Lab 5: Smart Home Monitoring using OM2M, Node-RED, Wyliodrin, and App Inventor 2

物聯網技術與應用(英) IoT/M2M Technologies and Applications

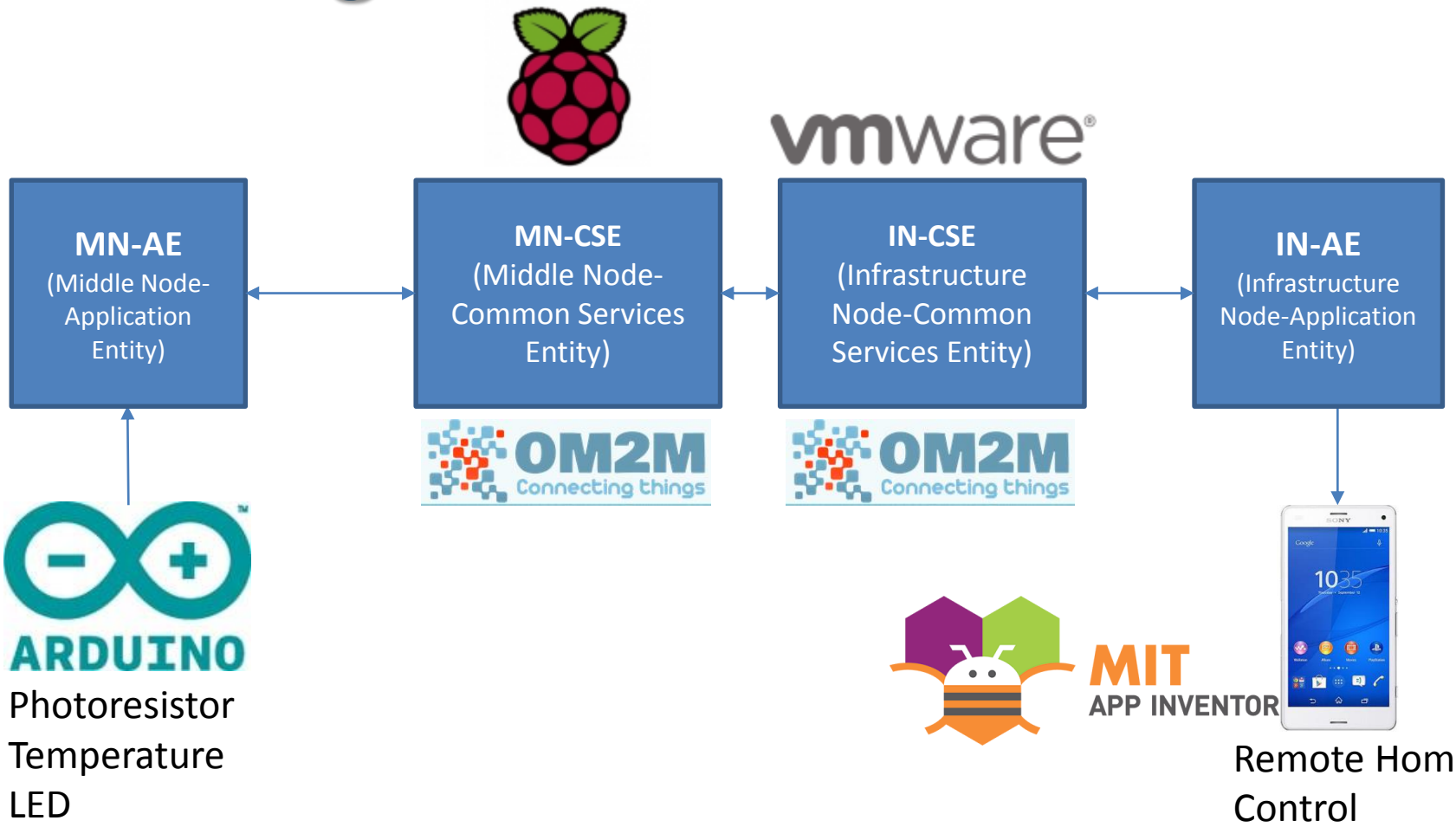
國立交通大學資訊工程系
Department of Computer Science
National Chiao Tung University

December 7, 2018

Outline

- High Level Architecture.
- Creating IN-AE and DATA Container to Monitor Smart Home Sensors with Node-Red. *(Checkpoint 1)*
- Modifying the Smart Home MN-AE Notification Handler Webservice using Node-Red. *(Checkpoint 2)*
- Creating a Subscription and Notification Handler for DATA Changes of Monitor Smart Home IN-AE using Node-Red. *(Checkpoint 3)*
- Using Point of Access (PoA) to Control Smart Home Actuators using Node-Red and Wyliodrin. *(Checkpoint 4)*
- Overview of App Inventor 2 and Hello World Android Application. *(Checkpoint 5)*
- Creating an Android Application for Smart Home Remote Control App using App Inventor 2. *(Checkpoint 6)*

High Level Architecture



Photoresistor
Temperature
LED

Remote Home
Control

Attention!!

Start your virtual machine and connect your Raspberry Pi, Arduino, and circuit board as you did in Lab 4!

Start the following software in your VM, in this order:

IN-CSE

Wyliodrin

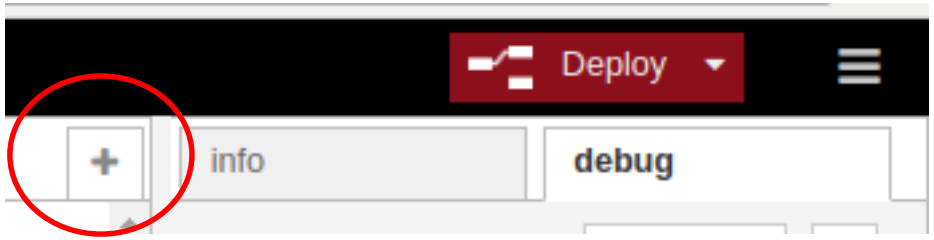
MN-CSE-PI

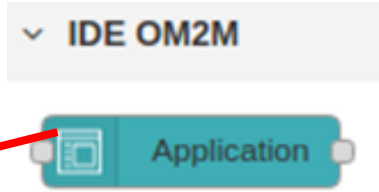
Node-Red

Trigger all the flows in Node-Red until reaching the state of Lab 4

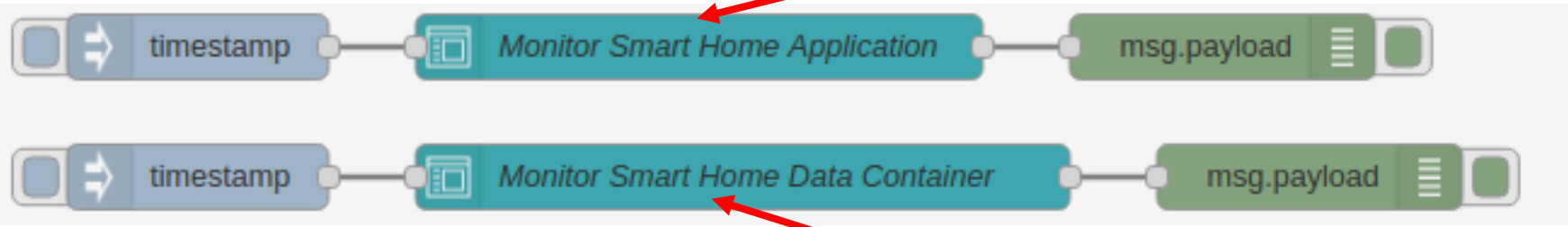
Create IN-AE to monitor smart home sensors

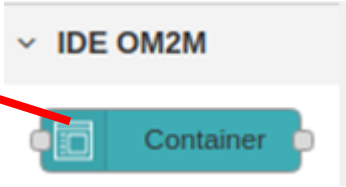
Add a new Flow to your workspace.





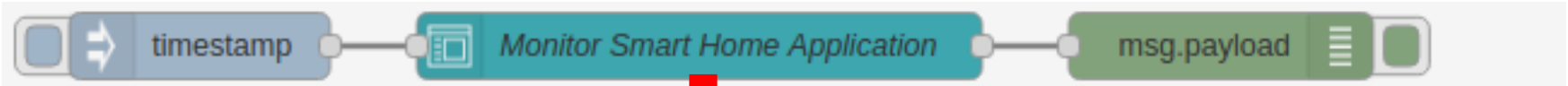
Create the following flows:

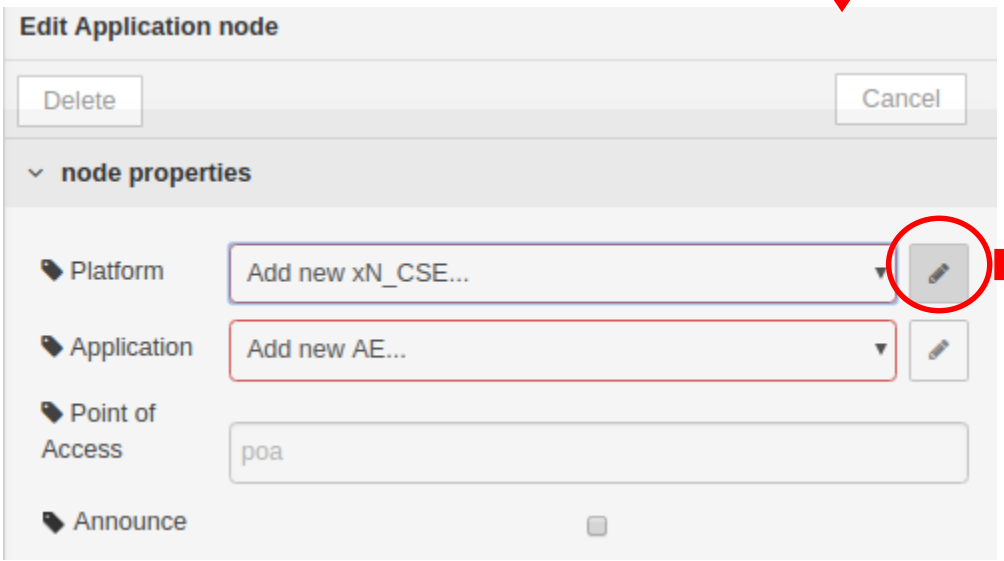




Create IN-AE to monitor smart home sensors

Add a new Platform to connect to your IN-CSE which is running in your VM.

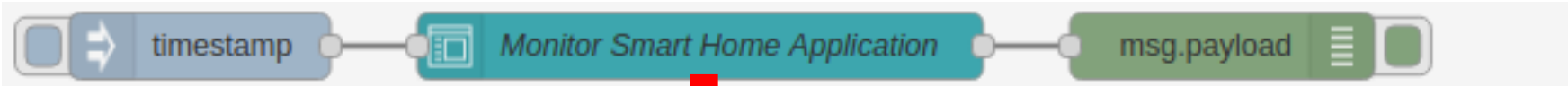


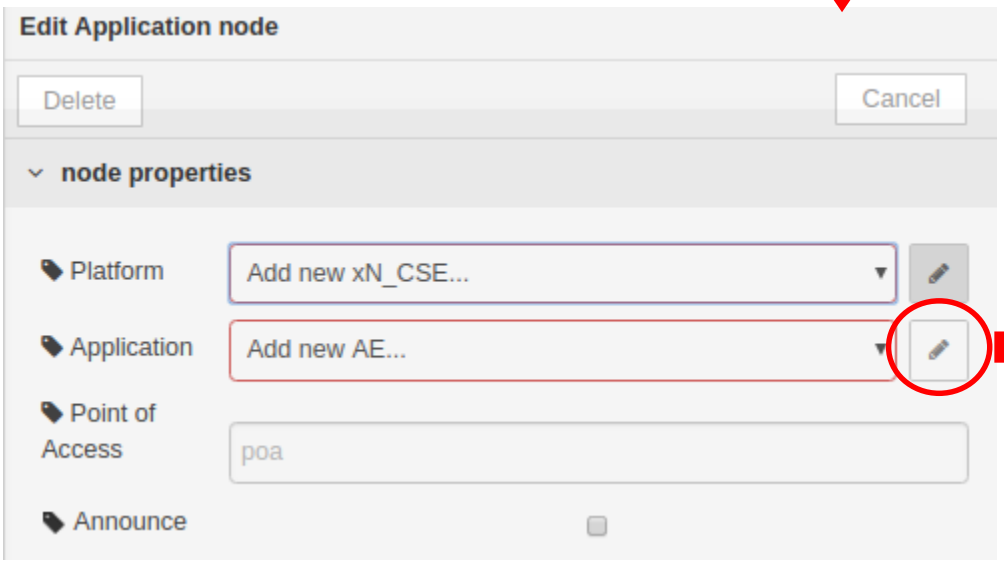


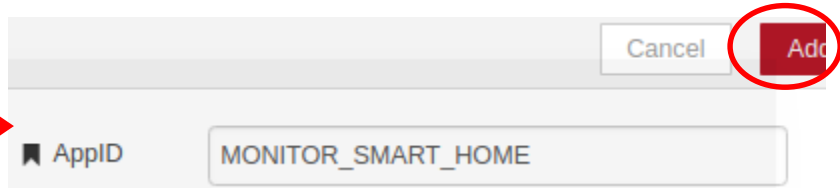
URLBase: http://**YOUR.VM.IP.ADRESS**:8080/~/in-cse/in-name
Username: admin
Password: admin

Create IN-AE to monitor smart home sensors

Set “MONITOR_SMART_HOME” as the AppID.

