DAIKIN

One More Solution

qube!

クイックガイド *3dsMax・コマンドライン編*

第6.0版 – v7.0.x対応

ダイキン工業株式会社

ご注意

このクイックガイドは、本製品の使用許諾契約書に基づいて使用することができます。
製品に付属するすべての資料の全部または一部を、ダイキン工業株式会社の書面による許可を得ることなく複写、複製、転用することはできません。
記載内容は、予告なく変更することがあります。

～ 目次 ～

1. はじめに

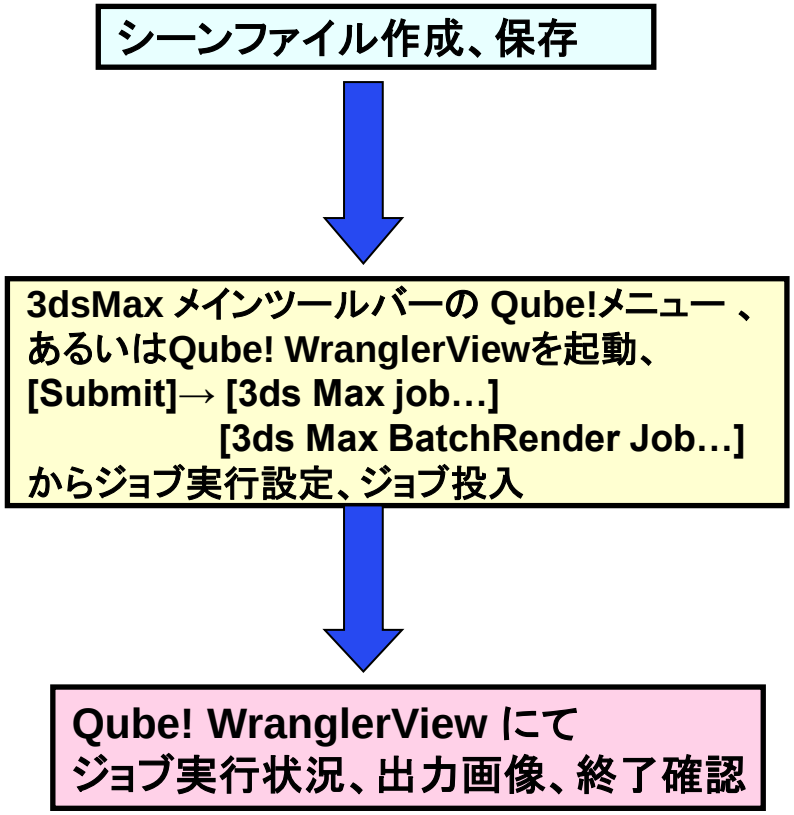
本ガイドは、Qube! クイックユーザーガイド 3dsMax編です。

詳細につきましては、3ds Max については 3ds Max オンラインヘルプ、また Qube! に関しましては、http://docs.pipelinefx.com 、

または、GUI の [Help]→[Qube User Guide](pdf) 、または、[スタート]→[プログラム]→[PipelineFX]→[User Manual](pdf)を参照ください。

2. ジョブ投入フロー (3ds Max編)

ジョブ投入フローの概要は以下の通りです。詳細説明については、左側で表示しましたページにてご覧いただけます。



3dsMax メニューでの操作 → 9 ～ 19 ページ

Qube! からの操作 → 20 ページ

21ページ

2－1. 3ds Max Jobtypeのインストールとセットアップ

＜クライアント＞

3ds Max で Qube!　Jobtypeを使用するには、qubeInstallerでClient Installを行います。*1
C:¥Program Files¥pfx¥jobtypes¥_3dsmax¥loadqube.mzp を
C:¥Program Files¥Autodesk¥3ds Max <ver>¥scripts¥Startup にコピーします。（3ds Maxインストール先に応じて変更してください）

※基本的にインストーラによりコピーされますが、後から3ds Maxをインストールした場合などでは手動でコピーが必要となります。
3ds Max を起動し、メインツールバー上に、Qube! メニューが表示されることを確認します。表示されない場合は、上記の
loadqube.mzpファイルが存在するかご確認ください。

レンダリングジョブを実行する前に、対象となるシーンの作成、保存を済ませます。
データ(プロジェクト、シーン、テクスチャ)の保存先としては、Qube! クライアント、Qube! Worker ホスト両方がアクセス可能となる
ネットワークドライブへ保存します。

*1: 詳細は、Qube! インストールガイドをご参照ください。

2－1. 3ds Max Jobtypeのインストールとセットアップ(Cont'd 1)

＜ワーカー＞（http://docs.pipelinefx.com/display/QUBE/Installing+and+Licensing+3ds+Max+on+a+Worker+Node より抜粋）

UACをOFF、Interactive Services Dialog Detection Serviceが動作していれば停止してください（恒久的に）。Microsoft Security Essentialsをご使用の場合は、通信が阻害されることがありますので、C:¥ProgramData¥Pfx¥Qube以下を除外する設定を行ってください。

管理者権限で、3ds Maxをフルインストールしてください。3ds Max 2018以前のバージョンでは、レンダラがBackburnerを参照していますので、3ds Maxのバージョンに対応したBackburnerもインストールしてください。インストール後、Backburner Serverを一度だけ起動し、終了させます。以後は起動させる必要はありません。また、Qube! Workerサービスと競合しないよう、Backburnerサービスは自動起動しないよう設定してください。

Qube!は、qubeInstallerでWorker Install*1を行います。各Jobtypeも一緒にインストールされますので、
C:¥Program Files¥pfx¥jobtypes¥_3dsmax¥loadqube.mzp を
C:¥Program Files¥Autodesk¥3ds Max <ver>¥scripts¥Startup にコピーします。（3ds Maxインストール先に応じて変更してください）
※基本的にインストーラによりコピーされますが、後から3ds Maxをインストールした場合などでは手動でコピーが必要となります。

3ds Maxの実行にはユーザプリファレンスとユーザディレクトリを必要とします。また、Windows側のユーザアカウント初期設定もありますので、ジョブ投入時に実際に使用するアカウントでWorkerにログインし（下記参照）、一度は3ds Maxを起動する必要があります。起動して、GUIまで表示されたら、メインツールバー上に、Qube! メニューが表示されることを確認します。表示されない場合は、上記のloadqube.mzpファイルが存在するかご確認ください。正常に表示されましたら終了して構いません。

※ジョブ実行モード*1(proxy_execution_mode)の設定により、以下の各アカウントでログインし、3ds Maxを起動し、終了させてください。一時的に3dsMaxのライセンスを使用しますが、レンダリング時は使用しません。

3ds Maxの制約上、アカウントには、管理者権限（Administrator権限、ドメインでご使用の場合は、Domain Admins権限）が必要です。各ユーザアカウントに管理者権限を与えるのが不都合な場合は、proxyモードでの運用をお勧めします。

（デフォルト）Desktop Userモードの場合、Desktop Workerとして実行したいユーザアカウントとパスワード

proxyモードの場合、Username : **qubeproxy**（表示名:Qube! Proxy）　Password : **Pip3lin3P@$$wd**

Userモードの場合、クライアント側でジョブ投入時に使用するユーザアカウントとパスワード

※Qube!は3ds Maxをスレーブモードで使用しますので、レンダリング中は3ds Maxのライセンスは消費しません。

*1: 詳細は、Qube! インストールガイドをご参照ください。

2－1. 3ds Max Jobtypeのインストールとセットアップ(Cont'd 2)

＜ワーカー(続)＞

ジョブ投入時には以下の状態になっているようにします。

(デフォルト) Desktop Userモードの場合、Desktop Workerとして実行したいユーザアカウントとパスワードでログインした状態にしておきます。

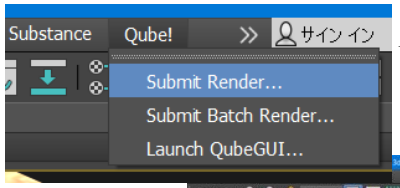
Proxyモードの場合、Username : **qubeproxy** (表示名:Qube! Proxy)　Password : **Pip3lin3P@$$wd** にて、ログインした状態(裏で良いです)にしておきます。

