PROJECT

KEYLESS DOOR ENTRY

VIA BLUETOOTH TECHNOLOGY

BY MATT WEPPLER

DATE

JUNE 10TH, 2015

CLIENT

MOBILE PAYMENTS, E-COMMERCE SECURITY, AND CRYPTOCURRENCY - 30319

Problem

- We live in an increasing & ever evolving mobile world.
- Yet some things are taking longer than other to catch up.

Implementation

- Bluetooth technology (NFC to follow)
- Mobile app Android (iOS to follow)
- Arduino (prototyping kit), microcontroller, bluetooth radio, servo
- Existing single cylinder deadbolt lock.

HARDWARE

Hardware

- Single Cylinder Deadbolt
- Mobile Phone w/Integrated Bluetooth
- Arduino
- Bluetooth Module
- Servo


ABC 04:01 PM
Bluetooth is Enabled
Disable Bluetooth
Scan for Devices
Paired Devices
ABC 04:01 PM
Front Door
00:11:22:33:44
Patio Door
Scanning
Scanning for Bluetooth Devices
Cancel
ABC 04:01 PM
Front Door
00:11:22:33:44
Patio Door
Pairing Request
Confirm passkey to pair with Front Door
Cancel
OK
ABC 04:01 PM
Front Door
00:11:22:33:44
Patio Door
00:11:22:33:45
Garage Door
00:11:22:33:46
ABC 04:01 PM
Front Door
00:11:22:33:44
Lock/Unlock Door

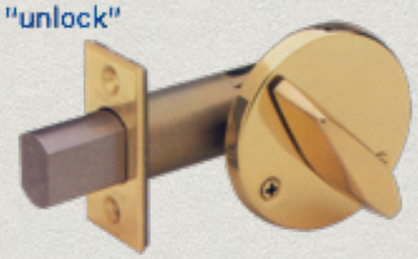
ANDROID APP

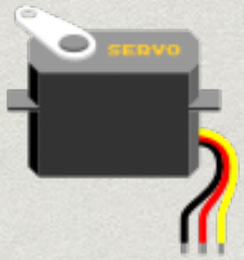
Android App

- Scan for device(s)
- Pair with device(s)
- Lock/Unlock deadbolt

Diagram

5 The servo is attached to the deadbolt lock handle. Rotation of the servo cause the hardware to "lock" or "unlock"



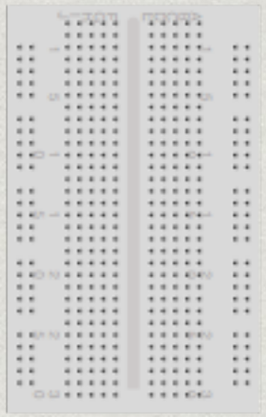


3 The microcontroller tells the servo to rotate N degrees, and waits N seconds for the servo to complete

1 A "command message" is sent from the android's bluetooth radio to the external bluetooth device's radio

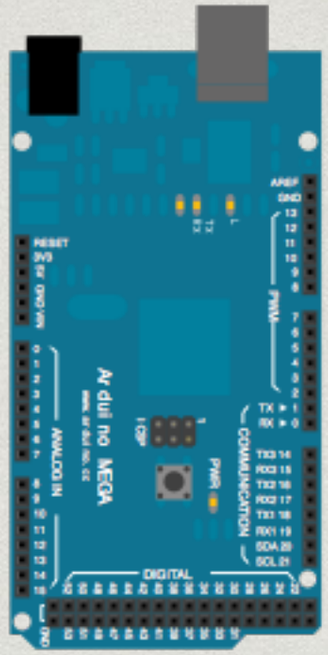
The microcontroller passes a "status message" to the android's bluetooth radio (via the external bluetooth devices radio)

4



The external bluetooth device's radio passes the "command message" along to the microcontroller