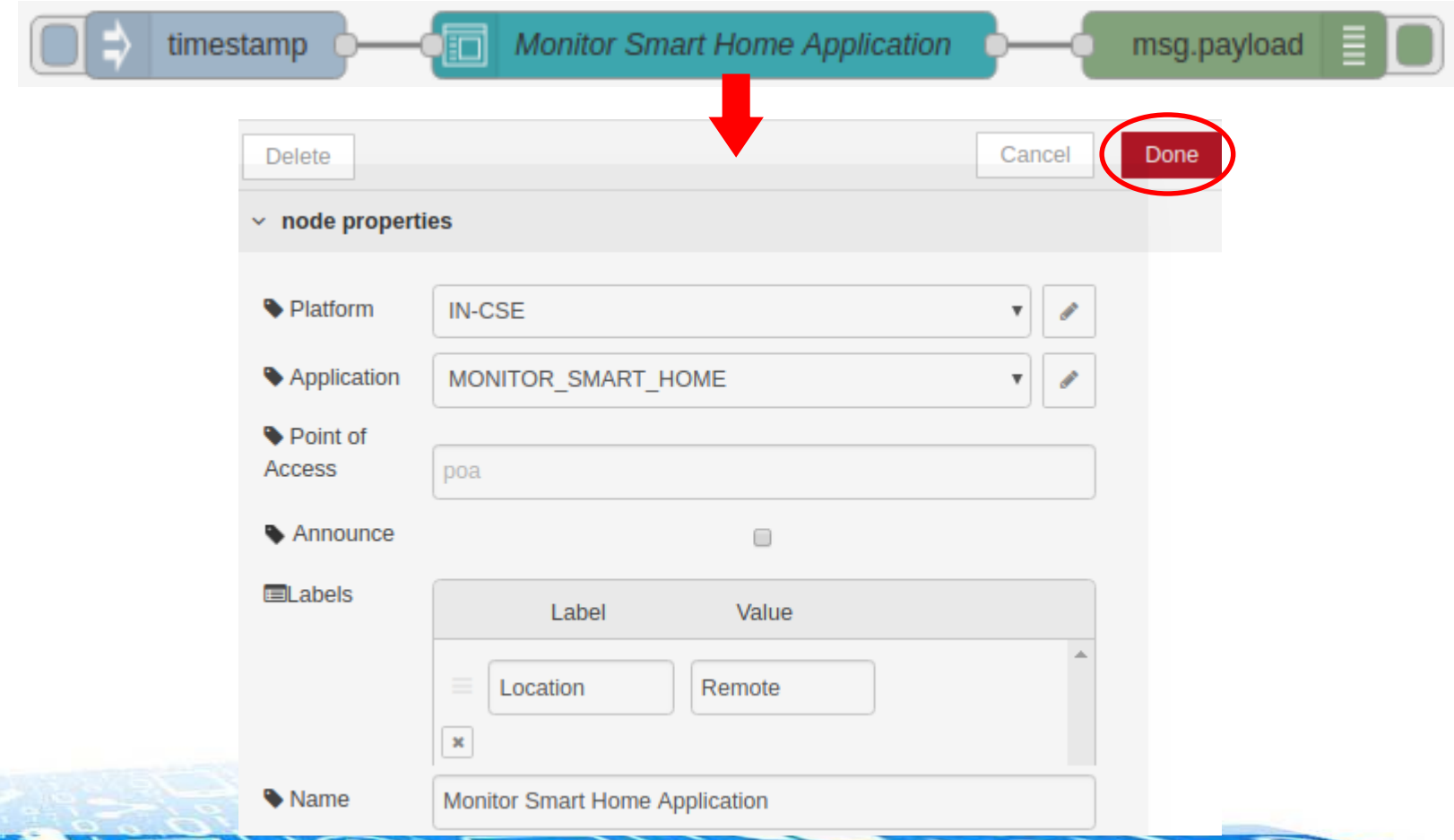






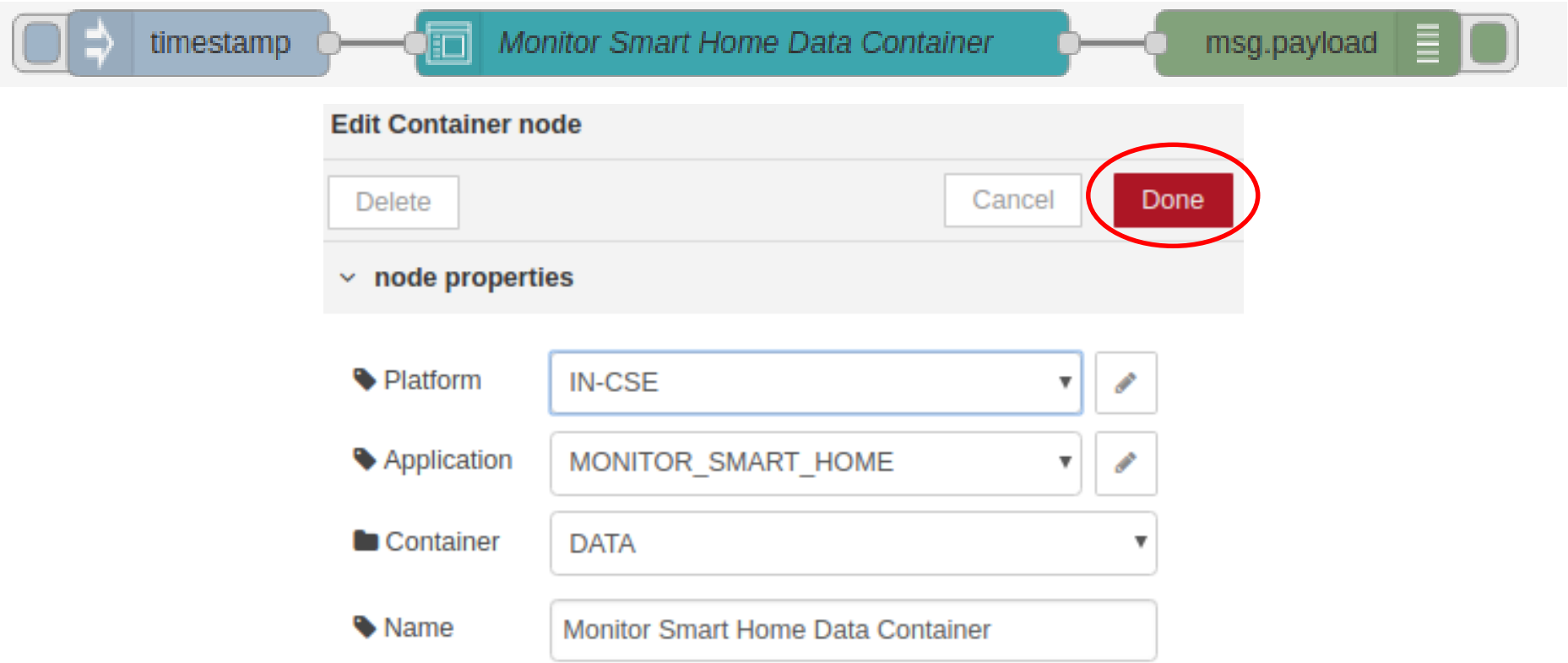
Create IN-AE to monitor smart home sensors

Complete the remaining attributes as shown in the picture below.



Create IN-AE to monitor smart home sensors

Create a DATA container.

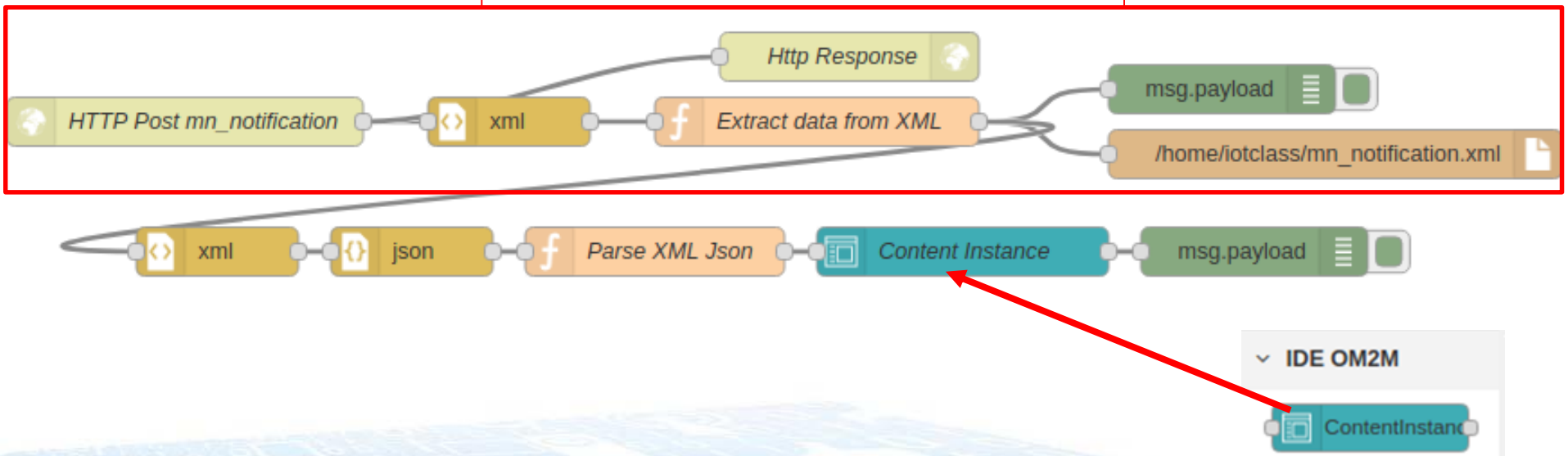


CHECKPOINT 1!

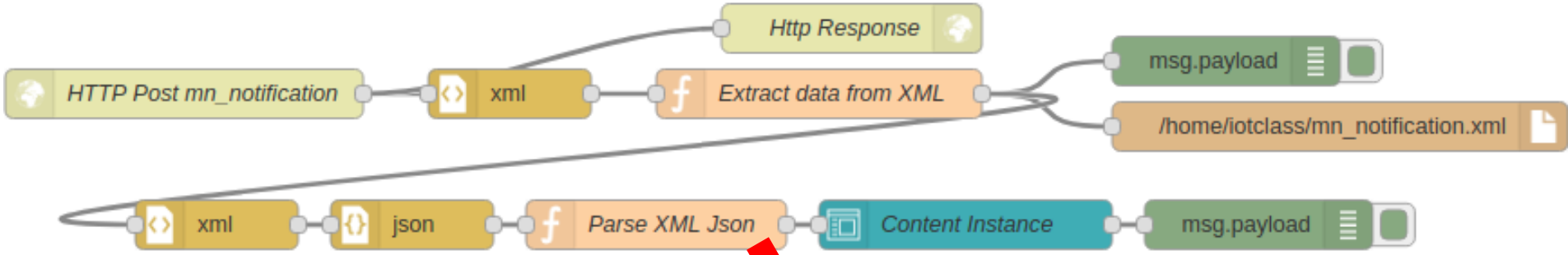
Modify HTTP Post mn_notification

We are going to combine parts of different flows we developed so far.
The purpose is to extract only the value of photoresistor from the notification message, and store this value in the DATA Container of the MONITOR_SMART_HOME application.

Do not need to be modified!

