raft是一种保证分布式一致性的算法。如果你了解zk的工作原理，可以一边想zk一边理解raft，会很好理解。

raft是一种强领导算法。

将存储节点分为3种角色：Leader、Follower、Candidate。即领导者、跟随者、议员。当集群启动时，所有节点都应该是Follower。

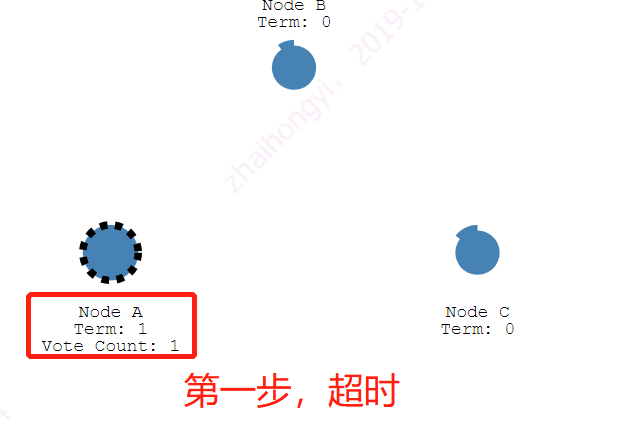
**Leader选举：**

超过选举超时Follower不能获得Leader的心跳，Follower变为Candidate，向所有节点发送投票消息。注意选举超时在不同节点上不同，150ms-300ms随机。不同会减少多个节点并发选举的概率。每次选举都会递增“选举周期（term）”。

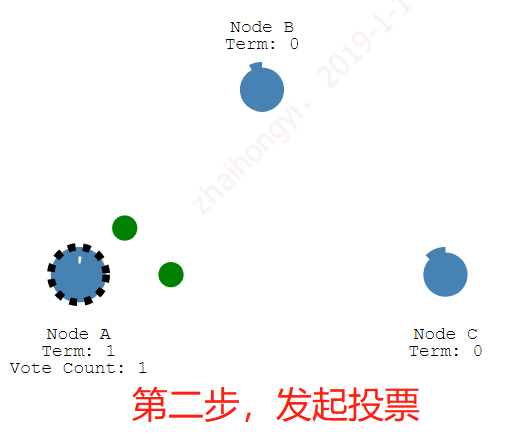
可以认为Candidate的投票消息意思是让其他节点选自己。节点会响应第一个第一个到达的投票请求，即“我认同你是leader”。

当Candidate收到超过半数的选票（包括自己），则成为leader。

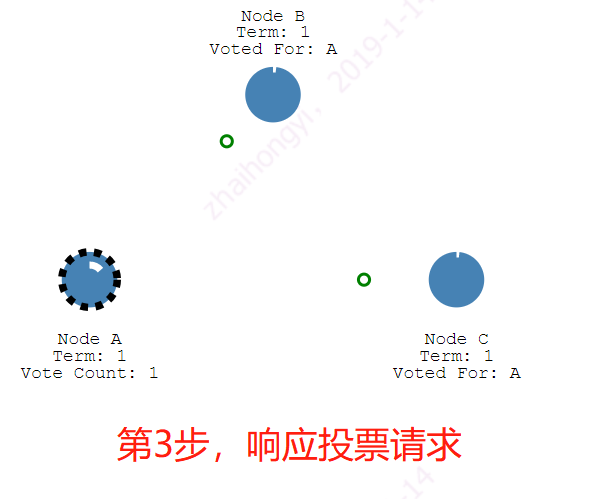
第一步：达到选举超时时间，Follower变为Candidate，递增Term。