2



When in range bluetooth devices pair to each other

Write the Android app

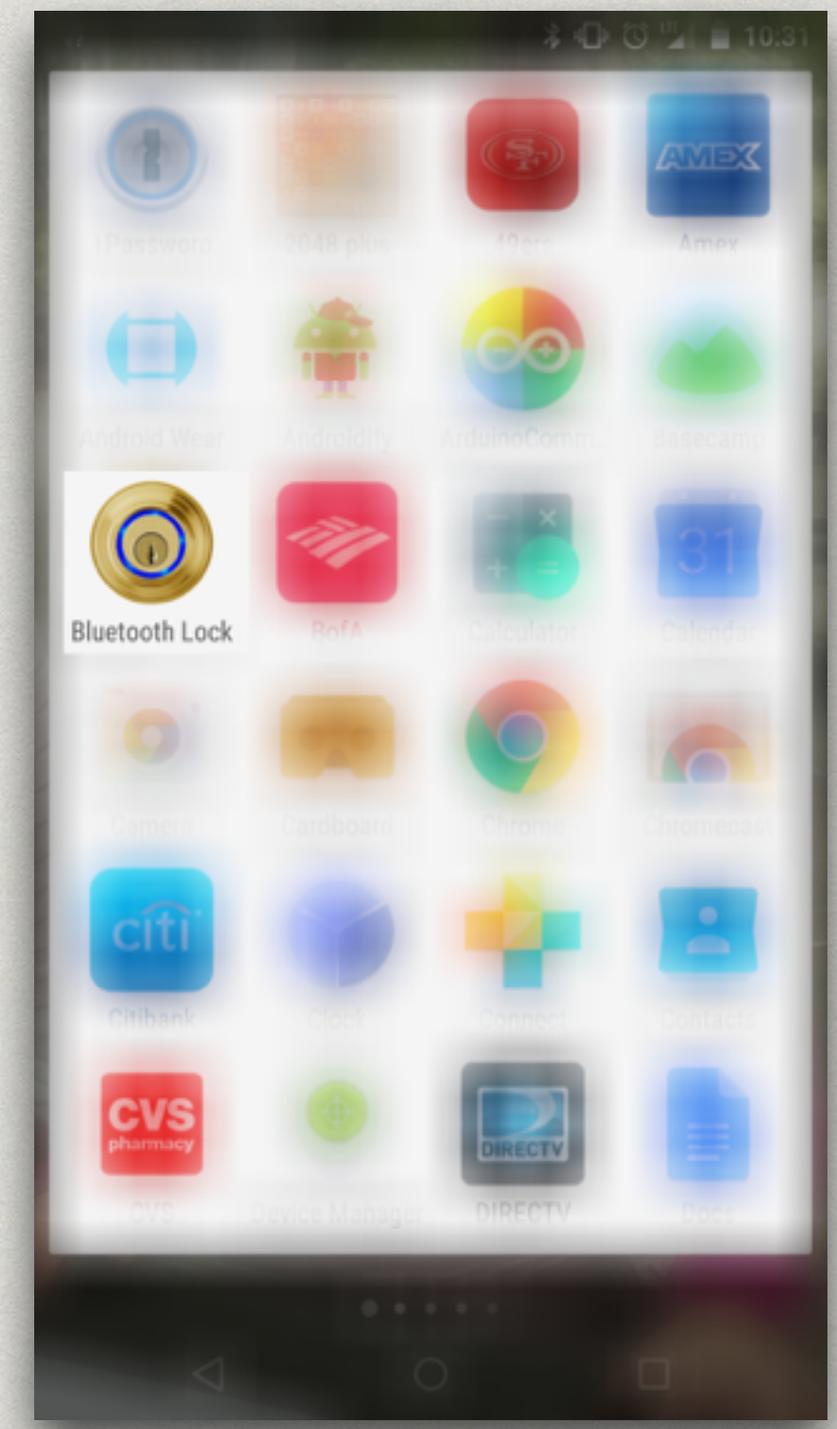
- **Open a socket to the bluetooth radio**

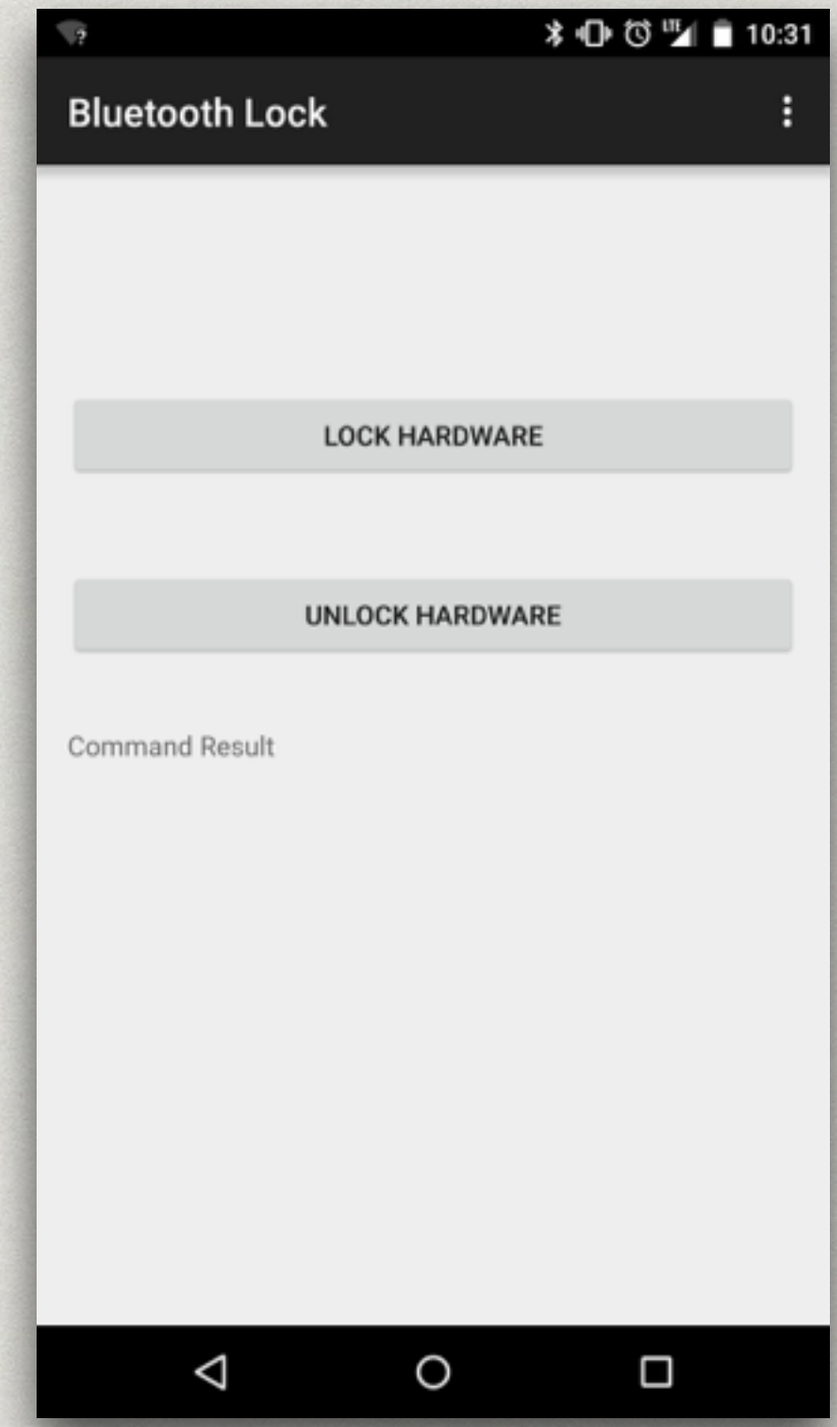
```
public void openBTSocket() throws IOException {
    mBTSocket = mBTDevice.createRfcommSocketToServiceRecord(uuid);
    mBTSocket.connect();
    mOutputStream = mBTSocket.getOutputStream();
    mInputStream = mBTSocket.getInputStream();
    listenForIncomingBTData();
    Toast.makeText(getApplicationContext(), “Ready to send commands", Toast.LENGTH_SHORT).show();
}

public void listenForIncomingBTData() {
  ...
  commandResult.setText(data);
  ...
}

public void sendLockCommandToBTHW(View view) throws IOException {
    mOutputStream.write(msg.getBytes());
    Toast.makeText(getApplicationContext(), "Lock Command Sent", Toast.LENGTH_SHORT).show();
}

public void sendUnlockCommandToBTHW(View view) throws IOException {
    mOutputStream.write(msg.getBytes());
    Toast.makeText(getApplicationContext(), "Unlock Command Sent", Toast.LENGTH_SHORT).show();
}
```


