step by step instructions for submitting 3dsMax jobs with Qube!

---

## Step 1

If you have been going through this Install Guide in order, you should be able to locate the "Qube!" menu in 3dsMax (picture). If you don't have that menu item, you need to install the JobType on the Worker - see [this page](#).

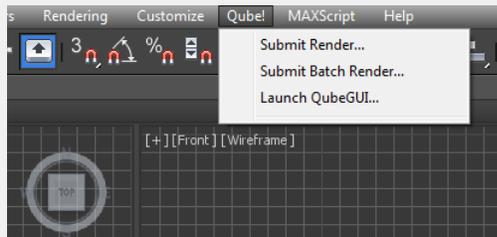
With a scene loaded in 3dsMax choose "Submit Render...".

Note that you could also choose "Submit Batch Render..." depending on your requirements.

### **Render vs Batch Render** What's the difference?

**Submit Render** will open a copy (instance) of 3dsMax on the Worker that picks up the job, and that instance will render frames as assigned by the Supervisor. Since the scene file is only being opened one time, this can be faster to generate the full sequence.

**Submit Batch Render** will use the command line to open a new instance of the 3dsMax file for each frame that it renders. Although this can be slower for large scenes, it can also be more flexible in large, complex environments.



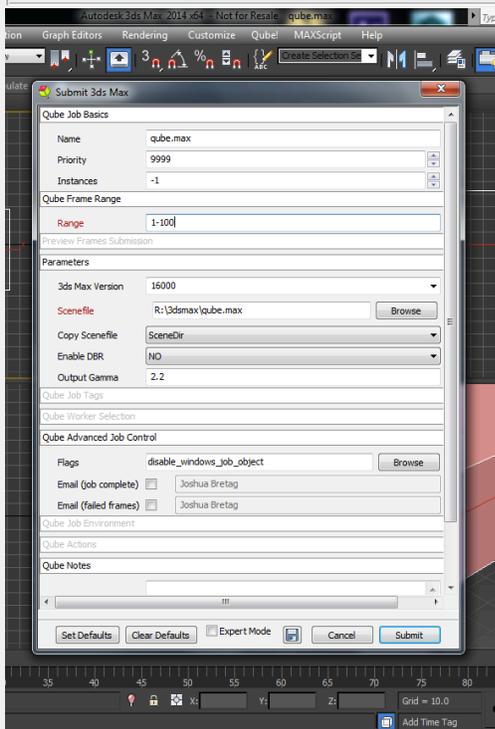
## Step 2

This will present a pre-filled submission UI like the one shown at right.

Ensure sections marked in red have the correct details.

Important: Because 3dsMax doesn't allow thread control, you need to make sure that each instance of 3dsMax occupies the full Worker. To do this, select "Expert Mode" at the bottom of the dialog. Scroll down to Qube Worker Selection and click on the Browse button next to Reservations. Check the box marked "All" and accept. You should end up with the statement "host.processors=1+" as shown here.

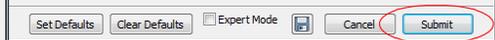
Qube Worker Selection	
Hosts	<input type="text"/> <input type="button" value="Browse"/>
Groups	<input type="text"/> <input type="button" value="Browse"/>
Omit Hosts	<input type="text"/> <input type="button" value="Browse"/>
Omit Groups	<input type="text"/> <input type="button" value="Browse"/>
Priority Cluster	<input type="text"/> <input type="button" value="Browse"/>
Host Order	+host.processors.avail <input type="button" value="Browse"/>
Requirements	<input type="text"/> <input type="button" value="Browse"/>
Reservations	host.processors=1+ <input type="button" value="Browse"/>
Restrictions	<input type="text"/> <input type="button" value="Browse"/>



## Step 3

Click 'Submit'

For further details on the submission UI see below.



## Job Submission Details



Not all sections need to be filled in in order to render. Only the fields marked in red are required.

Qube Job Basics	
Name	<input type="text"/>
Priority	9999
Instances	1
Max Instances	-1

[Click here for details...](#)

### Name

This is the name of the job of the job so it can be easily identified in the Qube! UI.

