Practical example Renderman job

The next step is to combine all of the information in this tutorial to create a dummy Renderman workflow. This tutorial requires an understanding of :

- Basic Python Job Submission I
- Basic Python Job Submission III
- Advanced Dependencies

Feel free to download and run the script below. It sets up a job that will :

- Create a Parent Rib Job that is set to run the command "sleep 20" with a range of 60
- Create a Blocked Child Renderman Job that is set to run the command "sleep 20" and waits for the Parent Rib job to complete before starting

Renderman_demo.py

```
#!/usr/bin/env python3
# Below are required imports for the script to run
import os, sys
# The below few lines of code are to determine the OS of the machine that your running

# this script from and then define the location of the Qube! API
if 'QBDIR' in os.environ:
 sys.path.append('%s/api/python' % os.environ['QBDIR']);
elif os.uname()[0] == 'Darwin':
 sys.path.append('/Applications/pfx/qube/api/python');
elif os.uname()[0] == 'Linux':
 sys.path.append('/usr/local/pfx/qube/api/python');
else:
 sys.path.append('c:/program files/pfx/qube/api/python');

# The below line of code is to import the above defined Qube! API
import qb
# Below is the main function to run in this script
def main():

    # ----------------Start creation of Parent
Job---------------------------------------

    # Below defines an empty list for combining all tasks in the dependency chain
    task = []

    # Below creates an empty dictionary to be filled by the following lines of code
    job = {}

    # Below defines a label for the dependency to be used internally within this
script
    job['label']= 'RibGenLabel'

    # Below defines the name of the Qube! job
    job['name'] = 'Maya Rib Generation Job TUTORIAL'

    # Below defines how many Instances/subjobs the job is to spawn
    job['cpus'] = 1

    # Below defines the internal Qube! jobtype to be used to execute the job
    job['prototype'] = 'cmdrange'
```

```
    # The below parameters are explained further in the "Job submission with job
package explained" page
    package = {}
    job['package'] = package
    job['package']['cmdline'] = 'sleep QB_FRAME_NUMBER'
    job['package']['simpleCmdType'] = 'Maya BatchRender (rib)'
 # Below defines the Agenda/Range of the job this will fill the Frames/Work section of
the Qube! GUI
 # "0-60" is range 0-60
    agendaRange = '0-60'

    # Below defines the internal command required to generate the agenda
    agenda = qb.genframes(agendaRange)

    # Below defines the job details for the agenda
    job['agenda'] = agenda

    # Below appends the details of this task to the job dictionary for later
submission
    task.append(job)

 # ----------------Start creation of Child Job--------------------------------------


 # Below creates an empty dictionary to be filled by the following lines of code
    job = {}

    # Below defines a label for the dependency to be used internally within this
script
    job['label']= 'PRmanLabel'

    # Below defines the dependency of this job see below for possible dependency
strings
    job['dependency'] = 'link-complete-job-RibGenLabel'

    # Below defines the name of the Qube! job
    job['name'] = 'Renderman Render Job TUTORIAL'

    # Below defines how many Instances/subjobs the job is to spawn
    job['cpus'] = 1

    # Below defines how many Instances/subjobs the job is to spawn
    job['prototype'] = 'renderman'

    # The below parameters are explained further in the "Job submission with job
package explained" page
    package = {}
    job['package'] = package
    job['package']['cmdline'] = 'sleep 20'
    job['package']['simpleCmdType'] = 'Renderman Job'

    # Below appends the details of this task to the job dictionary for later
submission
    task.append(job)

 # Below submits the task list to Qube!
    listOfSubmittedJobs = qb.submit(task)

    # Below prints out a list of jobs that have been submitted by name
```

```
    for job in listOfSubmittedJobs:
     print('%(name)15s: %(id)s' % job)

# Below runs the "main" function
if __name__ == "__main__":
    main()
```

```
    sys.exit(0)
```

This script differs very little from the Advanced Dependencies script.

```
job['package']['simpleCmdType'] = 'Maya BatchRender (rib)'
```

This was added to the Parent job to assign the UI type to the job.

```
job['prototype'] = 'renderman'
job['package']['simpleCmdType'] = 'Renderman Job'
```

This was added to the Child job to assign the UI type to the job.

Using the technique from the Basic Python Job Submission III page you can create you own custom code to fill in the required job/package details.

This concludes the Beginner "Python submission and dependencies".

Should you require any clarification of any of the tutorial please feel free to leave comments.