



사 이 버 보 안 캡 스톤 디 자 인

## 스마트폰과 스마트워치를 활용한 사용자 인증

1395030 엄시우  
1791224 심민주

<https://youtu.be/ksMr3n67U4g>

# 01

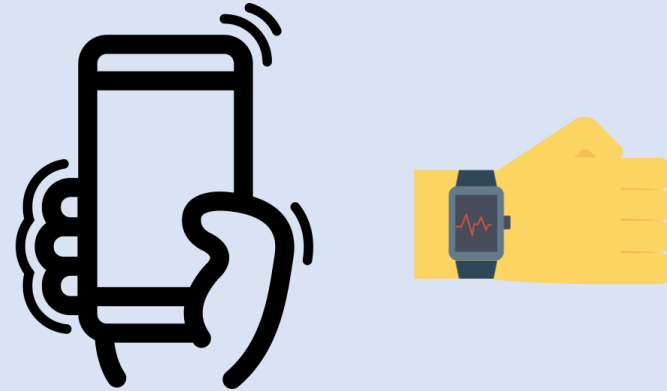
## 프로젝트 소개

### 1차 인증



지문 인식

### 2차 인증



스마트폰 & 스마트워치  
를 활용한 사용자 인증

단계 1. 기기를 흔들어 사용자를 인증

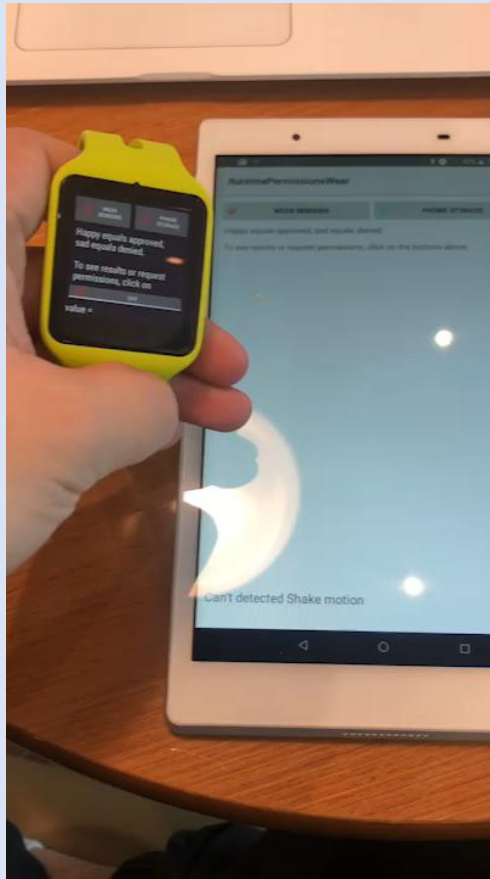
단계 2. 정해진 모션을 통해 사용자 인증

단계 3. 서로 다른 사용자가 같은 모션을 취해도 사용자를 구분하여 인증

1. 위치와 스마트폰의 움직임이 감지되면 인증되는 부분 구현
2. 스마트 워치에서 스마트폰으로 신호 전달
3. 스마트 워치와 스마트 폰의 센서값 수집(DB저장)