### Priority

Every job in Qube is assigned a numeric priority. Priority 1 is higher than priority 100. This is similar to 1st place, 2nd place, 3rd place, etc. The default priority assigned to a job is 9999.

### Instances

This is the number of copies of the application that will run at the same time across the network. The combination of "Instances=1" and "Max Instances=-1" means that this job will take as much of the farm as it can, and all jobs will share evenly across the farm.

Examples:

On a 12 slot(core) machine running Maya if you set

"Instances" to 4

"Reservations" to "host.processors=3"

Qube! will open 4 sessions of Maya on the Worker(s) simultaneously, which may consume all slots/cores on a given Worker.

if you set

"Instances" to 1

"Reservations" to "host.processors=1+"

Qube will open 1 session of Maya on a Worker, consuming all slots/cores ("host.processors=1+" is used for all slots/cores).

### Max Instances

If resources are available, Qube! will spawn more than 'Instances' copies of the application, but no more than 'Max Instances'. The default of -1 means there is no maximum. If this is set to 0, then it won't spawn more than 'Instances' copies.

More on [Instances & Reservations & SmartShare Studio Defaults](#)

Preview Frames Submission	
Use Preview Frames	<input type="checkbox"/>
Frame Numbers	<input type="text"/>
Preview Priority	0
Preview Subjobs	1

[Click here for details...](#)

### Use Preview Frames

Enabling preview frames will create 2 jobs:

- A primary dependent job with a higher priority that will render the selected frames first
- A secondary job with lower priority that will render the remaining frames. This will return the selected frames faster so that you can check the accuracy of your renders.

### Frame Numbers

Choose the frames that you wish to render first. If left blank the default is to render the first frame, the last frame and the middle frame in that order. You can select

specific frames by adding comma separated frame numbers e.g 1,2,10,15,75, or a range with, e.g., 1-100x5 (1 to 100, every 5th frame)

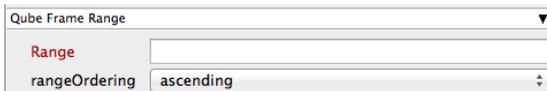
## Preview Priority

Choose the priority for the preview job. This can be set by the site admin.

## Preview Subjobs

Choose the number of instances / subjobs for the preview frames. By default, this is equal to the number of preview frames - that is, it will try to do all the preview frames at the same time.

Note that when you submit a job with preview frames enabled, it will actually submit 2 jobs—one with the preview frames list at a higher priority, and another with the rest of the agenda, at the normal priority (as specified in the job's **Priority** field). You will get, consequently, 2 job IDs for the submission.



[Click here for details...](#)

## Range

Frame range for the job (e.g 1-100, or 1-100x3, or 1,3,7,10). Most jobs require a frame range to execute on the Workers. You can set this range in a few different ways :

- "1-100" will just render the range between 1 and 100
- "1-100x3" will render every 3rd frame in the range 1 to 100; 1, 4, 7, ..., 94, 97, 100
- "1,3,7,10" will only render the selected frames 1,3,7,10

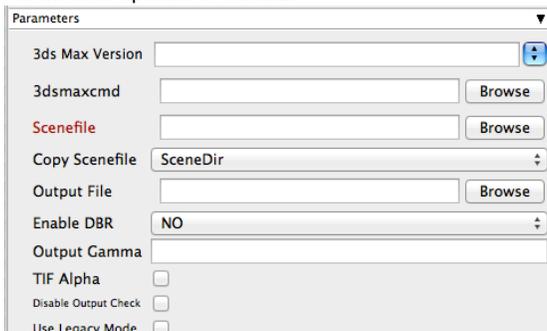
### rangeOrdering

Order to render the items.

(Ascending=1,2,3,4,5...,Descending=10,9,8...,Binary=first,middle,last...) You can set the order in which your frames are rendered. The drop down options are:

- "Ascending" - this will render the frames counting upwards from your start frame
- "Decending" - this will render the frames counting backwards from your end frame
- "Binary" - This will render the first, last, and middle frames of the range, then the middle frame of the first half and the middle frame of the second half, and so on. This is useful for sampling the frames in the sequence to make sure it is rendering correctly.