Userモードの場合、クライアント側でジョブ投入時に使用するユーザアカウントとパスワードにて、ログインした状態(裏で良いです)にしておきます。

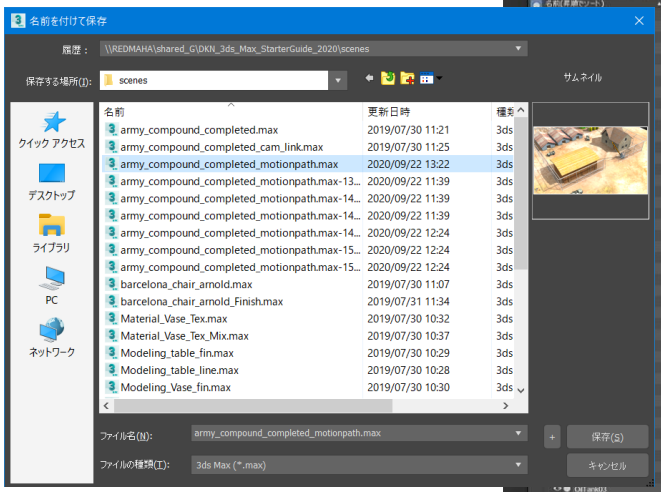
2－2. 3dsMax GUI（Qube! プラグイン）からの投入

レンダリングジョブを実行する前に、対象となるシーンの作成、保存を済ませます。

データ(プロジェクト、シーン、テクスチャ)の保存先としては、Qube! クライアント、Qube! Worker ホスト両方がアクセス可能となるネットワークドライブへ保存します。
データ保存の完了後、メニューの [Qube!]→[Submit Render...] または [Submit Batch Render…] をクリックします。Submit ダイアログが起動します。

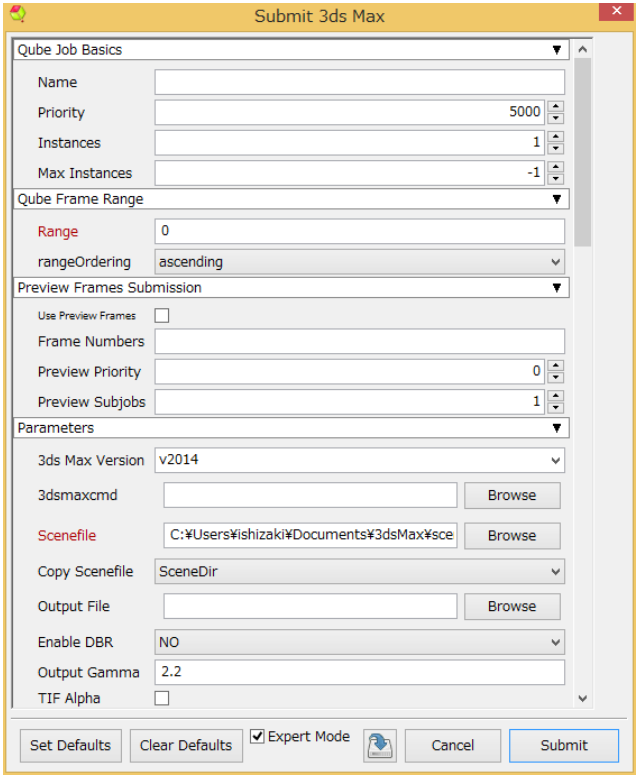


＜シーンを保存＞
UNCパス上のネットワークドライブに保存



以下のジョブ投入メニューは、”Expert Mode”で表示させたものです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。

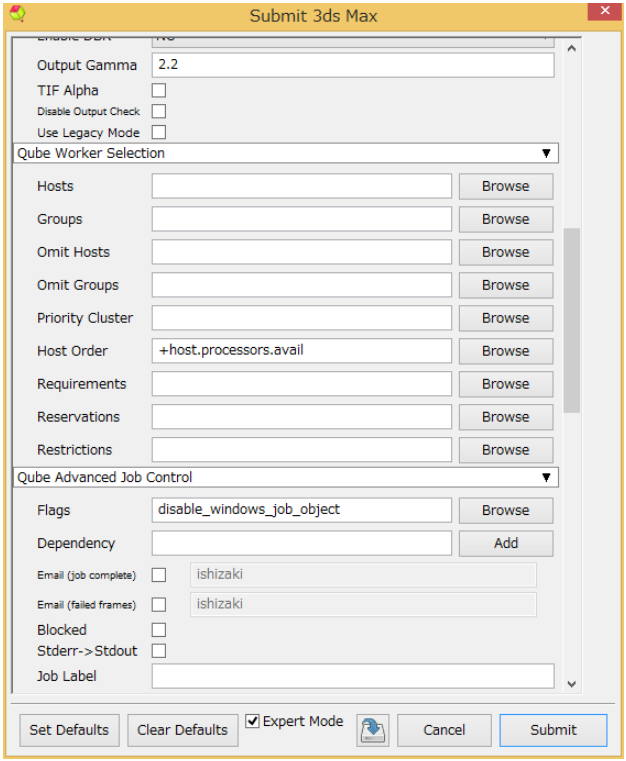


<Submit Render…>
ダイナミックアロケーションを使用したジョブ投入を行います。
フレーム毎にワーカーを割り当てますので、Submit Batch Renderにあるようなな Execution指定(Chunks設定)はありません。

== Qube Job Basics ==
Name ・・・ 実行ジョブの名称です。一意である必要はありません。
Priority・・・デフォルトは 9999 です。ジョブの優先度としては最低値となります。(値が低い方が優先度が高くなります。i.e. 9999 < 9998 ← ジョブ優先度)
Instances ・・・ 同時に実行されるタスク数。デフォルトは 1 です。
Max Instances・・・SmartShareが有効なとき(supervisorのqb.conf内の **supervisor_smart_share_mode** が”jobs”)、この数を上限に、可能な数までInstance数が自動的に拡張されます。(0:SmartShareを使用しない、-1:上限無し。ただし、supervisorのqb.conf内の **supervisor_max_cpus_limit** で制限可)
== Qube Frame Range ==
Range ・・・ レンダリングするフレーム範囲を指定します。(例:1-100 or 1-100x3 or 1,3,7,10)
1-100x3の x3はステップ値を表します。
rangeOrdering ・・・ 投入するフレームの順番を指定します。ascending(昇順)、descending(降順)、Binary(最初,最後,真ん中,残り..の順)
== Preview Frames Submission ==
Use Preview Frames ・・・ プレビュー用のフレームを投入します。
Frame Numbers ・・・ プレビュー用にレンダリングするフレーム番号を指定します。指定しない場合は、最初,最後,真ん中のフレームをレンダリングします。
Preview Priority・・・プレビュー用のフレームの優先度を指定します。デフォルトは0で最優先になります。
Preview Subjobs ・・・ プレビュー用に投入するサブジョブ数を指定します。
== Parameters ==
3ds Max Version ・・・ 3ds Maxのバージョンを指定します。実際にファイルに書き込まれるバージョン記述(例: 15000:v2013, 16000:v2014, 1700:v2015)での指定も可能です。各ワーカーでインストールパスが統一されていない場合など、3dsmaxcmdよりも、こちらで指定すると便利です。
3dsmaxcmd ・・・ 3dsmaxcmd.exeをフルパスで指定します。
Scenefile ・・・レンダリングするシーンファイル名を指定します。基本的にUNCパスで記述します。
Copy Scenefile・・・subjobの作業用にシーンをコピーするフォルダを指定します。(安定動作のため)
Output File ・・・ 出力ファイルを拡張子付きで指定します。基本的にUNCパスで記述します。
(例)¥¥serverA¥test¥imgA_.png → imgA_0001.png, imgA_0002.png,… となります。
Enable DBR・・・mentalrayまたはV-rayでDistributed Bucket Renderingを使用します。

ジョブ設定値の項目は以下の通りです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。



<Submit Render…>(Cont’d 1)

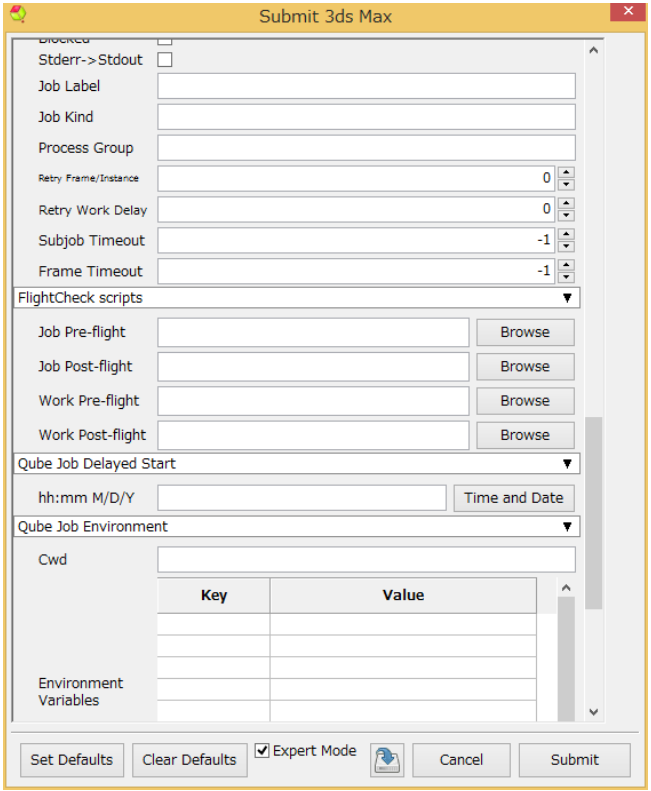
Output Gamma・・・指定した場合、出力画像へガンマ値が適用されます。
TIF Alpha・・・出力されるTIFフォーマットへアルファチャンネルを有効にします。
Disable Output Check・・・出力されるファイルの存在やサイズのチェックを無効にします。
Use Legacy Mode・・・3ds Max2008以前のバージョンではONにします。

