Queue NetLogo Extension (Version 1.0 beta)

What is it?

Queue is an extension for NetLogo version 5.x that introduces a new data type: a queue, which is a commonly used component in discrete-event simulation. Objects can be inserted into a queue at a particular point in simulated time according to a specified queueing strategy (default is "first in, first out" (FIFO), but "last in, first out" (LIFO) is also supported). Requests to remove an element from a queue always returns the top-most object.

Queues can be questioned as to their current status (length, empty?) and time-weighted statistics are calculated (mean length, max length, mean waiting time, max waiting time). Trying to insert an object with an associated simulation time lying in the past (i.e. a simulation time smaller than the last access time of the queue) throws an exception.

To make writing discrete-event simulations in NetLogo easier, we recommend using the queue extension together with the time extension (see https://github.com/colinsheppard/time/), which provides a discrete-event scheduler amongst other things.

Primitives

queue:create <strategy>

Reports a new queue with the specified queueing strategy (0 for FIFO, 1 for LIFO). Example:

```
set my-queue queue:create 0
```

queue:insert <queue> <object> <current-time>

Inserts the object into the given queue according to its strategy. The queue associates the object with the given time stamp in order to be able to calculate time-weighted statistics. It also sets its time of last access to the given time. Example:

```
queue:insert my-queue (turtle 2) ticks
```

If the given time lies in the past (i.e. is smaller than the current last access time of the queue), this command throws an exception.

queue:remove <queue> <current-time>

Reports the top-most object in the queue (or *nobody* if the queue is empty). The object is removed from the queue and the queue's time of last access is set to the given time. All time-weighted statistics are also updated (this is why you need to pass the current time as a parameter). Example:

```
set next-turtle queue:remove my-queue ticks
```

queue:empty? <queue>

Reports *true* if the given queue is empty; otherwise reports *false*. Example:

```
if (queue:empty? my-queue) [ ;; go to sleep ]
```

queue:length <queue>

Reports the current length of the given queue. Example:

```
show queue:length my-queue
```

queue:show <queue>

Reports a string representation of the given queue, where each element consists of a pair of time of entry and queued object. Example:

```
show queue:show my-queue
```

```
observer> "[[1.12705 turtle 6][2.00000 turtle 5]]"
```

queue:mean-length <queue>

Reports the average length of the given queue over the duration of its existence (time of creation to time of last access). Example:

```
show queue:mean-length my-queue
```

```
observer> 0.985
```

queue:max-length <queue>

Reports the maximum length of the given queue over the duration of its existence (time of creation to time of last access). Example:

```
show queue:max-length my-queue
```

```
observer> 4
```

queue:mean-wt <queue>

Reports the average wait time objects spent in the given queue over the duration of its existence (time of creation to time of last access). Example:

```
show queue:mean-wt my-queue
```

```
observer> 0.3863807458127657
```

queue:max-wt <queue>

Reports the maximum waiting time any of the objects spent in the given queue over the duration of its existence. Example:

```
show queue:max-wt my-queue

observer> 1.825
```

queue:get-stats <queue>

Returns a list of statistical measures of the specified queue in the following order: mean length, max length, mean wait time, max wait time, min wait time. Example:

```
show queue:get-stats my-queue

observer> [0.985 4 0.3863807458127657 1.825 0]
```

queue:reset <queue> <current-time>

Resets all statistical counters of the given queue to their initial values (0, except for min wait time, which is set to Double.MAX_VALUE). Both time of initialisation and time of last access to the queue are set to the given time. Any objects waiting in the queue are left untouched. Example:

```
queue:reset my-queue ticks
```