## Parameters Specific to 3dsMax



[Click here for details...](#)

### 3ds Max Version

Sets the version of 3d Studio Max on the worker. You may also use numeric version numbers, such as "13000" for v2011, "10000" for v2008, "9000" for v9, etc. This should normally be left blank.

### 3dsmaxcmd

Browse for or enter manually the location of 3dsmaxcmd.exe on the worker - e.g. C:/Program Files/Autodesk/3ds Max 2011/3dsmaxcmd.exe. This should normally be left blank.

### Scenefile

Browse or enter manually the location of the scene file to be rendered. This is a required field for submission. Important: Best practice is to ensure the scene file and all of its dependent files such as textures are on network storage accessible by the workers.

### Copy Scenefile

Whether or not every instance should create and work with its own copy of the scenefile to workaround an older 3dsmax limitation that would not allow a single copy of a scene file to be loaded by multiple computers at the same time. This limitation may have been resolved by Autodesk in more recent versions, but the options remains for legacy reasons.

### Output file

Browse for or manually enter the location of the output file you wish to generate. Important: Best practice is to ensure outputs are written to network storage accessible by the workers. This is normally already defined in the 3dsMax file.

### Enable DBR

Enable DBR for mental ray or V-Ray. See also [Distributed Renderers with mental ray or V-Ray](#).

### Output Gamma

Manually override the scene file gamma setting without opening 3dsMax.

### TIF Alpha

Manually override the scene file alpha-channel without opening 3dsMax.

### Disable Output Check

Ensures that the 3dsMax filecheck is not enabled. By default, 3dsMax will check for the existence of an output file after a render.

### Use Legacy Mode

Use legacy mode (required for 3dsMax prior to v2008. Will **not** work for v2013 and above). Only required if running 3ds Max < 2008.

Qube Job Tags	
Job Show	<input type="text"/>
Job Shot	<input type="text"/>
Job Seq	<input type="text"/>
Job Client	<input type="text"/>
Job Dept	<input type="text"/>
Job Custom1	<input type="text"/>
Job Custom2	<input type="text"/>
Job Custom3	<input type="text"/>
Job Custom4	<input type="text"/>
Job Custom5	<input type="text"/>

✓ [Click here for details...](#)

### Qube Job Tags

New in Qube 6.5

**Note:** The Job Tags section of the submission UI will not be visible unless they are turned on in the [Preferences](#) in the Wrangler View UI. Job Tags are explained in detail on the [Job Tags](#) page.

Qube Worker Selection		
Hosts	<input type="text"/>	<input type="button" value="Browse"/>
Groups	<input type="text"/>	<input type="button" value="Browse"/>
Omit Hosts	<input type="text"/>	<input type="button" value="Browse"/>
Omit Groups	<input type="text"/>	<input type="button" value="Browse"/>
Priority Cluster	<input type="text"/>	<input type="button" value="Browse"/>
Host Order	+host.processors.avail	<input type="button" value="Browse"/>
Requirements	<input type="text"/>	<input type="button" value="Browse"/>
Reservations	<input type="text"/>	<input type="button" value="Browse"/>
Restrictions	<input type="text"/>	<input type="button" value="Browse"/>

✓ [Click here for details...](#)

## Hosts

Explicit list of Worker hostnames that will be allowed to run the job (comma-separated).

## Groups

Explicit list of Worker groups that will be allowed to run the job (comma-separated). Groups identify machines through some attribute they have, eg, a GPU, an amount of memory, a license to run a particular application, etc. Jobs cannot migrate from one group to another. See [worker\\_groups](#).