10:31
Bluetooth Lock


10:31
Bluetooth Lock
LOCK HARDWARE
UNLOCK HARDWARE
Command Result

Write the Arduino sketch

- **Setup variables to store some key values.**

```
int UNLOCK_POSITION = 0;
int LOCK_POSITION   = 90;

int btRxPin  = 10; // bt RX-I pin <--> arduino d10 pin (green jumper)
int btTxPin  = 11; // bt TX-O pin <--> arduino d11 pin (yellow jumper)
int servoPin = 9;  // servo pin <--> arduino d9 pin (yellow jumper)
int servoPos = UNLOCK_POSITION;
```

- **Listen for input from a paired bluetooth device.**

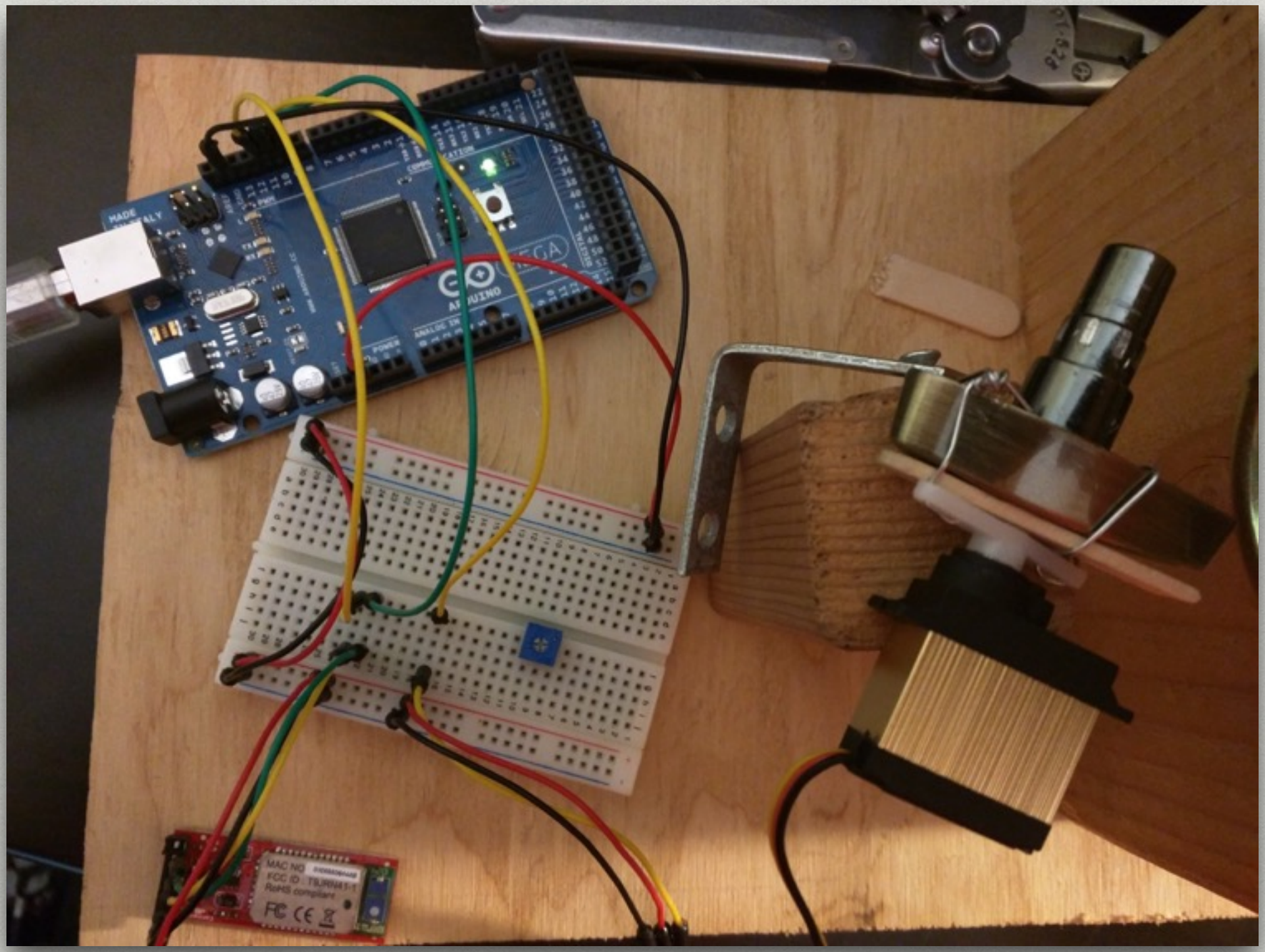
```
void listenForCommands() {
  if (btSerial.available()) {
    handleBluetoothCommand();
  } else if (Serial.available() > 0) {
    handleSerialCommand();
  } else {
    //Serial.println("UNKNOWN HANDLER");
  }
}
```