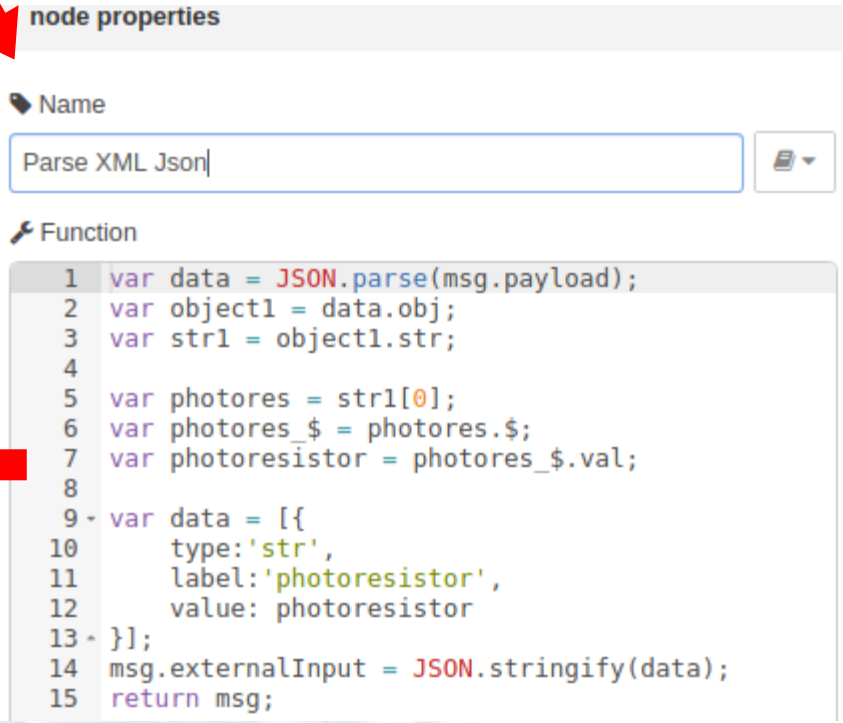


Modify HTTP Post mn_notification

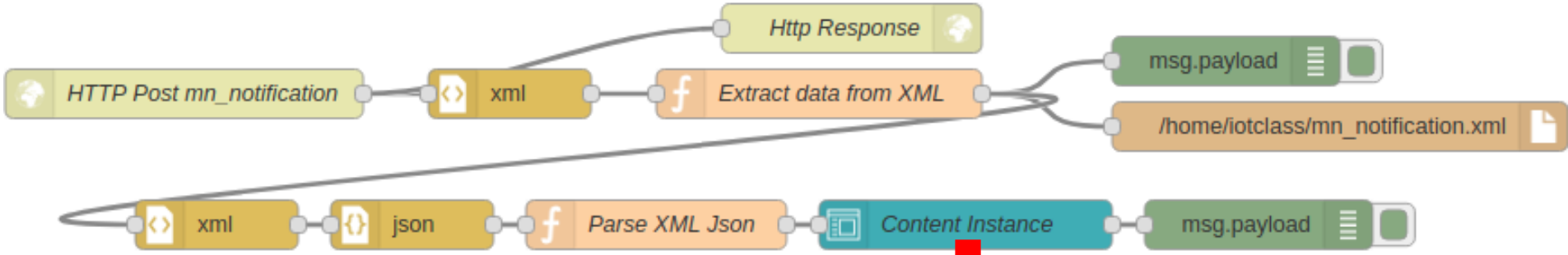


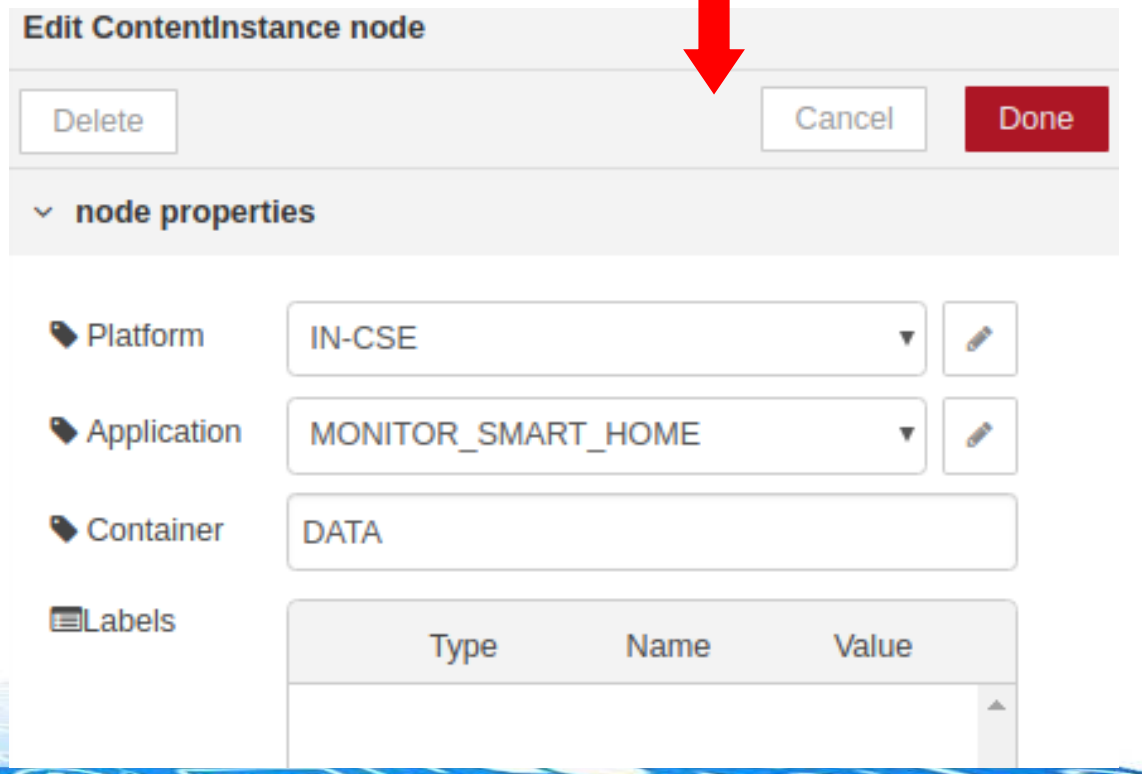
CODE (Copy & Paste in Node Red)

```
var data = JSON.parse(msg.payload);
var object1 = data.obj;
var str1 = object1.str;

var photores = str1[0];
var photores_$ = photores.$;
var photoresistor = photores_$.val;

var data = [{
    type:'str',
    label:'photoresistor',
    value: photoresistor
}];
msg.externalInput = JSON.stringify(data);
return msg;
```



Modify HTTP Post mn_notification





Modify HTTP Post mn_notification

Push the switch button in your breadboard. Verify the photoresistor value is stored into the DATA container of the MONITOR_SMART_HOME application.

http://192.168.1.51:8080/~/in-cse/cin-816072

- in-name
  - acp_admin
  - MONITOR_SMART_HOME
    - DATA
      - cin_816072674
  - mn-name-pi

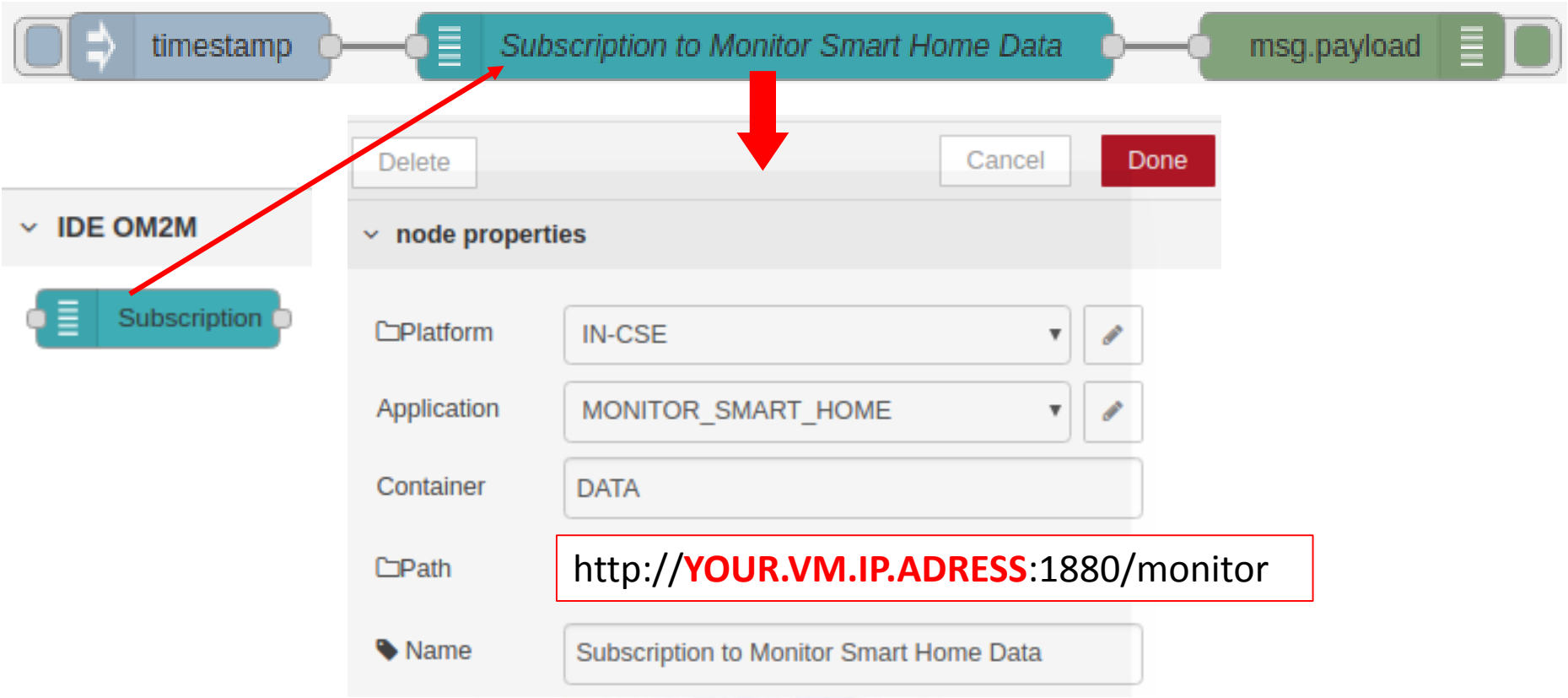
Attribute	Value
rn	cin_816072674
ty	4
ri	/in-cse/cin-816072674
pi	/in-cse/cnt-107923819
ct	20180705T192218
lt	20180705T192218
st	0
cnf	message
cs	93
con	

Attribute	Value
photoresistor	75.0

CHECKPOINT 2!

Create a subscription to Monitor Smart Home DATA container

Create a Subscription to DATA container according to the picture below.



Create a subscription to Monitor Smart Home DATA container

Trigger all the flows and verify the correct execution via the OM2M GUI.

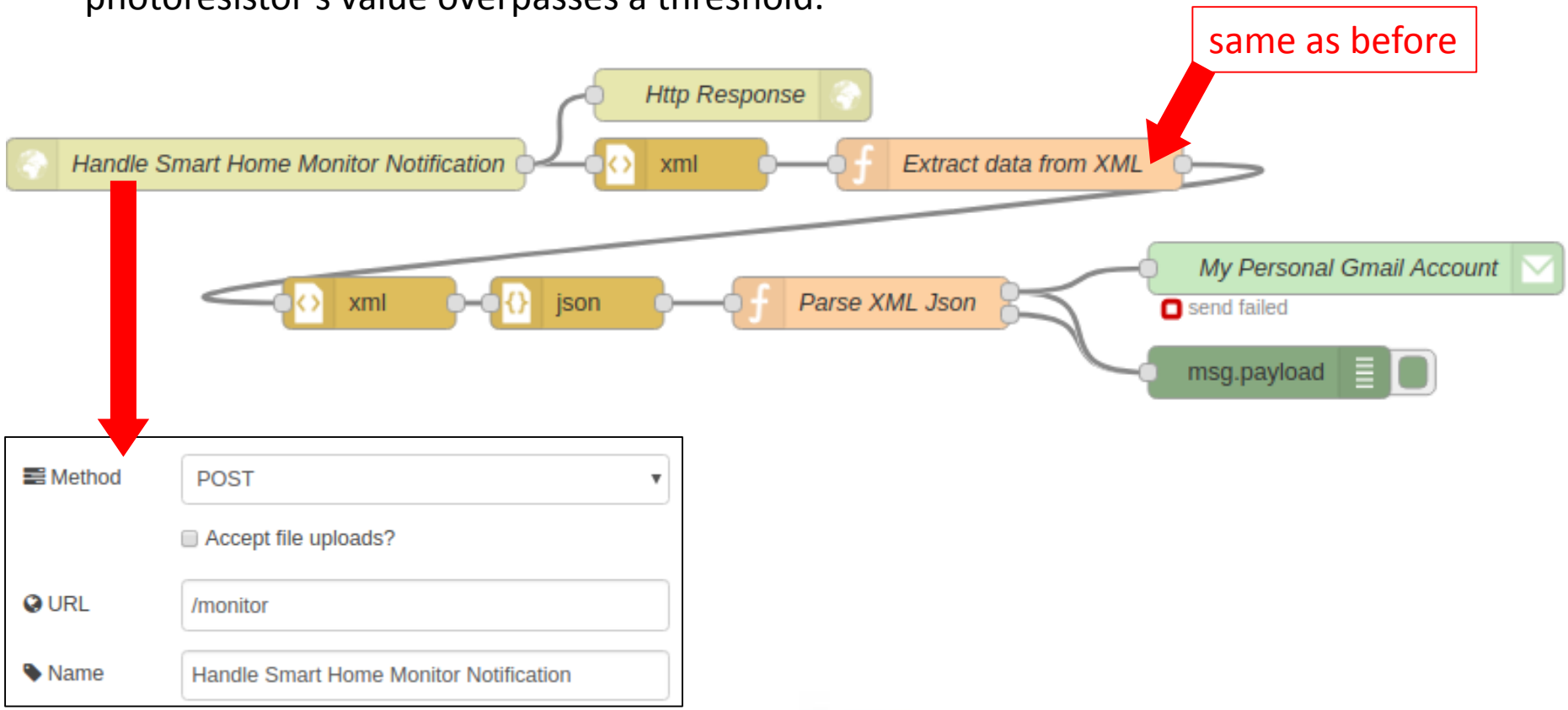
http://192.168.1.51:8080/~/in-cse/sub-916