== Qube Worker Selection ==
Hosts・・・ジョブ実行ホストを指定します。
Groups・・・ジョブ実行Workerグループを指定します。
Omit Hosts・・・ジョブ実行を抑制するホストを指定します。
Omit Group・・・ジョブ実行を抑制するWorkerグループを指定します。
Priority Cluster・・・ジョブ実行時の優先Workerクラスタを指定します。
Host Order・・・ジョブ実行時のホスト順番の優先度条件を指定します。+が付くと高くなり、-が付くと低くなります。
Requirements・・・ジョブ実行要件式を定義します。OSの種類、同種のジョブのみの投入制限ができます。(Job Kind欄参照)
Reservations・・・ジョブ実行要件指定を定義します。ジョブスロット数とメモリが指定できます。+が付いている場合は(AllがON)、使用可能なすべてのジョブスロットを使用します。
Restrictions・・・ジョブ実行抑制クラスタを指定します。

== Qube Advanced Job Control ==
Flags・・・ジョブ実行フラグを指定します。
Dependency・・・実行ジョブ依存関係の指定式を定義します。i.e. ジョブ A が終わった後に実行する
Mail(job complete)・・・チェックを入れた場合、右側の Email Address フィールドに指定したメールアドレス(コンマで複数指定可)にジョブ終了通知メールを送信します。(Success or Failure)
Mail(failed frames)・・・チェックを入れた場合、右側の Email Address フィールドに指定したメールアドレス(コンマで複数指定可)にジョブ異常終了通知メールを送信します。(Success or Failure)
Blocked・・・ ジョブを”Block”された状態で投入します。直ちに実行したくないときに使用し、手動で開始できます。
Stderr->Stfout・・・Stderrに出力されるエラーをStdoutにリダイレクトします。

ジョブ設定値の項目は以下の通りです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。



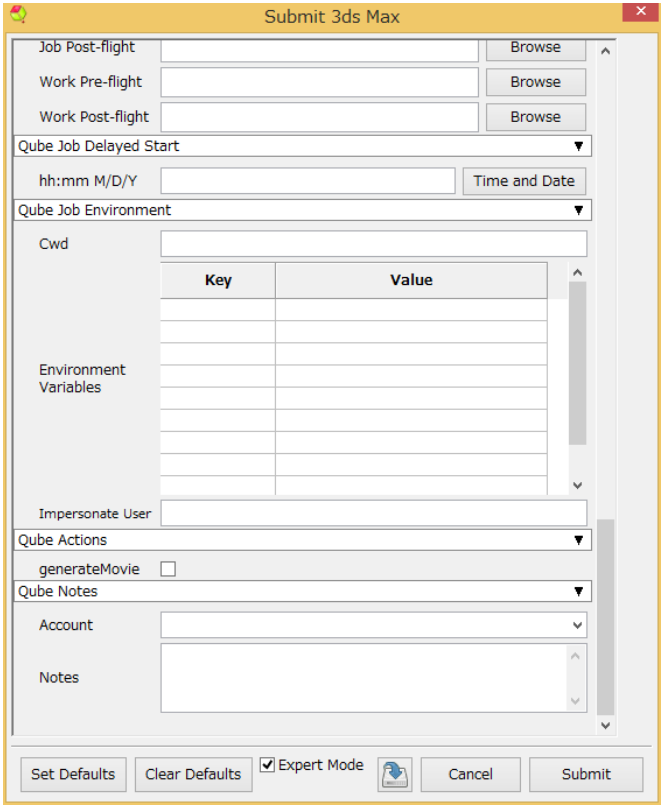
<Submit Render…>(Cont’d 2)

Job Label･･･ ジョブ識別のためのラベルを指定します。Process Group内でユニークな名前である必要があります。
Job Kind･･･ ジョブを識別するための任意の情報を設定します。Requirements欄の「Only 1 of a “kind” of job」が指定された際に、ジョブの種類を区別するために参照されます。
Process Group･･･ ジョブを組織的にまとめるためのグループ名を設定します。デフォルトはjobidです。Job Labelと組み合わせてユニークな名前にする必要があります。
Retry Frame/Instance･･･ フレーム/ジョブインスタンスがfailしたときに、リトライする回数を指定します。-1 を指定すると、studioのデフォルト値を使用します。(Preferences - Studio Defaults)
Retry Work Delay･･･ failしたframeに対して自動的にリトライをかける前の待ち時間を秒で指定します。
Subjob Timeout･･･ サブジョブがタイムアウトしてKillされる時間を秒で指定します。時間がかかりすぎるジョブを強制終了させることができます。-1 はこの機能を無効化します。
Frame Timeout･･･ フレーム単位の計算がタイムアウトしてKillされる時間を秒で指定します。時間がかかりすぎるジョブを強制終了させることができます。-1 はこの機能を無効化します。

== FlightCheck scripts == ※各スクリプトでゼロ以外を返すと、ジョブが”failed”になります。
Job Pre-flight･･･ 各ワーカーで、ジョブインスタンスが実行される前に実行されるスクリプトを指定します。
Job Post-flight･･･ 各ワーカーで、ジョブインスタンスが実行された後に実行されるスクリプトを指定します。
Work Pre-flight･･･各ワーカーで、各フレームまたはアジェンダが実行される前に実行されるスクリプトを指定します。
Work Post-flight･･･各ワーカーで、各フレームまたはアジェンダが実行された後に実行されるスクリプトを指定します。

ジョブ設定値の項目は以下の通りです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。



<Submit Render…>(Cont’d 3)

== Qube Job Delayed Start ==
hh:mm M/D/Y・・・ ジョブを実行する時間を指定します。

== Qube Job Environment ==
Cwd・・・ ジョブ実行時のカレントワークディレクトリを指定します。
Environment Variables・・・ ジョブ実行時に使用する環境変数を設定します。既存の環境変数に上書きできます。
Impersonate User・・・ ジョブを投入する際、指定したユーザとして投入します。デフォルトはカレントユーザです。指定する場合は、Qube! WranglerViewのUser Permissionsタブで、Impersonate権限を与える必要があります。

== Qube Actions ==
generateMovie・・・ 出力イメージからムービーを作成するためのジョブにリンクを追加します。

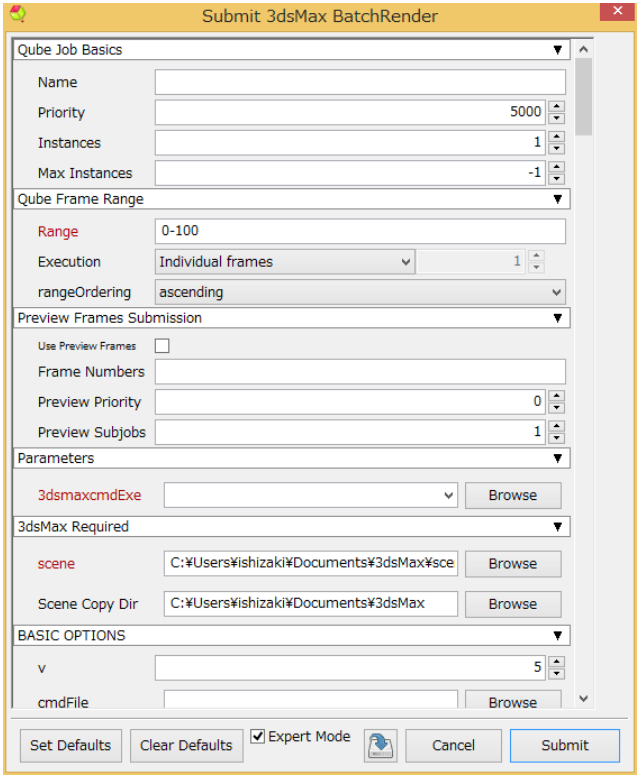
== Qube Notes ==
Account・・・任意のアカウントやプロジェクトデータを設定します。（ユーザ定義）
Notes・・・ このジョブについて、コメントを記述します。