## Omit Hosts

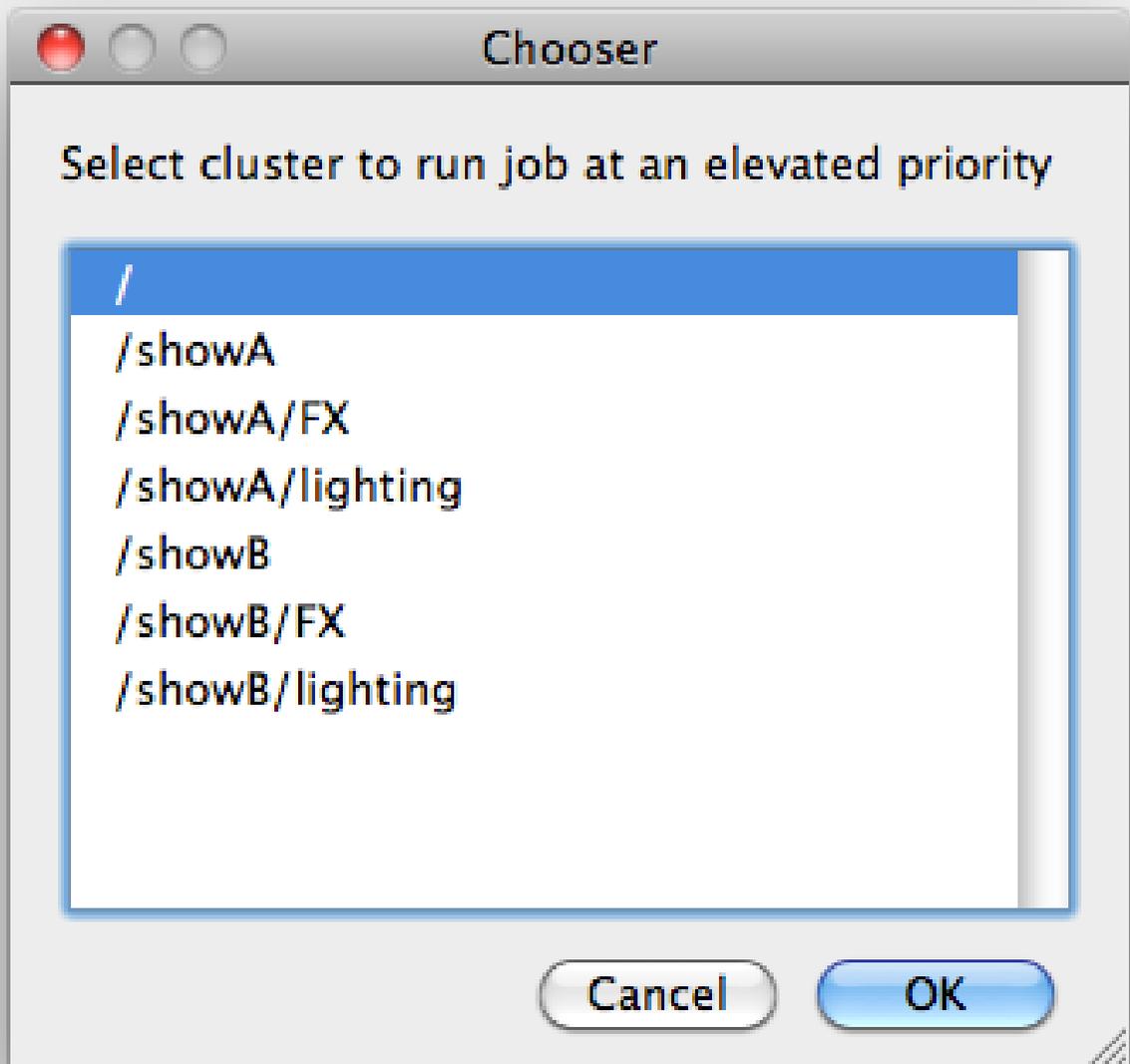
Explicit list of Worker hostnames that are **not** allowed run the job (comma-separated).

## Omit Groups

Explicit list of Worker groups that are **not** allowed to run the job (comma-separated).

## Priority Cluster

Clusters are non-overlapping sets of machines. Your job will run at the given priority in the given cluster. If that cluster is full, the job can run in a different cluster, but at lower priority. [Clustering](#)



Example:

- A job submitted to /showB/lighting will run with its given priority in /showB/lighting cluster.
- If /showB/lighting is full, that job can run in /showB/FX, but at a lower priority.
- If both /showB/lighting and /showB/FX are full, the job can run in /showA/\* at an even lower priority.