- in-name
  - acp_admin
  - MONITOR_SMART_HOME
    - DATA
      - SUBSCRIPTION
  - mn-name-pi

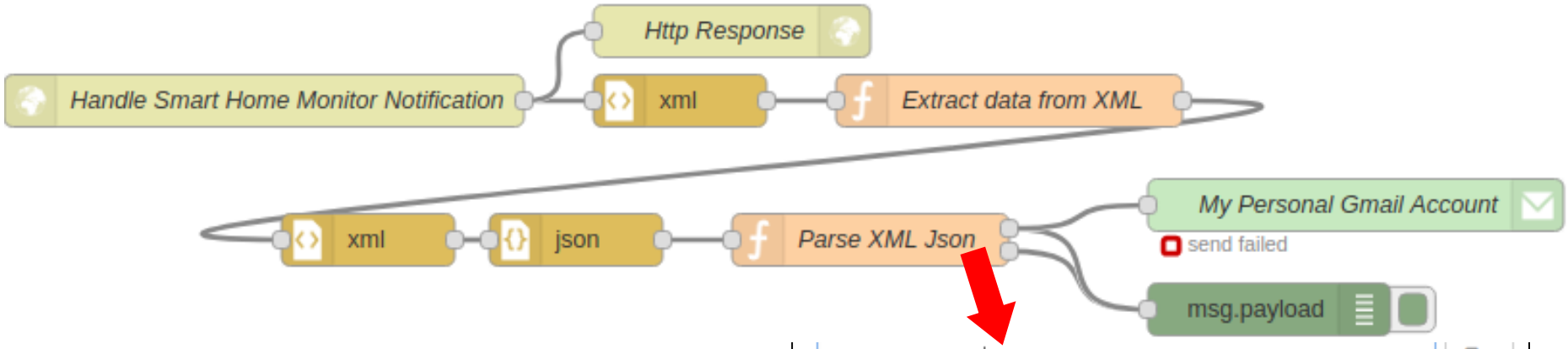
Attribute	Value
rn	SUBSCRIPTION
ty	23
ri	/in-cse/sub-916764550
pi	/in-cse/cnt-559137139
ct	20180705T165934
lt	20180705T165934
acpi	AccessControlPolicyIDs: /in-cse/acp-419143047
nu	• http://192.168.1.51:1880/monitor
nct	2

Create the “monitor” webservice

The purpose of this webservice is to send an notification email to a user if the photoresistor’s value overpasses a threshold.

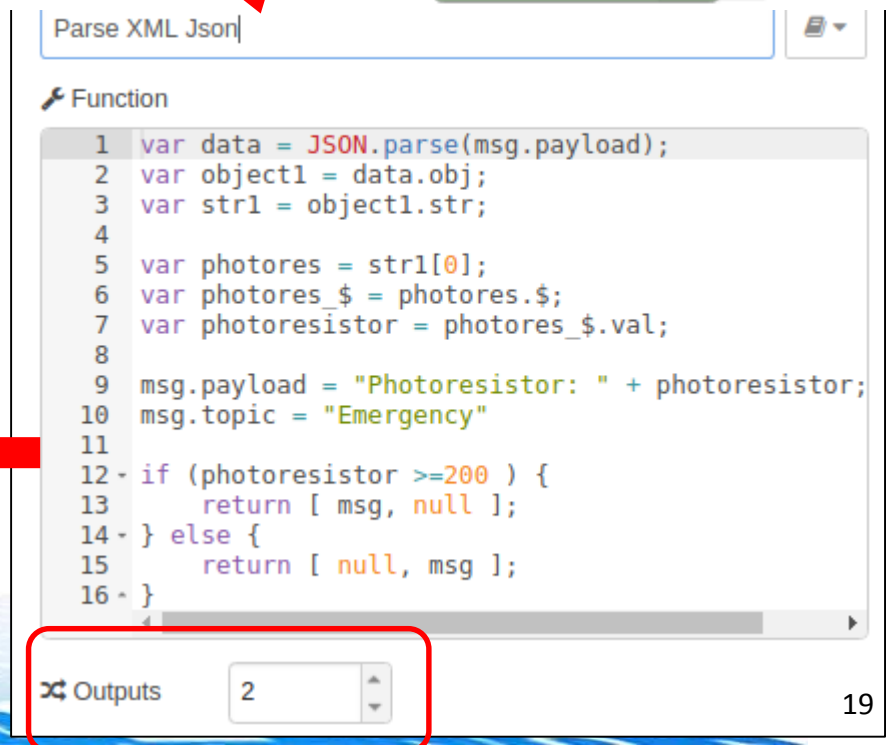


Create the “monitor” webservice

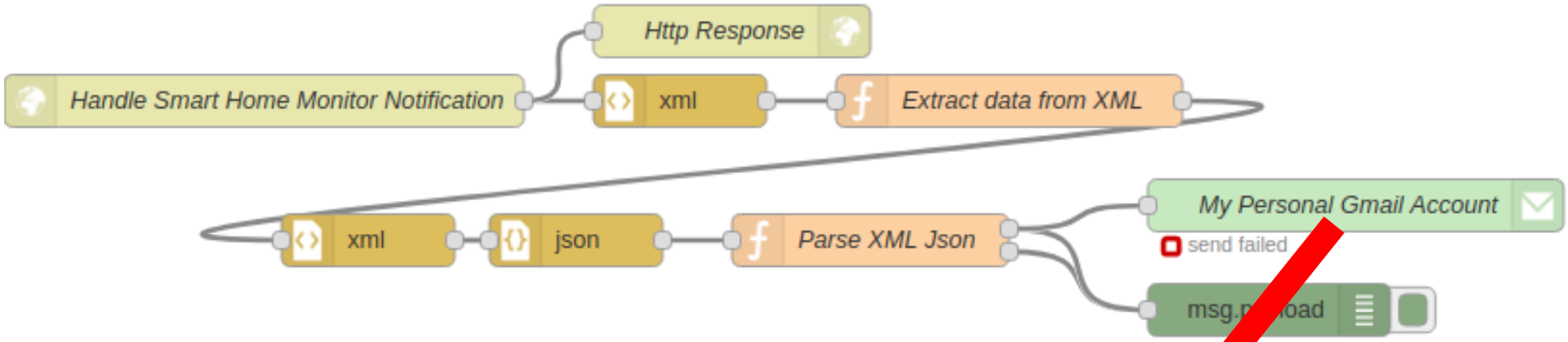


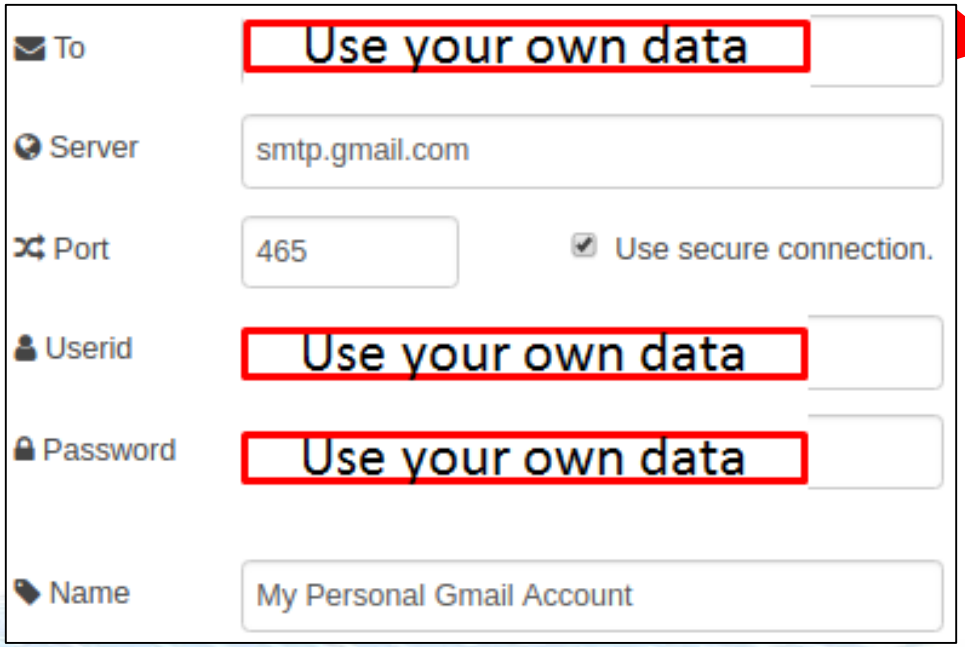
CODE (Copy & Paste in Node Red)

```
var data = JSON.parse(msg.payload);
var object1 = data.obj;
var str1 = object1.str;
var photores = str1[0];
var photores_$ = photores.$;
var photoresistor = photores_$.val;

msg.payload = "Photoresistor: " +
photoresistor;
msg.topic = "Emergency"

if (photoresistor >=200 ) {
    return [ msg, null ];
} else {
    return [ null, msg ];
}
```



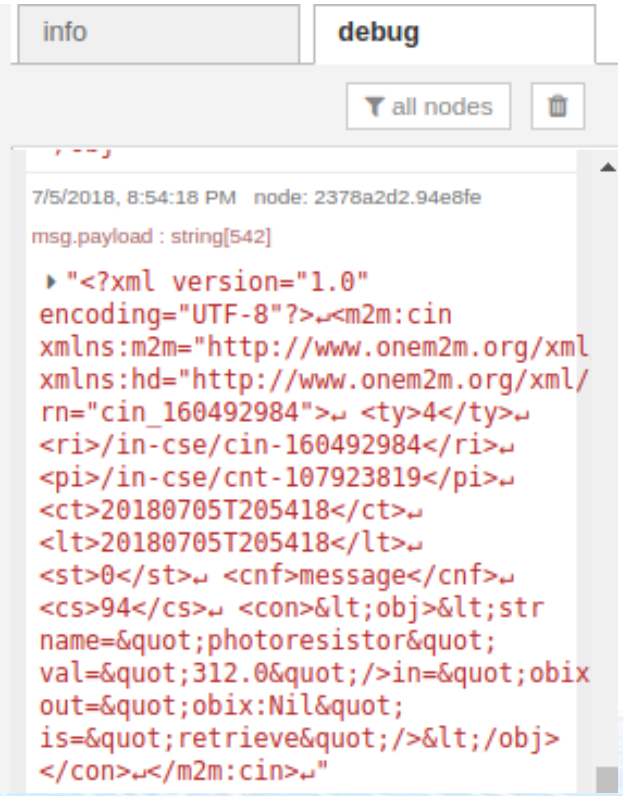
Create the “monitor” webservice

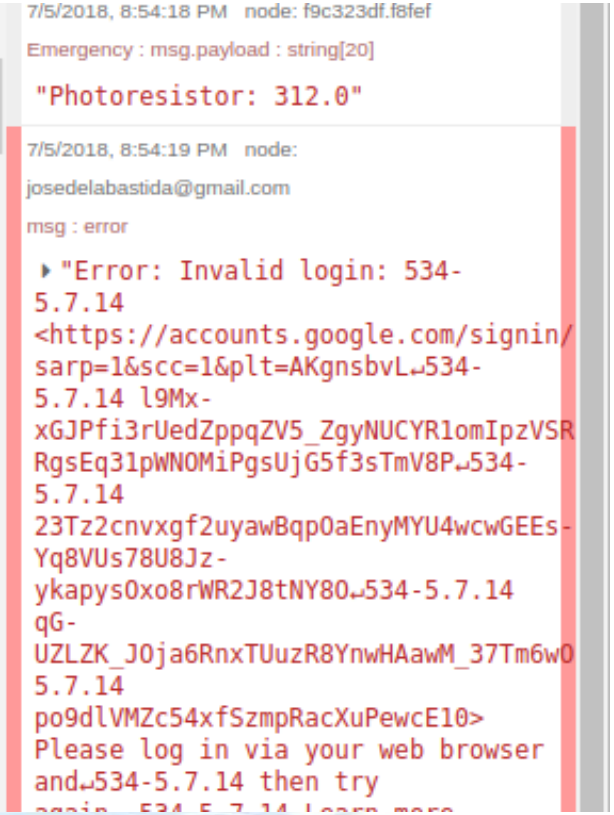




Try the whole system

The email notification may fail due to Google's security settings, somehow you can see the debug window of Node-Red to make sure the email was up to be sent.





