SAMSUNG Research

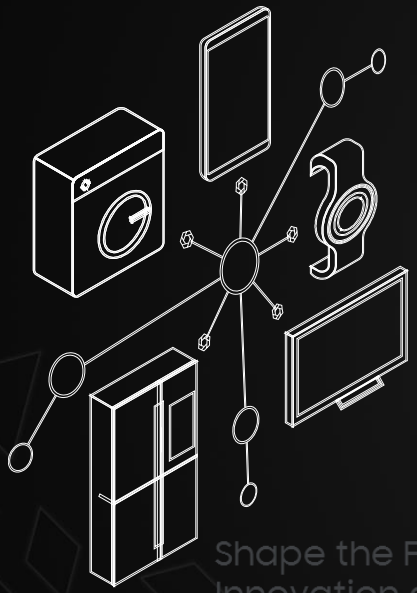
Seoul IoT x Tizen

Controlling LED Light with Motion Sensor

August 22, 2018

Boyeon Son

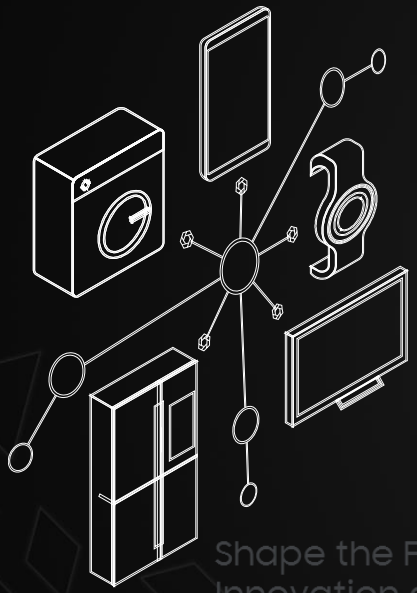
SAMSUNG Research



Shape the Future with
Innovation and Intelligence

Contents

SAMSUNG Research

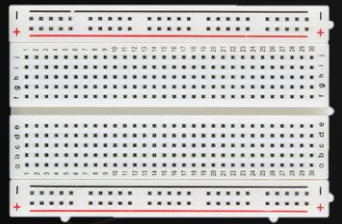


Shape the Future with
Innovation and Intelligence

I Introduction

Introduction: Today's Goal

Eagleye 530s



빵판과 전선 ☺

모션 센서

(HC-SR501)

5파이 LED 전구

330Ω 저항

+

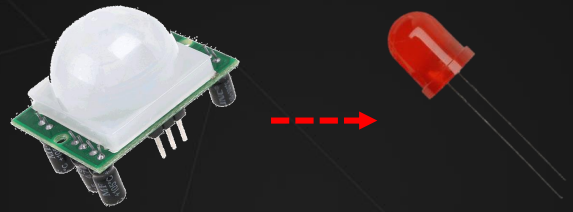
Tizen Studio

Tizen Studio

Introduction: Today's Goal

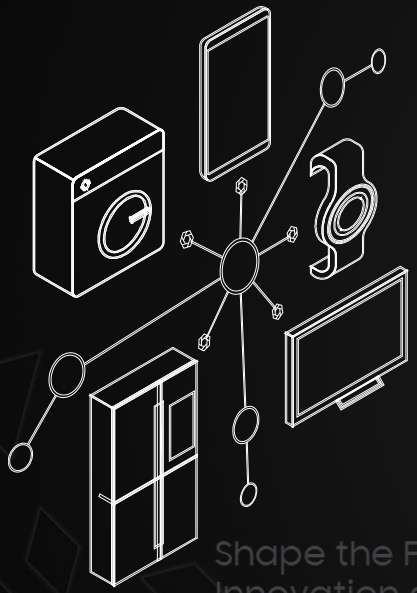
Introduction: Today's Goal

1. 모션 센서 값에 따라 LED 전구 On/Off 시키기



2. SmartThings App에서 디바이스(센서) 컨트롤하기

SAMSUNG Research



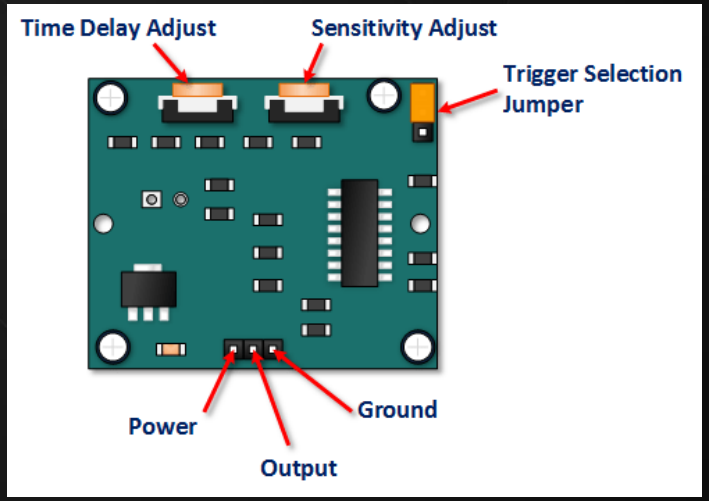
Shape the Future with
Innovation and Intelligence

II Motion Light Control App

Hardware Setup

Hardware Setup: The Motion Sensor

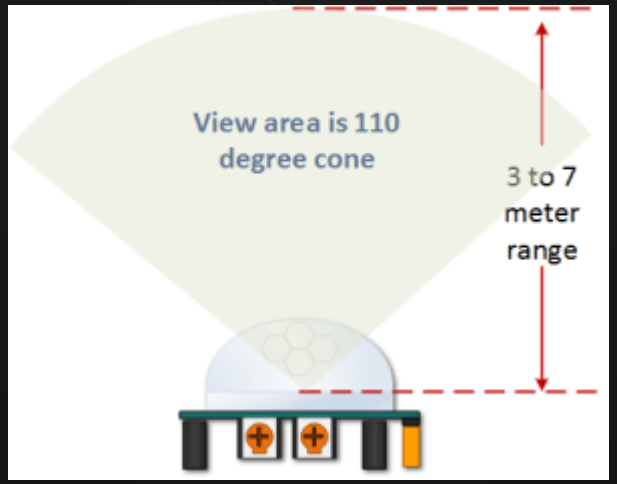
- **Passive Infrared Motion Sensor** (HC-SR501)
  - 사용자 메뉴얼 https://www.mpja.com/download/31227sc.pdf



Hardware Setup: The Motion Sensor

모션 감지 범위

- 3~7m 거리 안의 110도 부채꼴 모양 내 적외선 변화 감지



Hardware Setup: The Motion Sensor

감지 범위 조정

- 시계 방향으로 돌리면 감지 범위 감소 (min 3m)
- 반시계 방향으로 돌리면 감지 범위 증가 (max 7m)

Hardware Setup: The Motion Sensor

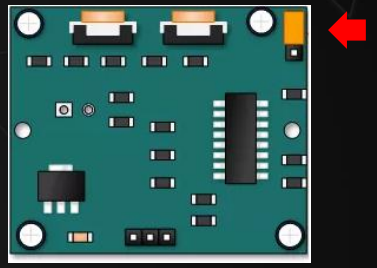
지연 시간 조정

- 움직임이 감지된 후부터 얼마동안 데이터가 'On'으로 머무를 지 결정
- 시계 방향으로 돌리면 지연 시간 증가 (max 5분)
- 반시계 방향으로 돌리면 지연 시간 감소 (min 3초)

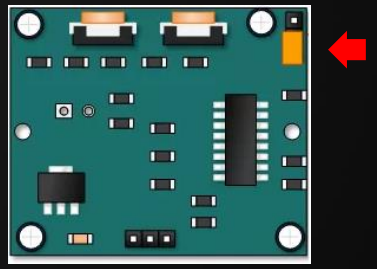
Hardware Setup: The Motion Sensor

트리거 모드 변경

- **Single 트리거 모드**: 최초 움직임이 감지되면 지연 시간이 시작되어 설정된 시간동안 움직임이 있어도 지연 시간이 갱신되지 않음



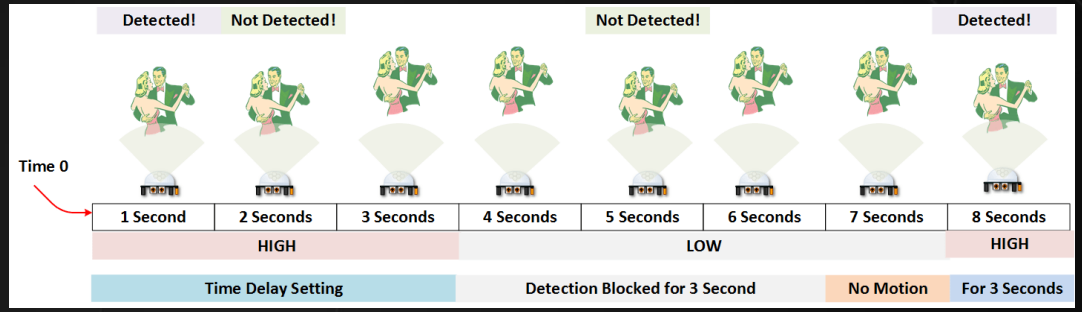
- **Repeatable 트리거 모드**: 움직임이 감지될 때마다 지연 시간이 갱신됨



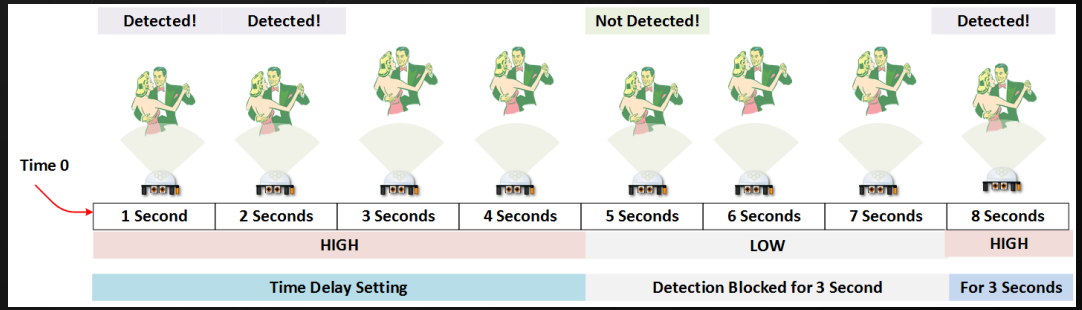
Hardware Setup: The Motion Sensor

⭘ 트리거 모드 / 지연 시간 예제

- Single 트리거
  지연시간 3초



- Repeatable 트리거
  지연시간 3초

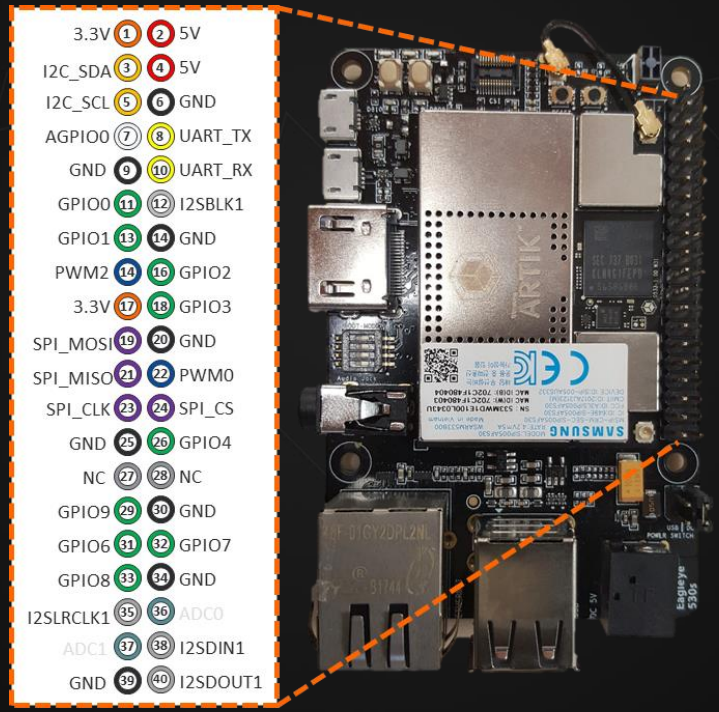


Hardware Setup: The Motion Sensor

실습 준비

Hardware Setup: The Eagleye 530s Board

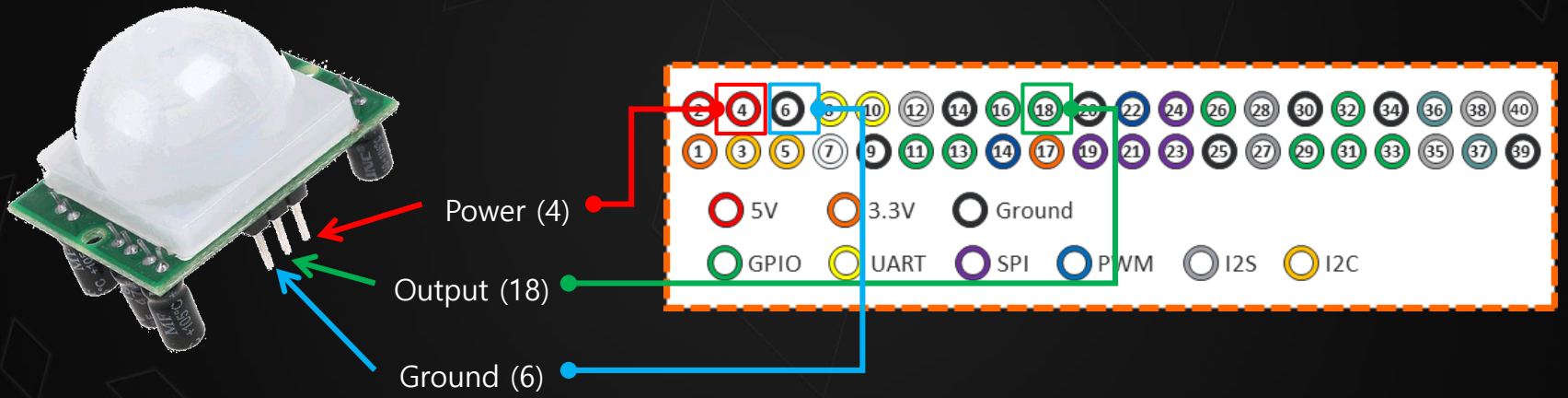
Eagleye 530s Pins



Hardware Setup: Motion Sensor - Board

SAMSUNG Research

Eagleye Board - Motion Sensor 연결



Hardware Setup: The LED Light

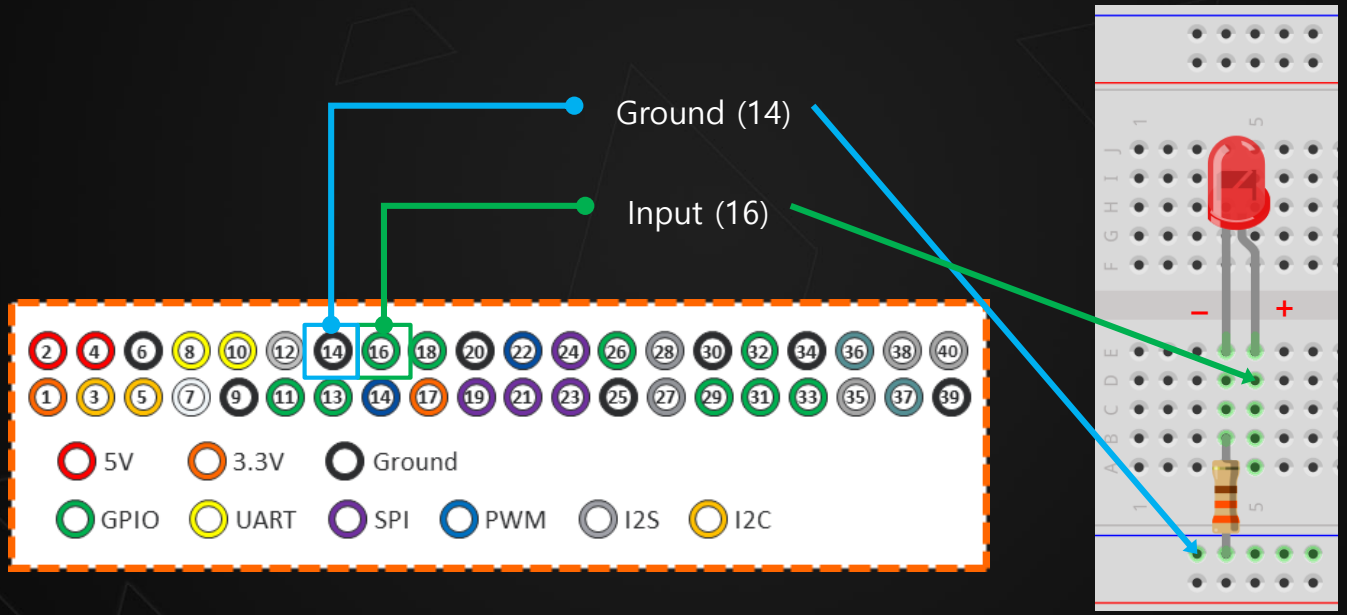
LED 전구 및 저항 연결

- 5파이 LED 전구 (전압 2.2V / 전류 50mA)