# 04

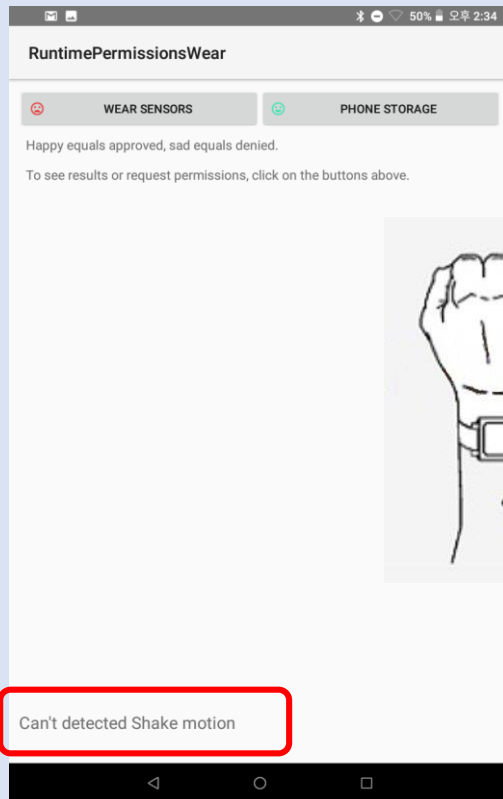
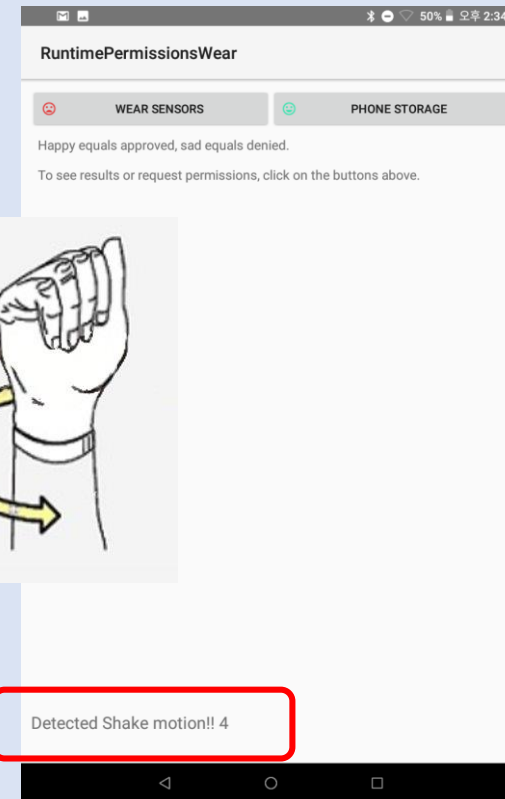
## 작동 영상