❋ Handle a “LOCK” command

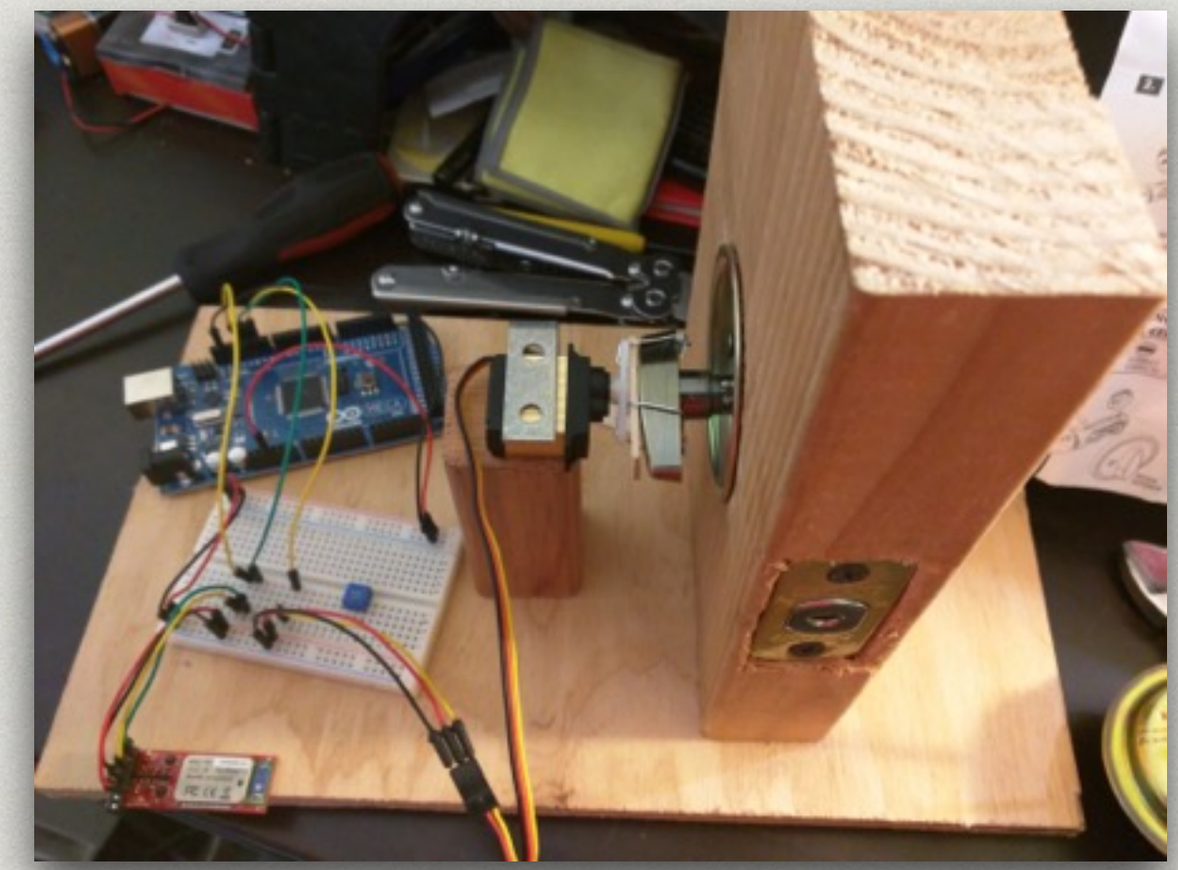
```
void lockHardware() {
  if (servoPos != LOCK_POSITION) {
    for (servoPos = 0; servoPos < LOCK_POSITION; servoPos++) {
      servo.write(servoPos);
      delay(5);
    }
    servoPos = LOCK_POSITION;
  }
}
```