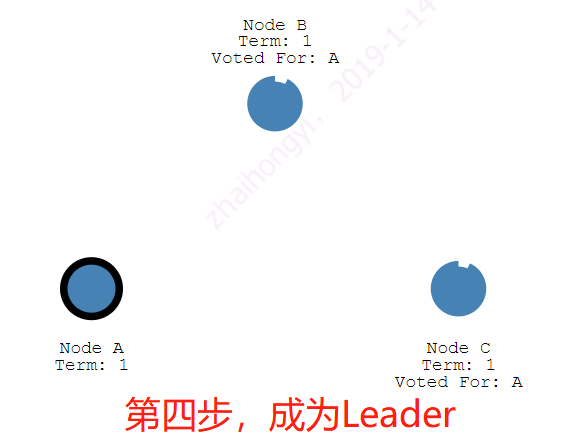
第二步：向其他节点发送投票消息，投票给我。



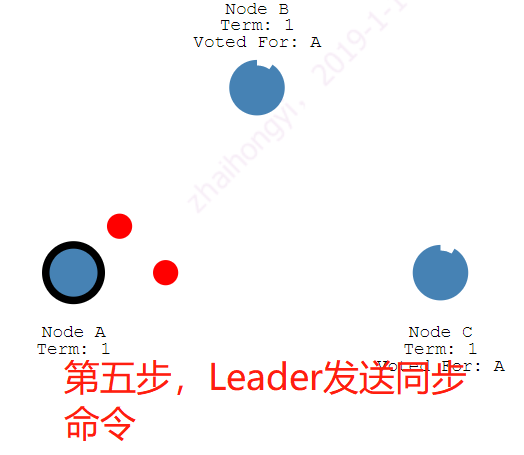
第三步：如果一个节点在这个投票周期没有投票，则会响应第一个请求投票的请求。注意此时Candidate也会维护一个心跳超时。其他节点会重置选举超时时间。



第四步：成为Leader



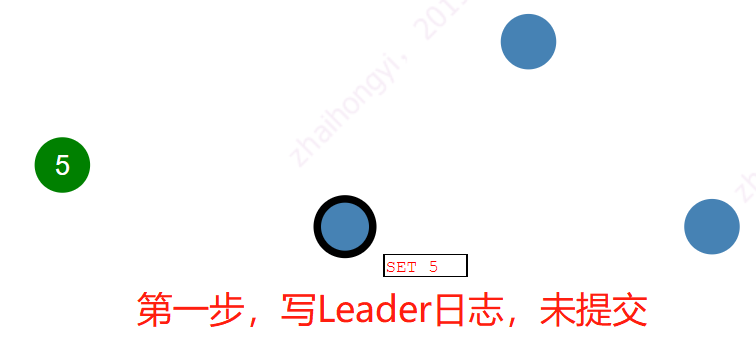
第五步：Leader发送消息同步命令，同步命令发送间隔是心跳超时时间。Leader响应心跳，并重置选举超时时间。



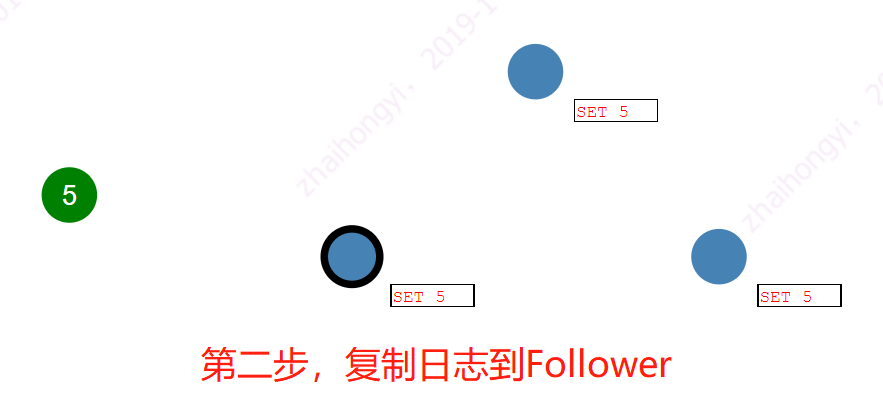
**写操作：**

当选出Leader后，所有写操作都由Leader完成。

第一步：写操作到达Leader，Leader先写日志，数据未提交。



第二步：Leader同步日志到Follower，等待超过半数的节点响应。



第三步：已经有超过半数节点响应，则Leader提交数据



第四步：Leader返回写成功

第五步：Leader通知Follower提交数据，集群数据保证一致性。第二步~第四步称为日志复制（Log Replication），图片步骤写错了。

