#include "mex.h"

#include "myo/myo.hpp"

// All callbacks invoke printMyoID to test existence of unique Myos

class DataCollector : public myo::DeviceListener

{

std::vector<myo::Myo\*> knownMyos; // Hold on to Myo\* for all unique myos

// Find myo in knownMyos OR add myo to knownMyos, print a message with

// Myo's id (one more than index into knownMyos) and pointer value.

void printMyoID(myo::Myo\* myo)

{

int ii = 0;

for (ii; ii < knownMyos.size(); ii++)

if (knownMyos[ii] == myo) { break; }

if (ii == knownMyos.size()) // myo not found

knownMyos.push\_back(myo);

mexPrintf("myo id = %d\tMyo\* = %p\n",ii+1,myo);

}

public:

DataCollector() {}

~DataCollector() {}

void onPair(myo::Myo\* myo, uint64\_t timestamp, myo::FirmwareVersion firmwareVersion) { printMyoID(myo); }

void onConnect(myo::Myo \*myo, uint64\_t timestamp, myo::FirmwareVersion firmwareVersion) { printMyoID(myo); }

void onDisconnect(myo::Myo\* myo, uint64\_t timestamp) { printMyoID(myo); }

void onLock(myo::Myo\* myo, uint64\_t timestamp) { printMyoID(myo); }

void onOrientationData(myo::Myo\* myo, uint64\_t timestamp, const myo::Quaternion<float>& q) { printMyoID(myo); }

void onGyroscopeData (myo::Myo\* myo, uint64\_t timestamp, const myo::Vector3<float>& g) { printMyoID(myo); }

void onAccelerometerData (myo::Myo\* myo, uint64\_t timestamp, const myo::Vector3<float>& a) { printMyoID(myo); }

void onEmgData(myo::Myo\* myo, uint64\_t timestamp, const int8\_t \*e) { printMyoID(myo); }

void onPose(myo::Myo\* myo, uint64\_t timestamp, myo::Pose p) { printMyoID(myo); }

void onUnpair(myo::Myo\* myo, uint64\_t timestamp) { printMyoID(myo); }

void onArmSync(myo::Myo\* myo, uint64\_t timestamp, myo::Arm arm, myo::XDirection xDirection) { printMyoID(myo); }

void onArmUnsync(myo::Myo\* myo, uint64\_t timestamp) { printMyoID(myo); }

};

void mexFunction(int nlhs, mxArray\* plhs[],

int nrhs, const mxArray\* prhs[])

{

mexPrintf("Instantiating DataCollector and Hub ...\n");

DataCollector dataCollector;

myo::Hub hub("com.mark-toma.myo\_connect\_test");

mexPrintf("Registering DataCollector with Hub ...\n");

hub.addListener(&dataCollector);

mexPrintf("Getting a Myo\* ...\n");

myo::Myo\* myo = hub.waitForMyo(1000);

mexPrintf("Running Hub ...\n");

hub.run(100);

mexPrintf("Done!\nExiting.\n");

}