### Host Order

Order to select Workers for running the job (comma-separated) [+ means ascending, - means descending].

---

---

Host Order is a way of telling the job how to select/order workers

- "+host.processors.avail" means prefer workers which have more slots available
- "+host.memory.avail" means prefer workers which have more memory available
- "+host.memory.total" means prefer workers which have more total memory
- "+host.processor\_speed" means prefer workers with higher cpu speeds
- "+host.cpus" means prefer workers with higher total cpu slots

## Requirements

Worker properties needed to be met for job to run on that Worker (comma-separated, expression-based). Click 'Browse' to choose from a list of Host Order Options.

Requirements is a way to tell the workers that this job needs specific properties to be present in order to run. The drop-down menu allows a choice of OS:

- "winnt" will fill the field with "host.os=winnt" which means only run on Windows based workers
- "linux" will fill the field with "host.os=linux" which means only run on Linux based workers
- "osx" will fill the field with "host.os=osx" which means only run on OSX based workers

You can also add any other Worker properties via plain text. Some examples:

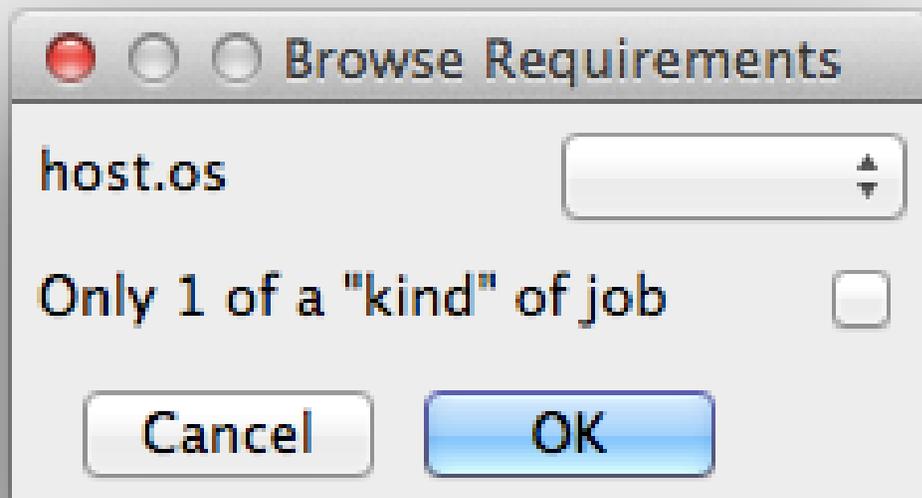
- "host.processors.avail.=4" means only run this job on workers that have 4 or more slots available
- "host.processors.used=0" means only run this job on workers with 0 slots in use
- "host.memory.avail=400" means only run this job on workers that have 400 memory available

With integer values, you can use any numerical relationships, e.g. =, <, >, <=, >=. This won't work for string values or floating point values. Multiple requirements can also be combined with AND and OR (the symbols && and || will also work).

The 'Only 1 of a "kind" of job' checkbox will restrict a Worker to running only one instance with a matching "kind" field (see below). The prime example is After Effects, which will only allow a single instance of AE on a machine. Using this checkbox and the "Kind" field, you can restrict a Worker to only one running copy of After Effects, while still leaving the Worker's other slots available for other "kinds" of jobs.

## Reservations

Worker resources to reserve when running job (comma-separated, expression-based).



**Browse Reservations**

host.memory	<input type="text" value="0"/>	<input type="button" value="↑"/> <input type="button" value="↓"/>	Mb
host.processors	<input type="text" value="1"/>	<input type="button" value="↑"/> <input type="button" value="↓"/>	<input type="checkbox"/> All
global.vray	<input type="text" value="0"/>	<input type="button" value="↑"/> <input type="button" value="↓"/>	
global_host.nuke	<input type="text" value="0"/>	<input type="button" value="↑"/> <input type="button" value="↓"/>	
license.prman	<input type="text" value="0"/>	<input type="button" value="↑"/> <input type="button" value="↓"/>	

Reservations is a way to tell the workers that this job will reserve the specific resources for this job.

