ZooKeeper是一个**分布式**的，开放源码的分布式**应用程序协调服务**，是Google的Chubby一个开源的实现，它是**集群的管理者**，**监视着集群中各个节点的状态根据节点提交的反馈进行下一步合理操作**。最终，将简单易用的接口和性能高效、功能稳定的系统提供给用户。  
客户端的**读请求**可以被集群中的**任意一台机器处理**，如果读请求在节点上注册了监听器，这个监听器也是由所连接的zookeeper机器来处理。对于**写请求**，这些请求会同**时发给其他zookeeper机器并且达成一致后，请求才会返回成功**。因此，随着**zookeeper的集群机器增多，读请求的吞吐会提高但是写请求的吞吐会下降**。  
有序性是zookeeper中非常重要的一个特性，所有的**更新都是全局有序的**，每个更新都有一个**唯一的时间戳**，这个时间戳称为**zxid（Zookeeper Transaction Id）**。而**读请求只会相对于更新有序**，也就是读请求的返回结果中会带有这个**zookeeper最新的zxid**。

## **ZooKeeper提供了什么？**

1、**文件系统**  
2、**通知机制**

Zookeeper提供一个多层级的节点命名空间（节点称为znode）。与文件系统不同的是，这些节点**都可以设置关联的数据**，而文件系统中只有文件节点可以存放数据而目录节点不行。Zookeeper为了保证高吞吐和低延迟，在内存中维护了这个树状的目录结构，这种特性使得Zookeeper**不能用于存放大量的数据**，每个节点的存放数据上限为**1M**。

## **四种类型的znode**

1、**PERSISTENT-持久化目录节点**   
客户端与zookeeper断开连接后，该节点依旧存在   
2、**PERSISTENT\_SEQUENTIAL-持久化顺序编号目录节点**  
客户端与zookeeper断开连接后，该节点依旧存在，只是Zookeeper给该节点名称进行顺序编号  
3、**EPHEMERAL-临时目录节点**  
客户端与zookeeper断开连接后，该节点被删除   
4、**EPHEMERAL\_SEQUENTIAL-临时顺序编号目录节点**  
客户端与zookeeper断开连接后，该节点被删除，只是Zookeeper给该节点名称进行顺序编号

## **Zookeeper通知机制**

client端会对某个znode建立一个**watcher事件**，当该znode发生变化时，这些client会收到zk的通知，然后client可以根据znode变化来做出业务上的改变等。

## **Zookeeper做了什么？**

1、命名服务  
2、配置管理  
3、集群管理  
4、分布式锁  
5、队列管理

<https://segmentfault.com/a/1190000014479433>

http://jm.taobao.org/2018/06/13/%E5%81%9A%E6%9C%8D%E5%8A%A1%E5%8F%91%E7%8E%B0%EF%BC%9F/