または

・・・クリックするとジョブ投入設定をファイルとして保存できます。
保存した設定でのSubmitは、Submit - [Job from file…]でそのファイルを選択します。

以下のジョブ投入メニューは、”Expert Mode”で表示させたものです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。



<Submit Batch Render…>
数フレームをまとめてワーカーに割り当てます。

== Qube Job Basics ==
Name ・・・ 実行ジョブの名称です。一意である必要はありません。
Priority・・・デフォルトは 9999 です。ジョブの優先度としては最低値となります。(値が低い方が優先度が高くなります。i.e. 9999 < 9998 ← ジョブ優先度)
Instances ・・・ 同時に実行されるタスク数。デフォルトは 1 です。
Max Instances・・・SmartShareが有効なとき（supervisorのqb.conf内の **supervisor_smart_share_mode** が”jobs”）、この数を上限に、可能な数までInstance数が自動的に拡張されます。(0:SmartShareを使用しない、-1:上限無し。ただし、supervisorのqb.conf内の **supervisor_max_cpus_limit** で制限可）

== Qube Frame Range ==
Range ・・・ レンダリングするフレーム範囲を指定します。(例：1-100 or 1-100x3 or 1,3,7,10)
1-100x3の x3はステップ値を表します。
Execution・・・ フレーム範囲をどのように分割するかを指定します。（Individual frames:1フレーム単位、Chunks with n frames: nフレーム単位、Split into n partitions: n個のかたまりに分割）
rangeOrdering ・・・ 投入するフレームの順番を指定します。ascending（昇順）、descending（降順）、Binary（最初,最後,真ん中,残り..の順）

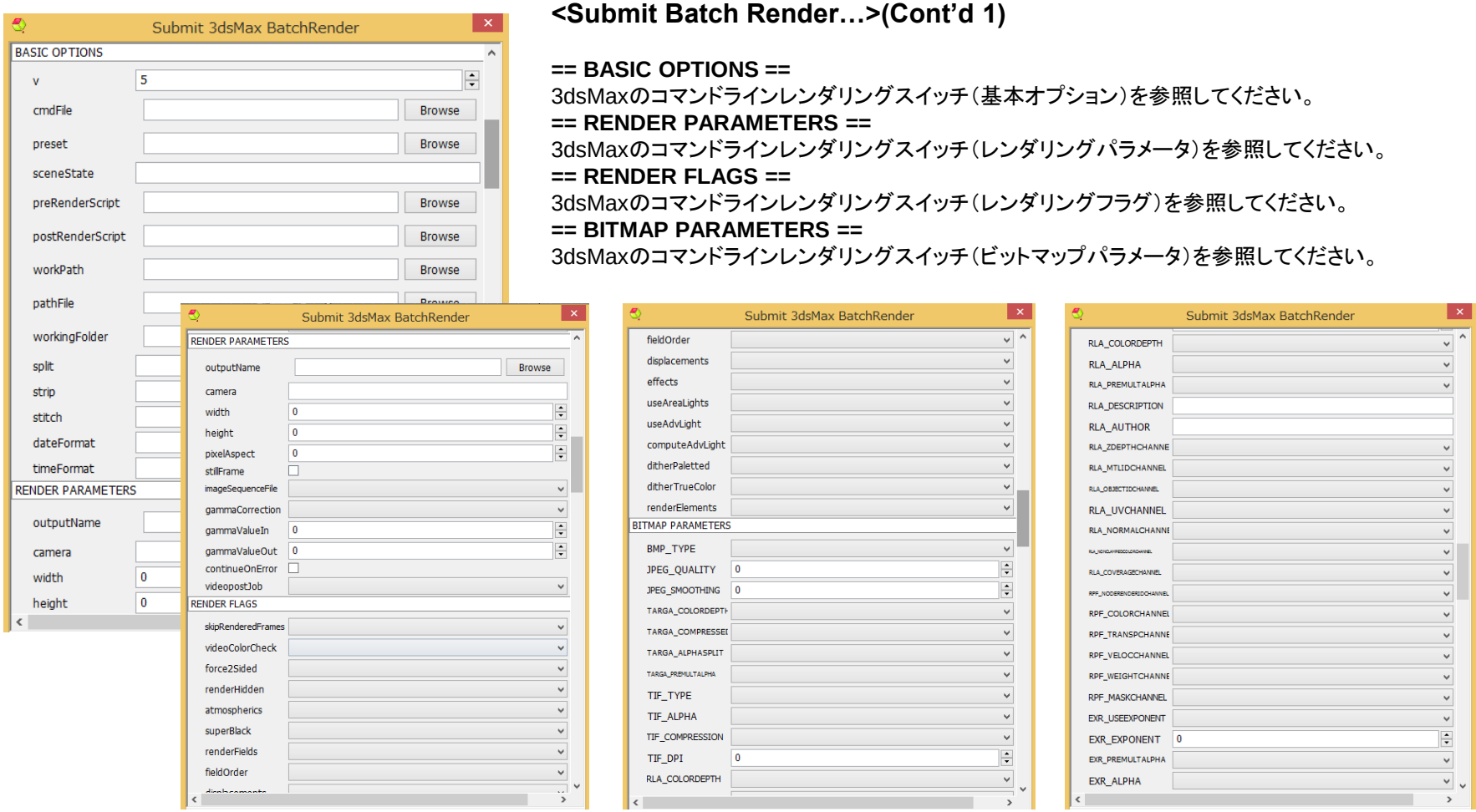
== Preview Frames Submission ==
Use Preview Frames ・・・ プレビュー用のフレームを投入します。
Frame Numbers ・・・ プレビュー用にレンダリングするフレーム番号を指定します。指定しない場合は、最初,最後,真ん中のフレームをレンダリングします。
Preview Priority・・・プレビュー用のフレームの優先度を指定します。デフォルトは0で最優先になります。
Preview Subjobs ・・・ プレビュー用に投入するサブジョブ数を指定します。

== Parameters ==
3dsmaxcmdExe ・・・ 3dsmaxcmd.exeをフルパスで指定します。

== 3dsMax Required ==
Scenefile ・・・レンダリングするシーンファイル名を指定します。基本的にUNCパスで記述します。
Scene Copy Dir・・・subjobの作業用にシーンをコピーするフォルダを指定します。

ジョブ設定値の項目は以下の通りです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。

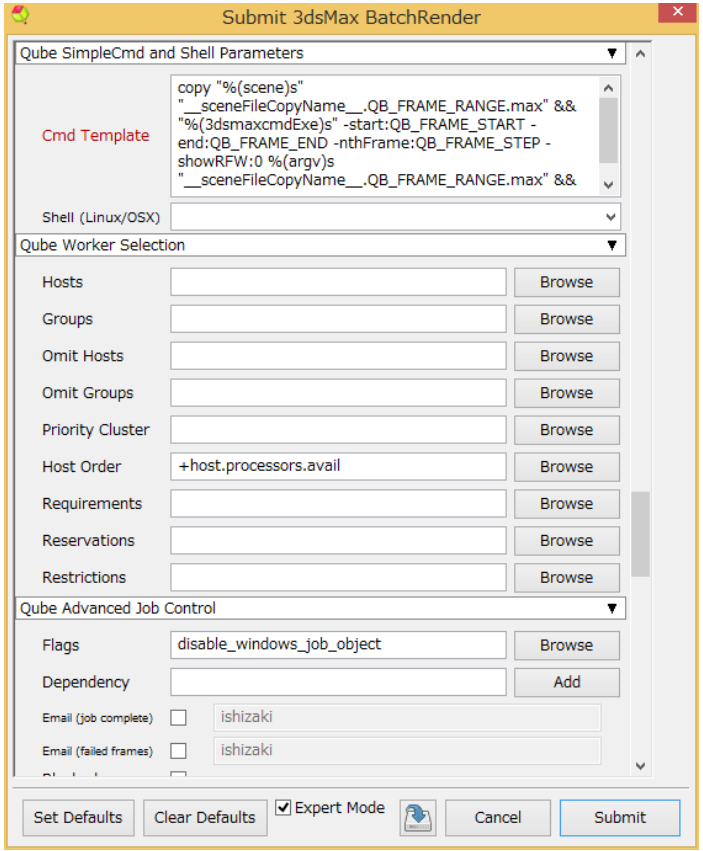


<Submit Batch Render…>(Cont’d 1)