CHECKPOINT 3!

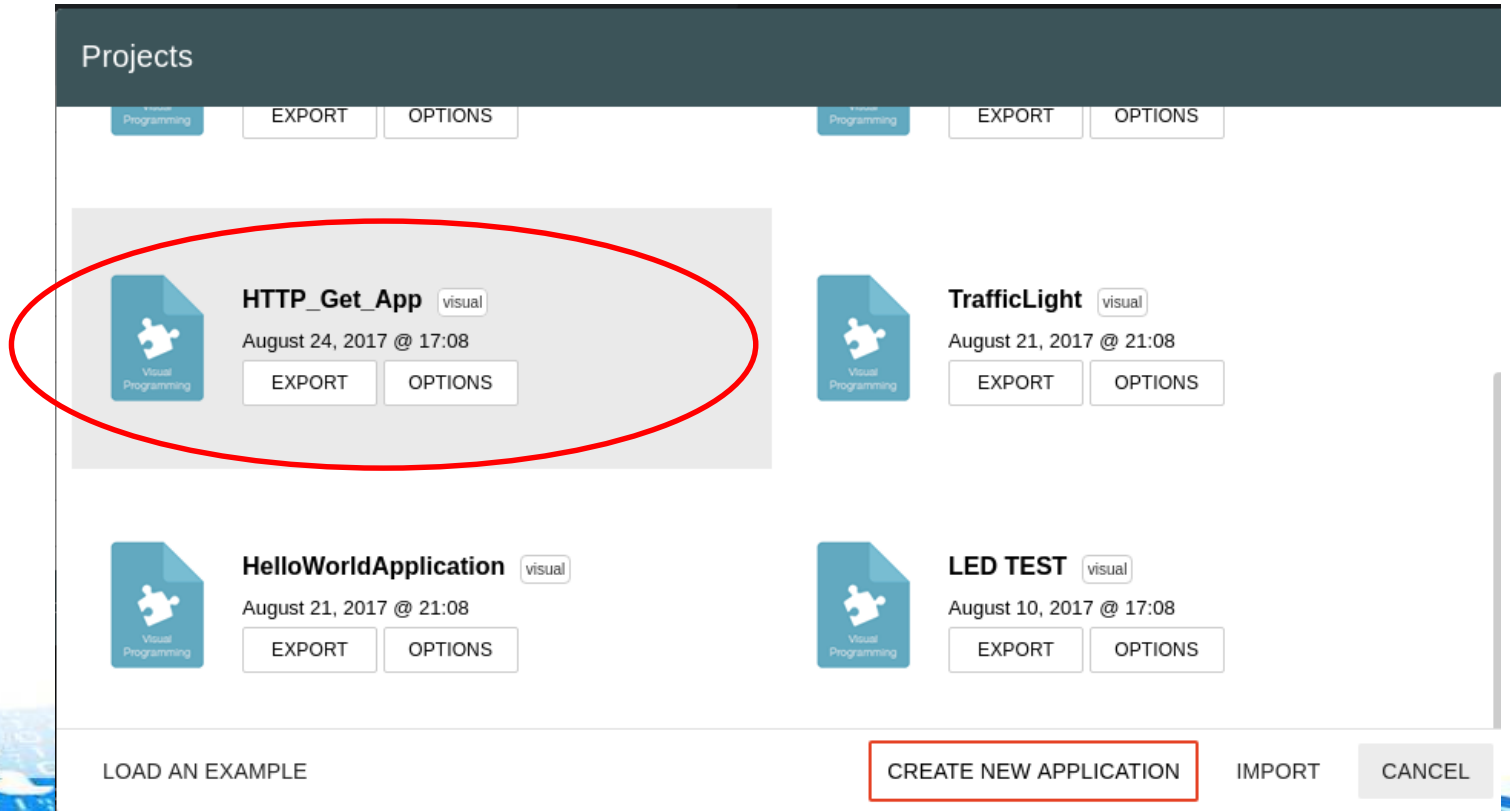
Attention!!

After demo checkpoint 3 to TA, please stop the current Wyliodrin App, and open the application “HTTP_Get_App” you created on Lab 2.

Open HTTP_Get_App

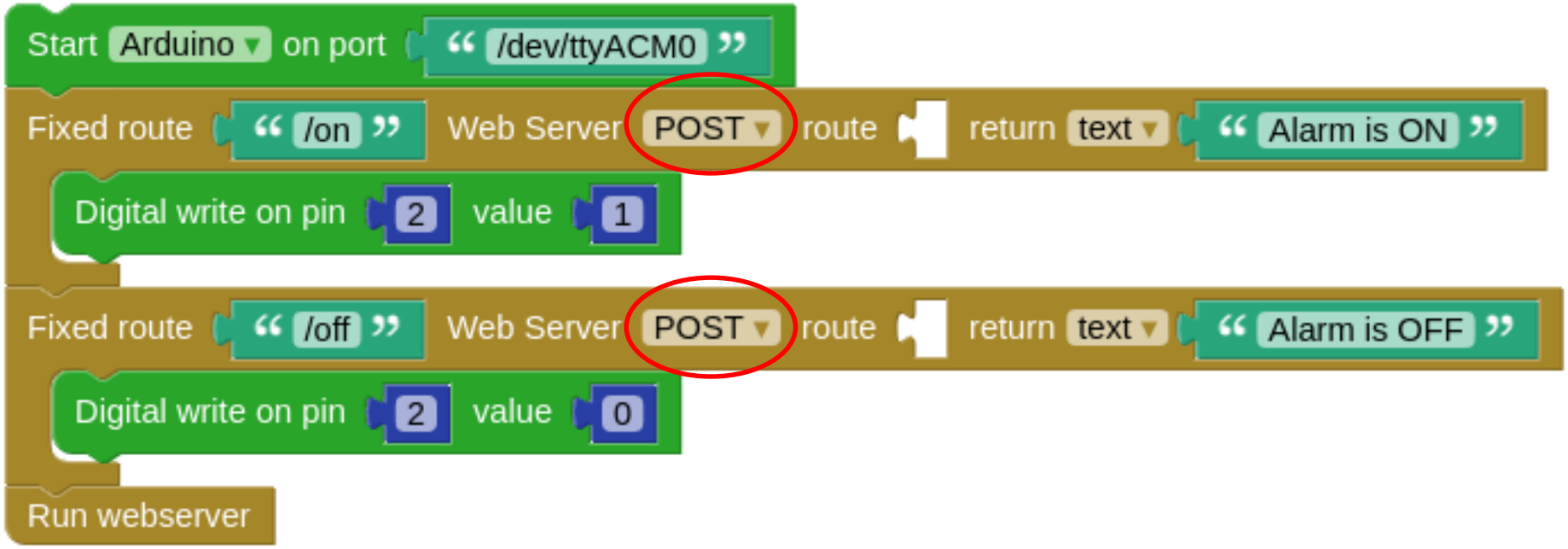
Stop the current app.

Open HTTP_Get_App



Modify and Run HTTP_Get_App

Change, Web Server from GET to POST.

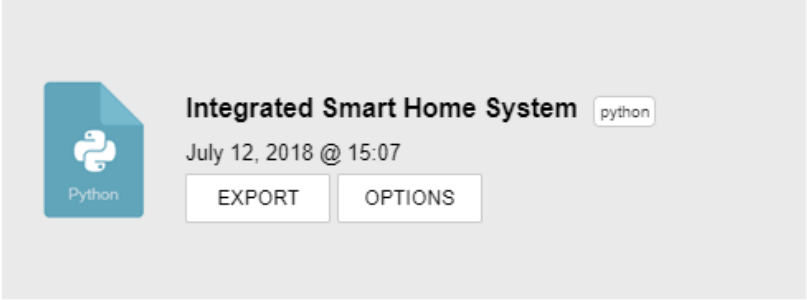


Run the application

Integrated System [Optional] (1)

To integrate the smart home and monitoring applications, we have written a single python script that achieves these tasks using threading. Threading is not provided in visual programming Wyliodrin.

Import and load the given python project “Integrated Smart Home System” from Wyliodrin.



Attention! : This task is optional and can be skipped. However it can substitute previous slide application “Modify and Run HTTP_Get_App”

Integrated System [Optional] (2)

- Change the ip_address in the url written inside **loopButton()** function in line 77 into your VM's ip_address

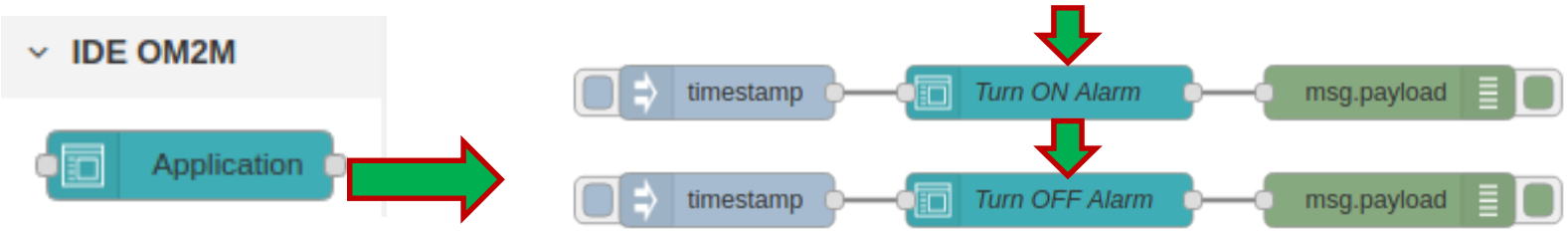
```
75 def loopButton():
76   if (digitalRead (15)) == 0:
77     http_response = requests.post('http://192.168.1.51:1880/postSmartHomeData',
78     print((http_response.text))
79   Timer(0.15, loopButton).start()
```

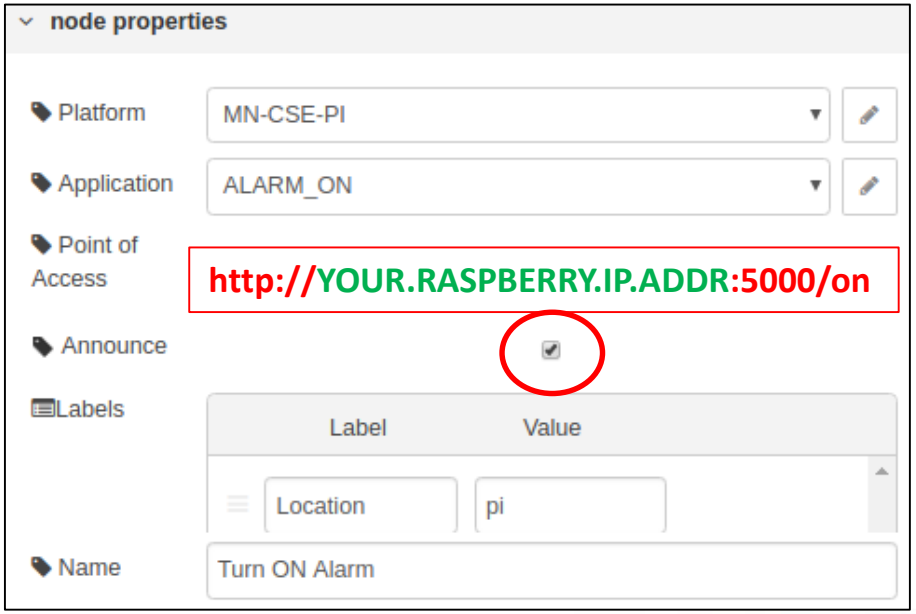
- Run the application

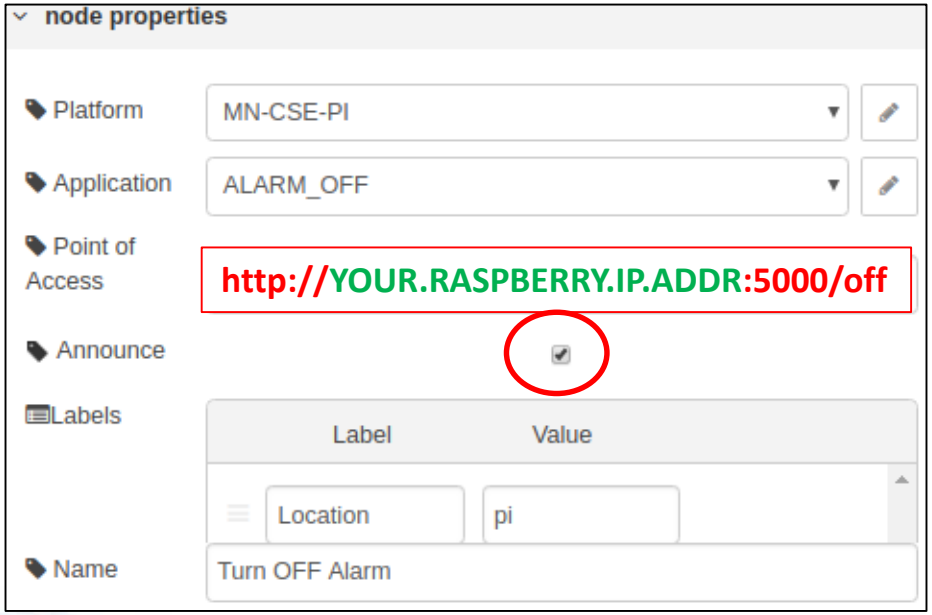
Attention! : This task is optional and can be skipped. However it can substitute previous slide application "Modify and Run HTTP_Get_App"

Create Two Applications in Node-Red

In the MN-CSE flow, create two additional MN-AE. Use the IDE-OM2M **Application** node:







Please deploy and trigger your applications.