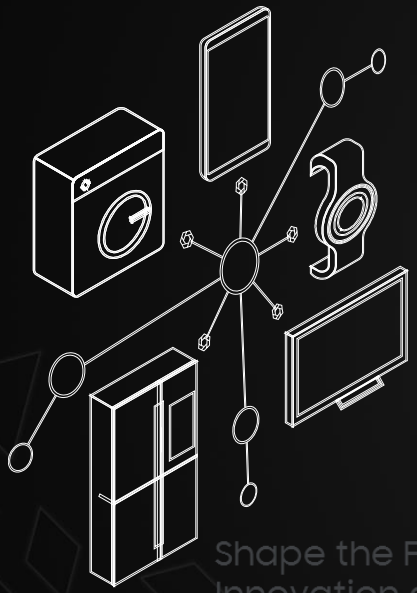
- 공급전압 5V 시, 100~400Ω 저항 사용

Hardware Setup: LED- Board

- Eagleye Board - LED 전구 및 저항 연결



SAMSUNG Research



Shape the Future with
Innovation and Intelligence

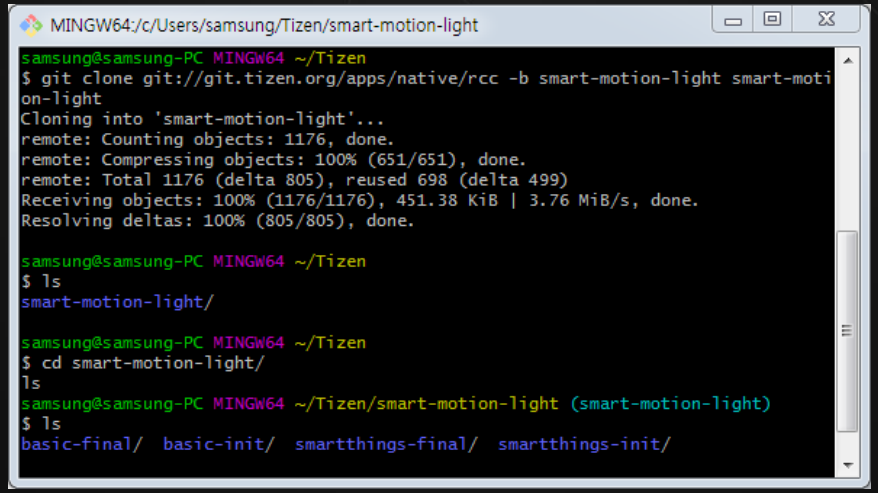
II Motion Light Control App

Code Implementation

Motion Light Control: Code Implementation

Git Bash에서 rcc/smart-motion-light 코드 clone 받기

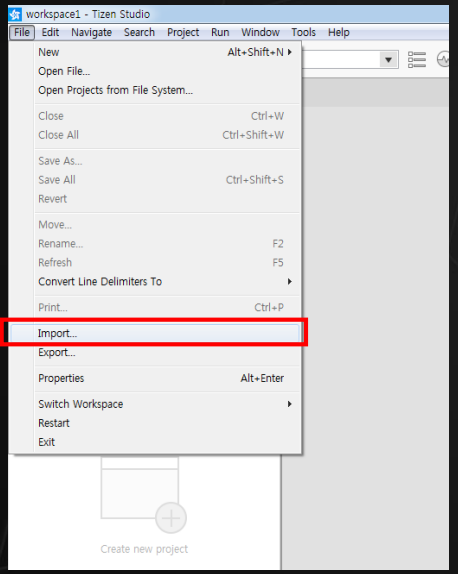
- git clone https://git.tizen.org/cgit/apps/native/rcc
  -b smart-motion-light smart-motion-light

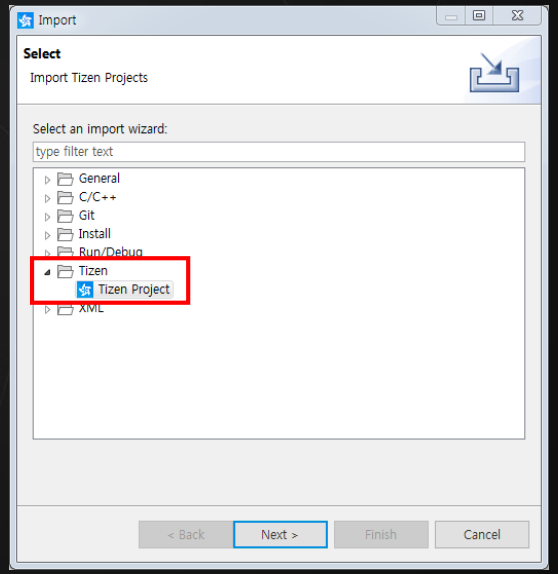


Motion Light Control: Code Implementation

Tizen Studio에서 motion-light-basic 프로젝트 불러오기

- File > Import 클릭 > Tizen > Tizen Project 선택

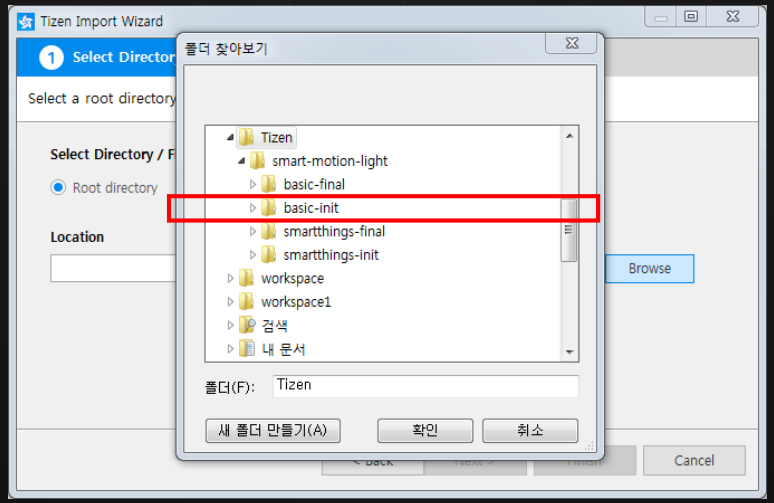




Motion Light Control: Code Implementation

Tizen Studio에서 motion-light-basic 프로젝트 불러오기

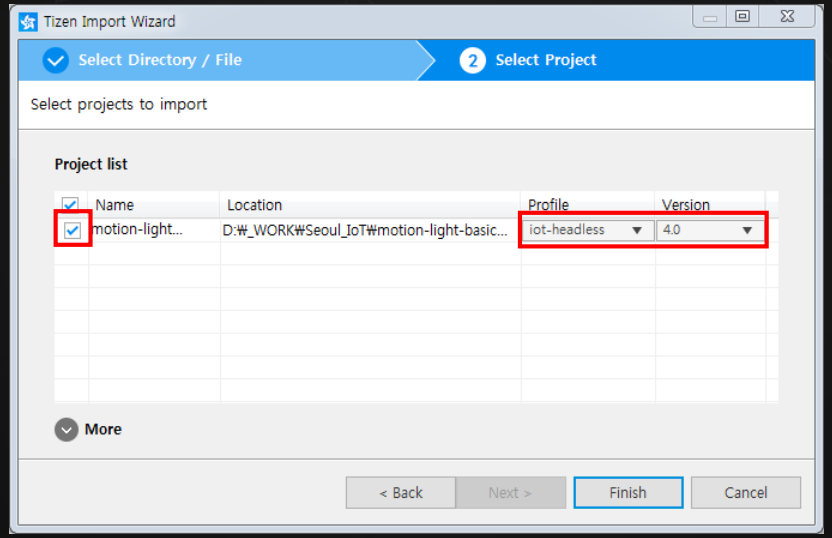
- Root directory 선택 > Browse 클릭
- smart-motion-light/basic-init 폴더 선택



Motion Light Control: Code Implementation

Tizen Studio에서 motion-light-basic 프로젝트 불러오기

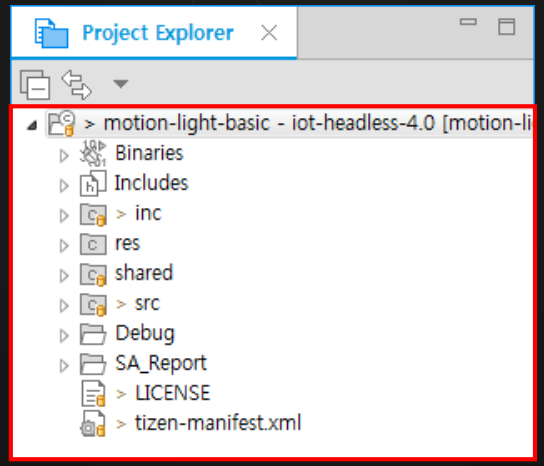
- Profile: iot-headless / Version: 4.0 선택 > 체크박스 선택



Motion Light Control: Code Implementation

Tizen Studio에서 motion-light-basic 프로젝트 불러오기

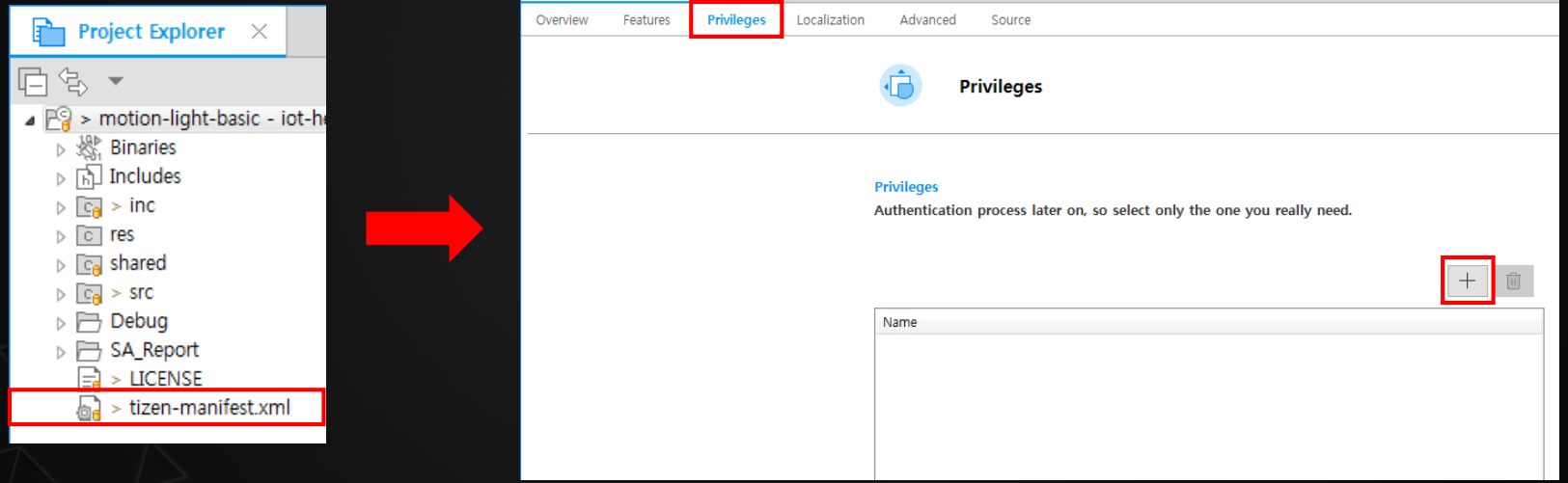
- Project Explorer에서 motion-light-basic 프로젝트 확인



Motion Light Control: Code Implementation

Peripheral IO API 사용을 위해 peripheralio privilege 추가

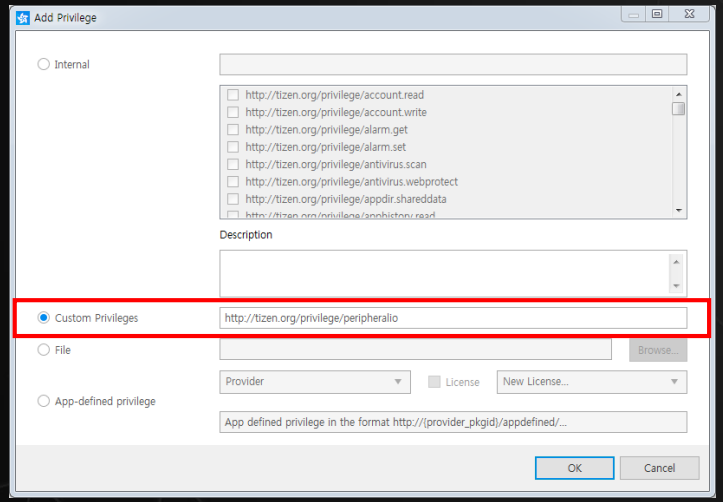
- tizen-manifest.xml 파일 선택 > Privileges 탭 > 추가(+) 버튼 클릭



Motion Light Control: Code Implementation

Peripheral IO API 사용을 위해 peripheralio privilege 추가

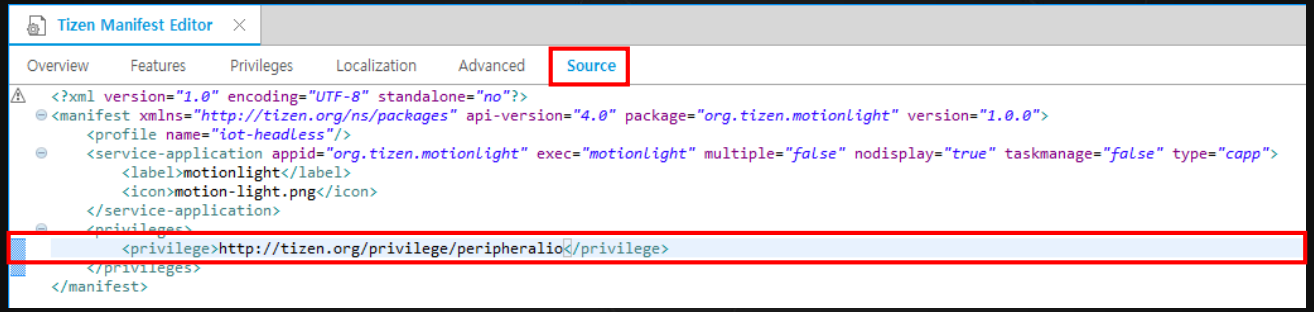
- Custom Privileges 선택 > http://tizen.org/privilege/peripheralio 추가