== BASIC OPTIONS ==
3dsMaxのコマンドラインレンダリングスイッチ（基本オプション）を参照してください。
== RENDER PARAMETERS ==
3dsMaxのコマンドラインレンダリングスイッチ（レンダリングパラメータ）を参照してください。
== RENDER FLAGS ==
3dsMaxのコマンドラインレンダリングスイッチ（レンダリングフラグ）を参照してください。
== BITMAP PARAMETERS ==
3dsMaxのコマンドラインレンダリングスイッチ（ビットマップパラメータ）を参照してください。

ジョブ設定値の項目は以下の通りです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。



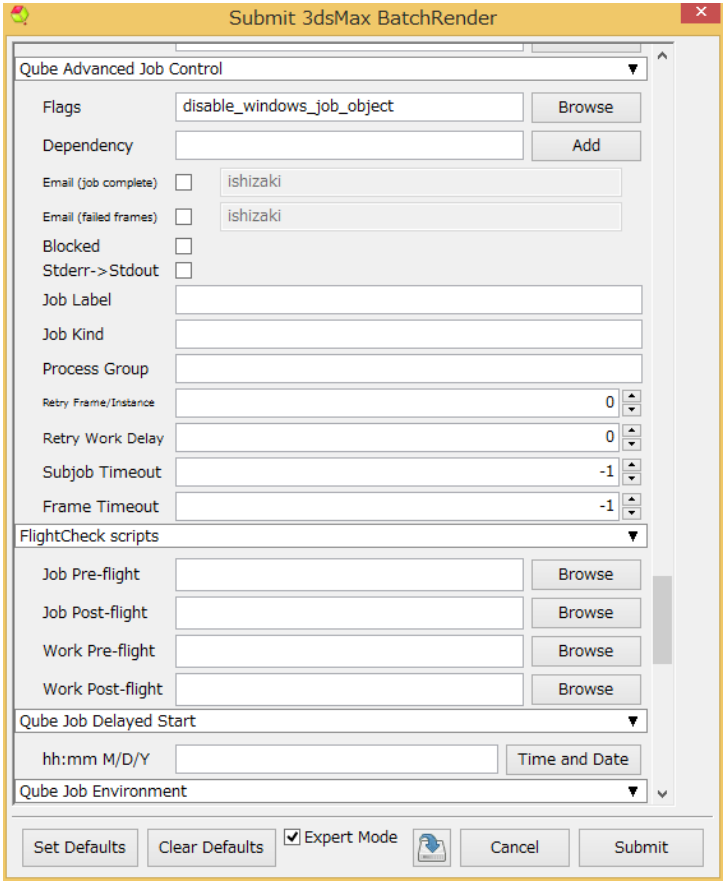
<Submit Batch Render…>(Cont’d 2)

== Qube SimpleCmd and Shell Parameters ==
Cmd Template・・・コマンドを生成するためのテンプレートを指定します。
Shell(Linux/OSX)・・・コマンド実行時のLinux/OSXシェルを指定します。

== Qube Worker Selection ==
Hosts・・・ジョブ実行ホストを指定します。
Groups・・・ジョブ実行Workerグループを指定します。
Omit Hosts・・・ジョブ実行を抑制するホストを指定します。
Omit Group・・・ジョブ実行を抑制するWorkerグループを指定します。
Priority Cluster・・・ジョブ実行時の優先Workerクラスタを指定します。
Host Order・・・ジョブ実行時のホスト順番の優先度条件を指定します。+が付くと高くなり、-が付くと低くなります。
Requirements・・・ジョブ実行要件式を定義します。OSの種類、同種のジョブのみの投入制限ができます。（Job Kind欄参照）
Reservations・・・ジョブ実行要件指定を定義します。ジョブスロット数とメモリが指定できます。+が付いている場合は（AllがON）、使用可能なすべてのジョブスロットを使用します。
Restrictions・・・ジョブ実行抑制クラスタを指定します。

ジョブ設定値の項目は以下の通りです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。



<Submit Batch Render…>(Cont’d 3)

== Qube Advanced Job Control ==

Flags・・・ジョブ実行フラグを指定します。

Dependency・・・実行ジョブ依存関係の指定式を定義します。i.e. ジョブ A が終わった後に実行する

Mail(job complete)・・・チェックを入れた場合、右側の Email Address フィールドに指定したメールアドレス（コンマで複数指定可）にジョブ終了通知メールを送信します。(Success or Failure)

Mail(failed frames)・・・チェックを入れた場合、右側の Email Address フィールドに指定したメールアドレス（コンマで複数指定可）にジョブ異常終了通知メールを送信します。(Success or Failure)

Blocked・・・ ジョブを”Block”された状態で投入します。直ちに実行したくないときに使用し、手動で開始できます。

Stderr->Stfout・・・Stderrに出力されるエラーをStdoutにリダイレクトします。

Job Label・・・ ジョブ識別のためのラベルを指定します。Process Group内でユニークな名前である必要があります。

Job Kind・・・ ジョブを識別するための任意の情報を設定します。Requirements欄の「Only 1 of a “kind” of job」が指定された際に、ジョブの種類を区別するために参照されます。

Process Group・・・ ジョブを組織的にまとめるためのグループ名を設定します。デフォルトはjobidです。Job Labelと組み合わせてユニークな名前にする必要があります。

Retry Frame/Instance・・・ フレーム/ジョブインスタンスがfailしたときに、リトライする回数を指定します。-1 を指定すると、studioのデフォルト値を使用します。（Preferences - Studio Defaults)

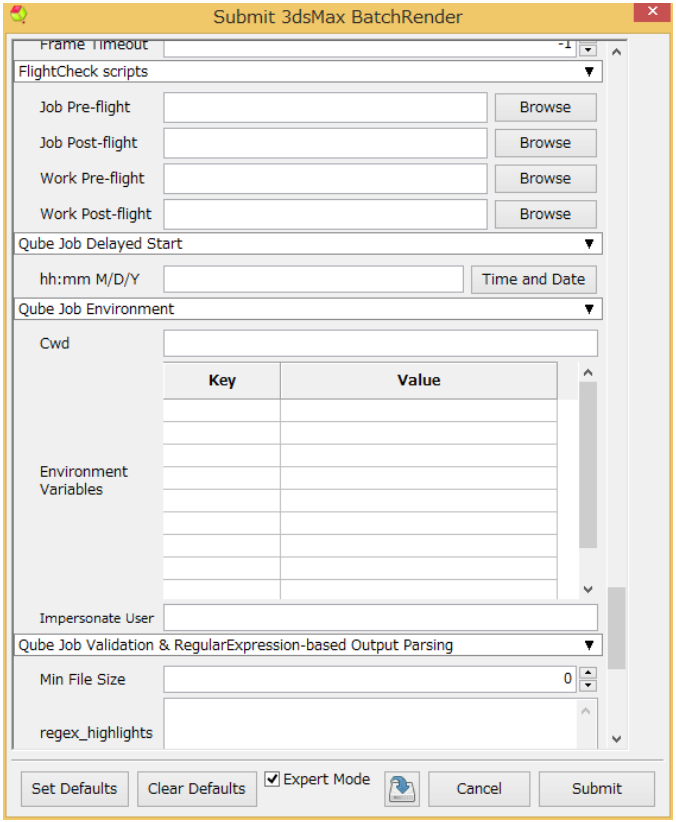
Retry Work Delay・・・ failしたframeに対して自動的にリトライをかける前の待ち時間を秒で指定します。

Subjob Timeout・・・ サブジョブがタイムアウトしてKillされる時間を秒で指定します。時間がかかりすぎるジョブを強制終了させることができます。-1 はこの機能を無効化します。

Frame Timeout・・・ フレーム単位の計算がタイムアウトしてKillされる時間を秒で指定します。時間がかかりすぎるジョブを強制終了させることができます。-1 はこの機能を無効化します。

ジョブ設定値の項目は以下の通りです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。



<Submit Batch Render…>(Cont’d 4)