## 05

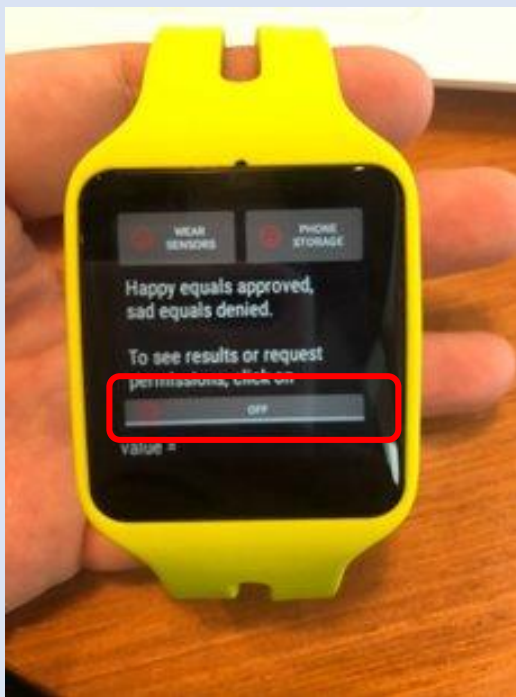
## 작동 이미지

## 스마트 워치 흔들때 마다 스마트폰으로 값 전달

워치 **shake** 전 스마트폰워치 **shake** 후 스마트폰

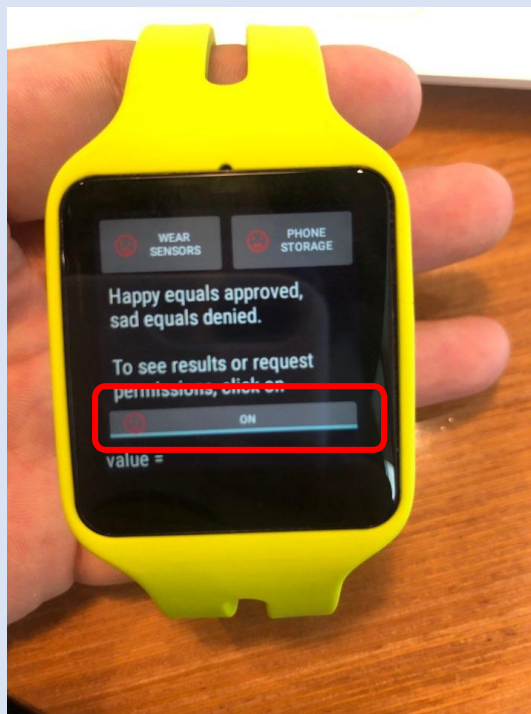
## 05

## 작동 이미지



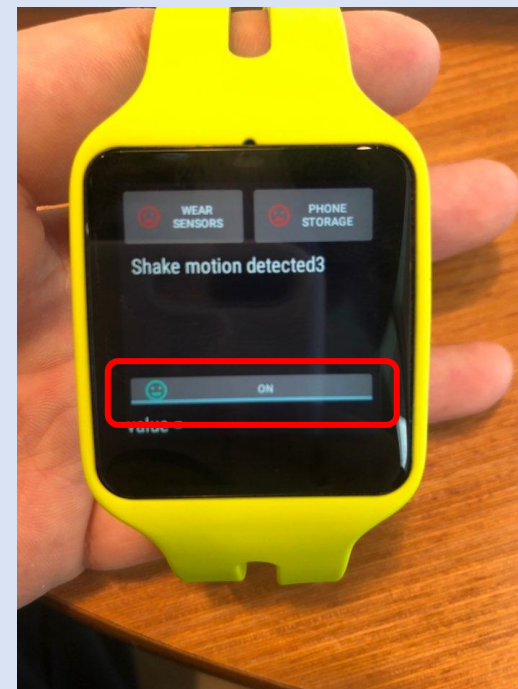
off 상태

워치 shake 신호 전달 불가



on 상태

워치 shake 신호 전달 가능 상태



on 상태

워치 shake 신호 전달 진행 중

## 가속도 센서 값 DB에 저장

DB Browser for SQLite - C:\Users\user\Downloads\testdb.db

파일(F) 편집(E) 뷰(V) Tools 도움말(H)

새 데이터베이스(N) 데이터베이스 열기(O) 변경사항 저장하기(W) 변경사항 취소하기(R) Open Project Save Project Attach Database 데이터베이스 닫기(C)

데이터베이스 구조 데이터 보기 Pragma 수정 SQL 실행

데이터(T): AccelsensorTable

	accXValues	accYValues	accZValues	angleXZ	angleYZ
필터	필터	필터	필터	필터	필터
1	-7.061050415...	4.5340728759...	4.5832061767...	-57.01309224...	44.691234248...
2	-7.061050415...	4.5340728759...	4.5832061767...	-57.01309224...	44.691234248...
3	-6.604614257...	3.8422088623...	5.442626953125	-50.509282558...	35.22008398008
4	-6.604614257...	3.8422088623...	5.442626953125	-50.509282558...	35.22008398008
5	-4.885025024...	2.57087707519...	8.1517486572...	-30.932612722...	17.504095298...
6	-4.885025024...	2.57087707519...	8.1517486572...	-30.932612722...	17.504095298...
7	-2.698471069...	2.9145202636...	9.4598236083...	-15.92111475...	17.1237810570...
8	-2.698471069...	2.9145202636...	9.4598236083...	-15.92111475...	17.1237810570...
9	2.5154571533...	6.63108825683...	6.1847991943...	22.1323650575...	46.994411337...
10	2.5154571533...	6.63108825683...	6.1847991943...	22.1323650575...	46.994411337...
11	4.7545928955...	8.8145446777...	2.0554504394...	66.6207662753...	76.873847888...
12	4.7545928955...	8.8145446777...	2.0554504394...	66.6207662753...	76.873847888...
13	-0.474319458...	8.6703033447...	1.2664794921...	-20.531833232...	81.689524267...
14	-0.474319458...	8.6703033447...	1.2664794921...	-20.531833232...	81.689524267...
15	-4.206115722...	7.2281951904...	3.4217834472...	-50.870757913...	64.667411332...
16	-4.206115722...	7.2281951904...	3.4217834472...	-50.870757913...	64.667411332...
17	-5.349212646...	4.7356109619...	5.5090026855...	-44.15689330...	40.682786104...
18	-5.349212646...	4.7356109619...	5.5090026855...	-44.15689330...	40.682786104...
19	-3.654067993...	3.6554260253...	8.76690673828...	-22.62649238...	22.634053685...
20	-3.654067993...	3.6554260253...	8.76690673828...	-22.62649238...	22.634053685...
21	-0.119018554...	4.8154296875...	9.1031646728...	-0.749066131...	27.878130071221
22	-0.119018554...	4.8154296875...	9.1031646728...	-0.749066131...	27.878130071221
23	0.5081481933...	6.1934204101...	6.9130249023...	4.2040177467...	41.857351914...
24	0.5081481933...	6.1934204101...	6.9130249023...	4.2040177467...	41.857351914...
25	-3.548324584...	4.0612182617...	7.73118591308...	-24.65334924...	27.7130966651...
26	-3.548324584...	4.0612182617...	7.73118591308...	-24.65334924...	27.7130966651...

