Start ZooKeeper Server (Single Node Mode)

```
docker run --name my-zookeeper -d zookeeper
```

Start ZooKeeper Client Process (ZooKeeper Shell)

```
docker run -it --rm --link my-zookeeper:zookeeper zookeeper zkCli.sh
-server zookeeper
```

Master ZooKeeper Client Process

The master watches for new workers and tasks and assigns tasks to available workers.

Worker ZooKeeper Client Process

Workers register with the master to make sure that the master sees that they are available to execute tasks, and then watch for new tasks.

Client ZooKeeper Client Process

Clients create new tasks and wait for responses.

Bootstrap procedure: Create basic znode structure

```
[zk: …] ls /
[zookeeper]
[zk: …] create /workers ""
Created /workers
[zk: …] create /tasks ""
Created /tasks
[zk: …] create /assign ""
Created /assign
[zk: …] ls /
[assign, tasks, workers, zookeeper]
```

Master election procedure: Every new process tries to become the master. If it fails to become the master the process operates as worker and watches for failure of the master in order to re-initiate the master election procedure.

Process 1: Tries to become the master and succeeds.

```
[zk: …] create -e /master "m1.example.com"
Created /master
[zk: …] ls /
[assign, master, tasks, workers, zookeeper]
[zk: …] get /master
m1.example.com
```

Process 2: Tries to become the master and fails.

```
[zk: …] create -e /master "m2.example.com"
Node already exists: /master
[zk: …] stat -w /master
cZxid = 0x29
ctime = Sat Jun 27 14:47:47 UTC 2020
mZxid = 0x29
mtime = Sat Jun 27 14:47:47 UTC 2020
pZxid = 0x29
cversion = 0
```

```
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x100000034f70004
dataLength = 14
numChildren = 0
```

Process 1: Crashes
Ctrl+C

Process 2: Detects crash of master and succeeds to become the new master.

```
WATCHER::
WatchedEvent state:SyncConnected type:NodeDeleted path:/master
[zk: …] ls /
[assign, tasks, workers, zookeeper]
[zk: …] create -e /master "m2.example.com"
Created /master
[zk: …] ls /
[assign, master, tasks, workers, zookeeper]
```

Process 1: Tries (after restart) to become the master, fails and continues as worker.

```
[zk: …] create -e /master "m1.example.com"
Node already exists: /master
[zk: …] stat -w /master
cZxid = 0x2d
ctime = Sat Jun 27 14:52:33 UTC 2020
mZxid = 0x2d
mtime = Sat Jun 27 14:52:33 UTC 2020
pZxid = 0x2d
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x100000034f70005
dataLength = 14
numChildren = 0
```

Master election procedure finished: System is now processing tasks

Master: Watches for new workers and tasks.

```
[zk: …] ls -w /workers
[]
[zk: …] ls -w /tasks
[]
```

Worker: Checks in as worker and watches for task assignments.

```
[zk: …] create -e /workers/worker1 "w1.example.com"
Created /workers/worker1
[zk: …] create /assign/worker1 ""
Created /assign/worker1
[zk: …] ls -w /assign/worker1
[]
```

Master: Detects new worker.
```
WATCHER::
WatchedEvent state:SyncConnected type:NodeChildrenChanged path:/workers
[zk: …] ls -w /workers
[worker1]
```

Client: Creates new tasks and watches for result.
```
[zk: …] create -s /tasks/task- "command-to-execute-task"
Created /tasks/task-0000000000
[zk: …] ls -w /tasks/task-0000000000
[]
```

Master: Gets notified of new task, looks for available worker, and assigns the task to the worker.
```
WATCHER::
WatchedEvent state:SyncConnected type:NodeChildrenChanged path:/tasks
[zk: …] ls -w /tasks
[task-0000000000]
[zk: …] ls -w /workers
[worker1]
[zk: …] create /assign/worker1/task-0000000000 ""
Created /assign/worker1/task-0000000000
```

Worker: Gets notified of new task and processes the task.
```
WATCHER::
WatchedEvent state:SyncConnected type:NodeChildrenChanged
path:/assign/worker1
[zk: …] ls -w /assign/worker1
[task-0000000000]
[zk: …] get /tasks/task-0000000000
command-to-execute-task
[zk: …] create /tasks/task-0000000000/status "computed-result"
Created /tasks/task-0000000000/status
```

Client: Gets notified that the task has been computed.
```
WATCHER::
WatchedEvent state:SyncConnected type:NodeChildrenChanged
path:/tasks/task-0000000000
[zk: …] get /tasks/task-0000000000/status
computed-result
```