Motion Light Control: Code Implementation

Peripheral IO API 사용을 위해 peripheralio privilege 추가

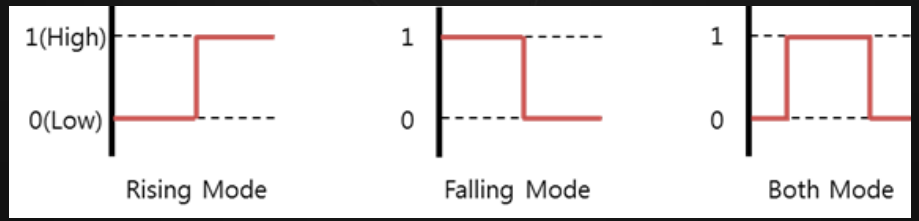
- 파일 저장 후, Source 탭에서 privilege 추가된 것을 재확인



Motion Light Control: Code Implementation

General Purpose Input / Output (GPIO)

- Binary input peripheral 상태 읽기/쓰기 가능한 interface
- Tizen Peripheral IO Native APIs - GPIO 가이드
  https://developer.tizen.org/development/iot-preview/iot-apis/tizen-peripheral-io-native-api/gpio



Motion Light Control: Code Implementation

peripheral_gpio_open()

- 특정 pin 넘버의 Peripheral GPIO 핸들을 열어줌

```
int pin = 26; /* ARTIK 530 : GPIO8, Raspberry Pi 3 : GPIO26 */
peripheral_gpio_h gpio_h;

Int ret = peripheral_gpio_open(pin, &gpio_h);
```

peripheral_gpio_close()

- 더 이상 사용되지 않는 Peripheral GPIO 핸들을 닫아줌

```
peripheral_gpio_close(gpio_h);
```

Motion Light Control: Code Implementation

- **peripheral_gpio_set_direction()**
  - 데이터 전송 방향을 설정함
    - *PERIPHERAL_GPIO_DIRECTION_IN* : 데이터 읽기 모드
    - *PERIPHERAL_GPIO_DIRECTION_OUT_INITIALLY_HIGH* : 데이터 쓰기 모드, 초기값을 high(1)로 설정
    - *PERIPHERAL_GPIO_DIRECTION_OUT_INITIALLY_LOW* : 데이터 쓰기 모드, 초기값을 low(0)로 설정

```
peripheral_gpio_set_direction(gpio_h, PERIPHERAL_GPIO_DIRECTION_OUT_INITIALLY_LOW);
```

Motion Light Control: Code Implementation

peripheral_gpio_read()

- Peripheral 핸들로부터 binary 데이터 값을 읽어옴

```
uint32_t value;

peripheral_gpio_read(gpio_h, &value);
```

peripheral_gpio_write()

- Peripheral 핸들에 binary 데이터 값을 입력함

```
uint32_t value = 1;

peripheral_gpio_write(gpio_h, value);
```

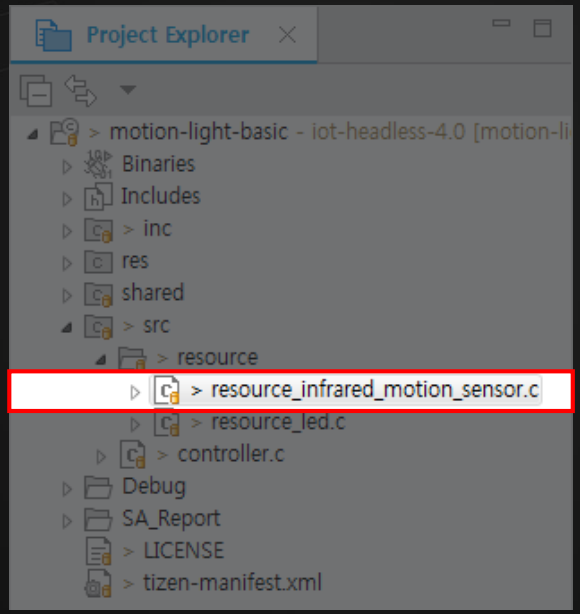
Motion Light Control: Code Implementation

GPIO Pin Number for Eagleye Board

Pin Order	Pin Param	Pin Order	GPIO Param
11	28	29	27
13	29	31	25
16	130	32	0
18	46	33	26
26	14		

Motion Light Control: Code Implementation

- src/resource/resource_infrared_motion_sensor.c



Motion Light Control: Code Implementation

resource_read_infrared_motion_sensor() 함수 내 GPIO 설정 확인 (1)

```
int resource_read_infrared_motion_sensor(int pin_num, uint32_t *out_value)
{
    int ret = PERIPHERAL_ERROR_NONE;

    if (!g_sensor_h) {
        peripheral_gpio_h temp = NULL;

        ret = peripheral_gpio_open(pin_num, &temp);          모션 센서가 연결된 특정 pin 넘버의 Peripheral GPIO 핸들 열기
        retv_if(ret, -1);
                                                             데이터 전송 방향 설정 (읽기 모드)
        ret = peripheral_gpio_set_direction(temp, PERIPHERAL_GPIO_DIRECTION_IN);
        if (ret) {
            peripheral_gpio_close(temp);                     방향 설정 실패 시 열어둔 GPIO 핸들 닫기
            _E("peripheral_gpio_set_direction failed.");
            return -1;
        }

        g_sensor_h = temp;
        g_pin_num = pin_num;
    }
    ...
```

Motion Light Control: Code Implementation

resource_read_infrared_motion_sensor() 함수 내 GPIO 설정 확인 (2)

```
int resource_read_infrared_motion_sensor(int pin_num, uint32_t *out_value)
{
    ...

    if (g_pin_num != pin_num) {
        _E("Invalid pin number.");
        return -1;
    }

    ret = peripheral_gpio_read(g_sensor_h, out_value);
    retv_if(ret < 0, -1);

    return 0;
}
```

GPIO 핸들을 통해 motion값 읽어오기

Motion Light Control: Code Implementation

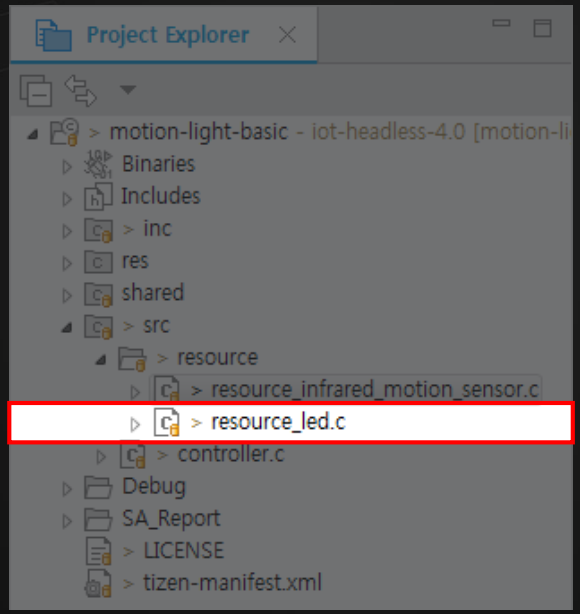
resource_close_infrared_motion_sensor() 함수 내 GPIO 설정 확인

```
void resource_close_infrared_motion_sensor(void)
{
    if (!g_sensor_h) return;

    _I("Infrared Motion Sensor is finishing...");

    peripheral_gpio_close(g_sensor_h);

    g_sensor_h = NULL;
    g_pin_num = -1;
}
```

모션 센서가 연결된 Peripheral GPIO 핸들 닫기

LED Light Control: Code Implementation

- src/resource/resource_led.c



Motion Light Control: Code Implementation

resource_write_led() 함수 내 GPIO 설정 확인 (1)

```
int resource_write_led(int pin_num, int write_value)
{
    int ret = PERIPHERAL_ERROR_NONE;

    if (!g_sensor_h) {
        peripheral_gpio_h temp = NULL;

        ret = peripheral_gpio_open(pin_num, &temp);
        retv_if(ret, -1);

        ret = peripheral_gpio_set_direction(temp, PERIPHERAL_GPIO_DIRECTION_OUT_INITIALLY_LOW);
        if (ret) {
            peripheral_gpio_close(temp);
            _E("peripheral_gpio_set_direction failed.");
            return -1;
        }

        g_sensor_h = temp;
        g_pin_num = pin_num;
    }
    ...
```

LED가 연결된 특정 pin 넘버의 Peripheral GPIO 핸들 열기

데이터 전송 방향 설정 (쓰기 모드)

방향 설정 실패 시 열어둔 GPIO 핸들 닫기

Motion Light Control: Code Implementation

resource_write_led() 함수 내 GPIO 설정 확인 (2)

```
int resource_write_led(int pin_num, int write_value)
{
    ...

    if (g_pin_num != pin_num) {
        _E("Invalid pin number.");
        return -1;
    }

    ret = peripheral_gpio_write(g_sensor_h, write_value);
    retv_if(ret < 0, -1);

    _I("LED Value : %s", write_value ? "ON":"OFF");

    return 0;
}
```

GPIO 핸들을 통해 LED값 설정하기

Motion Light Control: Code Implementation

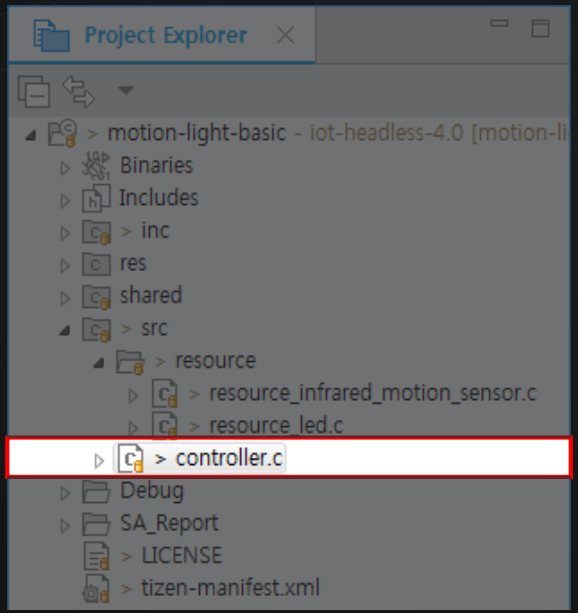
resource_close_led() 함수 내 GPIO 설정 확인

```
void resource_close_led(void)
{
    if (!g_sensor_h) return;

    _I("LED is finishing...");

    peripheral_gpio_close(g_sensor_h);

    g_sensor_h = NULL;
    g_pin_num = -1;
}
```

LED가 연결된 Peripheral GPIO 핸들 닫기

Motion Detection: Code Implementation

- src/controller.c



Motion Light Control: Code Implementation

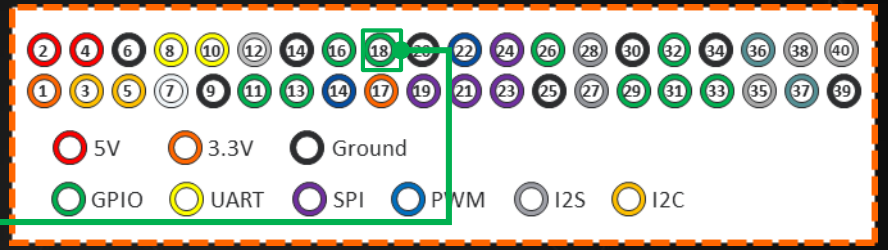
Motion Sensor 값 읽어오기

- 모션 센서가 연결된 특정 Pin에 접근하여 값을 읽어와 value에 저장

```
static Eina_Bool __read_motion_write_led(void *data)
{
    uint32_t value = 0;
    int ret = -1;

    /* Gets value from motion sensor */
    // TODO: Read data from motion sensor
    ret = resource_read_infrared_motion_sensor(SENSOR_MOTION_PIN_NUM, &value);   모션 센서 읽기 함수 호출

    if (ret != 0) _E("Cannot read sensor value");

    _D("Detected motion value is: %d", value);
    ...
}
```

Motion Light Control: Code Implementation

Output



GPIO Pin	Pin Number	GPIO Pin	Pin Number
GPIO 11	28	GPIO 29	27
GPIO 13	29	GPIO 31	25
GPIO 16	130	GPIO 32	0
GPIO 18	46	GPIO 33	26
GPIO 26	14		

Motion Light Control: Code Implementation

Motion Sensor 값 읽어오기

- 모션 센서가 연결된 특정 Pin에 접근하여 값을 읽어와 value에 저장

```
static Eina_Bool __read_motion_write_led(void *data)
{
    uint32_t value = 0;
    int ret = -1;

    /* Gets value from motion sensor */
    // TODO: Read data from motion sensor
    ret = resource_read_infrared_motion_sensor(46, &value);

    if (ret != 0) _E("Cannot read sensor value");

    _D("Detected motion value is: %d", value);
    ...
}
```

모션 센서 읽기 함수 호출

46번 Pin에 접근하여 값을 읽어오도록 설정

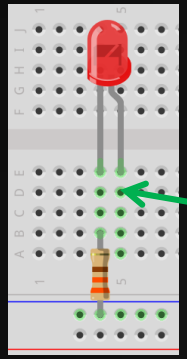
LED Light Control: Code Implementation

LED 값 변경하기

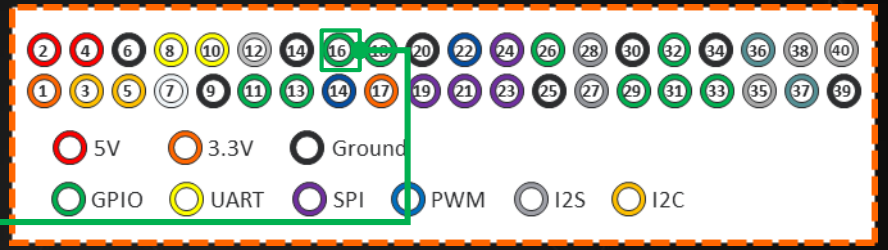
- LED가 연결된 특정 Pin에 접근하여 value 값을 전달

```
static Eina_Bool control_sensors_cb(void *data)
{
    ...

    /* Sends value to LED light */
    // TODO: Send data to LED sensor
    resource_write_led(SENSOR_LED_PIN_NUM, value);                    LED 쓰기 함수 호출

    return ECORE_CALLBACK_RENEW;
}
```

Motion Light Control: Code Implementation



Input



GPIO Pin	Pin Number	GPIO Pin	Pin Number
GPIO 11	28	GPIO 29	27
GPIO 13	29	GPIO 31	25
GPIO 16	130	GPIO 32	0
GPIO 18	46	GPIO 33	26
GPIO 26	14		

LED Light Control: Code Implementation

LED 값 변경하기

- LED가 연결된 특정 Pin에 접근하여 value 값을 전달

```
static Eina_Bool control_sensors_cb(void *data)
{
    ...

    /* Sends value to LED light */
    // TODO: Send data to LED sensor
    resource_write_led(130, value);                    LED 쓰기 함수 호출

    return ECORE_CALLBACK_RENEW;
}
```

