*// serial connection*  
**int** serialBaud = 9600;  
  
*// socket parameters*  
**int** serverPort = 23;  
  
*// start TCP servers*  
TCPServer server = TCPServer(serverPort);  
TCPClient client;  
  
**char** myIpString[24];  
  
**enum** tnetState {DISCONNECTED, CONNECTED};  
**int** telnetState = DISCONNECTED;  
  
**unsigned** **long** **activity\_t**imeout = 0;  
   
**void** **setup**() {  
 Serial.begin(serialBaud); *// open serial communications*  
 server.begin(); *// begin listening for TCP connections*  
  
 IPAddress myIP = WiFi.localIP();  
 sprintf(myIpString, "%d.%d.%d.%d", myIP[0], myIP[1], myIP[2], myIP[3]);  
 Spark.variable("devIP", myIpString, STRING);  
  
}  
  
**void** **loop**() {  
   
 **if** (client.connected()) {  
 **if** (telnetState == DISCONNECTED)  
 Serial.println("client connected"); *//Check for new connection on next loop()*  
  
 telnetState = CONNECTED;  
 *// echo all available bytes back to the client*  
 **int** incomingByte = 0;  
 **while** (client.available()) { *//Read incoming TCP data if available and copy to Serial port*  
 incomingByte = client.read();  
 Serial.write(**char**(incomingByte));  
 **activity\_t**imeout = millis();  
 }  
 **if** (incomingByte != 0) *//Make sure to flush outgoing serial data before looking at serial input*  
 Serial.flush();  
  
 **else** **if** (millis() - **activity\_t**imeout > 60000UL) { *// No data received so check 1 minute inactivity timeout*  
 Serial.println("client.stop"); *//Check for new connection on next loop()*  
 client.stop();  
 telnetState = DISCONNECTED;  
 }  
  
 **while** (Serial.available() > 0) { *//Read incoming serial data if available and copy to TCP port*  
 incomingByte = Serial.read();  
 client.write((**char**)incomingByte); *// write the char data to the client*  
 }  
 }  
 **else** {  
 *// if no client is yet connected, check for a new connection*  
 **if** (telnetState == CONNECTED) { *//If client WAS connected before, so stop() to close connection*  
 Serial.println("client.stop"); *//Check for new connection on next loop()*  
 client.stop();  
 telnetState = DISCONNECTED;  
 }  
 client = server.available(); *//Update TCP client status - "client" is declared globally*  
 }  
   
}