**zookeeper深入学习（2）---客户端监控服务器端的上下线**

原创 2016年12月19日 11:46:06

* 标签：
* [zookeeper](http://so.csdn.net/so/search/s.do?q=zookeeper&t=blog)
* 1489

1、基础

运行 zkCli.sh –server 进入命令行工具；

1、使用 ls 命令来查看当前 ZooKeeper 中所包含的内容：

[zk: 202.115.36.251:2181(CONNECTED) 1] ls /

* 1

2、创建一个新的 znode ，使用 create /zk myData 。这个命令创建了一个新的 znode 节点“ zk ”以及与它关联的字符串：

[zk: 202.115.36.251:2181(CONNECTED) 2] create /zk "myData“

* 1

3、我们运行 get 命令来确认 znode 是否包含我们所创建的字符串：

[zk: 202.115.36.251:2181(CONNECTED) 3] get /zk

* 1

#监听这个节点的变化,当另外一个客户端改变/zk时,它会打出下面的

#WATCHER::

#WatchedEvent state:SyncConnected type:NodeDataChanged path:/zk

[zk: localhost:2181(CONNECTED) 4] get /zk watch

* 1
* 2
* 3
* 4

4、下面我们通过 set 命令来对 zk 所关联的字符串进行设置：

[zk: 202.115.36.251:2181(CONNECTED) 4] set /zk "zsl“

* 1

5、下面我们将刚才创建的 znode 删除：

[zk: 202.115.36.251:2181(CONNECTED) 5] delete /zk

* 1

6、删除节点：rmr

[zk: 202.115.36.251:2181(CONNECTED) 5] rmr /zk

* 1

上面是对zookeeper的基础的使用。

2、zookeeper监听实例

业务要求：   
某分布式系统中，主节点可以有多台，可以动态上下线   
任意一台客户端都能实时感知到主节点服务器的上下线

代码如下：client端：

package NodeUpdate;

import org.apache.zookeeper.KeeperException;

import org.apache.zookeeper.WatchedEvent;

import org.apache.zookeeper.Watcher;

import org.apache.zookeeper.ZooKeeper;

import org.apache.zookeeper.data.Stat;

import java.io.IOException;

import java.io.UnsupportedEncodingException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* 某分布式系统中，主节点可以有多台，可以动态上下线；

\* 任意一台客户端都能实时感知到主节点服务器的上下线。

\* Created by tianjun on 2016/12/19 0019.

\*/

public class AppClient {

private String groupNode = "sgroup";

private ZooKeeper zk = null;

private Stat stat = new Stat();

private volatile List<String> serverList = null;

/\*\*

\* 连接zookeeper

\* @throws IOException

\*/

public void connectZookeeper() throws IOException, KeeperException, InterruptedException {

zk = new ZooKeeper("mini04:2181,mini05:2181,mini06:2181", 5000, new Watcher() {

public void process(WatchedEvent event) {

//如果发生了"/sgroup"节点下子节点变化事件，更新server列表，并重新注册监听

if(event.getType() == Event.EventType.NodeChildrenChanged && ("/" + groupNode).equals(event.getPath())){

try {

updateServerList();

} catch (KeeperException e) {

e.printStackTrace();

} catch (InterruptedException e) {

e.printStackTrace();

} catch (UnsupportedEncodingException e) {

e.printStackTrace();

}

}

}

});

updateServerList();

}

/\*\*

\* 更新server列表

\*/

private void updateServerList() throws KeeperException, InterruptedException, UnsupportedEncodingException {

List<String> newServerList = new ArrayList<String>();

//获取并监听groupNode子节点变化

//watch参数为true，表示监听子节点变化事件

//每次都需要重新注册监听，因为一次注册，只能监听一次事件，如果还想继续保持监听，必须重新注册

List<String> subList = zk.getChildren("/" + groupNode,true);

for (String subNode:subList){

byte[] data = zk.getData("/"+groupNode+"/"+subNode,false,stat);

newServerList.add(new String(data,"utf-8"));

}

//替换server列表

serverList = newServerList;

System.out.println("server list updated:"+serverList);

}

/\*\*

\* client的工作逻辑写在这个方法中

\* 此处不做任何处理，只让client sleep

\* @throws InterruptedException

\*/

public void handle() throws InterruptedException {

Thread.sleep(Long.MAX\_VALUE);

}

public static void main(String[] args) throws InterruptedException, IOException, KeeperException {

AppClient ac = new AppClient();

ac.connectZookeeper();

ac.handle();

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86

server端：

package NodeUpdate;

import org.apache.zookeeper.\*;

import java.io.IOException;

/\*\*

\* 某分布式系统中，主节点可以有多台，可以动态上下线；

\* 任意一台客户端都能实时感知到主节点服务器的上下线。

\* Created by tianjun on 2016/12/19 0019.

\*/

public class AppServer {

private String groupNode = "sgroup";

private String subNode = "sub";

/\*\*

\*链接zookeeper

\* @param address

\* @throws IOException

\*/

public void connectZookeeper(String address) throws IOException, KeeperException, InterruptedException {

ZooKeeper zk = new ZooKeeper("mini04:2181,mini05:2181,mini06:2181", 5000, new Watcher() {

public void process(WatchedEvent event) {

//不做处理

}

});

//在"/sgroup"下创建子节点

//子节点类型设置为EPHEMERAL\_SEQUENTIAL，表明这是一个临时节点，且在子节点的名称后面加上一串数字后缀

//将server的地址数据关联到新创建的子节点上

String createPath = zk.create("/"+groupNode+"/"+subNode,address.getBytes("utf-8"), ZooDefs.Ids.OPEN\_ACL\_UNSAFE, CreateMode.EPHEMERAL\_SEQUENTIAL);

System.out.println("create:"+createPath);

}

/\*\*

\* server的工作逻辑写在这个方法中

\* @throws InterruptedException

\*/

public void handle() throws InterruptedException {

Thread.sleep(Long.MAX\_VALUE);

}

public static void main(String[] args) throws InterruptedException, IOException, KeeperException {

if(args.length==0){

System.err.println("the first argument must be server address");

System.exit(1);

}

AppServer as = new AppServer();

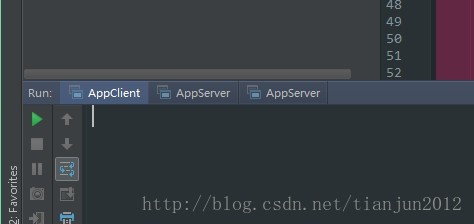
as.connectZookeeper(args[0]);

as.handle();

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56

结果：   
   
如图，我启动了一个client端，启动了两个服务器，之后在关闭一台server，如下：   


版权声明：本文为博主原创文章，未经博主允许不得转载。