Verify the two new applications are created in OM2M GUI

OM2M CSE Resource Tree

http://localhost:8080/~/mn-cse-pi/CAE917367037

- mn-name-pi
  - acp_admin
  - SMART_HOME
  - ALARM_ON
  - ALARM_OFF
  - in-name

Use Postman to test PoA (1)

Please verify that your LED is ON

POST http://**YOUR.VM.IP.ADDRESS**:8080/~/mn-cse-pi/mn-name-pi/ALARM_ON Send

Authorization | Headers | Body | Pre-request Script | Tests

	Key	Value	Description
☑	X-M2M-Origin	admin:admin	
☑	Content-Type	application/json	
☐			

Body | Cookies | Headers | Test Results — Status: 200 OK Time:

Pretty | Raw | Preview | HTML

```
1 Alarm is ON
2
```

Use Postman to test PoA (2)

Please verify that your LED is OFF

POST http://**YOUR.VM.IP.ADDRESS**:8080/~/mn-cse-pi/mn-name-pi/ALARM_OFF Send

Authorization | Headers | Body | Pre-request Script | Tests

	Key	Value	Description
☑	X-M2M-Origin	admin:admin	
☑	Content-Type	application/json	
☐			

Body | Cookies | Headers | Test Results — Status: 200 OK — Time

Pretty | Raw | Preview | HTML

```
1  Alarm is OFF
2
```

CHECKPOINT 4!

CREATING A REMOTE CONTROL APP IN APP INVENTOR 2

App Inventor 2 Overview (1)

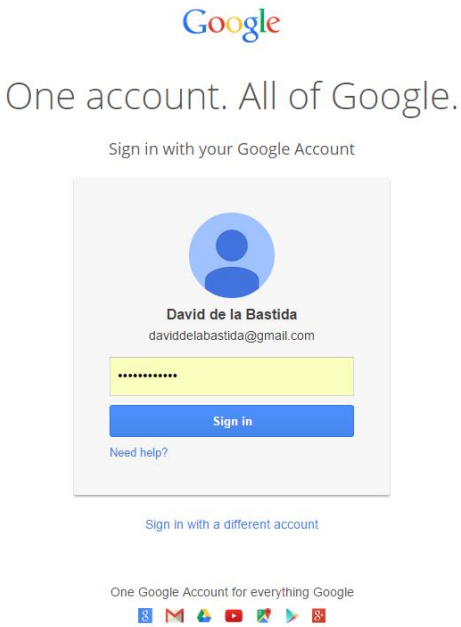
- Cloud-based tool for developing Android Apps.
- Originally provided by Google, and now maintained by the Massachusetts Institute of Technology (MIT).
- Apps can be built right in your web browser → Visual Programming.
- Url of the service: ai2.appinventor.mit.edu.
- Works with your google account.
- Focus on the Application not on Coding.

App Inventor 2 Overview (2)

- General Features:
  - A ***designer***, in which program's *components* are specified. This includes visible components, such as buttons and images, which are placed on a simulated screen, and non-visible components, such as sensors and web connections.
  - A ***blocks editor***, in which the program's logic is created.
  - A ***compiler*** based on the Kawa language framework.
  - ***Real-time deployment and debugging*** on a connected Android device.
  - ***Real-time deployment and debugging*** on a provided Emulator.
  - ***Deployment via .apk*** using QR code or direct download.

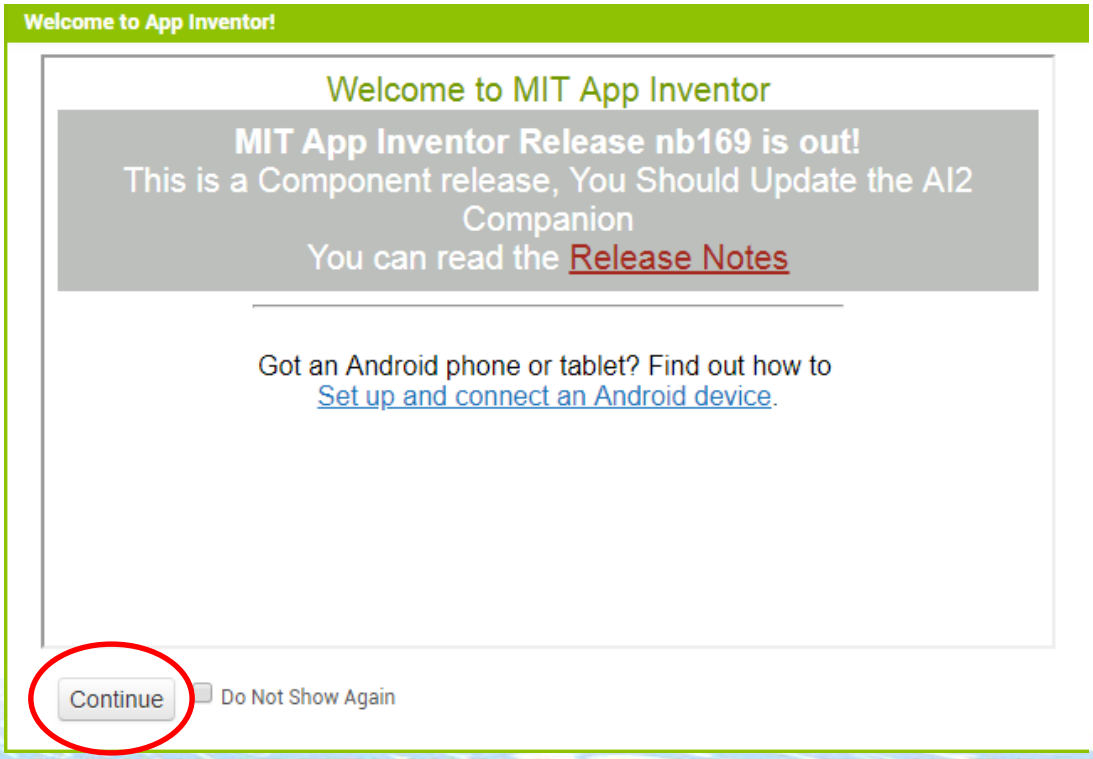
App Inventor 2 Login (1)

1. Type http://ai2.appinventor.mit.edu
2. Login using your google account.
3. Allow permission to MIT AppInventor Version 2 to access your google account (if required).



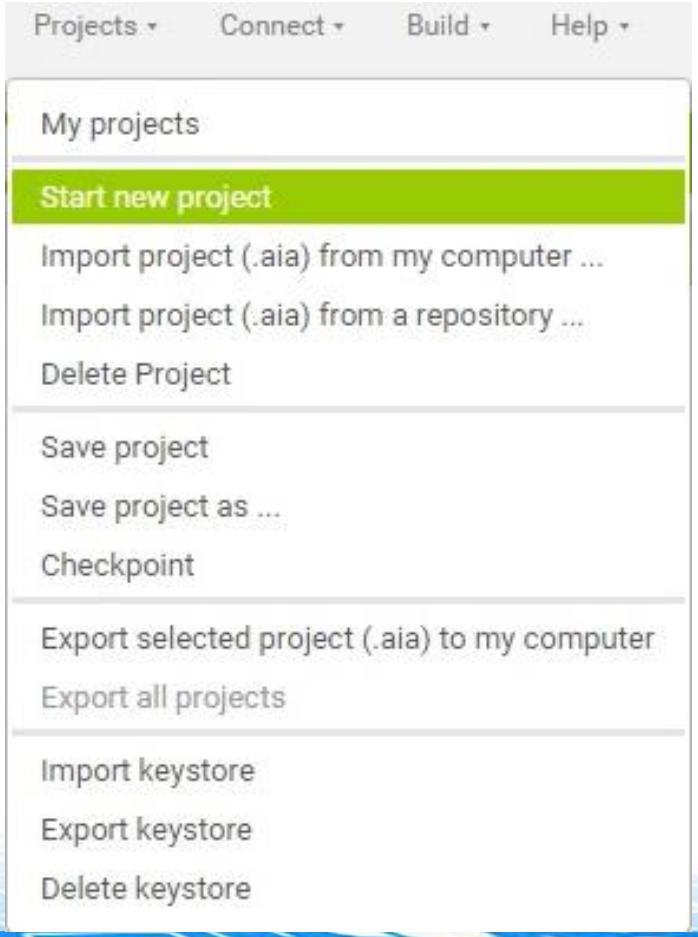
App Inventor 2 Login (2)

1. Type http://ai2.appinventor.mit.edu
2. Login using your google account.
3. Allow permission to MIT AppInventor Version 2 to access your google account (if required).
4. Click “continue”, uncheck “do not show again” (if required).



Hello World Application (1)

1. From menu “My projects” select “Start new project”.



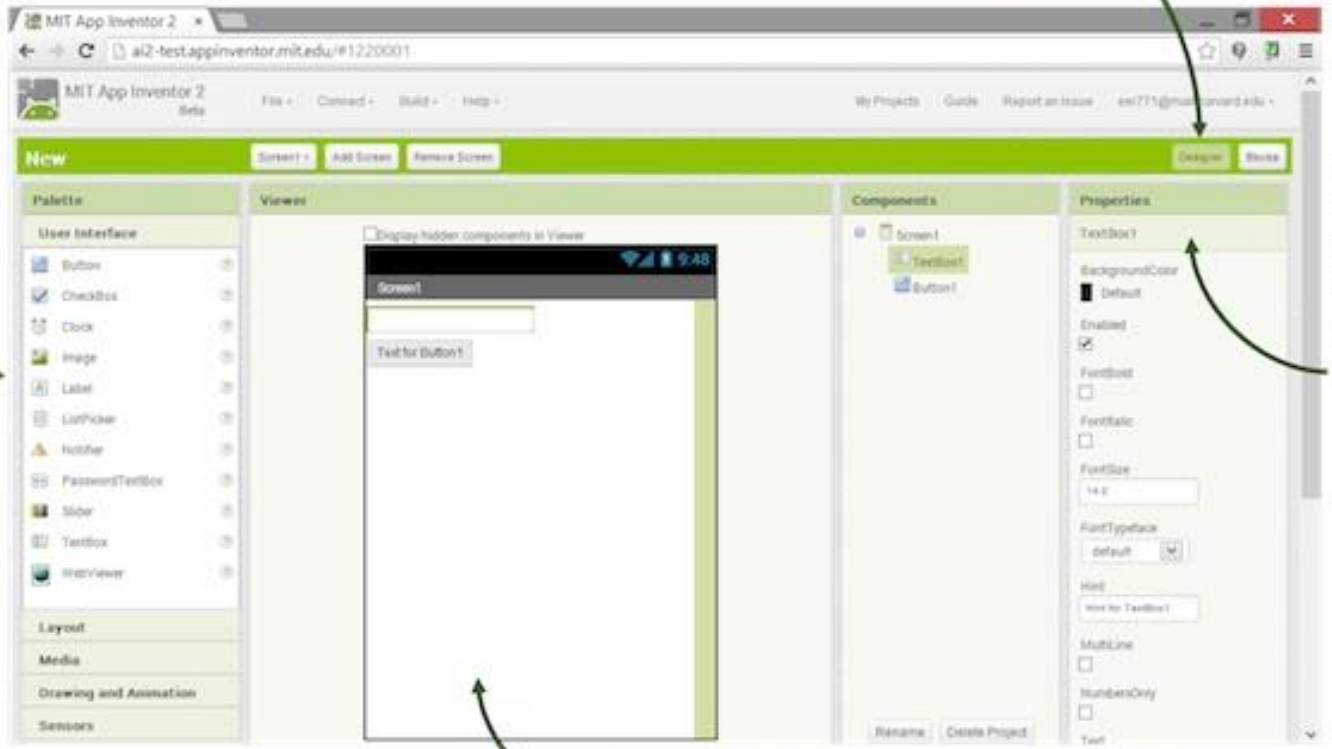
Hello World Application (2)

2. Provide a name to your project. (In this example we use “HelloWorldApp”).

App Inventor 2 GUI (1)

Palette: Find your components and drag them to the Viewer to add them to your app.