Menu items:

- "host.processors" this will fill the field with "host.processors=X" which means reserve X slots on the worker while running this job
- "host.memory" this will fill the field with "host.memory=X" which means only reserve X memory on the worker while running this job

Other options:

- "host.license.nuke=1" when a [Global Resources](#) entry has been made you can reserve any arbitrary named item. **New in 6.6:** Once you global resource, it will show up in this menu (eg global.vray above).
- See also [Job Reservations](#)

## Restrictions

Restrict job to run only on specified clusters ("|" -separated) [+]

means all below, \* means at that level]. Click 'Browse' to choose from a list of Restrictions Options.

Restrictions is a way to tell the workers that this job can only run on specific clusters. You can choose more than one cluster in the list.

Examples:

- Choosing /showA would restrict the job to machines that are only in the /showA cluster, and no other cluster, not even those below /showA.
- Choosing /showA/\* would restrict the job to the cluster(s) *below* /showA, but *not including* /showA
- Choosing /showA/+ would restrict the job to /showA and all the clusters below it.

## See Also

- [Controlling Host Selection](#)
- [How to use qbrwk.conf](#)
- [worker\\_groups](#)
- [worker\\_cluster](#)
- [How to use clustering for workers](#)

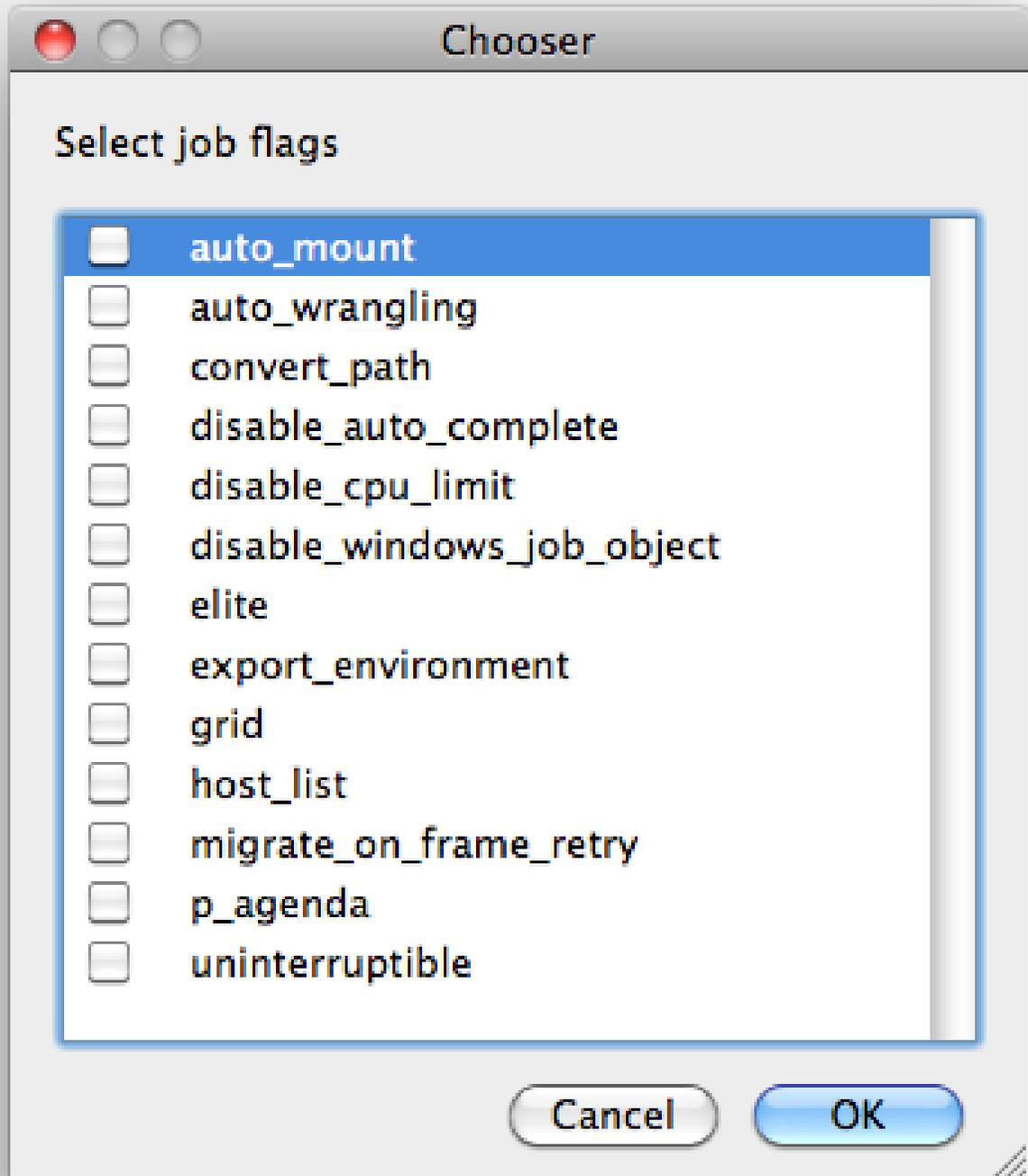
The screenshot shows the 'Qube Advanced Job Control' window. It contains several input fields and checkboxes:

