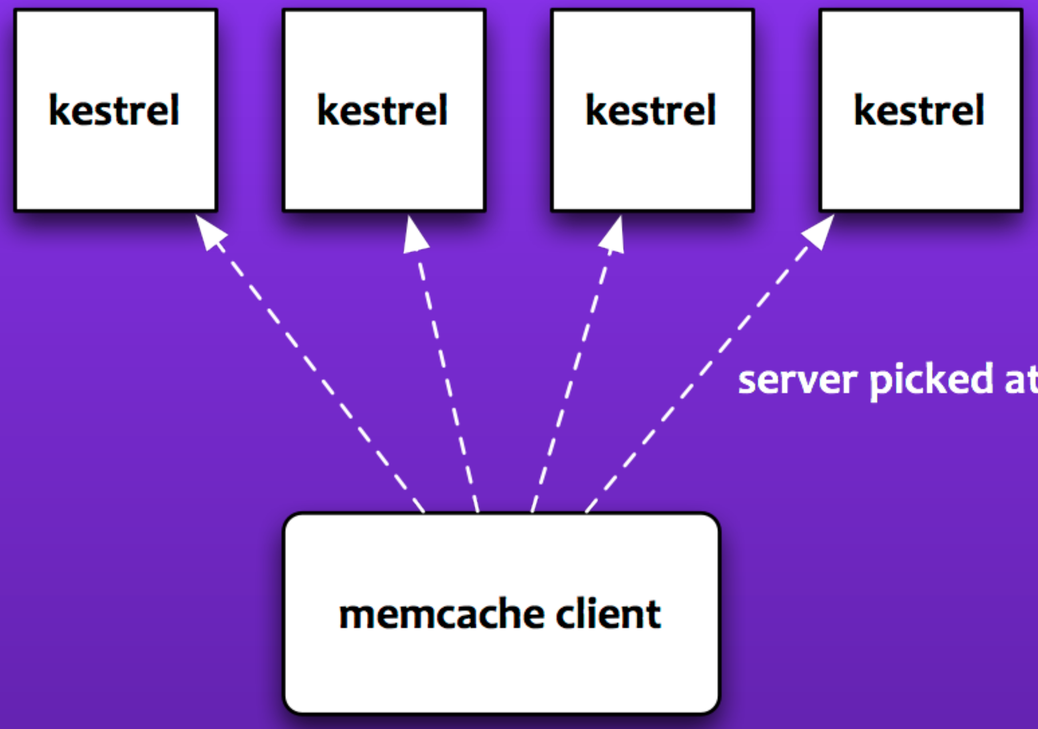