Designer Button: Click from any tab to go to the Designer tab.



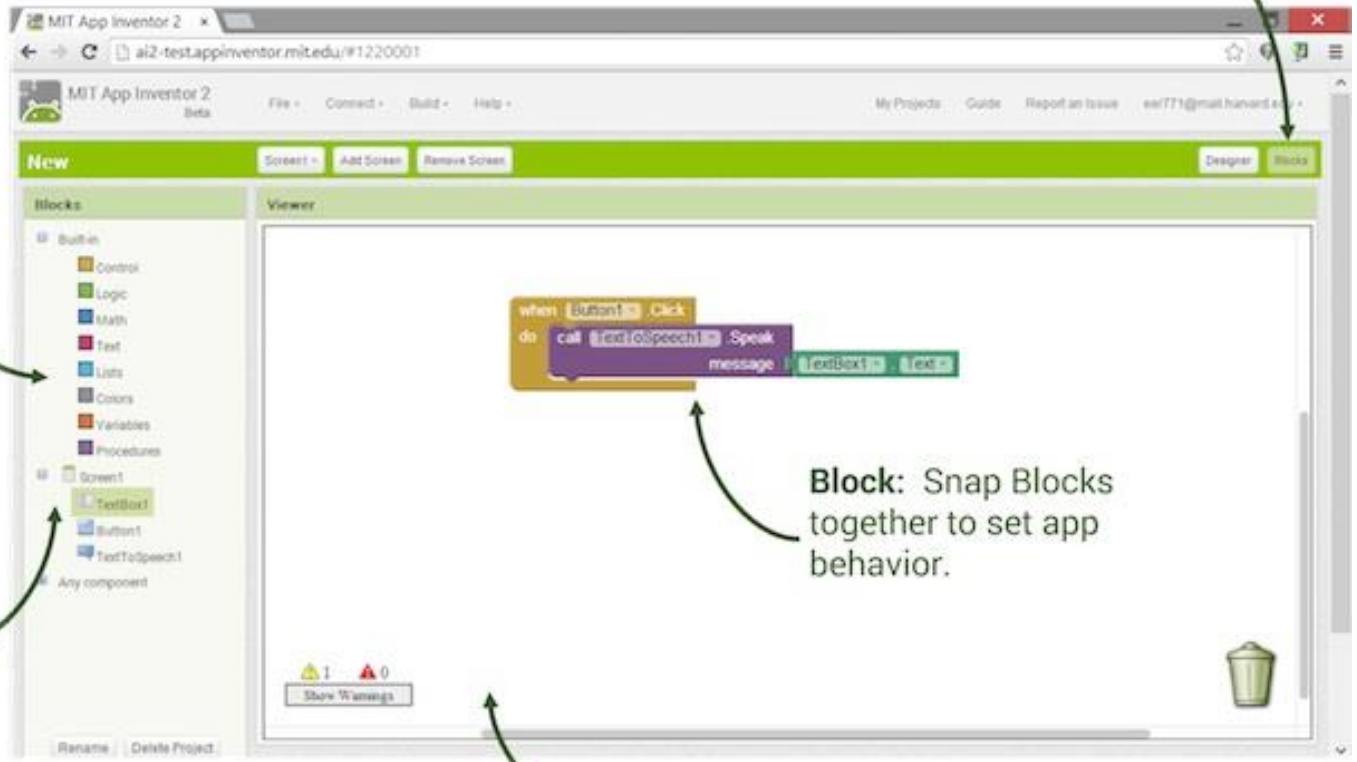
Properties: Select a Component in the Components List to change its properties (color, size, behavior) here.

Viewer: Drag components from the Palette to the Viewer to see what your app will look like.

App Inventor 2 GUI (2)

Built-In Drawers: Find Blocks for general behaviors you may want to add to your app and drag them to the Blocks Viewer.

Blocks Button: Click from any tab to go to the Blocks tab.



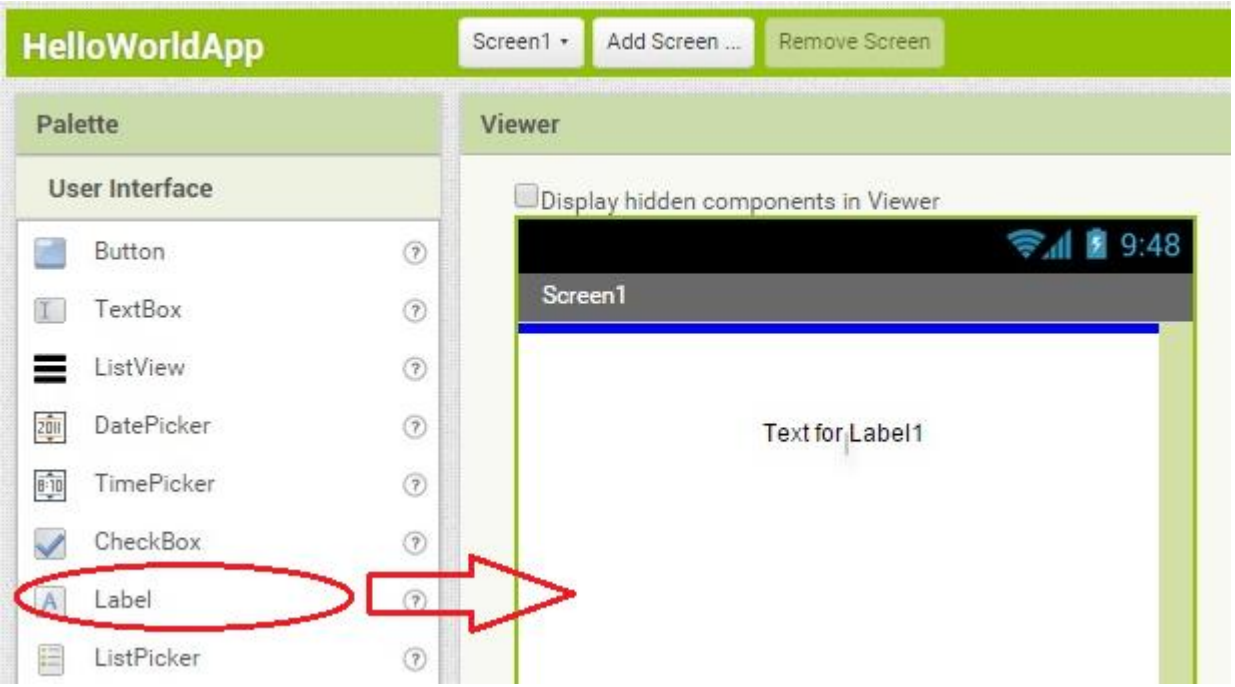
Component-Specific Drawers: Find Blocks for behaviors for specific Components and drag them to the Blocks Viewer.

Block: Snap Blocks together to set app behavior.

Viewer: Drag Blocks from the Drawers to the Blocks Viewer to build relationships and behavior.

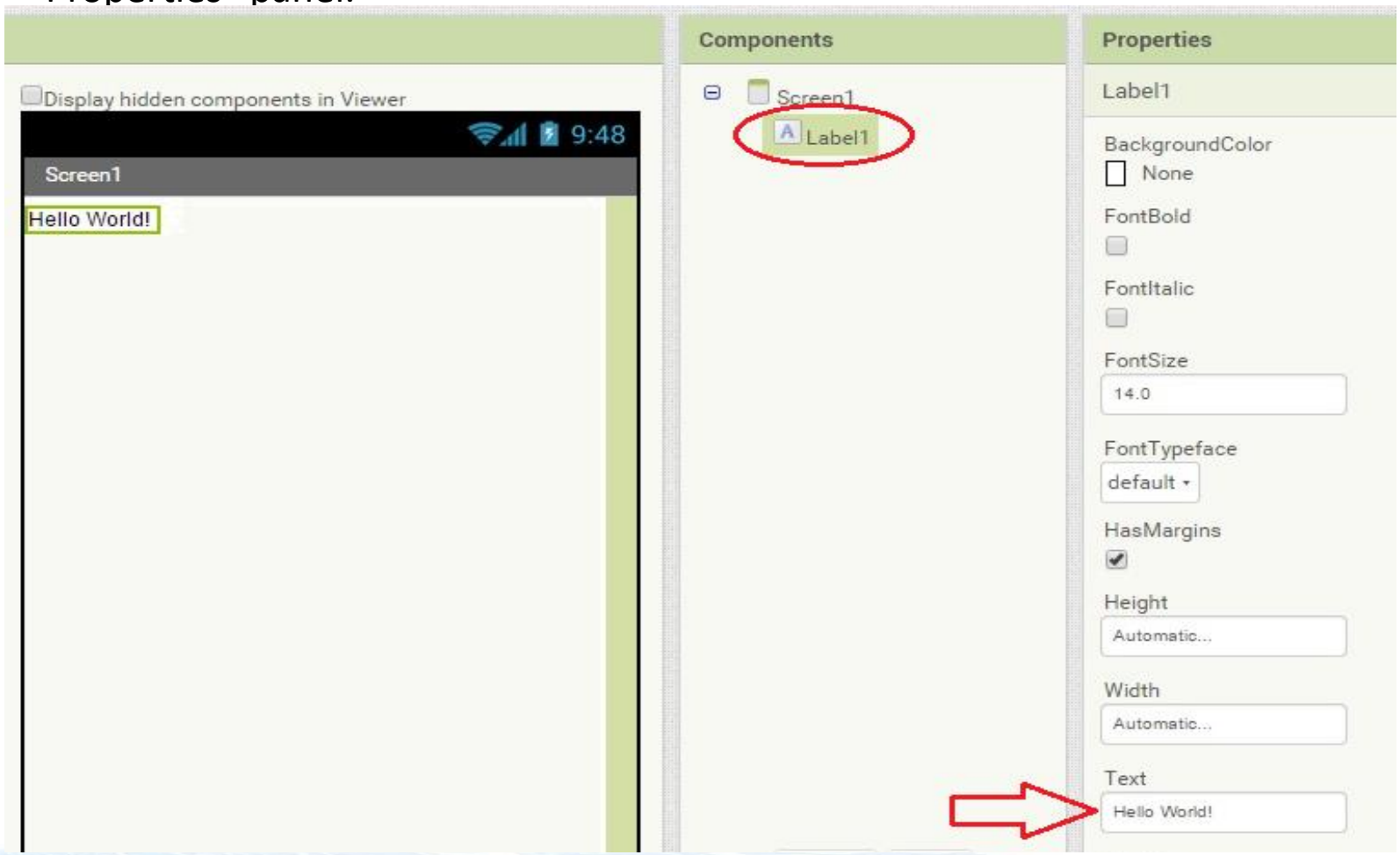
Hello World Application (3)