- Flags:** A text input field with a 'Browse' button to its right.
- Dependency:** A text input field with an 'Add' button to its right.
- Email (job complete):** A checkbox followed by a text input field containing 'briank'.
- Email (failed frames):** A checkbox followed by a text input field containing 'briank'.
- Blocked:** A checkbox.
- Stderr->Stdout:** A checkbox.
- Job Label:** A text input field.
- Job Kind:** A text input field.
- Process Group:** A text input field.
- Retry Frame/Instance:** A text input field with '0' and a spinner icon.
- Retry Work Delay:** A text input field with '0' and a spinner icon.
- Subjob Timeout:** A text input field with '-1' and a spinner icon.
- Frame Timeout:** A text input field with '-1' and a spinner icon.

[Click here for details...](#)

## Flags

List of submission flag strings (comma separated). Click 'Browse' to choose required job flags.

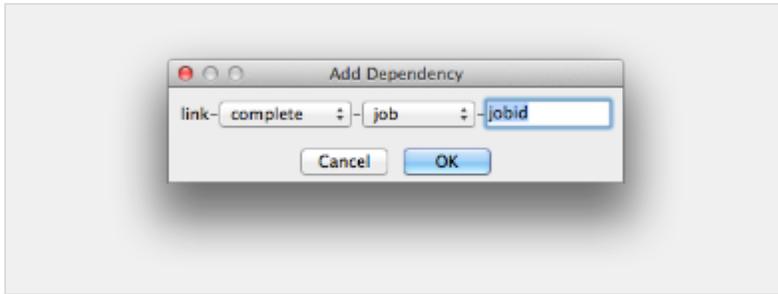


See [this page](#) for a full explanation of flag meanings

### Dependency

Wait for specified jobs to complete before starting this job

(comma-separated). Click 'Add' to create dependent jobs.



You can link jobs to each other in several ways:

- "complete" means only start this job after designated job completes
- "failed" means only start this job if the designated job fails
- "killed" means only start this job if the designated job has been killed
- "done" means start this job if the designated job is killed/failed/complete

The second menu chooses between "job" (the entire set of frames) and "work" (typically a frame). So to link frame 1 of one job to frame 1 of a second, job, you would choose "work" in this menu. If you want to wait for all the frames of one job to complete before starting a second, then choose "job". The other option, "subjob", refers to the instance of a job. This is much less common, but means that, for example, the instance of Maya that was running frames has completed.

For a complete description on how to define complex dependencies between jobs or frames, please refer to the [Callbacks](#) section of the Developers Guide.

### **Email (job complete)**

Send email on job completion (success or failure). Sends mail to the designated user.

### **Email (failed frames)**

Sends mail to the designated user if frames fail.