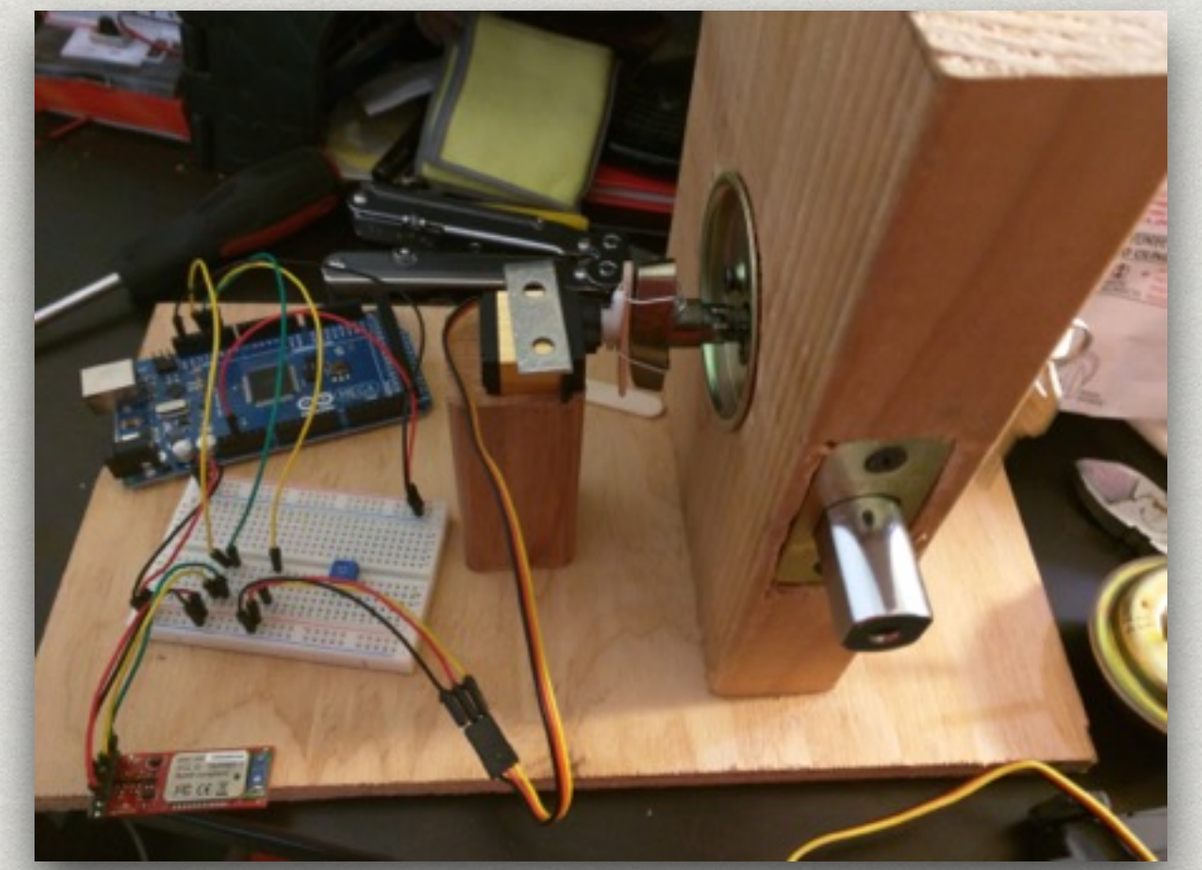
❋ Handle an “UNLOCK” command

```
void unlockHardware() {
  if (servoPos != UNLOCK_POSITION) {
    for (servoPos = LOCK_POSITION; servoPos > UNLOCK_POSITION; servoPos--) {
      servo.write(servoPos);
      delay(5);
    }
    servoPos = UNLOCK_POSITION;
  }
}
```