1 - 27 of 60

특정 레코드 행으로 가기: 1

데이터베이스 셀 수정하기(C)

모드: 문자열

가져오기(O) 내보내기(E) NULL로 만들기

현재 데이터 타입: 문자열 / 숫자  
1 문자

적용

원격

아이디

이름 커밋 마지막 수정 크기

SOL 로그 쿼리 DB 스키마 원격

UTF-8



# 06

## 코드 ( 위치 )

```
float gravityX = axisX / SensorManager.GRAVITY_EARTH;
float gravityY = axisY / SensorManager.GRAVITY_EARTH;
float gravityZ = axisZ / SensorManager.GRAVITY_EARTH;
Float f = gravityX * gravityY * gravityZ * gravityX * gravityY * gravityZ;
double squareadD = Math.sqrt(f.doubleValue());
gForce = (float) squareadD;

if(gForce>SHAKE_THRESHOLD_GRAVITY){
    long currentTime = System.currentTimeMillis();
    Log.e( tag: "shake", Float.toString(gForce));
    if(mShakeTime + SHAKE_SKIP_TIME > currentTime){
        return;
    }

    mShakeTime = currentTime;
    if(oncheckedtbutton){
        Shakecount += 1;
        logToUi("Shake motion detected" + Shakecount);
        DataMap dataMap = new DataMap();
        dataMap.putInt(Constants.KEY_COMM_TYPE, Constants.COMM_TYPE_SHAKE_RESULT_DATA );
        mshake_result_button.setCompoundDrawablesWithIntrinsicBounds(
            R.drawable.ic_permission_approved, top: 0, right: 0, bottom: 0);
        sendMessage(dataMap);
    }
}
```

## 06

## 코드 ( 위치 )

```
float gravityX = axisX / SensorManager.GRAVITY_EARTH;  
float gravityY = axisY / SensorManager.GRAVITY_EARTH;  
float gravityZ = axisZ / SensorManager.GRAVITY_EARTH;  
Float f = gravityX * gravityY * gravityZ * gravityX * gravityY * gravityZ;  
double squareadD = Math.sqrt(f.doubleValue());  
gForce = (float) squareadD;
```

- X, Y, Z 값 이용
- 위치를 흔들었을 때, 중력가속도 계산
- 계산한 값 > 중력, 위치를 흔든 것으로 가정하여 중력가속도를 계산

\*gForce 가 중력가속도 값을 의미

## 06

## 코드 ( 위치 )

```
float gravityX = axisX / SensorManager.GRAVITY_EARTH;
float gravityY = axisY / SensorManager.GRAVITY_EARTH;
float gravityZ = axisZ / SensorManager.GRAVITY_EARTH;
Float f = gravityX * gravityY * gravityZ * gravityX * gravityY * gravityZ;
double squareadD = Math.sqrt(f.doubleValue());
gForce = (float) squareadD;

if(gForce>SHAKE_THRESHOLD_GRAVITY){
    long currentTime = System.currentTimeMillis();
    Log.e( tag: "shake", Float.toString(gForce));
    if(mShakeTime + SHAKE_SKIP_TIME > currentTime){
        return;
    }

    mShakeTime = currentTime;
    if(oncheckedtbutton){
        Shakecount += 1;
        logToUi("Shake motion detected" + Shakecount);
        DataMap dataMap = new DataMap();
        dataMap.putInt(Constants.KEY_COMM_TYPE, Constants.COMM_TYPE_SHAKE_RESULT_DATA );
        mshake_result_button.setCompoundDrawablesWithIntrinsicBounds(
            R.drawable.ic_permission_approved, top: 0, right: 0, bottom: 0);
        sendMessage(dataMap);
    }
}
```

## 06

## 코드 ( 위치 )

```
if(gForce>SHAKE_THRESHOLD_GRAVITY){  
    long currentTime = System.currentTimeMillis();  
    Log.e( tag: "shake", Float.toString(gForce));  
    if(mShakeTime + SHAKE_SKIP_TIME > currentTime){  
        return;  
    }  
  
    mShakeTime = currentTime;  
}
```