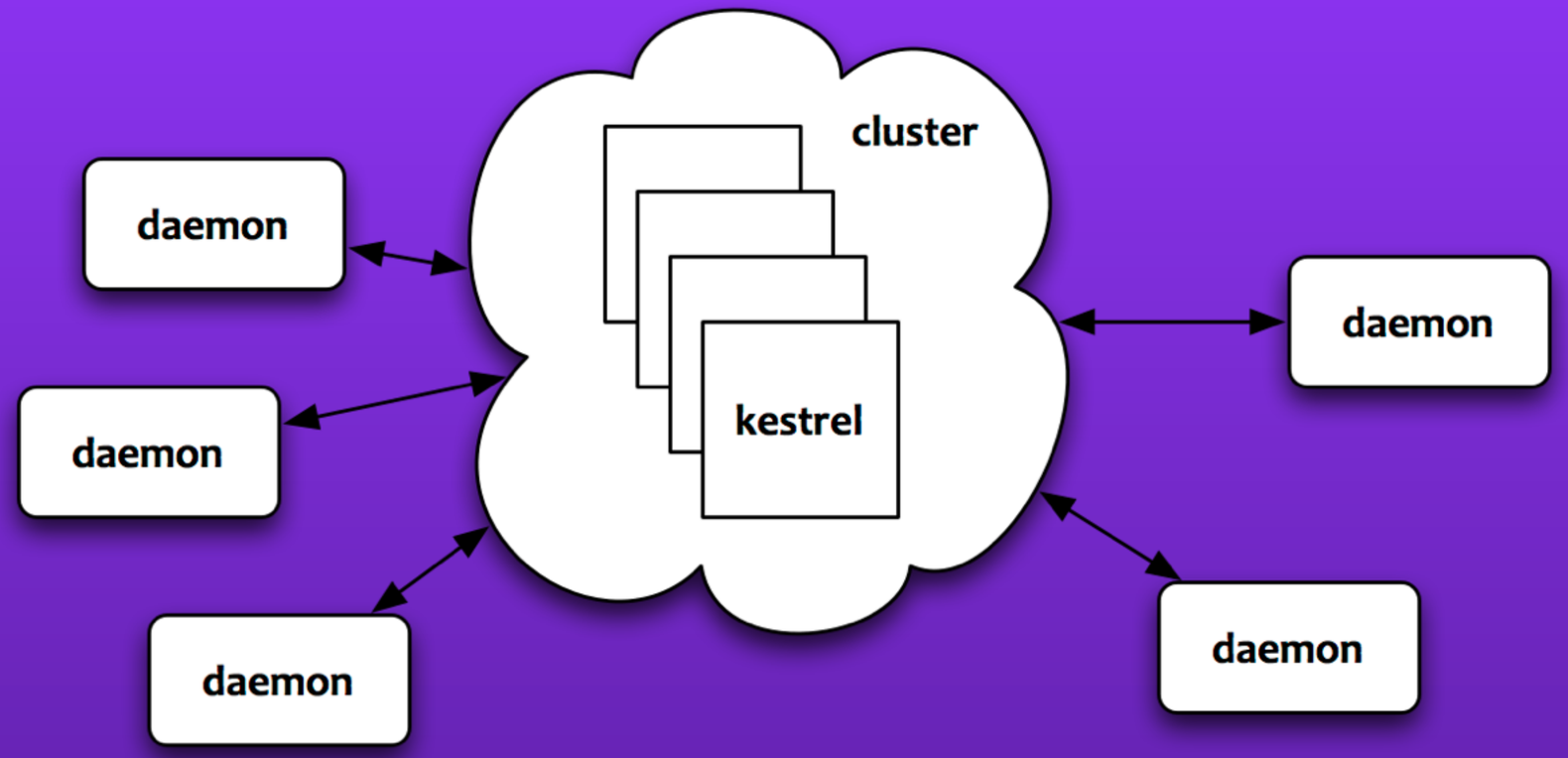
kestrel