130번 Pin에 접근하여 값을 쓰도록 설정

Motion Light Control: Code Implementation

service_app_create() 함수 내 타이머 생성

- 특정 주기마다 모션 센서 값을 불러오고 LED 값을 설정해주는 **__read_motion_write_led** 함수 호출

```
static bool service_app_create(void *data)
{
    ...

    ad->getter_timer = ecore_timer_add(1.0f, __read_motion_write_led, ad);
    if (!ad->getter_timer) {
        _E("Failed to add infrared motion getter timer");
        return false;
    }
    ...
}
```

타이머 생성

주기는 1초(1.0f)로 설정

Motion Light Control: Code Implementation

service_app_terminate() 함수 내 리소스 정리

- App 종료 시, 사용하던 리소스 정리

```
static void service_app_terminate(void *data)
{
    app_data *ad = (app_data *)data;

    // TODO: Delete timer
    if (ad->getter_timer)
        ecore_timer_del(ad->getter_timer);

    // TODO: Close infrared motion & led resources
    resource_close_infrared_motion_sensor();
    resource_close_led();

    // TODO: Free data resource
    free(ad);
}
```

타이머 삭제

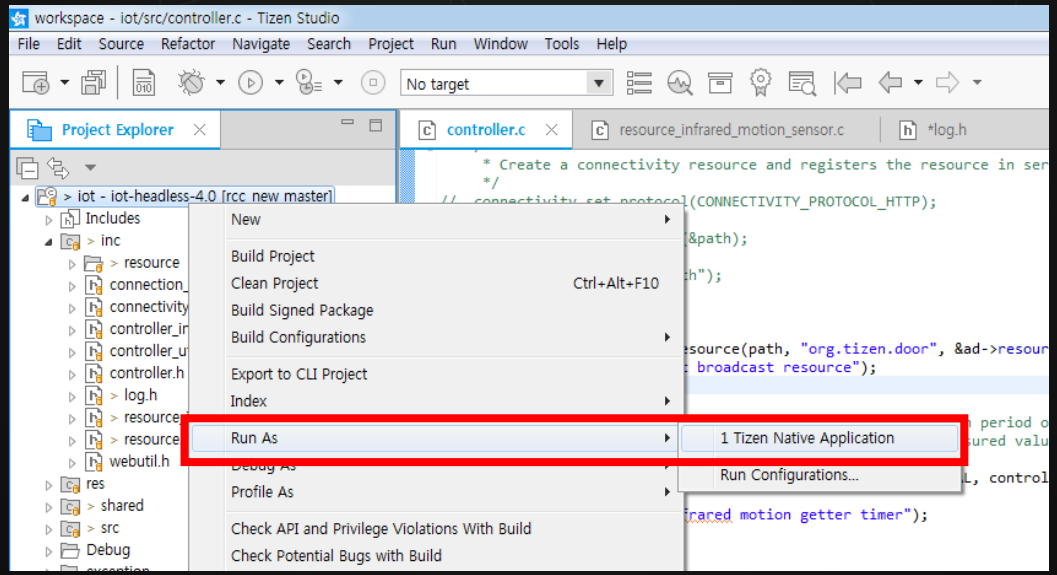
리소스 종료 함수 호출

데이터 리소스 해제

Motion Light Control: Code Implementation

Tizen App 실행

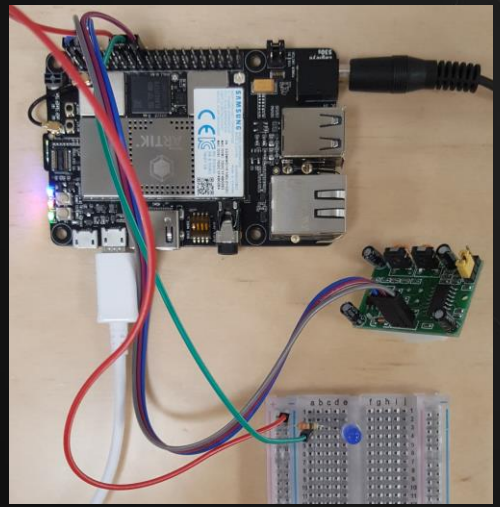
- Project Explorer 내 프로젝트 우클릭 > Run As > Tizen Native App

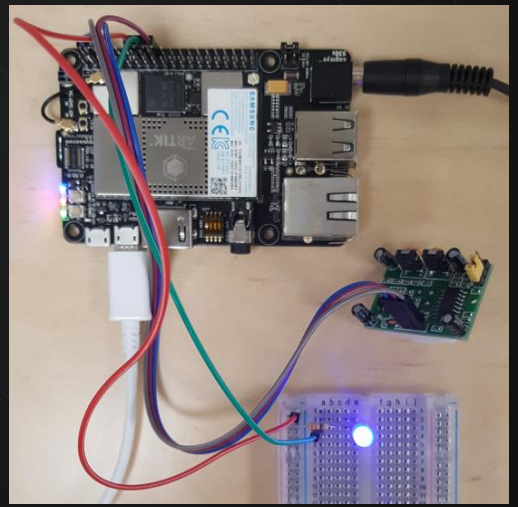


LED Light Control: Code Implementation

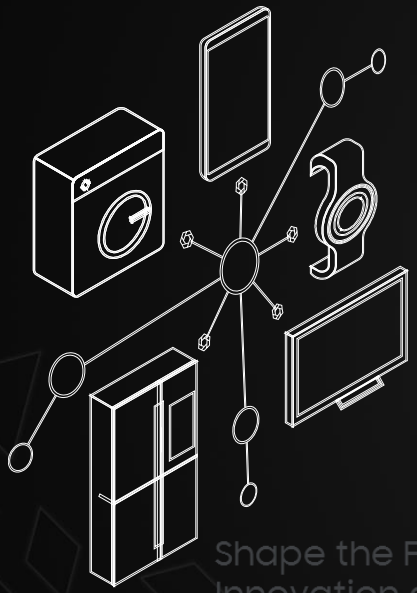
Motion - LED Light 확인

- Motion sensor 값에 따른 LED Light 변화 확인





SAMSUNG Research



Shape the Future with
Innovation and Intelligence

Ⅲ SmartThings App

Overview

SmartThings

SmartThings

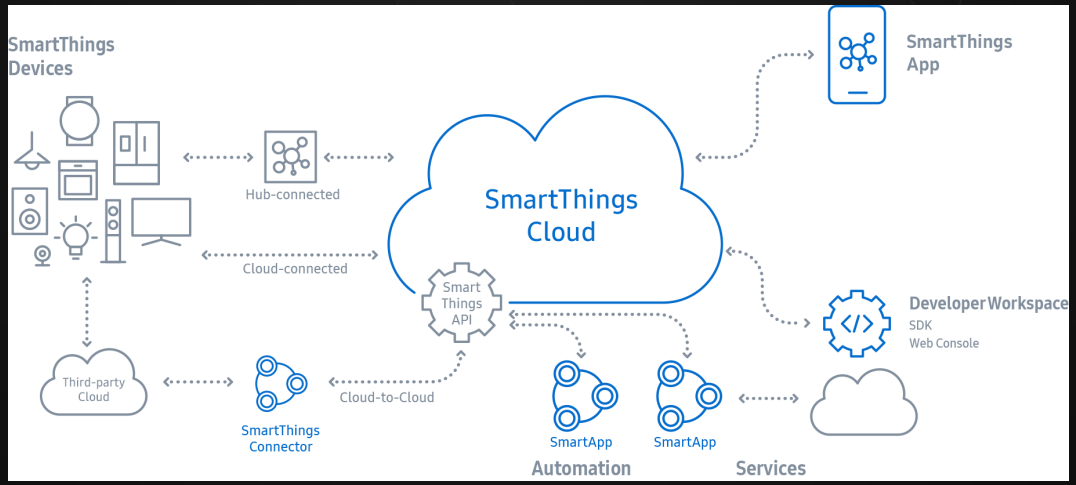
- SmartThings 앱으로 스마트 디바이스를 연결하여 관리 및 제어할 수 있도록 제공되는 IoT 플랫폼
- 높은 확장성을 가져 삼성전자의 제품 외에도 다양한 제품군을 지원함
- **타이젠을 통해 보다 쉽게 SmartThings 앱 개발이 가능**

- SmartThings 공식 홈페이지
- SmartThings Developers 홈페이지

SmartThings

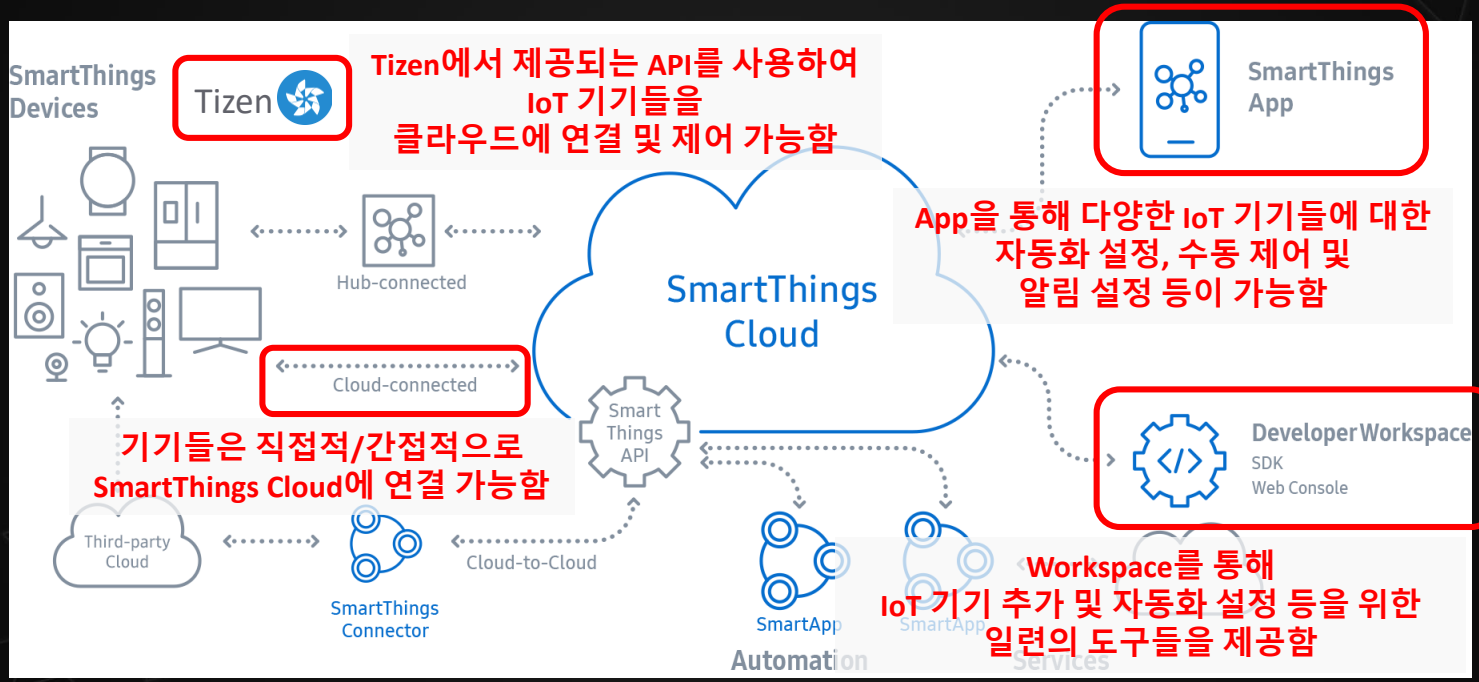
SmartThings Ecosystem

- 다양한 스마트 기기와 SmartThings App 등이 SmartThings Cloud를 통해 서로 연결되고, 제공되는 SmartThings API를 통해 제어 가능



SmartThings

SmartThings Ecosystem



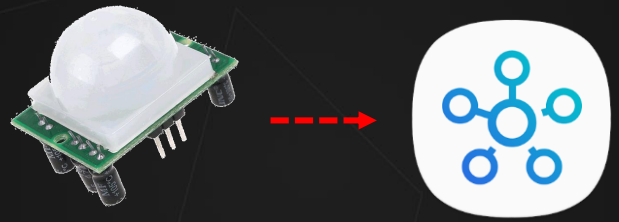
SmartThings

Cloud-connected Device

- Open Connectivity Foundation (OCF) spec의 통신 protocol에 기반하여 SmartThings Cloud와 직접적으로 통신하는 Cloud-connected Tizen App 개발

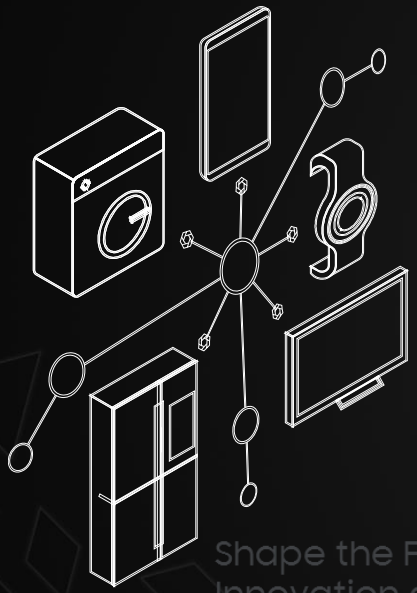
- Today's SmartThings App

1. 모션 센서에 움직임이 감지되면 SmartThings App을 통해 알림 받기



2. SmartThings App을 통해 LED Light를 on/off 시키기

SAMSUNG Research



Shape the Future with
Innovation and Intelligence

Ⅲ SmartThings App
Developer Workspace

SmartThings: Developer Workspace

Developer Workspace 가입

- https://devworkspace.developer.samsung.com/smartthingsconsole
- 삼성 계정으로 가입 및 로그인

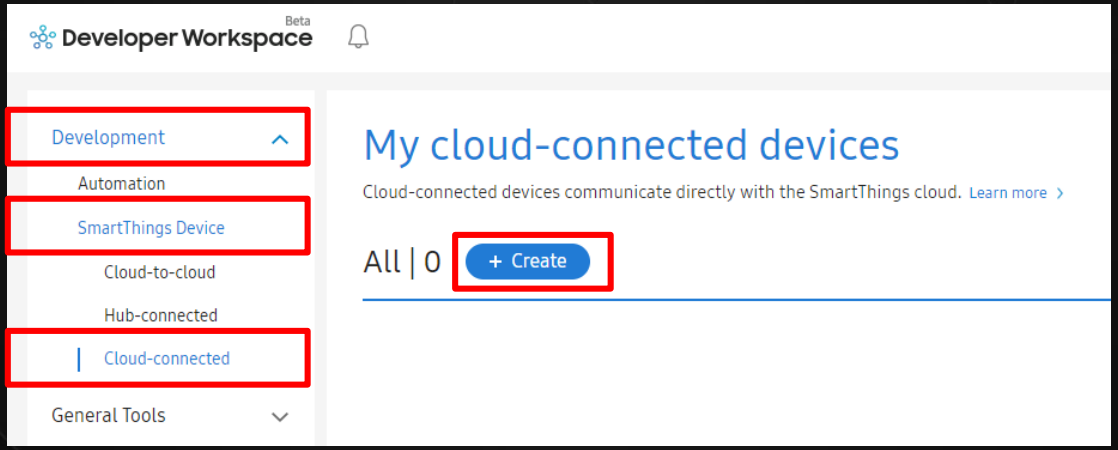
SmartThings App 다운로드

- 안드로이드 모바일 앱 다운로드
- SmartThings (* SmartThings Classic 아님)