- 위치를 흔들때마다 계속 카운트 X
- 한번 흔들 후, 0.5초의 시간간격을 두고 흔들어야 카운트가 증가

## 06

## 코드 ( 위치 )

```
float gravityX = axisX / SensorManager.GRAVITY_EARTH;
float gravityY = axisY / SensorManager.GRAVITY_EARTH;
float gravityZ = axisZ / SensorManager.GRAVITY_EARTH;
Float f = gravityX * gravityY * gravityZ * gravityX * gravityY * gravityZ;
double squareadD = Math.sqrt(f.doubleValue());
gForce = (float) squareadD;

if(gForce>SHAKE_THRESHOLD_GRAVITY){
    long currentTime = System.currentTimeMillis();
    Log.e( tag: "shake", Float.toString(gForce));
    if(mShakeTime + SHAKE_SKIP_TIME > currentTime){
        return;
    }

    mShakeTime = currentTime;
    if(oncheckedtbutton){
        Shakecount += 1;
        logToUi("Shake motion detected" + Shakecount);
        DataMap dataMap = new DataMap();
        dataMap.putInt(Constants.KEY_COMM_TYPE, Constants.COMM_TYPE_SHAKE_RESULT_DATA );
        mshake_result_button.setCompoundDrawablesWithIntrinsicBounds(
            R.drawable.ic_permission_approved, top: 0, right: 0, bottom: 0);
        sendMessage(dataMap);
    }
}
```

## 06

## 코드 ( 위치 )

```
if(oncheckedtgbbutton){
    Shakecount += 1;
    logToUi("Shake motion detected" + Shakecount);
    DataMap dataMap = new DataMap();
    dataMap.putInt(Constants.KEY_COMM_TYPE, Constants.COMM_TYPE_SHAKE_RESULT_DATA );
    mshake_result_button.setCompoundDrawablesWithIntrinsicBounds(
        R.drawable.ic_permission_approved, top: 0, right: 0, bottom: 0);
    sendMessage(dataMap);
}
```

- oncheckedtgbbutton -> 토글버튼 on/off( on=true / off = false) 상태 확인 후, on 경우, Shakecount를 1 증가
- SendMessage를 통해 스마트폰으로 데이터를 전송
- 전송 데이터는 datamap을 활용하여 <key, value>처럼 활용합니다.
- Count 값 전달 X, 흔들렸다는 신호 전송

## 06

## 코드(스마트폰)

```
public void onMessageReceived(MessageEvent messageEvent) {
    Log.d(TAG, msg: "onMessageReceived(): " + messageEvent);

    String messagePath = messageEvent.getPath();

    if (messagePath.equals(Constants.MESSAGE_PATH_PHONE)) {
        DataMap dataMap = DataMap.fromByteArray(messageEvent.getData());

        int commType = dataMap.getInt(Constants.KEY_COMM_TYPE, 0);

        if (commType == Constants.COMM_TYPE_SHAKE_RESULT_DATA){
            shakecount += 1;
            mShake_result_textView.setText("Detected Shake motion!! " + shakecount);
        } else {
            Log.d(TAG, msg: "Unrecognized communication type received.");
        }
    }
}
```

워치로 부터 데이터를 전달받고 카운트 증가시키고 출력

### # 개발 목표

1단계 (흔들기) -> 2단계 (모션) -> 3단계 (사용자 인식)

### # 개발 진척도

- 모든 단계에 공통적으로 필요한 위치와 폰의 통신 부분 해결에 많은 시간 소비 -> 개발 진척도 제자리 걸음

- 현재, 통신부분 문제점 해결

- 1단계(흔들기) 부분의 마무리 작업을 진행 중

**\*\* 통신부분이 해결된 만큼 앞으로 진행속도가 빨리질 것이라 예상**



## 08

## 향후 개발 계획

		10주차	11주차	12주차	13주차	14주차	15주차
1단계	구현	○					
	테스트	○	○				
2단계	조사	○	○				
	구현		○	○			
	테스트			○	○		
3단계	조사		○	○			
	구현			○	○		
	테스트				○	○	
UI					○	○	
최종 발표 준비						○	○

**THANK  
YOU**