kestrel is a distributed queue



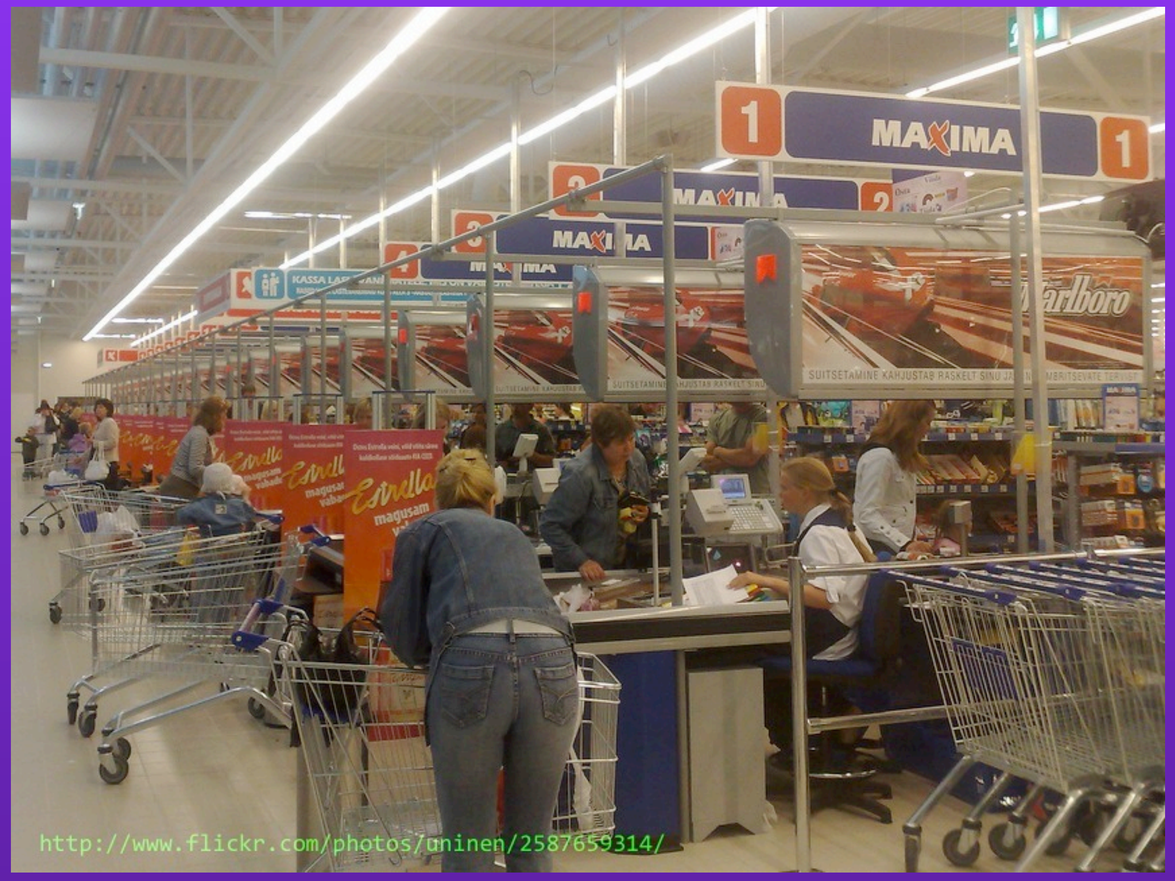
each server stands alone

kestrel is a distributed queue



each server is strongly ordered
... making the cluster loosely ordered

kestrel is a distributed queue



“infinite” horizontal scaling

kestrel is a distributed queue

http://cloc.sourceforge.net v 1.53 T=0.5 s (30.0 files/s, 5684.0 lines/s)

Language	files	blank	comment	code
Scala	15	314	393	2135
SUM:	15	314	393	2135

~~paxos~~ ~~replication~~

simple to reason about

features

```
$ telnet localhost 22133
Trying ::1...
Connected to localhost.
Escape character is '^]'.
set kitty 0 0 6
commie
STORED
get kitty
VALUE kitty 0 6
commie
END
```

memcache protocol

features

```
$ ./scripts/qdump.sh /var/spool/kestrel/kitty
Queue: /var/spool/kestrel/kitty
00000000  ADD 6
0000001b  REM
Journal size: 28 bytes, with 2 operations.
0 items totalling 0 bytes.
```

journal

features

```
put kitty:
commie

+1
get kitty+huey
:commie
get kitty+dewey
:commie
```

fanout queues

features

```
put kitty 1000:
i expire in one second.

+1
...

get kitty
*
```

expiring items (with handoff)

features

```
new QueueBuilder {
  name = "jobs_pending"
  expireToQueue = "jobs_ready"
  maxAge = 30.seconds
}
```

```
put jobs_pending:
{command:"do_work"}

+1

...

get jobs_pending
*
get jobs_ready
:{command:"do_work"}
```

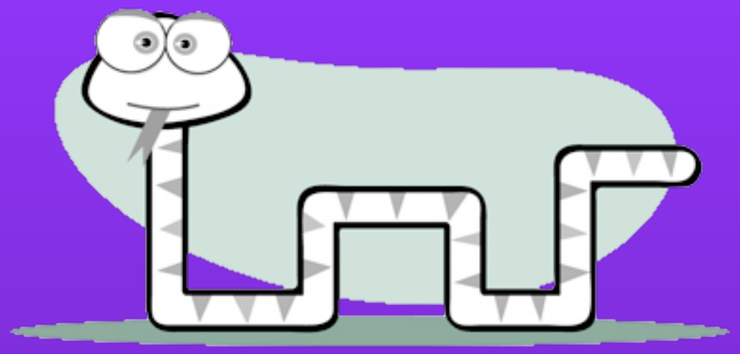
expiring items (with handoff)

features

```
set kitty 0 0 6
commie
STORED
get kitty/open
VALUE kitty 0 6
commie
END
get kitty/abort
END
get kitty/open
VALUE kitty 0 6
commie
END
```

reliable reads

features



in journal only

in memory

tail
(put)

filled in as needed

head
(get)

read-behind

drawbacks

- not strictly ordered
- jvm GC: better than matzruby, but not perfect
- round-trip on each GET and PUT
- read-behind + falling-behind queue =
  infinite journal on disk
- fanout mode is not magickal:
  disk/memory = (# fanout queues) * items