SmartThings: Developer Workspace

Cloud-connected 디바이스 생성

- Developer Workspace > Development > SmartThings Device > Cloud-connected > + Create 버튼 클릭



SmartThings: Developer Workspace

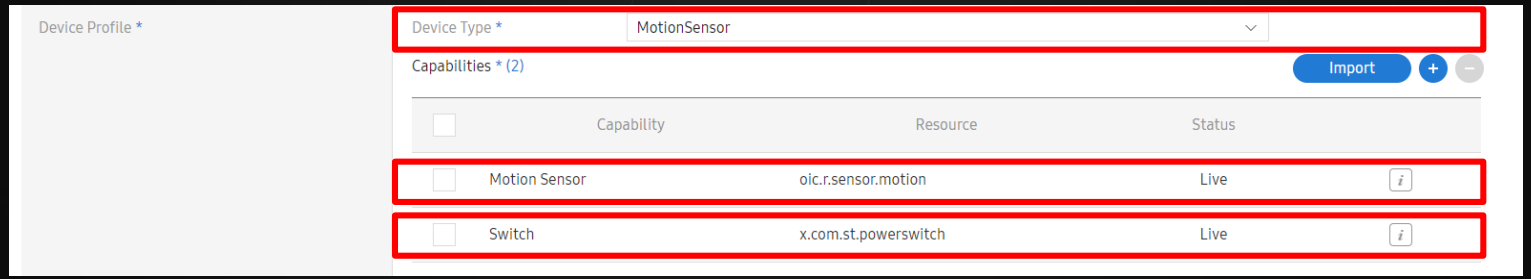
Device Information – 기본 정보 입력

- Device Name 및 VID 정보는 Tizen App의 정보와 같아야 함으로 유의

SmartThings: Developer Workspace

Device Information – Device Profile 입력

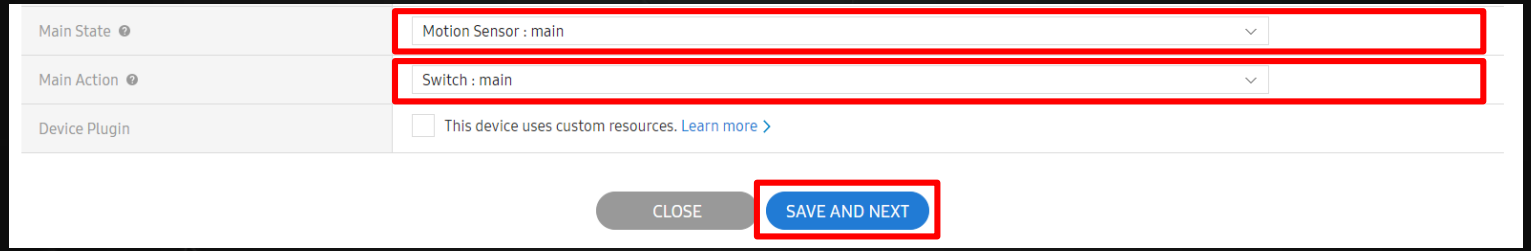
- Sensor에 따라 device type과 필요한 capabilities 선택
  - Device Type: MotionSensor
  - Capabilities: Motion Sensor & Switch



SmartThings: Developer Workspace

Device Information – Main State & Action 설정

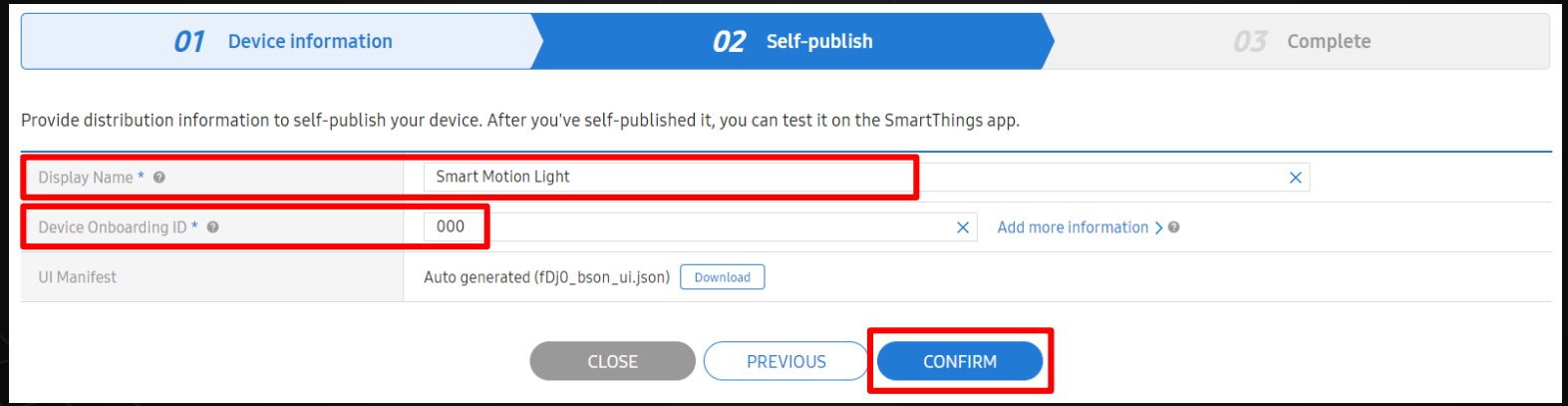
- SmartThings App 메인 화면에 표시될 State와 Action 설정
  - Main State: Motion Sensor : main
  - Main Action: Switch : main
  - App의 UI 변경이 필요하다면 Custom Device Plugin 사용



SmartThings: Developer Workspace

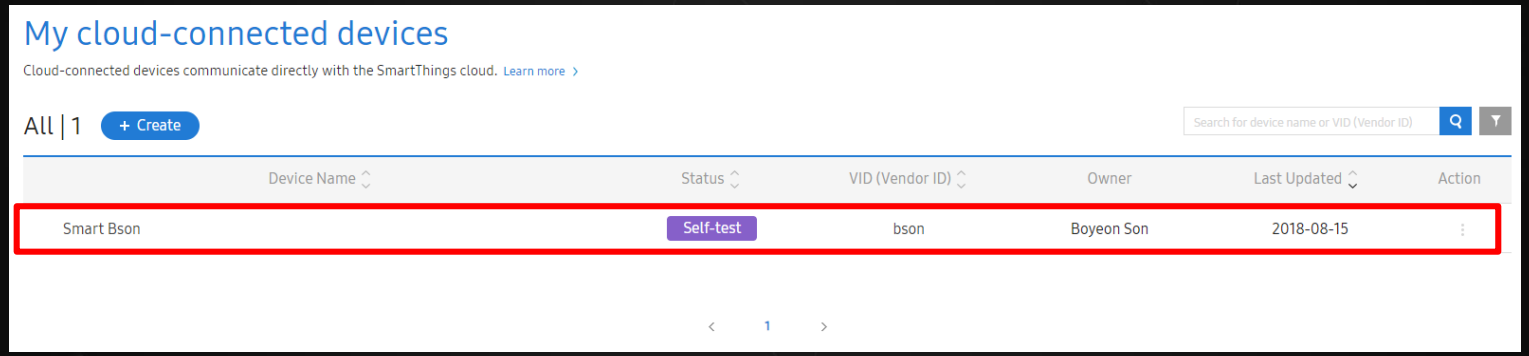
Self-publish

- Display Name: 연결 시 표시 될 디바이스 이름
- Device Onboarding ID: SmartThings App과의 easy setup 시 디바이스의 soft AP 이름에 사용될 임의의 숫자 3개