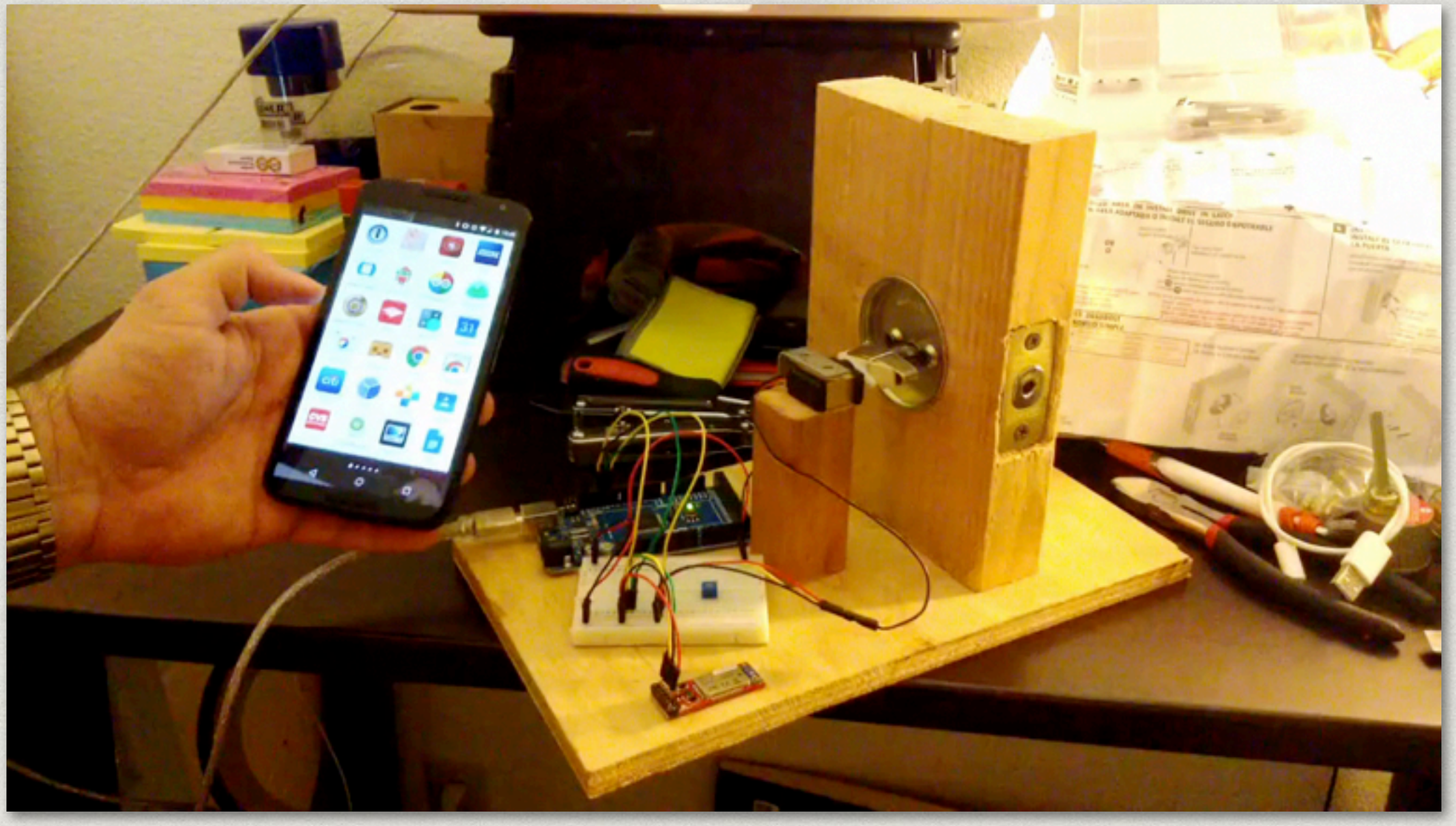

MADE
ARDUINO
MEGA
POWER
MAC NO
FCC ID : TBJRN41-1
RoHS compliant

Support frame to hold the Lock & Electronics





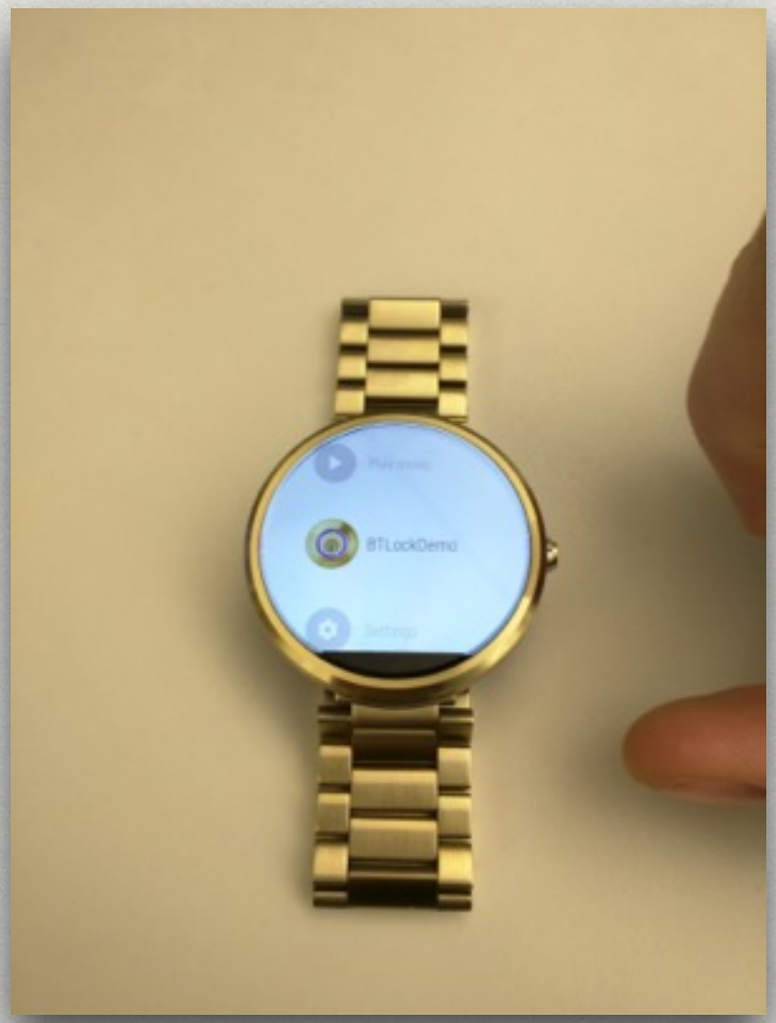
…and now the moment we've all been waiting for