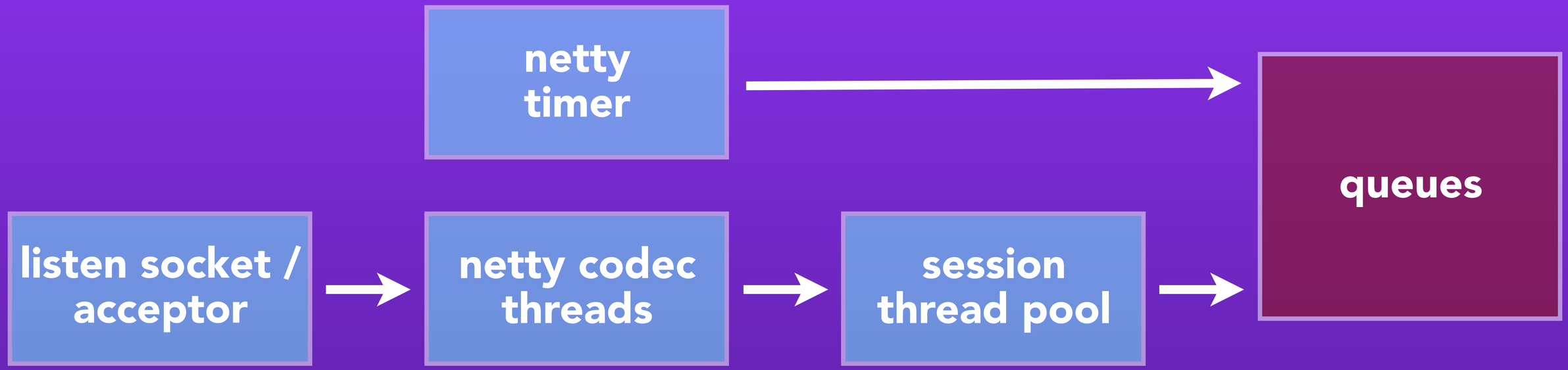
new in 2.0

Test	Mean
MySQL JDBC	23.03ms
MySQL JDBC/Thread-pool	69.89ms
MySQL ADBCJ/MINA	7.74ms
MySQL ADBCJ/Netty	4.84ms
Postgresql JDBC	25.95ms
Postgresql JDBC/Thread-pool	176.98ms
Postgresql ADBCJ/MINA	6.9ms
Postgresql ADBCJ/Netty	5.65ms

... if you believe
random benchmarks
found on the web

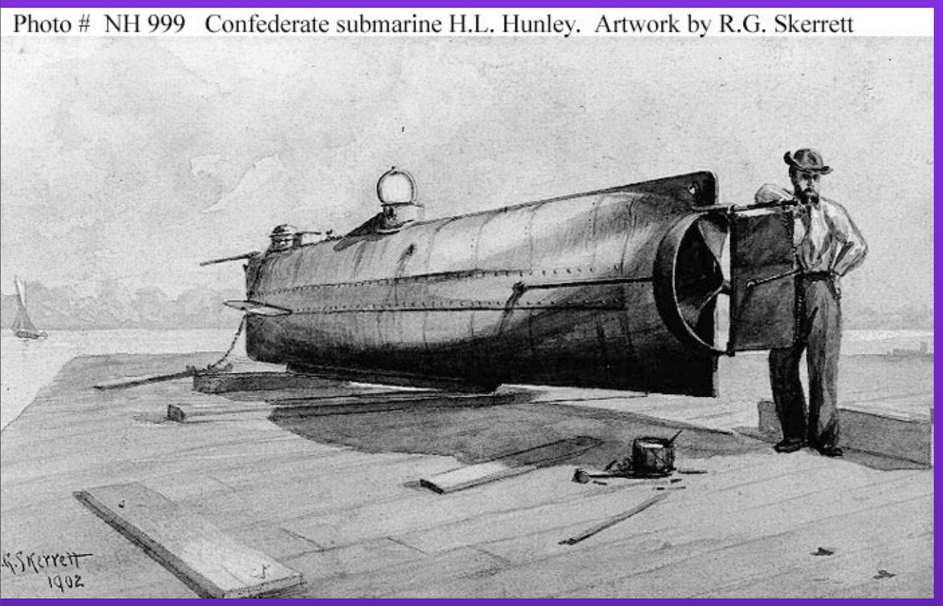
mina -> netty

new in 2.0



removing actors

new in 2.0




```
monitor kitty 60

...

VALUE kitty 0 6
commit
END

...

VALUE kitty 0 8
garfield
END

...

END
confirm kitty 2
END
```