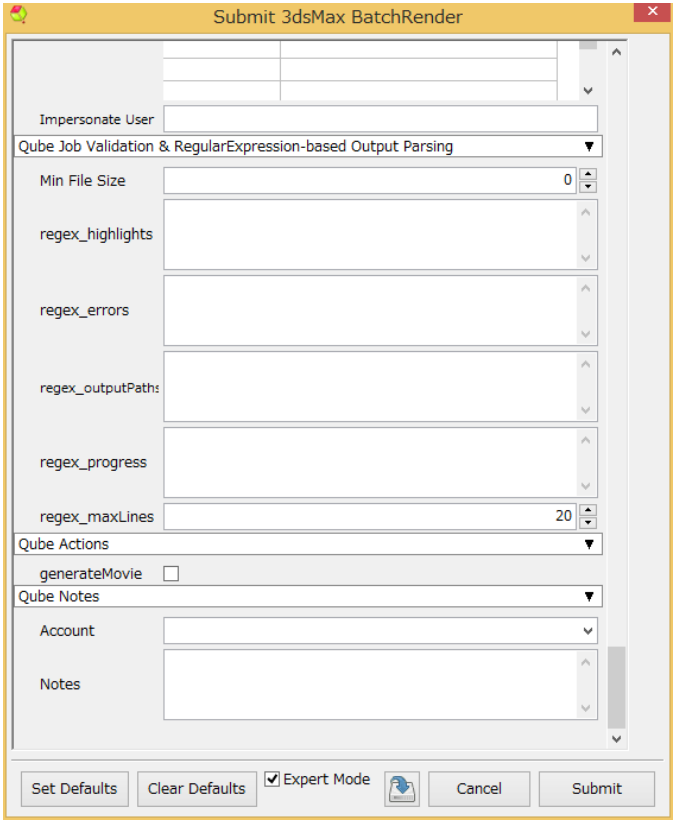
== FlightCheck scripts ==　※各スクリプトでゼロ以外を返すと、ジョブが”failed”になります。
Job Pre-flight･･･ 各ワーカーで、ジョブインスタンスが実行される前に実行されるスクリプトを指定します。
Job Post-flight･･･ 各ワーカーで、ジョブインスタンスが実行された後に実行されるスクリプトを指定します。
Work Pre-flight･･･各ワーカーで、各フレームまたはアジェンダが実行される前に実行されるスクリプトを指定します。
Work Post-flight･･･各ワーカーで、各フレームまたはアジェンダが実行された後に実行されるスクリプトを指定します。

== Qube Job Delayed Start ==
hh:mm M/D/Y･･･ ジョブを実行する時間を指定します。

== Qube Job Environment ==
Cwd･･･ ジョブ実行時のカレントワークディレクトリを指定します。
Environment Variables･･･ ジョブ実行時に使用する環境変数を設定します。既存の環境変数に上書きできます。
Impersonate User･･･ ジョブを投入する際、指定したユーザとして投入します。デフォルトはカレントユーザです。指定する場合は、Qube! WranglerViewのUser Permissionsタブで、Impersonate権限を与える必要があります。

ジョブ設定値の項目は以下の通りです。***下線項目は必須入力項目です。***

指定が必要な全ての項目に入力した上で、”Submit” ボタンをクリックし、ジョブを投入します。



<Submit Batch Render…>(Cont’d 5)

== Qube Job Validation & RegularExpression-based Output Parsing ==
Min File Size・・・ 出力パスとして認識される最小サイズを指定します。(0だと無効になります)
regex_highlights・・・ stdout/stderrメッセージで、ハイライト表示される正規表現を指定します。
regex_errors・・・ stdout/stderrメッセージで、エラー表示される正規表現を指定します。
regex_outputPaths・・・ stdout/stderrメッセージで、画像の出力パスとして認識される正規表現を指定します。
regex_progress・・・ stdout/stderrメッセージで、frame/chunk進行中として認識される正規表現を指定します。
regex_maxLines・・・ stdout/stderrメッセージで、正規表現とパターンマッチする最大行数を指定します。

== Qube Actions ==
generateMovie・・・ 出力イメージからムービーを作成するためのジョブにリンクを追加します。

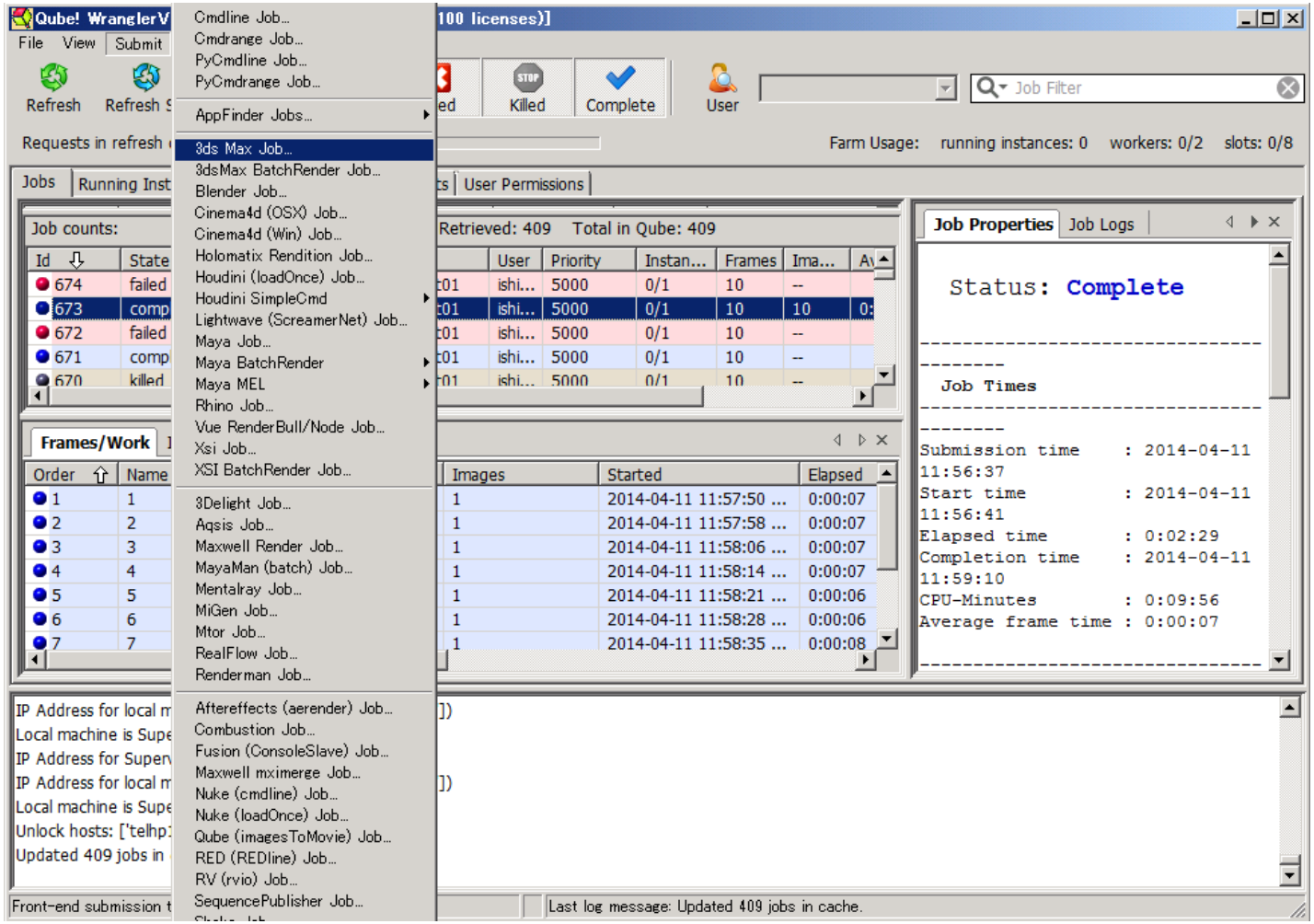
== Qube Notes ==
Account・・・任意のアカウントやプロジェクトデータを設定します。(ユーザ定義)
Notes・・・ このジョブについて、コメントを記述します。


または

・・・クリックするとジョブ投入設定をファイルとして保存できます。
保存した設定でのSubmitは、Submit - [Job from file…]でそのファイルを選択します。

2－3. Qube! Wrangler View からの投入

[スタート]→[プログラム]→[PipelineFX]→[Qube!] をクリック、Qube! Wrangler View を起動、[Submit]→[3ds Max Job…] または [3dsMax BatchRender Job]を選択します。設定項目は、3ds Max GUIからのものと同じです。



2－4. ジョブ投入後のジョブ確認方法について

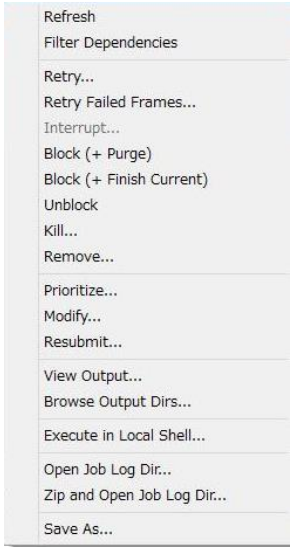
投入したジョブが完了したかについては、完了通知メール送信設定をされている場合でも、必ず Qube! WranglerView を起動し確認するかたちになります。([スタート]→[プログラム]→[PipelineFX]→[Qube!])

