Zookeeper Client API

Zookeeper Class API	
ZooKeeper(String connectString, **int** sessionTimeout, Watcher watcher) **ZooKeeper**(String connectString, **int** sessionTimeout, Watcher watcher, **boolean** canBeReadOnly) **ZooKeeper**(String connectString, **int** sessionTimeout, Watcher watcher, **long** sessionId, **byte**[] sessionPasswd) **ZooKeeper**(String connectString, **int** sessionTimeout, Watcher watcher, **long** sessionId, **byte**[] sessionPasswd, **boolean** canBeReadOnly)	To create a ZooKeeper client object, the application needs to pass at least : • a connection string containing a comma separated list of host:port pairs, each corresponding to a ZooKeeper server • a session timeout • a watcher object which will be notified of state changes
void close()	Close this client object. Once the client is closed, its session becomes invalid. All the ephemeral nodes in the ZooKeeper server associated with the session will be removed. The watches left on those nodes (and on their parents) will be triggered.
String create(String path, **byte** data[], List<ACL> acl, CreateMode createMode) **void** create(String path, **byte** data[], List<ACL> acl, CreateMode createMode, StringCallback cb, Object ctx)	Create a node with the given path. The node data will be the given data, and node acl will be the given acl. This operation, if successful, will trigger all the watches left on the node of the given path by exists and getData API calls, and the watches left on the parent node by getChildren API calls. If a node is created successfully, the ZooKeeper server will trigger the watches on the path left by exists calls, and the watches on the parent of the node by getChildren calls. The maximum allowable size of the data array is 1 MB
void delete(String path, **int** version) **void** delete(String path, **int** version, VoidCallback cb, Object ctx)	Delete the node with the given path. The call will succeed if such a node exists, and the given version matches the node's version (if the given version is -1, it matches any node's versions). This operation, if successful, will trigger all the watches on the node of the given path left by exists API calls, and the watches on the parent node left by getChildren API calls.
List<OpResult> multi(Iterable<Op> ops) **void** multi(Iterable<Op> ops, MultiCallback cb, Object ctx)	Executes multiple ZooKeeper operations or none of them. On success, a list of results is returned. On failure, an exception is raised which contains partial results and error details.
Stat exists(String path, **boolean** watch) Stat exists(String path, Watcher watcher) **void** exists(String path, **boolean** watch, StatCallback cb, Object ctx) **void** exists(**final** String path, Watcher watcher, StatCallback cb, Object ctx)	Return the stat of the node of the given path. Return null if no such a node exists. If the watch is non-null or is true and the call is successful (no exception is thrown), a watch will be left on the node with the given path. The watch will be triggered by a successful operation that creates/delete the node or sets the data on the node.
byte[] getData(String path, **boolean** watch, Stat stat) **byte**[] getData(String path, Watcher watcher, Stat stat) **void** getData(String path, **boolean** watch, DataCallback cb, Object ctx) **void** getData(**final** String path, Watcher watcher, DataCallback cb, Object ctx)	Return the data and the stat of the node of the given path. If the watch is non-null or true and the call is successful (no exception is a watch will be left on the node with the given path. The watch will be triggered by a successful operation that sets data on the node, or deletes the node.
Stat setData(**final** String path, **byte** data[], **int** version) **void** setData(**final** String path, **byte** data[], **int** version, StatCallback cb, Object ctx)	Set the data for the node of the given path if such a node exists and the given version matches the version of the node (if the given version is-1, it matches any node's versions). Return the stat of the node. This operation, if successful, will trigger all the watches on the node of the given path left by getData calls.A KeeperException with error code KeeperException.NoNode will be thrownif no node with the given path exists.A KeeperException with error code

	KeeperException.BadVersion will bethrown if the given version does not match the node's version.

`List<String> getChildren(String path, boolean watch)`
`List<String> getChildren(final String path, Watcher watcher)`
 	
`List<String> getChildren(String path, boolean watch, Stat stat)`
`List<String> getChildren(final String path, Watcher watcher, Stat stat)`
 	
`void getChildren(String path, boolean watch, ChildrenCallback cb, Object ctx)`
`void getChildren(final String path, Watcher watcher, ChildrenCallback cb, Object ctx)`
	Return the list of the children of the node of the given path. If the watch is non-null or is true and the call is successful (no exception is thrown)a watch will be left on the node with the given path. The watch willbe triggered by a successful operation that deletes the node of the given path or creates/delete a child under the node. The list of children returned is not sorted and no guarantee is provided s to its natural or lexical order. A KeeperException with error code KeeperException.NoNode will be thrown if no node with the given path exists.

`void sync(final String path, VoidCallback cb, Object ctx)`	Asynchronous sync. Flushes channel between process and leader.

CreateMode values :

- **PERSISTENT** : The znode will not be automatically deleted upon client's disconnect.
- **PERSISTENT_SEQUENTIAL** : The znode will not be automatically deleted upon client's disconnect, and its name will be appended with a monotonically increasing number.
- **EPHEMERAL :** The znode will be deleted upon the client's disconnect.
- **EPHEMERAL_SEQUENTIAL** : The znode will be deleted upon the client's disconnect, and its name will be appended with a monotonically increasing number.

Watcher Interface

This interface specifies the public interface an event handler class must implement. A ZooKeeper client will get various events from the ZooKeeper server it connects to. An application using such a client handles these events by registering a callback object with the client. The callback object is expected to be an instance of a class that implements Watcher interface. The unique method to implement is :

```
void process(final WatchedEvent event)
```

WatchedEvent class

A WatchedEvent represents a change on the ZooKeeper that a Watcher is able to respond to. The WatchedEvent includes exactly what happened, the current state of the ZooKeeper, and the path of the znode that was involved in the event. The methods offered by this class are :

- `String getPath()` : the path of the concerned znode
- `EventType getType()` : the type of the event. Possible values are :
  - `NodeCreated` : the znode has been created.
  - `NodeDataChanged` : The content of the znode has changed.
  - `Node Deleted` : the znode has been deleted.
  - `NodeChildrenChanged` : the set of children of the znode has changed.
- `KeeperState getState()` : the connection state between the client and zookeeper