**Zookeeper节点类型**

ZooKeeper 节点是有生命周期的，这取决于节点的类型。在 ZooKeeper 中，节点类型可以分为持久节点（PERSISTENT ）、临时节点（EPHEMERAL），以及时序节点（SEQUENTIAL ），具体在节点创建过程中，一般是组合使用，可以生成以下 4 种节点类型。  
 ****持久节点（PERSISTENT）****  
所谓持久节点，是指在节点创建后，就一直存在，直到有删除操作来主动清除这个节点——不会因为创建该节点的客户端会话失效而消失。  
  
****持久顺序节点（PERSISTENT\_SEQUENTIAL）****  
这类节点的基本特性和上面的节点类型是一致的。额外的特性是，在ZK中，每个父节点会为他的第一级子节点维护一份时序，会记录每个子节点创建的先后顺序。基于这个特性，在创建子节点的时候，可以设置这个属性，那么在创建节点过程中，ZK会自动为给定节点名加上一个数字后缀，作为新的节点名。这个数字后缀的范围是整型的最大值。  
  
****临时节点（EPHEMERAL）****  
和持久节点不同的是，临时节点的生命周期和客户端会话绑定。也就是说，如果客户端会话失效，那么这个节点就会自动被清除掉。注意，这里提到的是会话失效，而非连接断开。另外，在临时节点下面不能创建子节点。

****临时顺序节点（EPHEMERAL\_SEQUENTIAL）****  
可以用来实现分布式锁  
客户端调用create()方法创建名为“\_locknode\_/guid-lock-”的节点，需要注意的是，这里节点的创建类型需要设置为EPHEMERAL\_SEQUENTIAL。  
客户端调用getChildren(“\_locknode\_”)方法来获取所有已经创建的子节点，注意，这里不注册任何Watcher。  
客户端获取到所有子节点path之后，如果发现自己在步骤1中创建的节点序号最小，那么就认为这个客户端获得了锁。  
如果在步骤3中发现自己并非所有子节点中最小的，说明自己还没有获取到锁。此时客户端需要找到比自己小的那个节点，然后对其调用exist()方法，同时注册事件监听。  
之后当这个被关注的节点被移除了，客户端会收到相应的通知。这个时候客户端需要再次调用getChildren(“\_locknode\_”)方法来获取所有已经创建的子节点，确保自己确实是最小的节点了，然后进入步骤3。