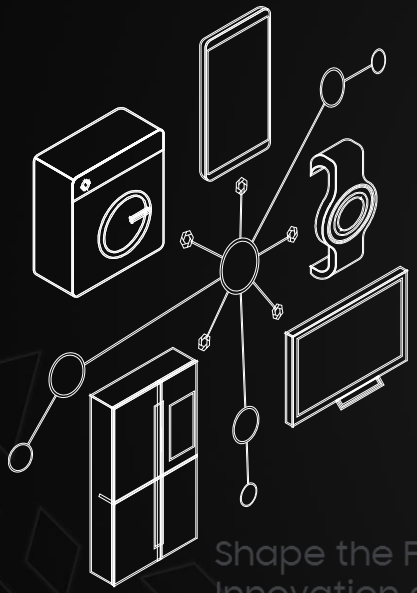


SmartThings: Developer Workspace

- 새로 생성된 디바이스 확인
  - My cloud-connected devices 목록에서 추가된 디바이스 확인



SAMSUNG Research



Shape the Future with
Innovation and Intelligence

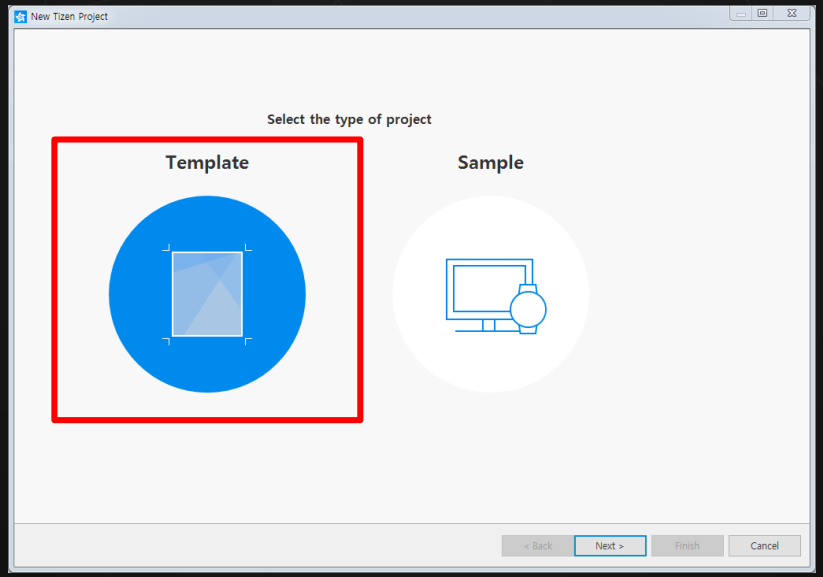
Ⅲ SmartThings App

Tizen Development

SmartThings: Tizen Development

Tizen Studio 내 Headless things App Template 생성

- File > New > Tizen Project > Template 선택

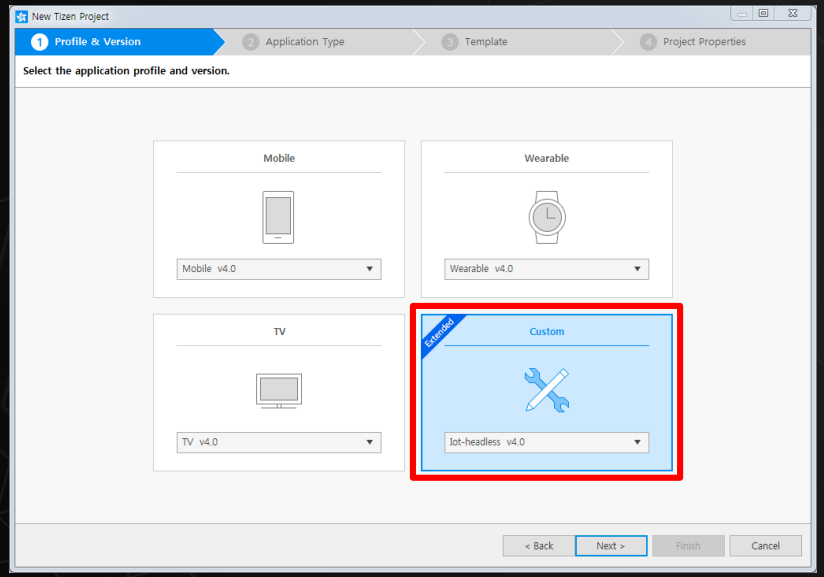


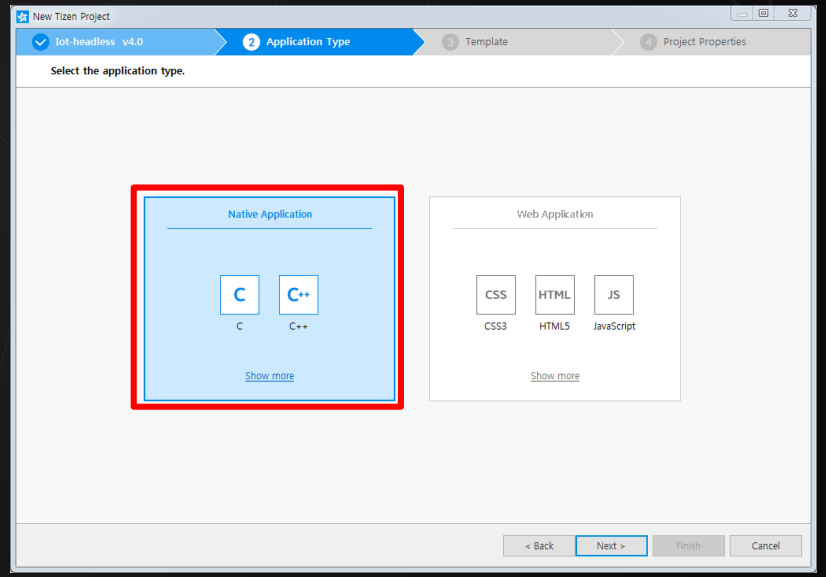
SmartThings: Tizen Development

SAMSUNG Research

Tizen Studio 내 Headless things App Template 생성

- Iot-headless v4.0 선택 > Native Application 선택

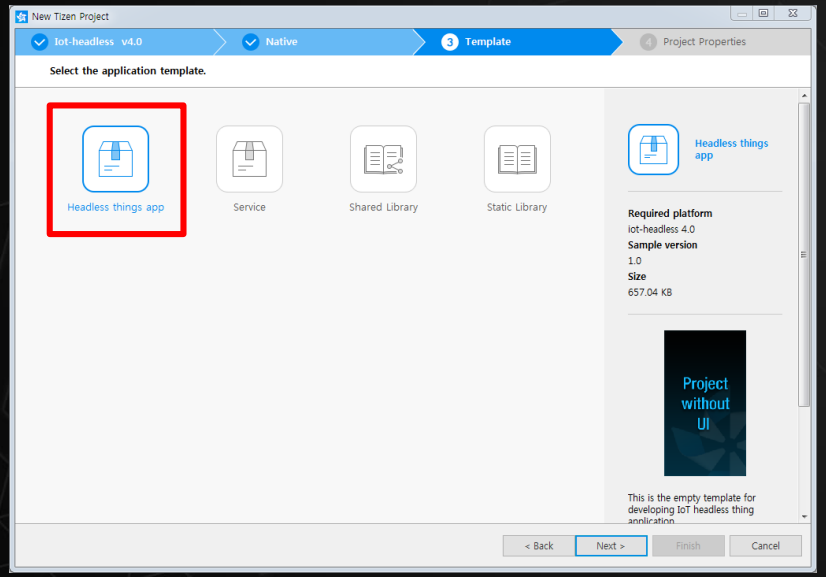


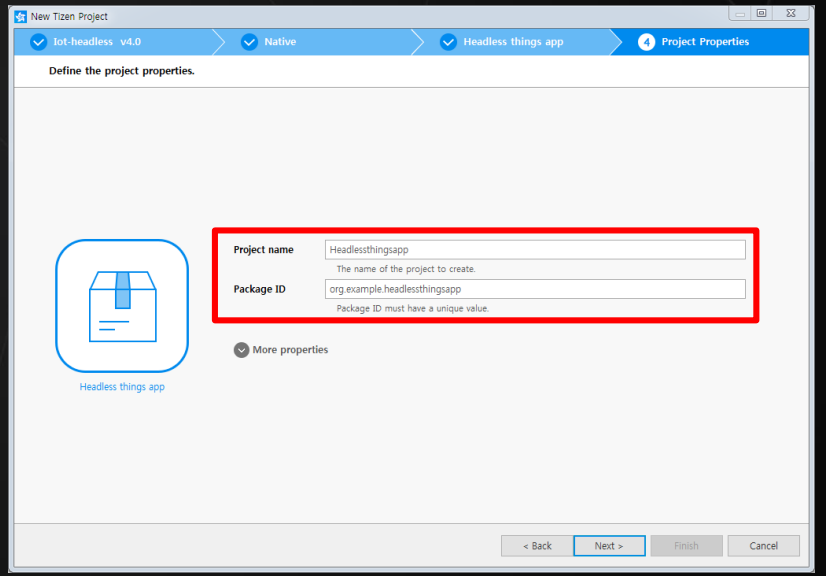


SAMSUNG Research

Tizen Studio 내 Headless things App Template 생성

- Headless things app 선택 > Project Name & Package ID 설정

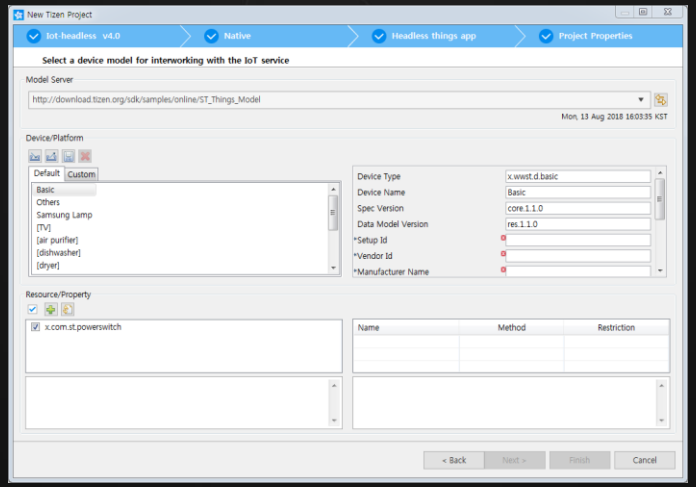




SmartThings: Tizen Development

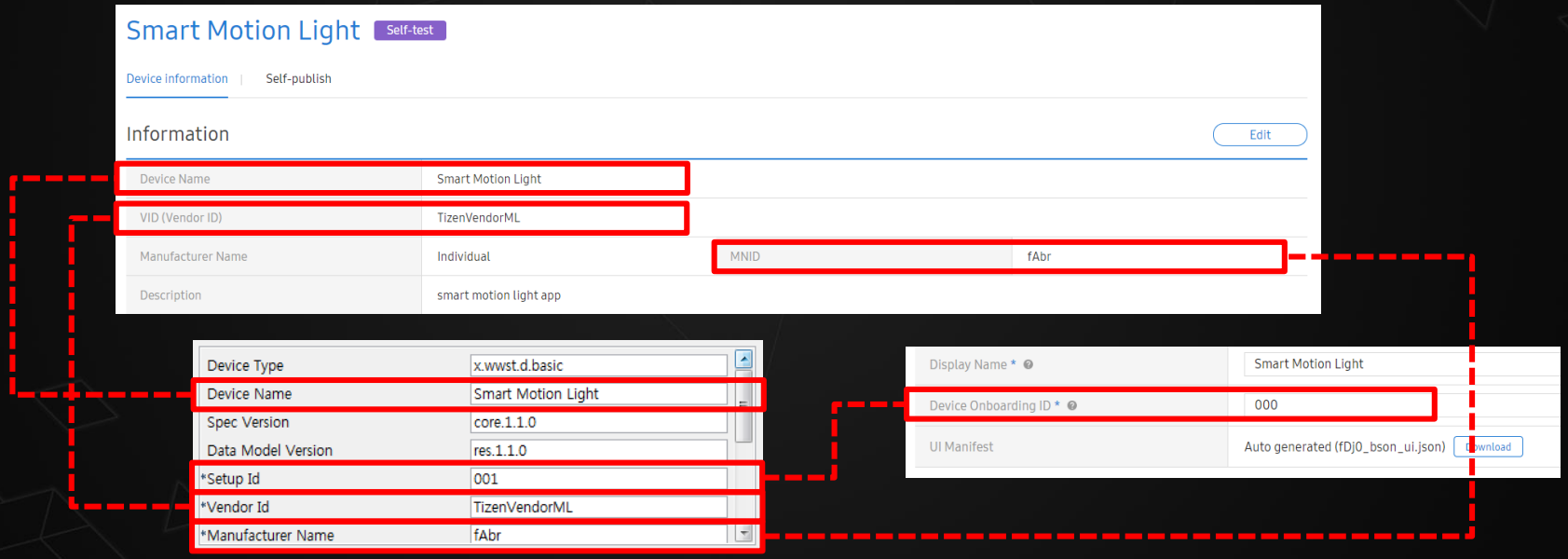
Tizen Studio 내 Headless things App Template 생성

- Device 관련 정보 입력
- Developer Workspace 내 Device 정보와 동일하게 입력



Tizen Studio 내 Headless things App Template 생성

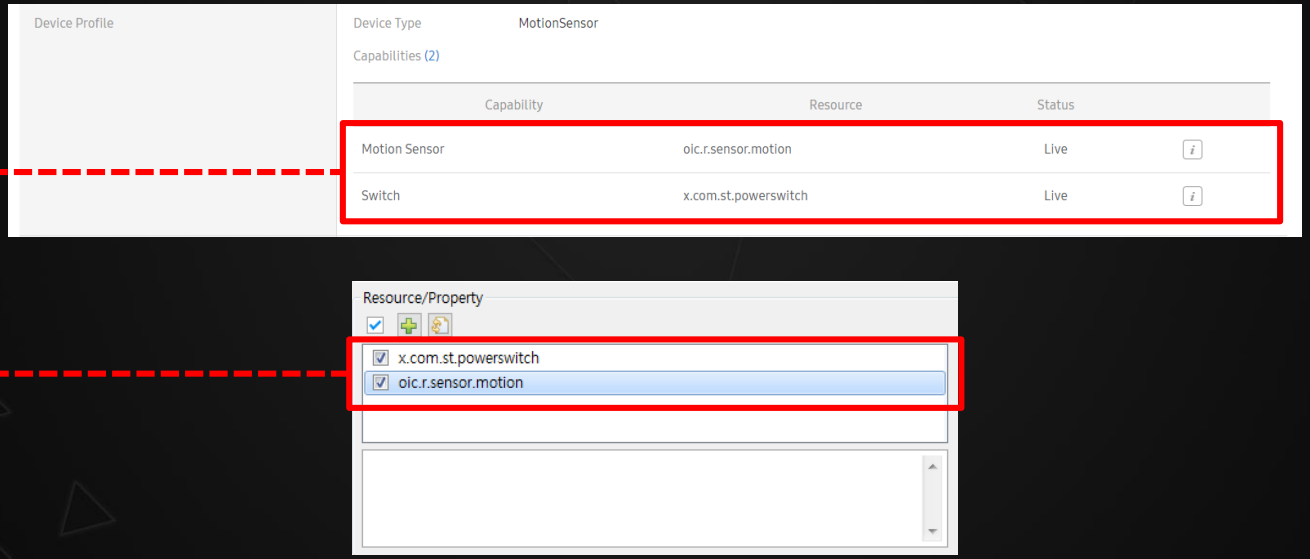
- Developer Workspace 내 Device 정보와 동일하게 입력



SmartThings: Tizen Development

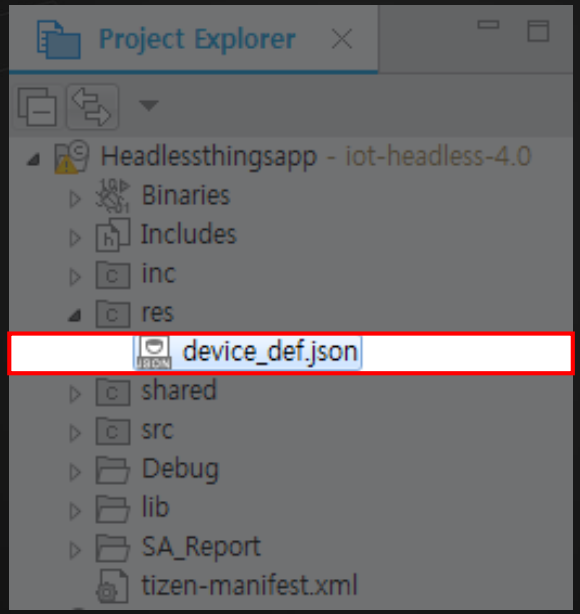
Tizen Studio 내 Headless things App Template 생성

- Developer Workspace 내 Device 정보와 동일하게 입력



SmartThings: Tizen Development

- res/device_def.json



SmartThings: Tizen Development

- device_def.json에서 설정 값 확인

device_def.json

```
{
  "device": [
    {
      "specification": {
        "device": {
          "deviceType": "x.wwst.d.basic",
          "deviceName": "Smart Motion Light",
          "specVersion": "core.1.1.0",
          "dataModelVersion": "res.1.1.0"
        },
        "platform": {
          "manufacturerName": "fAbr",
          "manufacturerUrl": "http://www.samsung.com/sec/",
          "manufacturingDate": "2017-11-29",
          "modelNumber": "NWSP-01",
          "platformVersion": "1.0",
          "osVersion": "1.0",
          "hardwareVersion": "1.0",
          "firmwareVersion": "1.0",
          "vendorId": "TizenVendorML"
        }
      },
      "resources": {
        "single": [
          {
            "uri": "/capability/switch/main/0",
            "types": [
              "x.com.st.powerswitch"
            ],
```

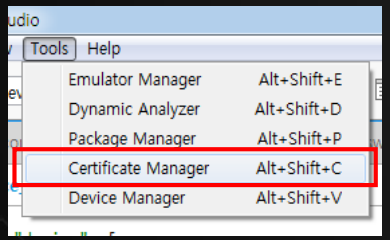
device_def.json

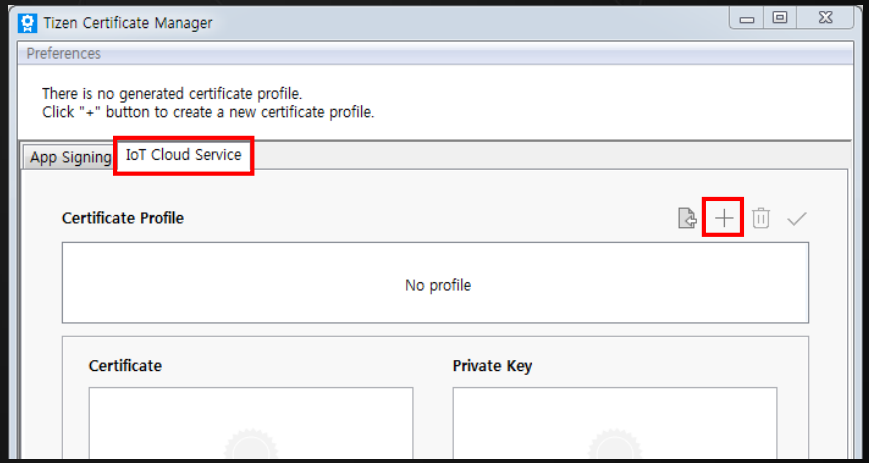
```
      }
    ],
    "resourceTypes": [
      {
        "type": "x.com.st.powerswitch",
        "properties": [
          {
            "key": "power",
            "type": 3,
            "mandatory": true,
            "rw": 3
          }
        ]
      },
      {
        "type": "oic.r.sensor.motion",
        "properties": [
          {
            "key": "value",
            "type": 0,
            "mandatory": false,
            "rw": 1
          }
        ]
      }
    ],
    "configuration": {
      "easySetup": {
        "connectivity": {
```

SmartThings: Tizen Development

IoT Cloud Certificate 생성

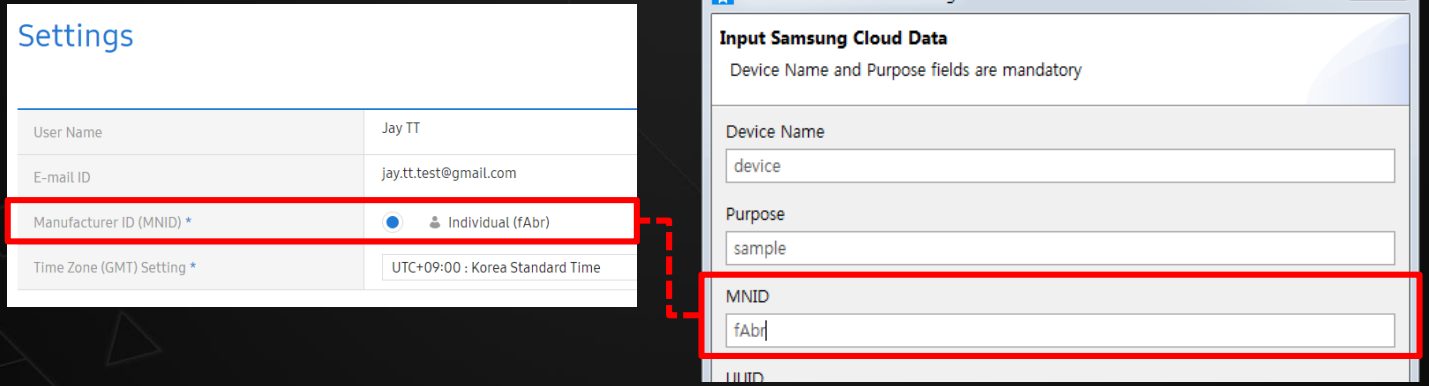
- SmartThings App은 별도의 Iot Cloud용 인증서가 필요
- Tools > Certificate Manager 선택 > IoT Cloud Service 탭 선택





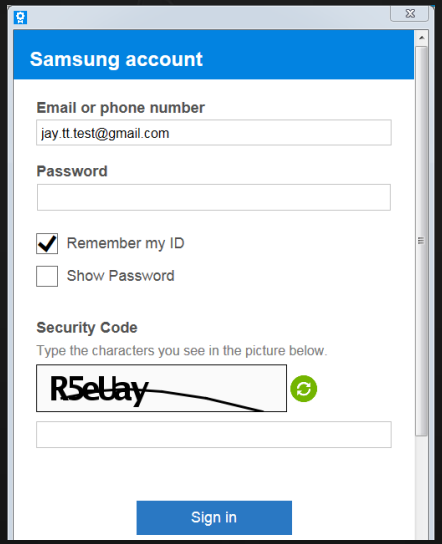
IoT Cloud Certificate 생성

- 개인 정보 입력 / 디바이스 정보 입력
  - MNID는 Developer Workspace의 MNID와 동일해야함으로 유의
  - 본인의 MNID는 Worksapce의 계정 settings에서 확인 가능