### **Blocked**

Set initial state of job to "blocked".

### **Stderr->Stdout**

Redirect and consolidate the job stderr stream to the stdout stream. Enable this if you would like to combine your logs into one stream.

### **Job Label**

Optional label to identify the job. Must be unique within a Job Process Group. This is most useful for submitting sets of dependent jobs, where you don't know in advance the job IDs to depend on, but you do know the labels.

### **Job Kind**

Arbitrary typing information that can be used to identify the job. It is commonly used to make sure only one of this "kind" of job runs on a worker at the same time by setting the job's requirements to

include "not (job.kind in host.duty.kind)". See [How to restrict a host to only one instance of a given kind of job, but still allow other jobs](#)

### Process Group

Job Process Group for logically organizing dependent jobs. Defaults to the jobid. Combination of "label" and "Process Group" must be unique for a job. See [Process group labels](#)

### Retry Frame/Instance

Number of times to retry a failed frame/job instance. The default value of -1 means don't retry.

### Retry Work Delay

Number of seconds between retries.

### Subjob Timeout

Kill the subjob process if running for the specified time (in seconds). Value of -1 means disabled. Use this if the acceptable instance/subjob spawn time is known.

### Frame Timeout

Kill the agenda/frame if running for the specified time (in seconds). Value of -1 means disabled. Use this if you know how long frames should take, so that you can automatically kill those running long.

FlightCheck scripts		
Job Pre-flight	<input type="text"/>	Browse
Job Post-flight	<input type="text"/>	Browse
Work Pre-flight	<input type="text"/>	Browse
Work Post-flight	<input type="text"/>	Browse

[Click here for details...](#)

#### Job Pre-flight

Some executable (e.g. a script) that will run *on the Worker* before the job runs. If the pre-flight fails, the job will not run on this Worker, but will be moved to another. Useful for setting up environments, copying required images, checking the existence of fonts on the Worker, etc.

#### Job Post-flight

An executable that will run *on the Worker* after the job has finished on that Worker. If this fails, the instance is failed.

#### Work Pre-flight

Some executable (e.g. a script) that will run *on the Worker* before the frame/work runs. If this fails, the frame/work will be marked as failed and rendering will be skipped. If there are retries configured, the frame/work will be treated according to those settings.

#### Work Post-flight

An executable that will run *on the Worker* after the frame/work runs. If this fails, the frame/work will be marked as failed, even if the render, itself, was successful. If there are retries configured, the frame/work will be treated according to those settings.

Qube Job Delayed Start		
hh:mm M/D/Y	<input type="text"/>	Time and Date

[Click here for details...](#)

### hh:mm M/D/Y

The job is blocked on the Supervisor until this day and time. Click on "Time and Date" to get a calendar and time UI.

Qube Job Environment

Cwd

Key	Value

Environment Variables

Impersonate User

[Click here for details...](#)

### Cwd

Current Working Directory to use when running the job.

### Environment Variables

Environment variables override when running a job. You can specify key/value pairs of environment variables

This is useful when you might need different settings for your render applications based on different departments or projects.

### Impersonate User

You can specify which user you would like to submit the job as. The default is the current user. The format is simply <username>. This is useful for troubleshooting a job that may fail if sent from a specific user.

Example:

Setting "josh" would attempt to submit the job as the user "josh" regardless of your current user ID.

Note: In order to do this, the submitting user must have "impersonate user" permissions.

Qube Actions

generateMovie

[Click here for details...](#)

### GenerateMovie

Select this option to create a secondary job that will wait for the render to complete then combine the output files into a movie.

Note: For this to work correctly the "Qube (ImagesToMovie) Job..." has to be setup to use your studios transcoding application.

Qube Notes

Account

Notes

[Click here for details...](#)

### Account

Arbitrary accounting or project data (user-specified). This can be

used for creating tags for your job.

You can add entries by typing in the drop-down window or select already created accounts from the drop-down.

See also [Qube! Job Tags](#)

**Notes**

Freeform text for making notes on this job. Add text about the job for future reference. Viewable in the Qube UI.