“Jobs” タブで実行中、あるいは実行完了したジョブの概要をブラウズします。

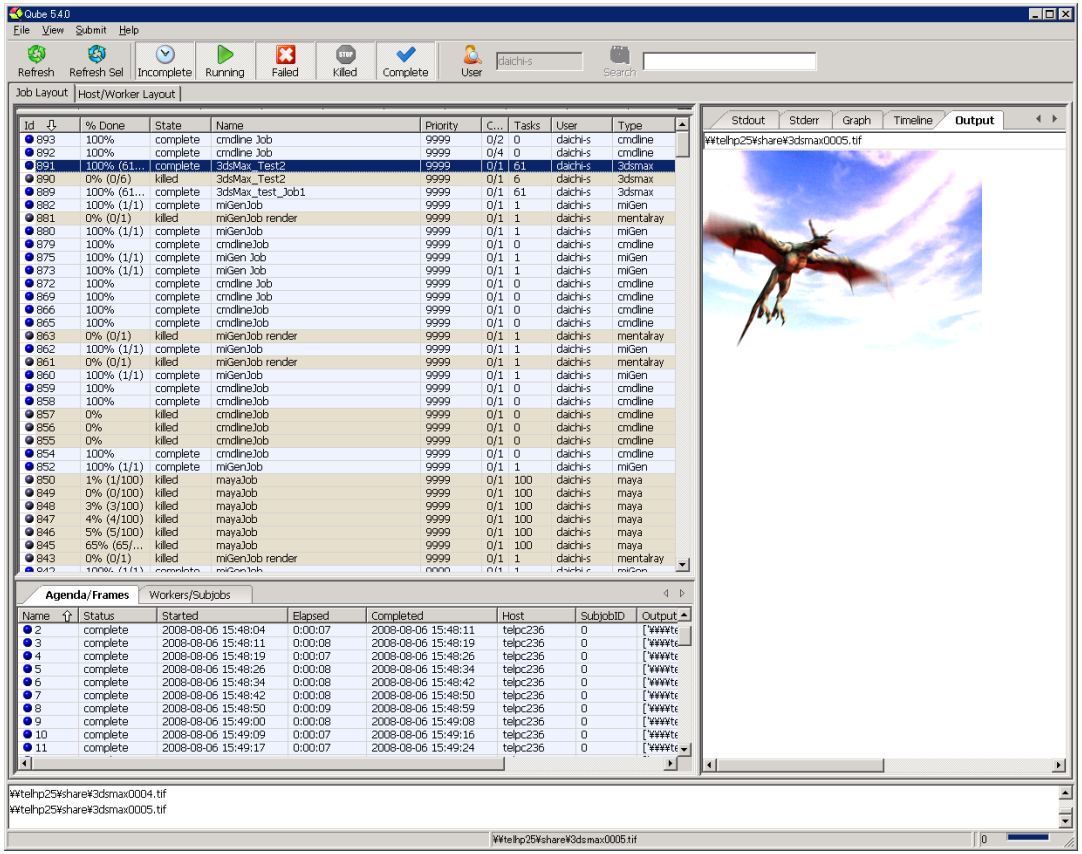
また各タブをクリックする事で、[Properties] (ジョブ概要)、[Stdout] (ジョブ標準出力)、[Stderr] (ジョブエラー出力)、[Output] (出力画像確認※) などが行えます。

ジョブの状態は “State” 表示で “running” (実行中)、”complete” (完了)、”failed” (失敗) であるか確認します。

実行中のジョブをキャンセルする場合には、ジョブ ID 項目を選択の上、マウス右クリックし、メニューの “Kill” を選択します。



Frames/Work から目的の Work を選択の上、左側　ウインドウの”Output” タブをクリックする事で、画像表示をさせます。対応画像フォーマット以外を表示させるには、File > PreferencesのImage Viewerにて外部ビューアが指定できます。



※ 対応画像フォーマット～ TIFF(標準フォーマット)、PNG、JPG、BMP、GIF

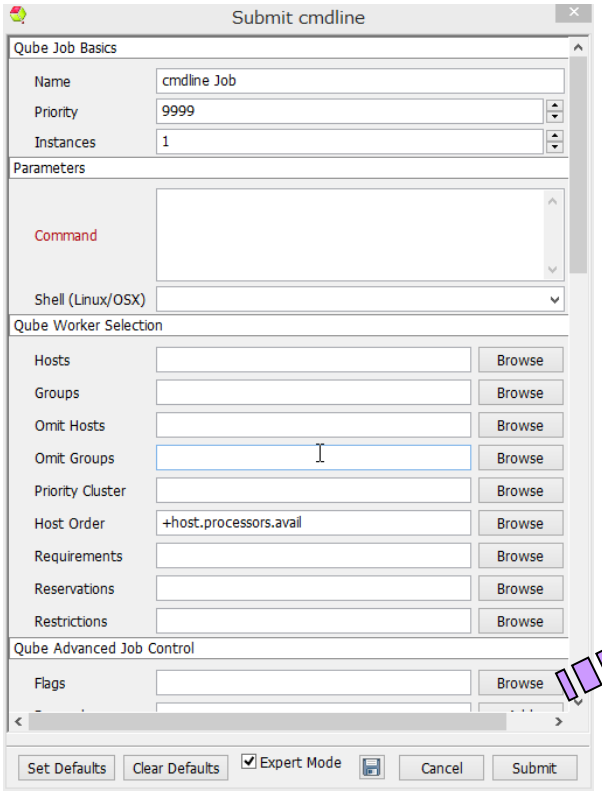
3. コマンドラインジョブ投入フロー

3－1. コマンドラインジョブ

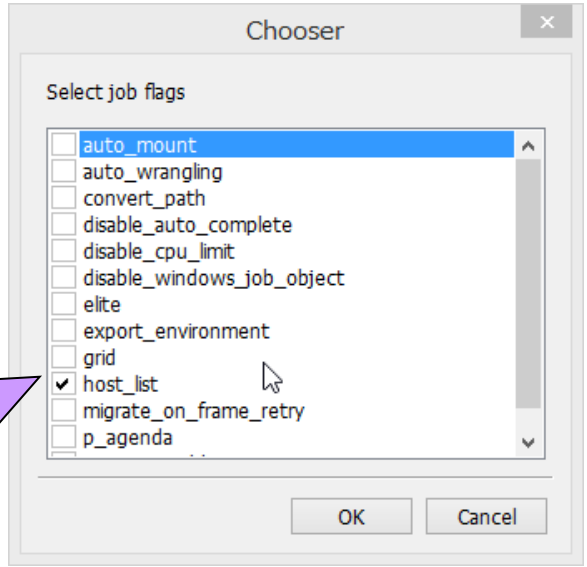
Qube! WranglerView からレンダリングジョブ以外に使用可能なコマンドジョブの実行が可能です。この機能を活用する事により、レンダーファームの管理～リブート、シャットダウン、システム設定ファイルの差し替え、ソフトウェアのインストールなど～が行えます。※

[Submit]→[Cmdline Job…] をクリックすると以下のウインドウが開きます。

Command フィールドに実行コマンドラインを入力、Hosts で実行ホストの指定、あるいは Flags フィールドで Browse ボタンを押し以下にある Chooser ウインドウで “host_list” にチェックを入れた上でジョブ実行を行うと、全ての Worker ホストで指定されたコマンドを実行します。

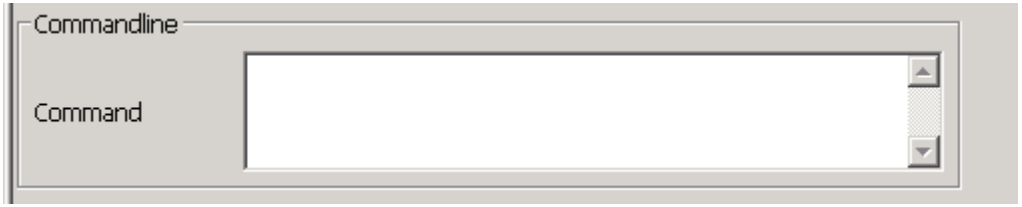


※ 実行条件は、proxy mode が User である場合、Administrators あるいは Superuser 権限でのジョブ投入、また proxy である場合は Proxy ユーザーが Administrators あるいは Superuser 権限 である事が必要になります。（ドメインの場合は、Domain Admins権限が必要です）