IoT Cloud Certificate 생성

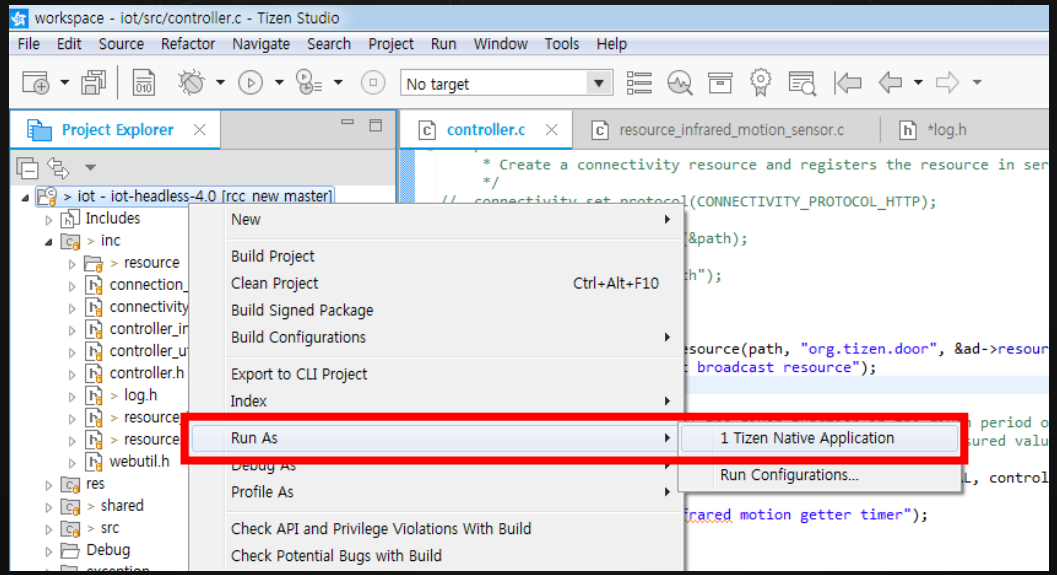
- Developer Workspace에 등록된 Samsung 계정으로 로그인



SmartThings: Tizen Development

Tizen App 실행

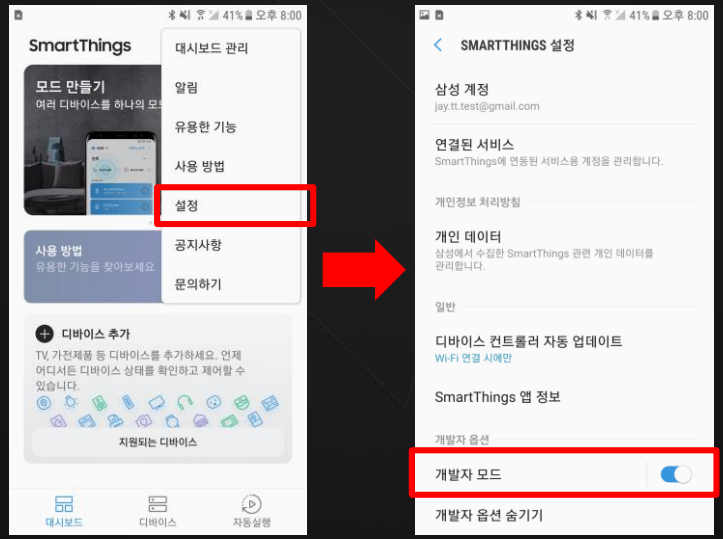
- Project Explorer 내 프로젝트 우클릭 > Run As > Tizen Native App



SmartThings: Easy Setup

SmartThings Mobile App 개발자 모드 설정

- 우측 상단 More(...) 버튼 클릭 > 설정 > 개발자 모드 ON



SmartThings: Easy Setup

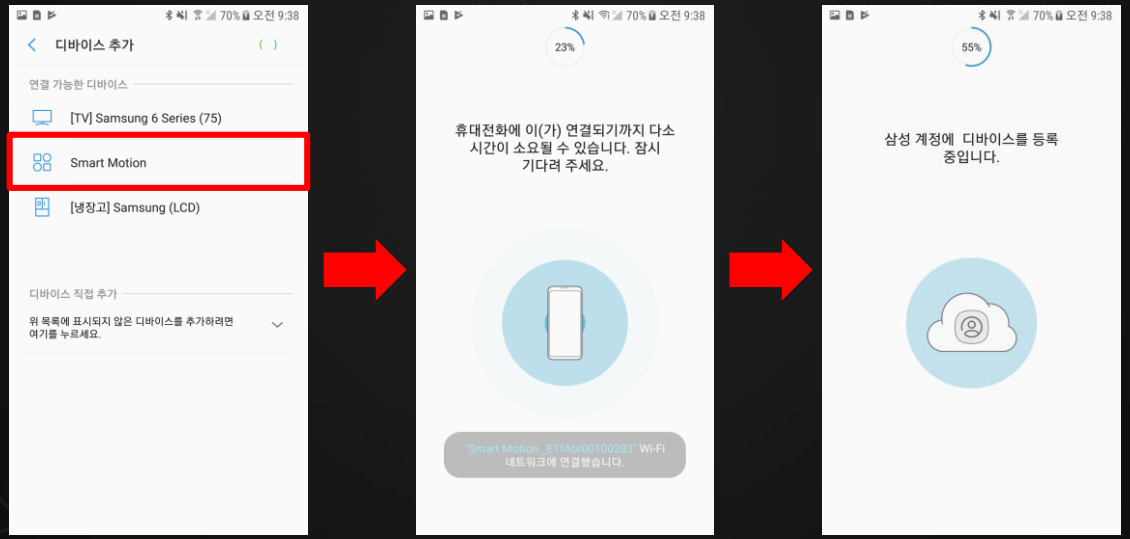
Easy Setup: SmartThings App – Device 연결

- 디바이스 탭 선택 > 디바이스 추가

SmartThings: Easy Setup

Easy Setup: SmartThings App – Device 연결

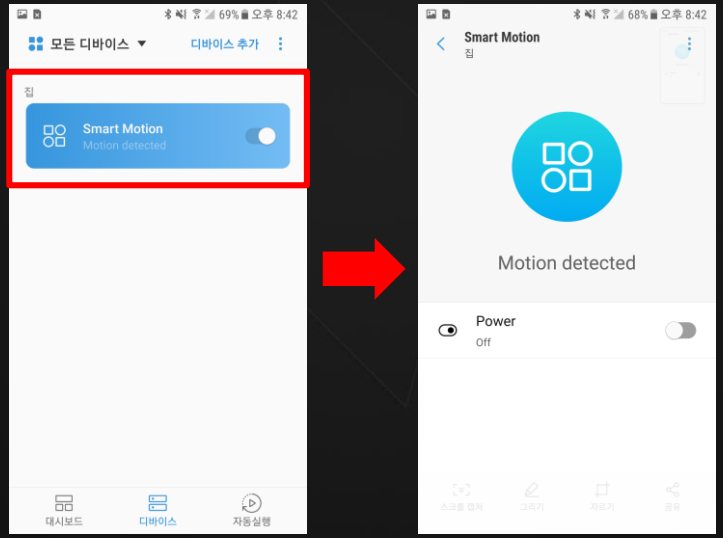
- 본인의 디바이스 선택



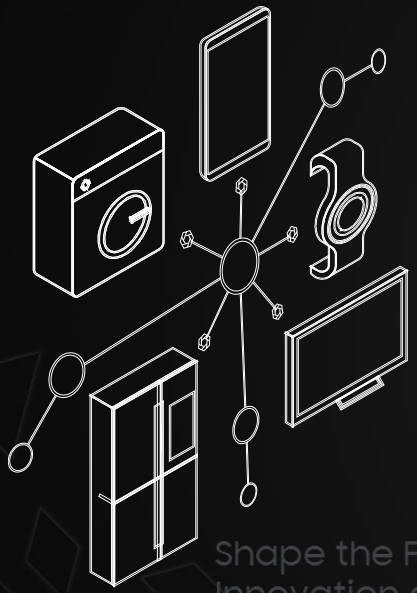
SmartThings: Easy Setup

Easy Setup: SmartThings App – Device 연결

- App UI (Motion 표시창, Power 버튼) 확인



SAMSUNG Research



Shape the Future with
Innovation and Intelligence

Ⅲ SmartThings App
Code Implementation

SmartThings: Code Implementation

- **Tizen에서 제공되는 Things SDK API를 활용**
  - Tizen Things SDK API
    https://developer.tizen.org/development/iot-preview/iot-apis/things-sdk-api
  - Tizen Things SDK API Usage
    https://developer.tizen.org/development/iot-preview/iot-apis/things-sdk-api/api-usage
  - ST Things SDK Documentation
    https://developer.tizen.org/dev-guide/things-sdk/index.html

SmartThings: Code Implementation

디바이스 설정 파일 설정

- 디바이스 설정 파일이 저장되는 경로를 지정

```
int st_things_set_configuration_prefix_path(const char* ro_path, const char* rw_path);
```

디바이스 초기화

- SmartThings와의 연동을 위한 초기화 진행 & easy setup status 확인

```
int st_things_initialize(const char *json_path, bool *easysetup_complete);
```

디바이스 자원 정리

- SmartThings와의 연동이 끝날 때 detinitialize 프로세스 진행

```
int st_things_deinitialize(void);
```

SmartThings: Code Implementation

주요 콜백 등록

- Request Callback: SmartThings 클라우드로부터의 요청 처리
- Reset Callback: 디바이스 초기화 관련 요청 처리
- User Confirm Callback: 상호인증 방식으로 easy setup 진행 처리
- Things Status Change Callback: SmartThings 클라우드와의 연동 과정에서 벌어지는 상태 변화를 확인

```
int st_things_register_request_cb(st_things_get_request_cb get_cb, st_things_set_request_cb set_cb);
int st_things_register_reset_cb(st_things_reset_confirm_cb confirm_cb, st_things_reset_result_cb result_cb);
int st_things_register_user_confirm_cb(st_things_user_confirm_cb confirm_cb);
int st_things_register_reset_cb(st_things_reset_confirm_cb confirm_cb, st_things_reset_result_cb result_cb);
```

SmartThings: Code Implementation

Get Request Callback

- 특정 리소스의 property에 대하여 Get 요청 처리

```
typedef bool (*st_things_get_request_cb)(st_things_get_request_message_s *req_msg,
st_things_representation_s *resp_rep);
```

Set Request Callback

- 특정 리소스의 property에 대하여 Set 요청 처리

```
typedef bool (*st_things_set_request_cb)(st_things_set_request_message_s *req_msg,
st_things_representation_s *resp_rep);
```

SmartThings: Code Implementation

Notifying Observers

- 특정 리소스(센서)에 변동사항이 생기면 클라우드에 알려줌

```
int st_things_notify_observers(const char *resource_uri);
```

클라우드 연동 시작

- SmartThings Cloud와의 연동 시작

```
int st_things_start(void);
```

클라우드 연동 종료

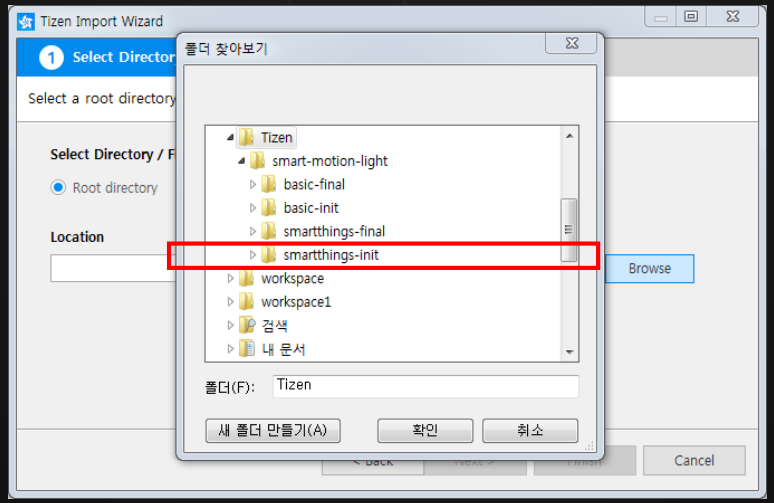
- SmartThings Cloud와의 연동 종료

```
int st_things_stop(void);
```

SmartThings: Code Implementation

Tizen Studio에서 motion-light-smartthings 프로젝트 불러오기

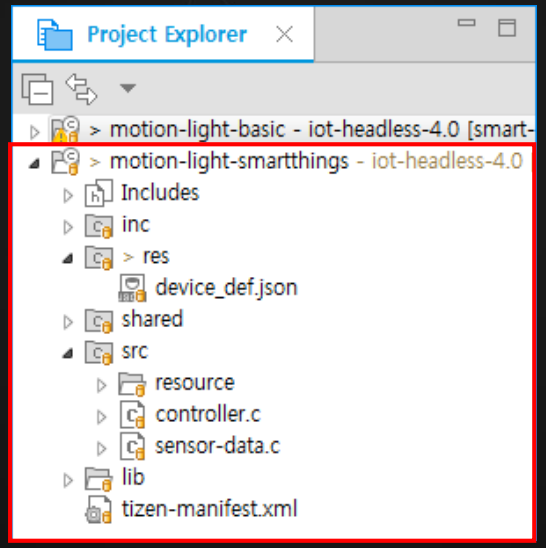
- Root directory 선택 > Browse 클릭
- smart-motion-light/smartthings-init 폴더 선택



SmartThings: Code Implementation

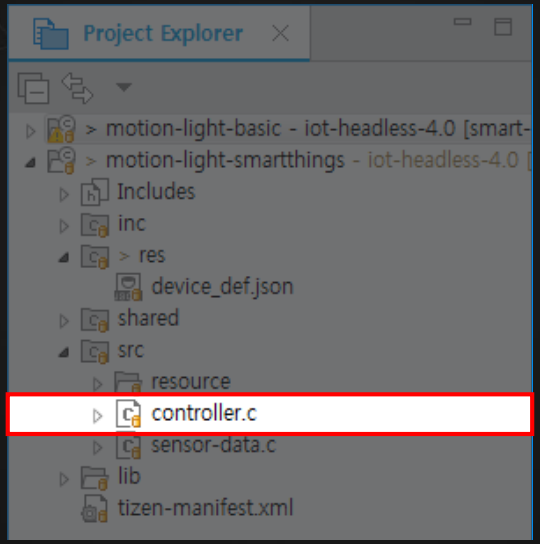
Tizen Studio에서 motion-light-smartthings 프로젝트 불러오기

- Project Explorer에서 motion-light-smartthings 프로젝트 확인



SmartThings: Code Implementation

- src/controller.c



SmartThings: Code Implementation

SmartThings API 사용 시 필요한 Macro 값 확인

device_def.json

```
"resources": {
  "single": [
    {
      "uri": "/capability/switch/main/0",
      "types": [
        "x.com.st.powerswitch"
      ],
      "interfaces": [
        "oic.if.a",
        "oic.if.baseline"
      ],
      "policy": 3
    },
    {
      "uri": "/capability/motionSensor/main/0",
      "types": [
        "oic.r.sensor.motion"
      ],
      "interfaces": [
        "oic.if.s",
        "oic.if.baseline"
      ],
      "policy": 3
```

```
#define JSON_NAME "device_def.json"
#define SENSOR_MOTION_URI "/capability/motionSensor/main/0"
#define SENSOR_MOTION_KEY "value"
#define SENSOR_LED_URI "/capability/switch/main/0"
#define SENSOR_LED_KEY "power"
#define SENSOR_LED_INIT "off"
```