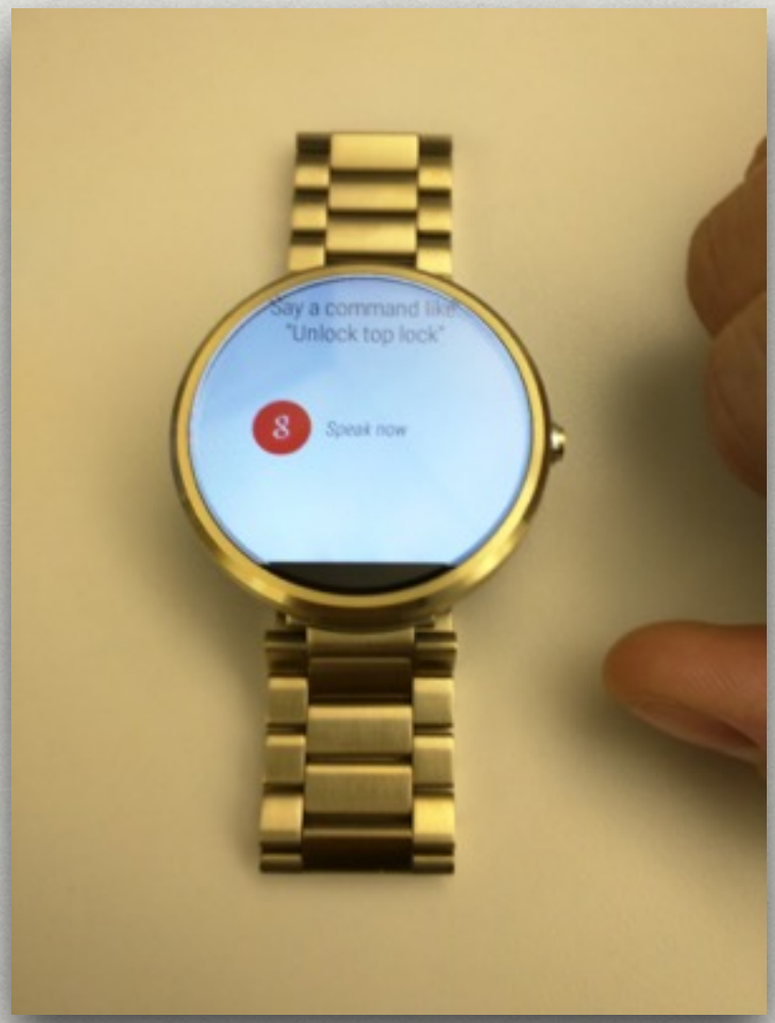


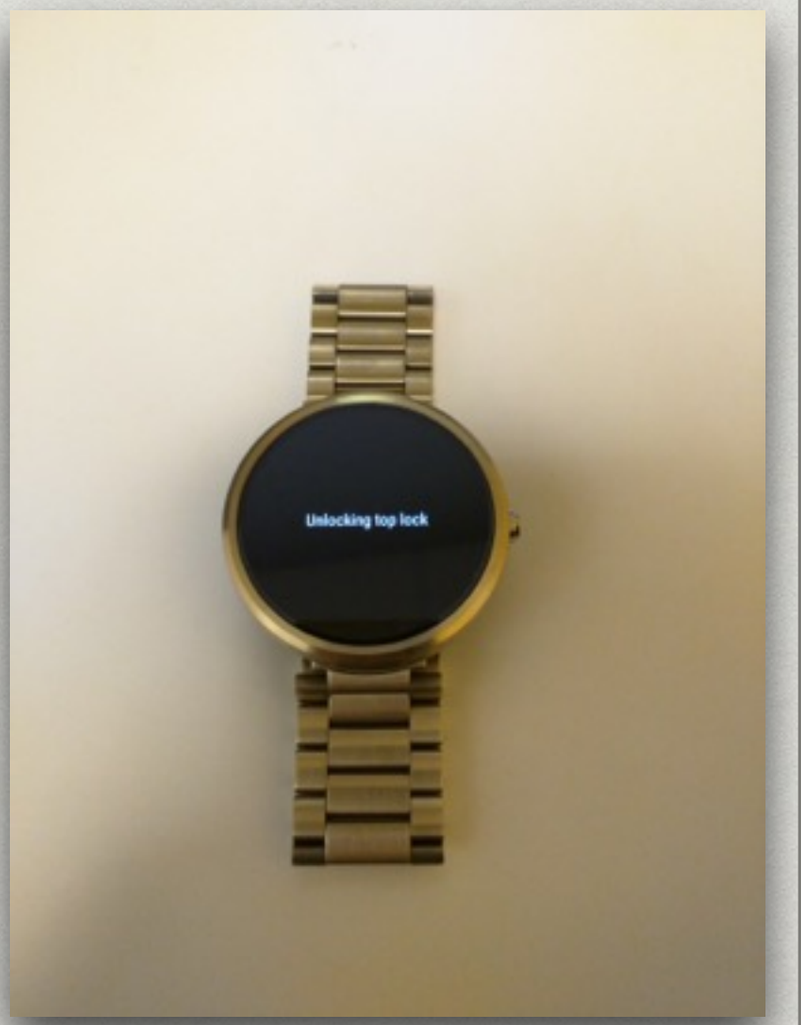
Challenges Faced

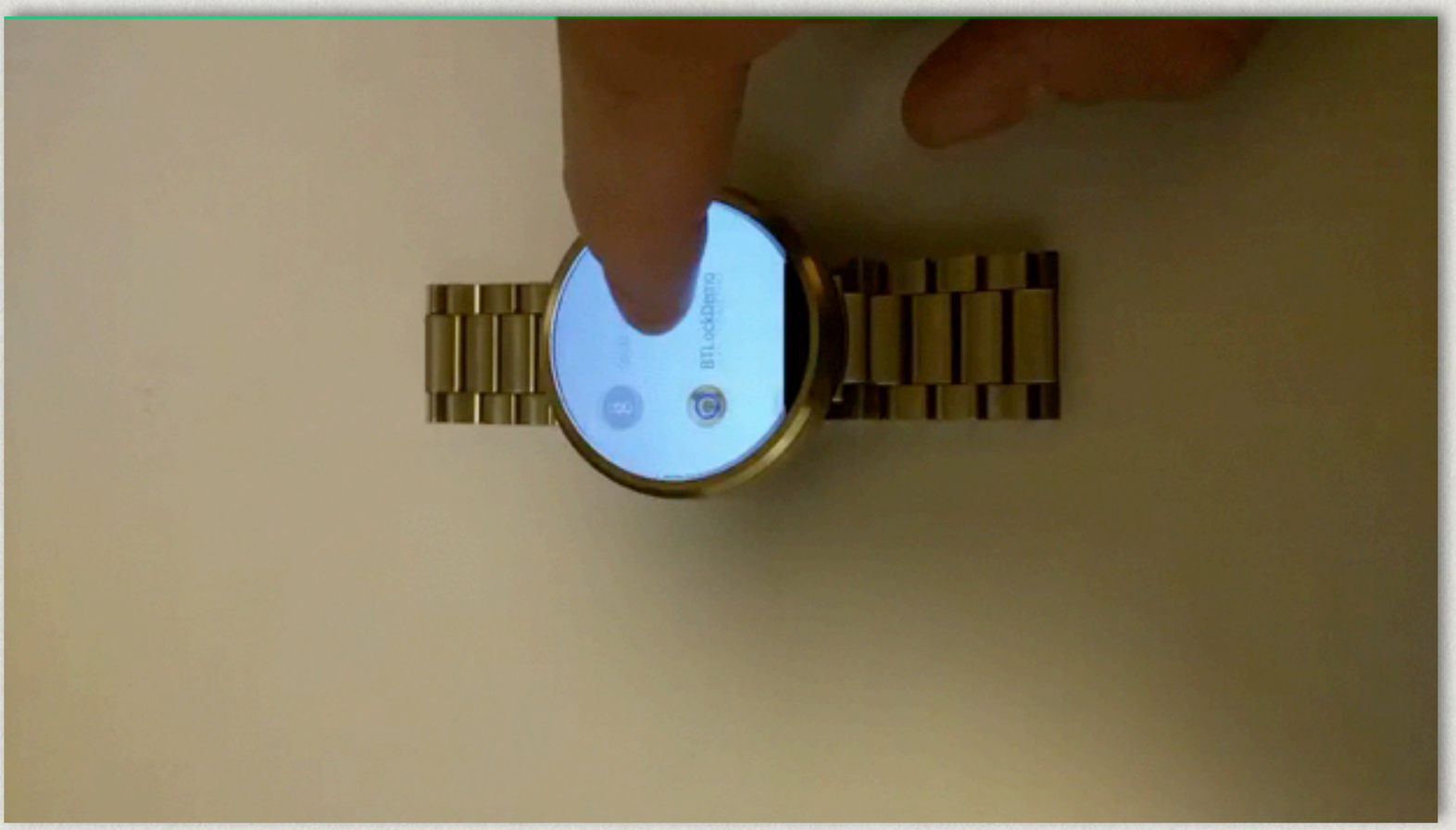
- No experience developing with bluetooth technologies.
- No Arduino experience, but the community is very helpful.
- Servos… I went through 3 of them (stripped the gears)
- Prototype Costs $$$

Android Wear (coming soon).








BTLockDemo