ZeroMQ - Fun Part II

MARKDOWN In this sample: * send data directly from **target to service1** * send data from **service2 to target and back** ```java package org.tt.zmq.tests; import org.junit.Test; import org.zeromq.ZContext; import org.zeromq.ZMQ; import org.zeromq.ZMsg; /** * Here I will keep some tests to understand the behaviour of ZMQ-Socket-Types * * @author dertom95 * */ public class ZMQBTo2Dealers2 { private static void doProxy(ZMQ.Socket from, ZMQ.Socket to) { ZMsg msg = ZMsg.recvMsg(from); msg.send(to); } @Test public void testBoundRountTo2Dealers() { // create one context for each logical section to "simulate" different // computers // ------------ setup target ---------------------- ZContext ctxTarget = new ZContext(); ZMQ.Socket target = ctxTarget.createSocket(ZMQ.ROUTER); // give the socket an identity. this name will be used as address if // this socket connects to a router. // this is important if you want to route to a socket that didn't // contact you before target.setIdentity("target".getBytes()); target.bind("tcp://127.0.0.1:9001"); // ------------ setup proxy ------------------------ ZContext ctxProxy = new ZContext(); ZMQ.Socket dealer = ctxProxy.createSocket(ZMQ.DEALER); // give the dealer an identity. here it is actually important dealer.setIdentity("dealer".getBytes()); dealer.connect("tcp://127.0.0.1:9001"); ZMQ.Socket proxy = ctxProxy.createSocket(ZMQ.ROUTER); proxy.setIdentity("proxy".getBytes()); proxy.bind("tcp://127.0.0.1:9000"); // ------------ setup service ---------------------- ZContext ctxServices = new ZContext(); ZMQ.Socket service1 = ctxServices.createSocket(ZMQ.DEALER); // give the dealer an identity. here it is actually important service1.setIdentity("service1".getBytes()); service1.connect("tcp://127.0.0.1:9000"); ZMQ.Socket service2 = ctxServices.createSocket(ZMQ.DEALER); // give the dealer an identity. here it is actually important service2.setIdentity("service2".getBytes()); service2.connect("tcp://127.0.0.1:9000"); // ------------ actions ---------------------------- // send data from server over the proxy to service1 // let's create the message and tell which way it should go. // 1. go to socket that's identity is 'dealer' on the first // router-socket target.sendMore("dealer"); // 2. go to socket that's identity is 'service1' on the 2nd // router-socket target.sendMore("service1"); // 3. Give it the data and send this to target. // target-socket is our start-socket which is a router // this will pop the first line of our msg ('dealer') and checks if // there // is a socket called 'dealer' known to the router. If yes pass the rest // of // the message (without the 'dealer'-address) to this socket target.send("target greets you service1"); // the target-router should have routed automatically to the // 'dealer'-socket // so we should be able receive the message // the job of the dealer is to get the message and to pass it as it is // to the // proxy which is a router socket. ZMsg dealerIncomingMsg = ZMsg.recvMsg(dealer); // check if the routing-address is "service1" String dealerIncomingAddress = dealerIncomingMsg.peekFirst().toString(); assert (dealerIncomingAddress.equals("service1")); // send the message as it is to the proxy-router. this will take the // first-line as routing-address // checks also if this address is known and if yes it will pass the rest // of the message automatically to this socket dealerIncomingMsg.send(proxy); // Since service1-socket is connected to proxy-router and has the // identiy "service1" it should be able to receive // the message now. ZMsg service1Incoming = ZMsg.recvMsg(service1); String service1IncomingData = service1Incoming.popString(); assert (service1IncomingData.equals("target greets you service1")); // now let's send a message from 'serivce2' to 'target'. // send the message to our own socket 'service2' which is connected to // proxy service2.send("hello target!"); // therefore the data we send via service2 will end up in the // incoming-queue of proxy-socket ZMsg proxyIncomingMsg = ZMsg.recvMsg(proxy); // receiving(?) on a router-socket adds the address of the socket that // sent the message // to check the should beside the data there should now be an additional // address-line 'service2' String service2Address = proxyIncomingMsg.peekFirst().toString(); assert (service2Address.equals("service2")); // send the message without doing something further to our dealer-proxy // which is connected to the target // therefore sending through 'dealer'-socket should make the message // available on target which is a router-socket. proxyIncomingMsg.send(dealer); // receive the message ZMsg targetIncomingMsg = ZMsg.recvMsg(target); // since target is also a router there should be a 2nd additional // address-line that points to the dealer, who sent // data from the dealer-socket to this target-socket String

dealerAddress = targetIncomingMsg.peekFirst().toString(); assert (dealerAddress.equals("dealer")); String data = targetIncomingMsg.getLast().toString(); System.out.println("Got message from service2:" + data); // To send data BACK we can reuse the message // 1. get rid of the old data targetIncomingMsg.removeLast(); // 2. add our new data targetIncomingMsg.addLast("Heyho, let's go!"); // since in this message all routing information were created // automatically on the way from service1 to target // the message has everything its need to send data back to the caller targetIncomingMsg.send(target); // pass data from dealer to proxy (as we did when send directly from // target to service1 at the beginning) dealerIncomingMsg = ZMsg.recvMsg(dealer); dealerIncomingMsg.send(proxy); // now we should get the data again on our service2-socket ZMsg service2IncomingMsg = ZMsg.recvMsg(service2); data = service2IncomingMsg.popString(); assert (data.equals("Heyho, let's go!")); System.out.println("Get some data back from target:" + data); ctxProxy.close(); ctxServices.close(); ctxTarget.close(); } } ```

Published by Thomas Trocha
Fri Jan 06 13:57:00 CET 2017
http://thomas.trocha.com:80/pebble/zeromq-fun-part-ii