Vamos a crear nuestro set de tres réplicas en docker para evitar tener que levantar muchas máquinas a la vez, por ello será necesario tener instalado docker (https://docs.docker.com/engine/installation/) en nuestra máquina. El proceso, si se prefiere, puede realizarse con máquina distintas pero que sean visibles en red.

Creamos un red docker para simplificar el proceso

```
~$ docker network create openWebinarsReplicaSetCluster
~$ docker network ls
```

Levantamos la primera máquina de las tres que usaremos: mongo1

```
~$ docker run -p 3001:27017 --name mongo1 --net openWebinarsReplicaSetCluster -d mongo mongod --replSet my-mongo-set
```

`-p` se emplea para indicarle que mapee el puerto de nuestra máquina local 3001 al puerto 27017 de nuestro contenedor.

`--name` nos permite escoger un nombre para nuestro contenedor (que se emplea como dominio posteriormente).

`--net` se utiliza para determinar la red de docker que se empleará.

Y el comando `--replSet` es el que debe ejecutarse al levantar una instancia de mongoDB para configurarla como Replica SET.

Levantamos la segunda máquina de las tres que usaremos: mongo2

```
~$ docker run -p 3002:27017 --name mongo2 --net openWebinarsReplicaSetCluster -d mongo mongod --replSet my-mongo-set
```

Levantamos la tercera máquina de las tres que usaremos: mongo3

```
~$ docker run -p 3003:27017 --name mongo3 --net openWebinarsReplicaSetCluster -d mongo mongod --replSet my-mongo-set
```

En mongo1 ejecutamos los siguientes comandos para configurar el set de réplicas

```
>db = (new Mongo('localhost:27017')).getDB('test')
>config={"_id":"my-mongo-set",members:[{"_id":0,"host":"mongo1:27017"},{"_id":1,"host":"mongo2:27017"},{"_id":2,"host":"mongo3:27017"}]}
>rs.initiate(config)
>db.coleccion1.insert({name:"Pablo Campos"})
```

En mongo2 y mongo3 comprobamos que todo ha salido correctamente

```
>db.setSlaveOk()
>show collections
>db.coleccion1.find()
```