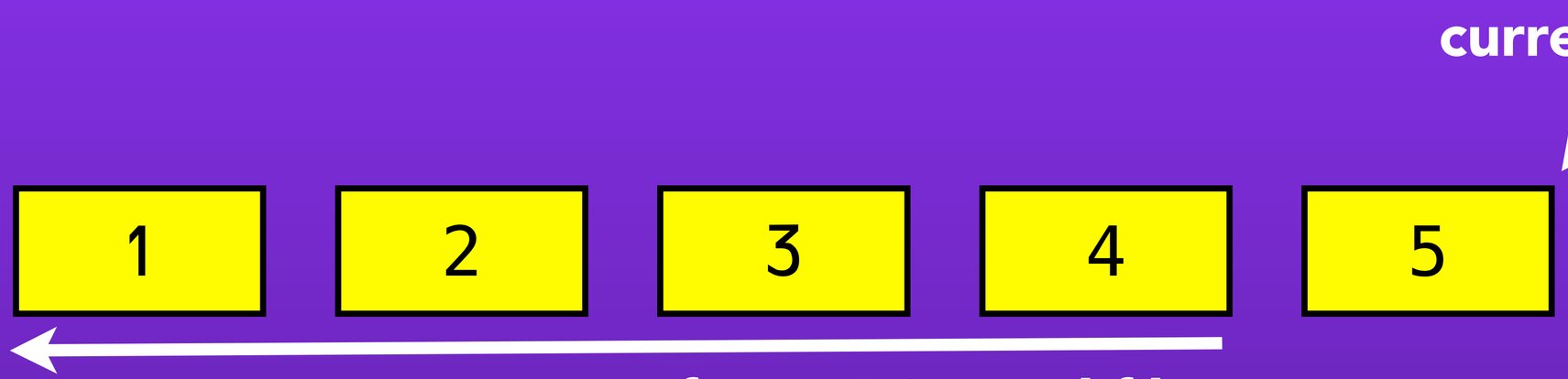
monitor / confirm

new in 2.0

```
kestrel: Packing journals for 'spam': spam.0, spam.1298062157535
kestrel: Packing 'spam': 1024.1 KiB so far (570.6 KiB trailing)
kestrel: Packing 'spam': 1.5 MiB so far (1024.1 KiB trailing)
kestrel: Packing 'spam': 2.0 MiB so far (1.4 MiB trailing)
...
kestrel: Packing 'spam': 16.0 MiB so far (15.8 MiB trailing)
kestrel: Packing 'spam': 16.3 MiB so far (16.0 MiB trailing)
kestrel: Packing 'spam' into new journal.
kestrel: Packing 'spam' -- erasing old files.
kestrel: Packing 'spam' done: spam.0, spam
```

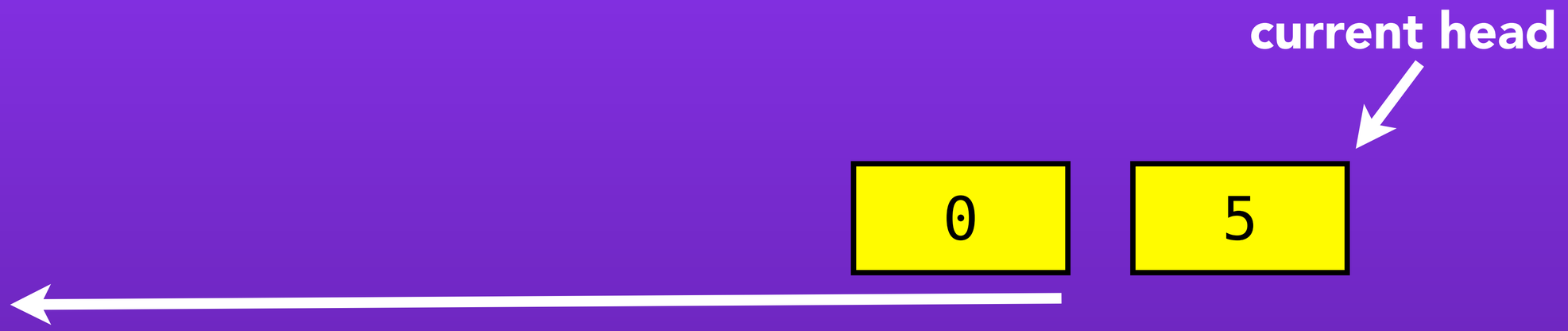
multi-file journals

new in 2.0



multi-file journals

new in 2.0



multi-file journals

new in 2.0

```
$ curl localhost:2223/stats.txt
...
  q/spam/age_msec: 0
  q/spam/bytes: 0
  q/spam/items: 0
q/spam/journal_size: 13544659
q/spam/mem_bytes: 0
q/spam/mem_items: 0
q/spam/open_transactions: 0
q/spam/waiters: 0
```

ostrich 4 stats collecting

work in 2.2/3.0 (done)

use finagle server

work in 2.2/3.0 (in progress)

```
class JournaledBlockingQueue[T] extends BlockingQueue[T]
```

journaled queue -> library

work in 2.2/3.0 (in progress)

```
service Kestrel {
  void put(1: string item)
  ...
}
```

thrift API