SmartThings: Code Implementation

SmartThings API 사용 시 필요한 Macro 값 확인

device_def.json

```
    }
  ],
  "resourceTypes": [
    {
      "type": "x.com.st.powerswitch",
      "properties": [
        {
          "key": "power",
          "type": 3,
          "mandatory": true,
          "rw": 3
        }
      ]
    },
    {
      "type": "oic.r.sensor.motion",
      "properties": [
        {
          "key": "value",
          "type": 0,
          "mandatory": false,
          "rw": 1
        }
      ]
    }
```

```
#define JSON_NAME "device_def.json"
#define SENSOR_MOTION_URI "/capability/motionSensor/main/0"
#define SENSOR_MOTION_KEY "value"
#define SENSOR_LED_URI "/capability/switch/main/0"
#define SENSOR_LED_KEY "power"
#define SENSOR_LED_INIT "off"
```

SmartThings: Code Implementation

__thing_init() 함수 내 디바이스 초기화

- 디바이스 설정 파일이 저장되는 경로를 지정

```
static int __things_init(void)
{
  ...
  app_res_path = app_get_resource_path();
  ...

  app_data_path = app_get_data_path();
  ...

  // TODO: Specify the read-only and read-write path
  if (0 != st_things_set_configuration_prefix_path(app_res_path, app_data_path)) {
    _E("st_things_set_configuration_prefix_path() failed!!");
    free(app_res_path);
    free(app_data_path);
    return -1;
  }
  ...
}
```

read-only 파일을 저장할 경로 (app_res_path) 및
read-write 파일을 저장할 경로 (app_data_path) 지정

SmartThings: Code Implementation

__thing_init() 함수 내 디바이스 초기화

- SmartThings와의 연동을 위한 초기화 진행

```
static int __things_init(void)
{
  bool easysetup_complete = false;
  ...

  // TODO: Specify the device configuration JSON file and change the status of easysetup_complete
  if (0 != st_things_initialize(app_json_path, &easysetup_complete)) {
    _E("st_things_initialize() failed!!");
    return -1;
  }
  ...
}
```

디바이스와 리소스에 대해 정의하고 있는 JSON 파일의 경로 및 easysetup 완료 여부를 확인할 수 있는 boolean 변수의 주소 지정

SmartThings: Code Implementation

__thing_init() 함수 내 주요 콜백 등록

- request / restet / user_confirm / status_change에 대한 콜백 함수 등록

```
static int __things_init(void)
{
  ...

  // TODO: Register callback for handling request get (handle_get_request) and request set (handle_set_request)
messages
  st_things_register_request_cb(handle_get_request, handle_set_request);
  // TODO: Register callback for reset confirmation (handle_reset_request) and reset result(handle_reset_result)
functions
  st_things_register_reset_cb(handle_reset_request, handle_reset_result);
  // TODO: Register callback for getting user confirmation for ownership transfer (handle_ownership_transfer_request)
  st_things_register_user_confirm_cb(handle_ownership_transfer_request);
  // TODO: Register callback for getting notified when ST Things state changes (handle_things_status_change)
  st_things_register_things_status_change_cb(handle_things_status_change);

  ...
}
```

SmartThings: Code Implementation

handle_get_request()

- Request 메시지의 resource_uri에 따라 get request 처리 함수 호출

```
static bool handle_get_request(st_things_get_request_message_s* req_msg, st_things_representation_s* resp_rep)
{
  ...
  _D("resource_uri [%s]", req_msg->resource_uri);
  retv_if(!g_ad, false);

  if (0 == strcmp(req_msg->resource_uri, SENSOR_MOTION_URI)) {
    _D("query : %s, property: %s", req_msg->query, req_msg->property_key);

    // TODO: Call handle get request function for motion sensor
    ret = __handle_get_request_on_motion(req_msg, resp_rep);
  } else if (0 == strcmp(req_msg->resource_uri, SENSOR_LED_URI)) {
    _D("query : %s, property: %s", req_msg->query, req_msg->property_key);

   // TODO: Call handle get request function for LED
    ret = __handle_get_request_on_led(req_msg, resp_rep);
  }
  ...
}
```

get request 메시지의 resource_uri가 SENSOR_MOTION_URI와 같으면 Motion sensor에 대한 get_request 처리 함수 호출

get request 메시지의 resource_uri가 SENSOR_LED_URI와 같으면 LED에 대한 get_request 처리 함수 호출

SmartThings: Code Implementation

__handle_get_request_on_motion()

- Motion property 관련 데이터 값을 변경

```
static bool __handle_get_request_on_motion (st_things_get_request_message_s* req_msg, st_things_representation_s*
resp_rep)
{
  if (req_msg->has_property_key(req_msg, SENSOR_MOTION_KEY)) {
    bool value = false;

    sensor_data_get_bool(g_ad->motion_data, &value);

    // TODO: Update the response representation about the Sensor property which is sent to the client
    resp_rep->set_bool_value(resp_rep, SENSOR_MOTION_KEY, value);

    _D("Value : %d", value);

    return true;
  } else {
    _E("not supported property");

    return false;
  }
}
```

SmartThings: Code Implementation

__handle_get_request_on_led()

- LED property 관련 데이터 값을 변경

```
static bool __handle_get_request_on_led (st_things_get_request_message_s* req_msg, st_things_representation_s*
resp_rep)
{
  if (req_msg->has_property_key(req_msg, SENSOR_LED_KEY)) {
    const char *str = NULL;

    sensor_data_get_string(g_ad->led_data, &str);

    if (!str) {
      str = SENSOR_LED_INIT;
    }

    // TODO: Update the response representation about the LED property which is sent to the client
    resp_rep->set_str_value(resp_rep, SENSOR_LED_KEY, str);

    ...
  }
  ...
}
```

SmartThings: Code Implementation

handle_set_request()

- Request 메시지의 resource_uri에 따라 set request 처리 함수 호출

```
static bool handle_set_request(st_things_set_request_message_s* req_msg, st_things_representation_s* resp_rep)
{
  bool ret = false;

  _D("resource_uri [%s]", req_msg->resource_uri);
  retv_if(!g_ad, false);

  if (0 == strcmp(req_msg->resource_uri, SENSOR_LED_URI)) {
    // TODO: Call handle set request function for LED
    ret = __handle_set_request_on_led(req_msg, resp_rep);
  } else {
    _E("not supported uri");
  }

  return ret;
}
```

set request 메시지의 resource_uri가 SENSOR_LED_URI와 같으면
LED에 대한 set_request 처리 함수 호출

SmartThings: Code Implementation

__handle_set_request_on_led()

- LED property 관련 데이터 값 변경 및 LED의 상태 변경

```
static bool __handle_set_request_on_led (st_things_set_request_message_s* req_msg, st_things_representation_s*
resp_rep)
{
  int ret = 0;
  char *str = NULL;

  if (req_msg->rep->get_str_value(req_msg->rep, SENSOR_LED_KEY, &str)) {
    ...
    // TODO: Update the response representation about the LED property which is sent to the client
    resp_rep->set_str_value(resp_rep, SENSOR_LED_KEY, str);

    // Turn on LED light with __change_led_data()
    ret = __change_led_data(g_ad, strdup(str));

    retv_if(ret != 0, false);
  } else {
    ...
  }
  ...
}
```

SmartThings: Code Implementation

__change_led_data() 함수 내 notify 설정

- LED 데이터 변경 시, 클라우드에 notify하도록 설정

```
static int __change_led_data(void *data, char *state) {
  ...

  sensor_data_set_string(g_ad->led_data, state, strlen(state));

  if (0 == strcmp(state, LED_ON)) {
    ret = resource_write_led(130, 1);
  } else {
    ret = resource_write_led(130, 0);
  }

  retv_if(ret != 0, -1);

  // TODO: Notify observers of the LED resource
  st_things_notify_observers(SENSOR_LED_URI);

return 0;
}
```

SmartThings: Code Implementation

__change_motion_sensor_data() 함수 내 notify 설정

- Motion sensor 데이터 변경 시, 클라우드에 notify하도록 설정

```
static Eina_Bool __change_motion_sensor_data(void *data)
{
  uint32_t value = 0;

  /* Gets value from motion sensor */
  int ret = resource_read_infrared_motion_sensor(46, &value);

  if (ret != 0) _E("Cannot read sensor value");

  sensor_data_set_bool(g_ad->motion_data, value);

  _D("Detected motion value is: %d", value);

  // TODO: Notify observers of the Sensor motion resource
  st_things_notify_observers(SENSOR_MOTION_URI);

  return ECORE_CALLBACK_RENEW;
}
```

SmartThings: Code Implementation

기존 설치된 app package 삭제

- sdb shell 접속

  Device Manager > 폴더 표시창에서 우클릭 > Open shell 선택
- 설치된 패키지 확인

  pkgcmd -l -t tpk
- 패키지 삭제

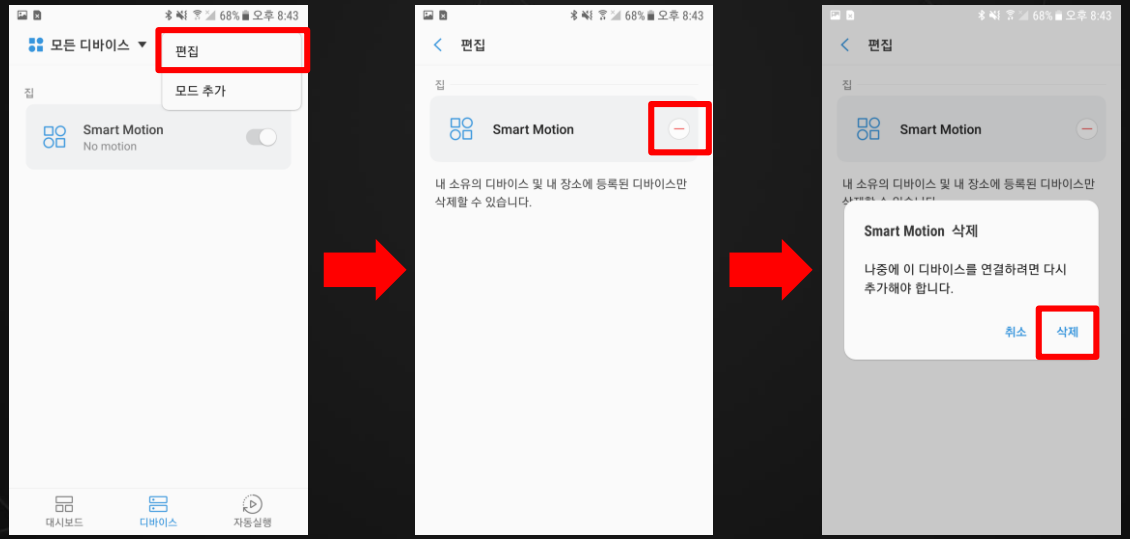
  pkgcmd -u -n org.tizen.motionlight

  pkgcmd -u -n org.example.headlessthingsapp

SmartThings: Code Implementation

⭘ 기존 연결된 디바이스 삭제

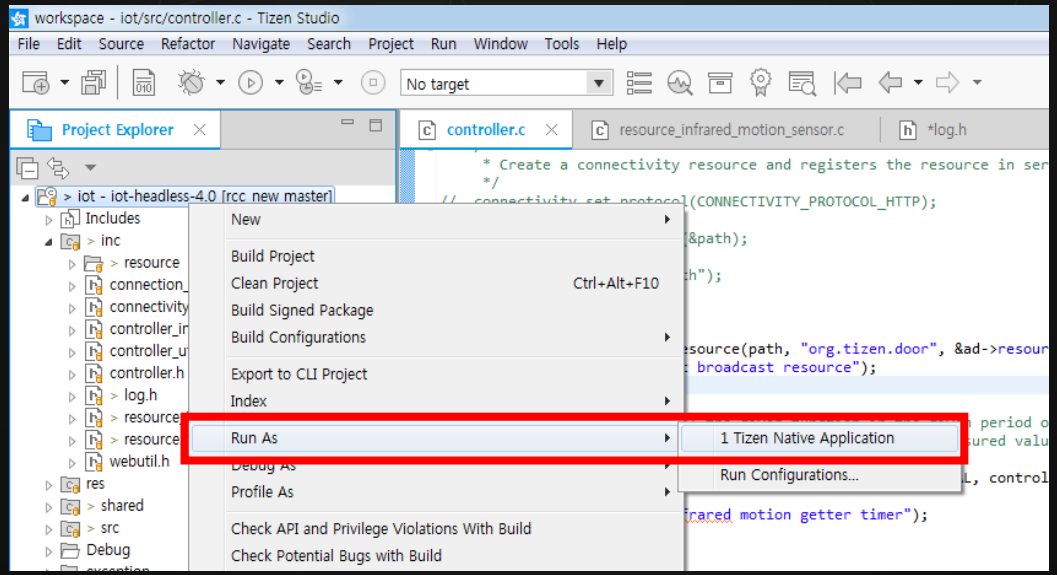
- 우측 상단 More(...) 버튼 클릭 > 편집 > 삭제(-) 버튼 클릭 > 삭제



SmartThings: Code Implementation

Tizen App 실행

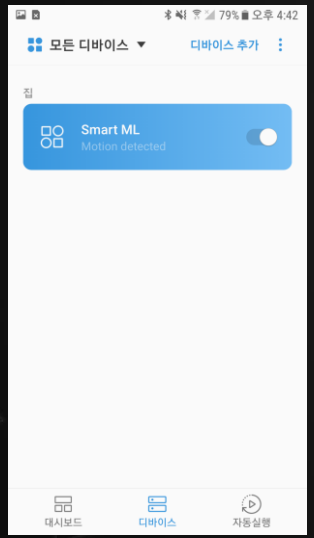
- Project Explorer 내 프로젝트 우클릭 > Run As > Tizen Native App

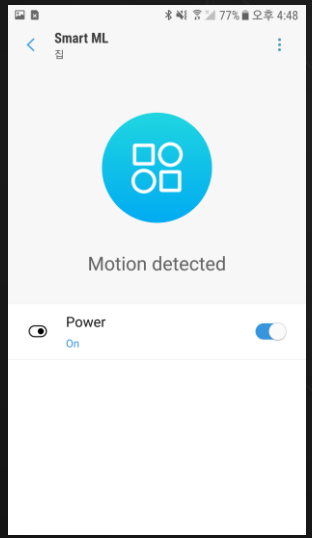


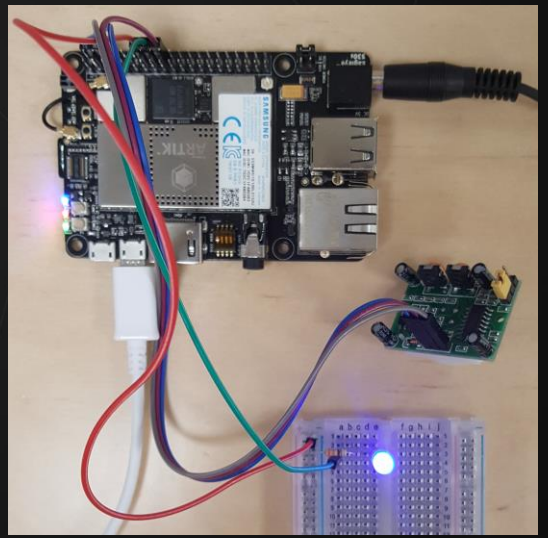
SmartThings: Code Implementation

Easy Setup: SmartThings App – Device 연결

- App UI (Motion 표시창, Power 버튼) 확인







SAMSUNG Research

Thank you