Python job submission with agenda based timeout

In this example, we demonstrate adding a timeout to the job's agenda items and then a callback that will fail the agenda item based on the triggering of the timeout.

This script will submit a job with a 30 second, agenda-based timeout that runs a bunch of "sleep" commands. Each agenda item will do exactly nothing for a number of seconds equal to the frame number. Timeouts have a resolution of about 15 seconds, so you should start seeing agenda items timeout at around frame 30-50, and then timeout consistently beyond that (or until the job is auto-wrangled away, assuming auto-wrangling is enabled).

```
import qb
# a job can have several callbacks, so they are stored in a list
callbacks = []
# each callback is a Qube Callback object.  Callback objects (like
# all Qube objects in the Python API) are dictionaries at the end
# of the day, so we'll define a blank dict.
cb = {}
cb['language'] = 'qube'                  # requires the "qube"
supervisor_language_flag be enabled
cb['triggers'] = 'timeout-work-self-*'   # this is the trigger for which we'll be
watching - a timeout event on any of our work (agenda) items
cb['code']     = 'fail-work-self'        # this is the thing we will do when the
trigger fires - fail the work item that triggered the event
# append all individually created callbacks to the callback list
callbacks.append(cb)
# now create a basic job object
j = {}
j['name'] = 'agenda timeout test - 30 second timeout (+/- 15 seconds)'
j['agendatimeout'] = 30                  # number of seconds before timeout is
triggered - this can be edited after submission in Qube 6.6+
j['prototype'] = 'cmdrange'
j['package'] = {'cmdline':'qbping -sleep QB_FRAME_NUMBER','range':'1-100'}  # 'qbping
-sleep X' is a cross-platform method of sleeping
j['callbacks'] = callbacks               # use our previously created callbacks list
j['agenda'] = qb.genframes(j['package']['range'])  # don't forget to generate frames.
# finally, submit the job.
submitted = qb.submit(j)
for s in submitted:
 print('submitted job %d' % s.get("id"))
```

⚠ If "qbping" is not in your system PATH env var, then you may have to specify the full path to it. qbping lives in $QBDIR/bin.