3－2. 応用編 ～ Qube！ジョブとしてのインストールパッケージ処理

コマンドラインジョブを応用していただくことにより、レンダーファームホストのソフトウェア管理工数を削減する事も可能です。共有ファイルシステムへインストールパッケージを保存し、特定のレンダーファームホストに対して、無人インストールを行うかたちで対応可能です。
実行にあたっては、Command フィールドに直接コマンドラインを記述する、あるいはそれらの記述を行っている実行バッチを指定する、などの方法となります。



各プラットフォーム毎の代表的なインストールコマンドとそのオプション指定について、以下に示します。

詳細はそれぞれのコマンドのヘルプをご参照ください。

-Windows (msiパッケージ)の場合-

msiexec /i "Z:¥XYZ.msi" /qn /liwearcmov Z:¥install.log

-OS X (インストーラーパッケージ) の場合-

/usr/sbin/installer -pkg /foo/share/XYZ.pkg -target "/Volumes“

-Linux (rpmパッケージ)の場合-

rpm –ivh /foo/share/XYZ.rpm

4. トラブルシューティング

■ failedになる、pendingのまま進まないなど、うまくレンダリングができない場合

ログにエラーメッセージ等が表示されることもありますが、基本的な事項として先ずは以下の件をご確認ください。
•ワーカーでレンダリング時に使用するログインアカウントに管理者権限があるか
（ドメインの場合、Domain Admins権限が必要です）
•UACがオフになっているか
•“Interactive Services Dialog Detection Service” が存在している場合、停止状態になっているか
•使用するアカウントで、ワーカーにログインし、一度 3dsMax を立ち上げ、終了しているか。ユーザプロファイルが作成され、メインツールバーにQube!メニューが組み込まれている必要があります。
•MSE(Microsoft Security Essentials)を使用している場合は、C:¥ProgramData¥Pfx¥Qube を除外登録しているか（登録していない場合、supervisorとworker間の通信が不安定になる現象が報告されています）
•各ワーカーから、同じパスでシーンやテクスチャ、出力フォルダ等にアクセスできるか。基本的にUNCパスで記述しますが、各ワーカーでパスが異なる場合は、worker_drive_mapやworker_path_map等の設定が必要となります。
•ワーカーの動作モードに応じて、ワーカーがログイン状態になっているかご確認ください。Serviceモードでもログイン状態にする必要がありますが、これは、3dsMaxの特性によります。P.8をご参照ください。
•Output Fileフィールドを使用している場合は、拡張子付きでファイル名を指定しているかご確認ください。
（例）¥¥serverA¥test¥imgA_.png → imgA_0001.png, imgA_0002.png,… と出力されます。

■ 投入時の設定の注意事項
一台のマシンで複数の 3ds Max ジョブを起動すると不安定になる場合があります。
回避するには、投入する際に Reservations で host.processors=1+ と指定してから投入します。（Browse > host.processorsのAllにチェックを入れます）。これでひとつのインスタンスがワーカーノード上の全てのスロットを専有し実行します。
これは、3ds Max ではマルチスレッドの制御自体に制約があるため、基本的に各インスタンス毎にコア数分のスレッドを立ち上げてレンダリングする、1台＝1インスタンスという形が望ましいためです。

4. トラブルシューティング(Cont'd 1)

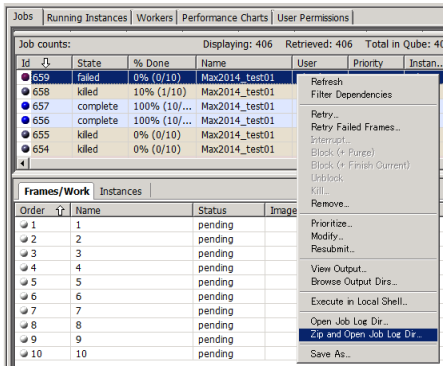
■ エラー解析に必要なログについて

● ジョブログ(http://docs.pipelinefx.com/display/QUBE/Job+logs)

位置：スーパーバイザマシン上の、C:¥ProgramData¥Pfx¥Qube¥logs¥job

各ジョブ単位で出力されるログです。投入時の設定、Qube!およびレンダラからの情報が出力されています。

スーパーバイザにて、ジョブを右クリック > Open Job Log Dir... でフォルダへアクセスでき、Zip and Open Job Log Dir... にて、Zipファイルにアーカイブ化できます。メールに添付して送付する際などに便利です。

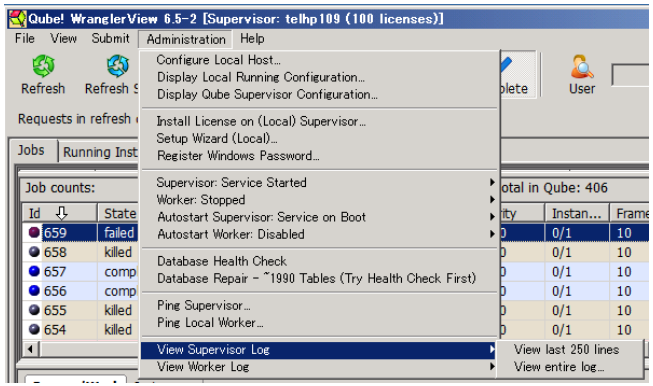


●Supervisor Logs(supelog)(http://docs.pipelinefx.com/display/QUBE/Supervisor+Logs)

位置：スーパーバイザマシン上の、C:¥ProgramData¥Pfx¥Qube¥logs¥supelog

スーパーバイザが出力するログです。各ジョブの割り振り状況やワーカーからのエラー、リトライ情報などファーム全体のマネジメントに関する情報が出力されています。

スーパーバイザにて、Qube! WranglerView > Administration > View Supervisor Log にて見ることができます。放っておくと大きなファイルになりがちですので、定期的に作成し直すことをお勧めします。（ファイル名を変更したり、削除したりすると、自動的に作成されます）



4. トラブルシューティング(Cont'd 2)

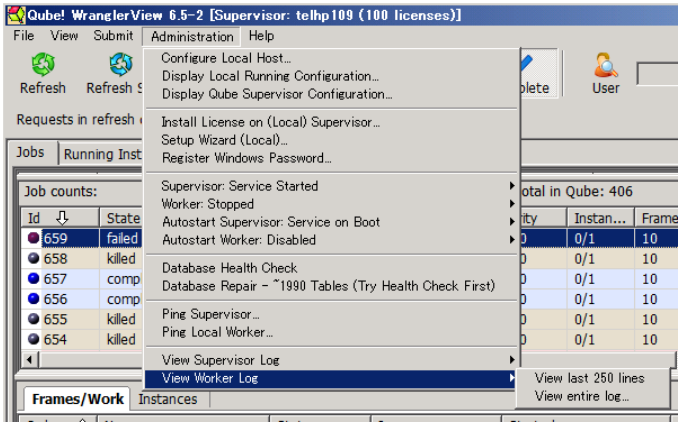
■ エラー解析に必要なログについて

● Worker Logs(http://docs.pipelinefx.com/display/QUBE/Worker+Logs)

位置：ワーカーマシン上の、C:¥ProgramData¥Pfx¥Qube¥logs¥workerlog

各ワーカー単位で出力されるログです。ワーカーのステータスや割り振られたジョブに対し、本ワーカーがどのようにレンダリング資源を割り当てたかなどの情報が出力されています。

ワーカーにて、Qube! WranglerView > Administration > View Worker Log にて見ることができます。放っておくと大きなファイルになりがちですので、定期的に作成し直すことをお勧めします。（ファイル名を変更したり、削除したりすると、自動的に作成されます）



■ Qube! サポートサイトもご参照ください。
https://www.comtec.daikin.co.jp/DC/prd/qube/support/
FAQ: https://www.comtec.daikin.co.jp/faq-qube/
開発元サポートサイト（英語）：https://www.pipelinefx.com/supportpage/

Qube!クイックガイド *3dsMax・コマンドライン編*　第6.0版

2020年　9月24日

ダイキン工業株式会社 電子システム事業部　営業部　MCグループ
https://www.comtec.daikin.co.jp　qb-support@daikin.co.jp
〒104-0028
東京都中央区八重洲二丁目2番1号　東京ミッドタウン八重洲　八重洲セントラルタワー