3. Drag and Drop a “Label” control from the “User Interface Panel” to the screen of your app.



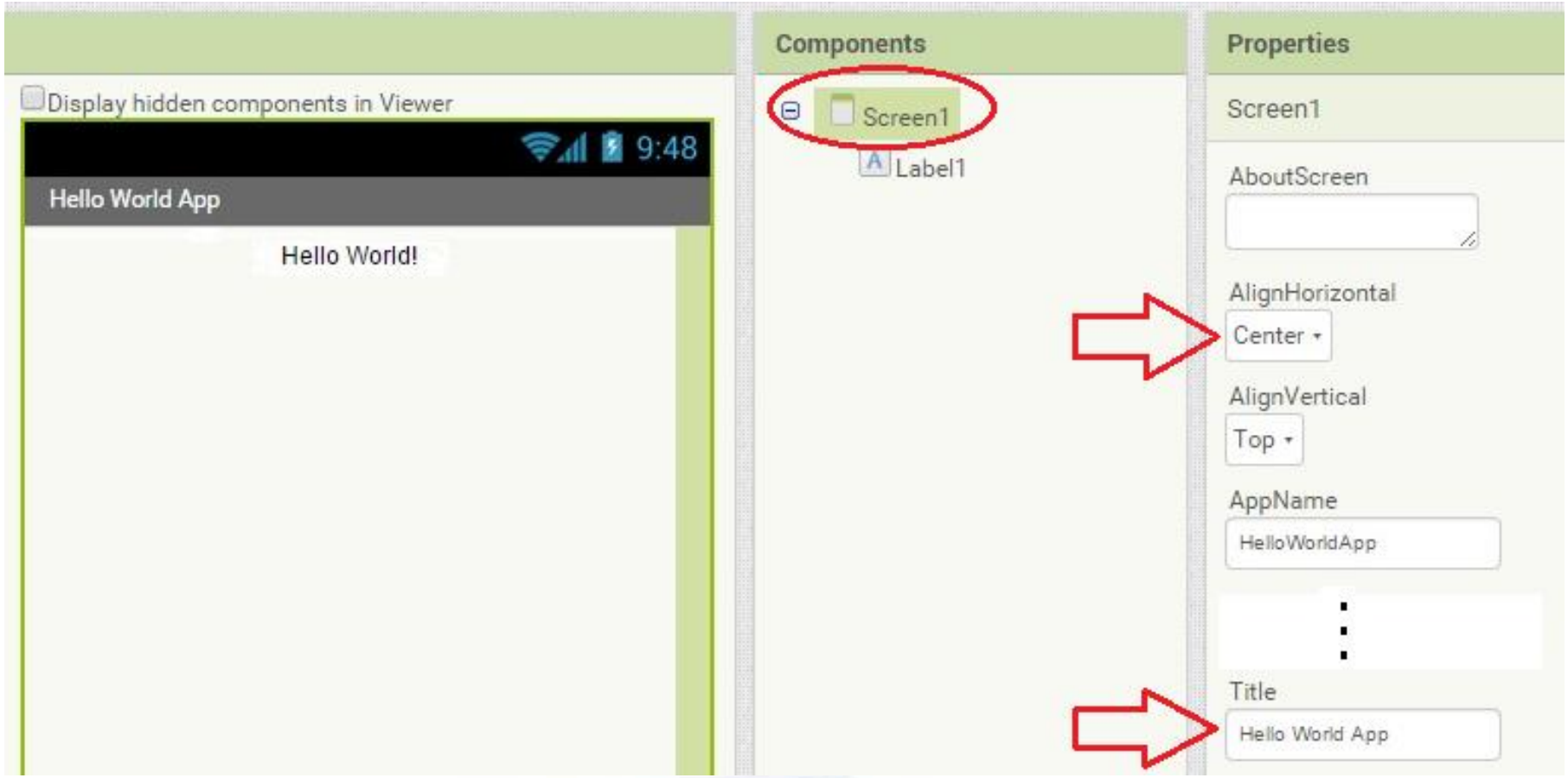
Hello World Application (4)

4. Change the text of the label component. (Click “Label1” in the “Components” panel, then type “Hello World!” in the “Text” field in the “Properties” panel.



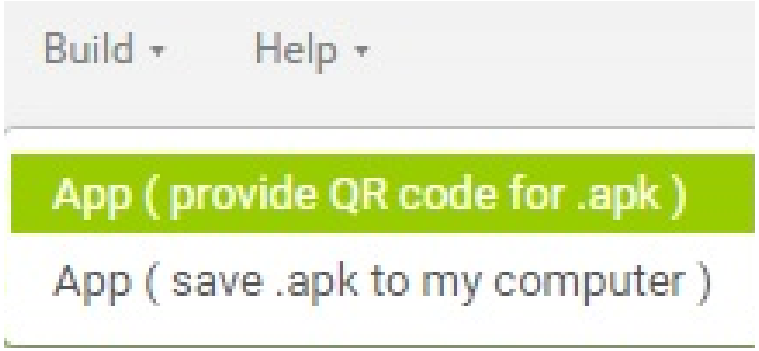
Hello World Application (5)

5. Change settings of main screen. (Click “Screen1” in the “Components” panel, then in “Properties” panel change “AlignHorizontal” field to “Center” and type “Hello World App” in the “Title” field.



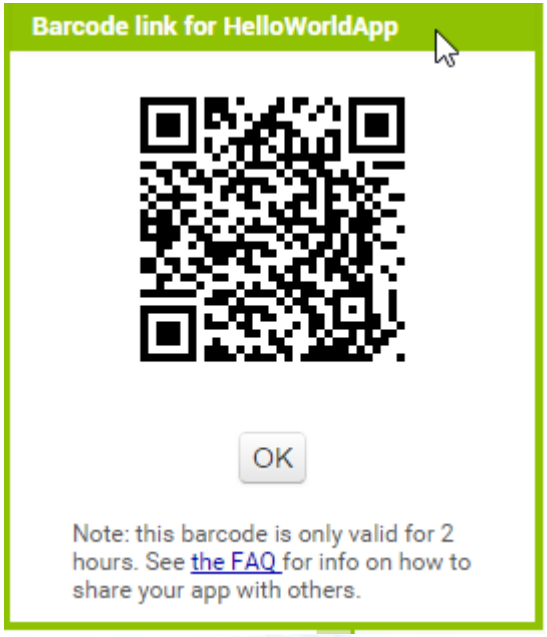
Hello World Application (6)

6. Create an .apk and install the application in your phone. (Select “App (provide QR code for .apk)” from Build menu.



Hello World Application (7)

7. Read the provided QR code using any QR code reader app in your phone. This action will download an .apk file into your phone. After download just install it and run it.



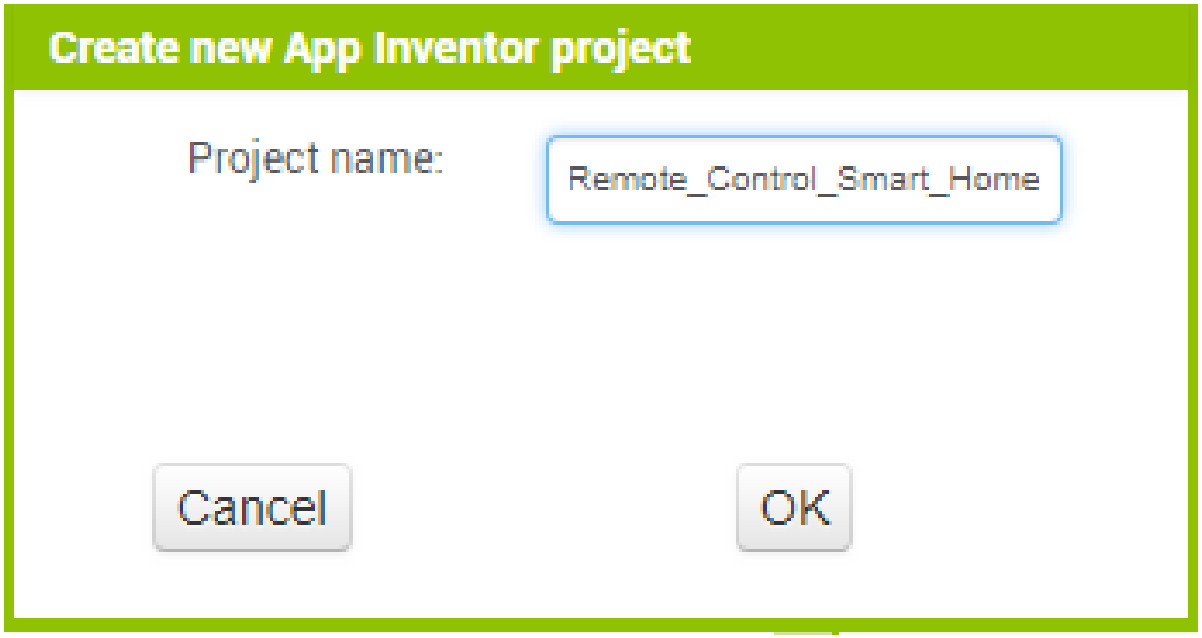
CHECKPOINT 5!

Remote Control Smart Home App

Create a new project

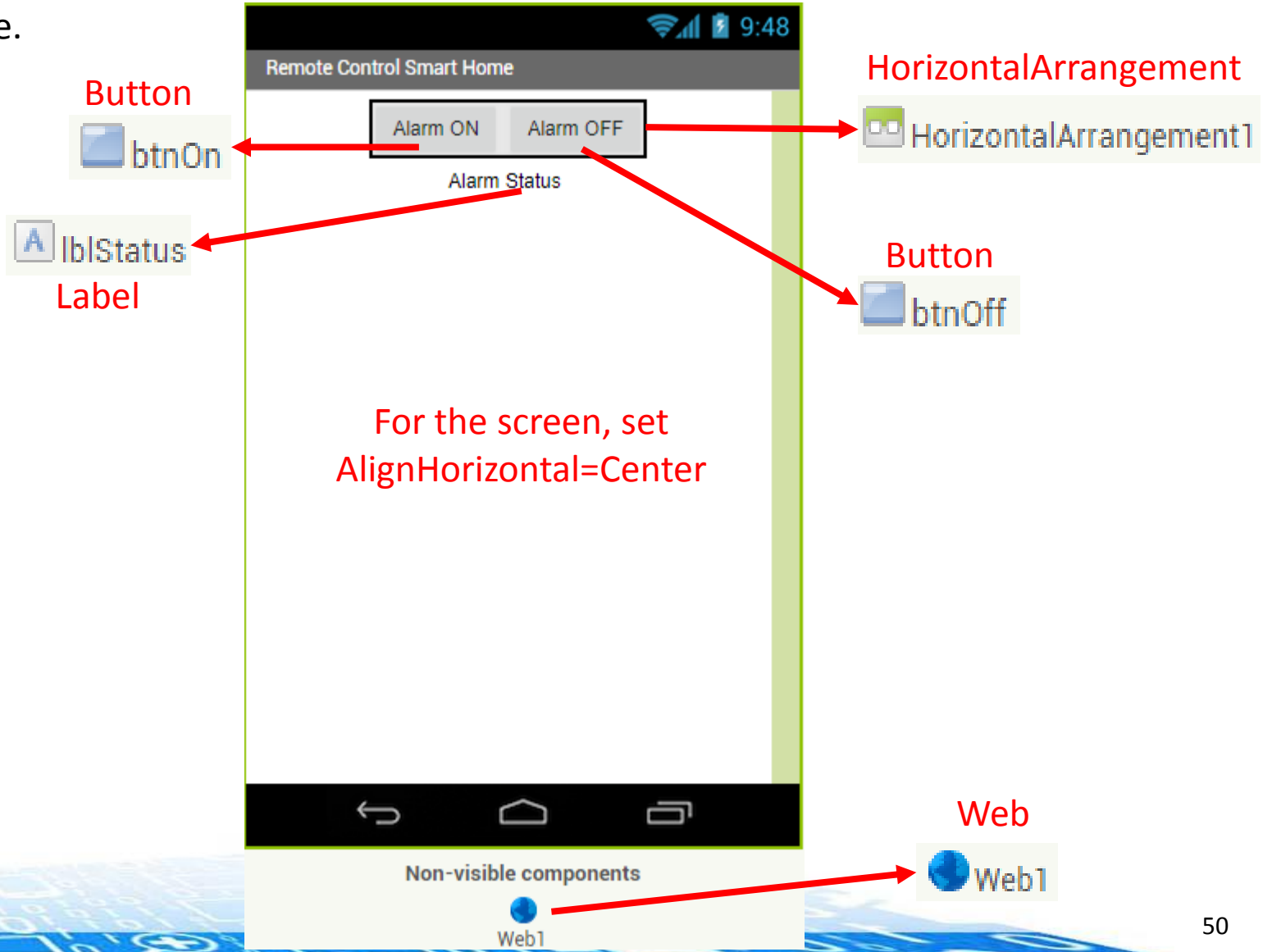
Remote Control Smart Home App

Create a new project called “Remote_Control_Smart_Home”.



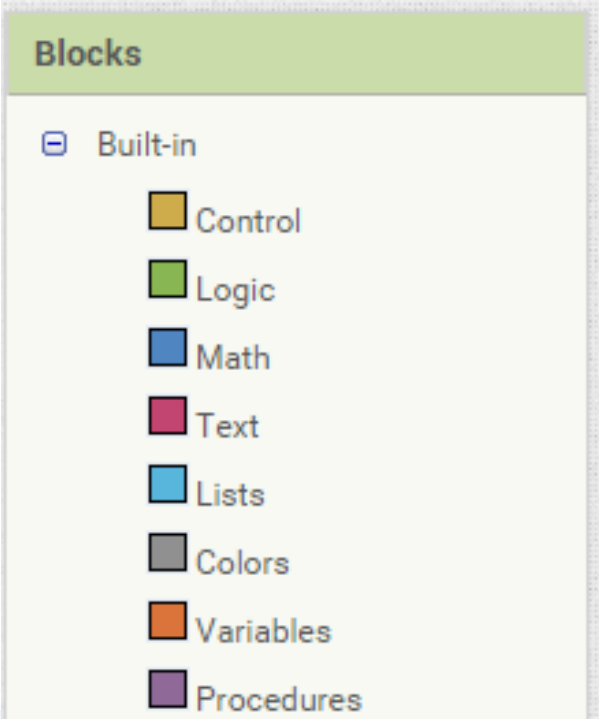
Remote Control Smart Home App

3. Draw the GUI components and change their properties as showed in the picture.



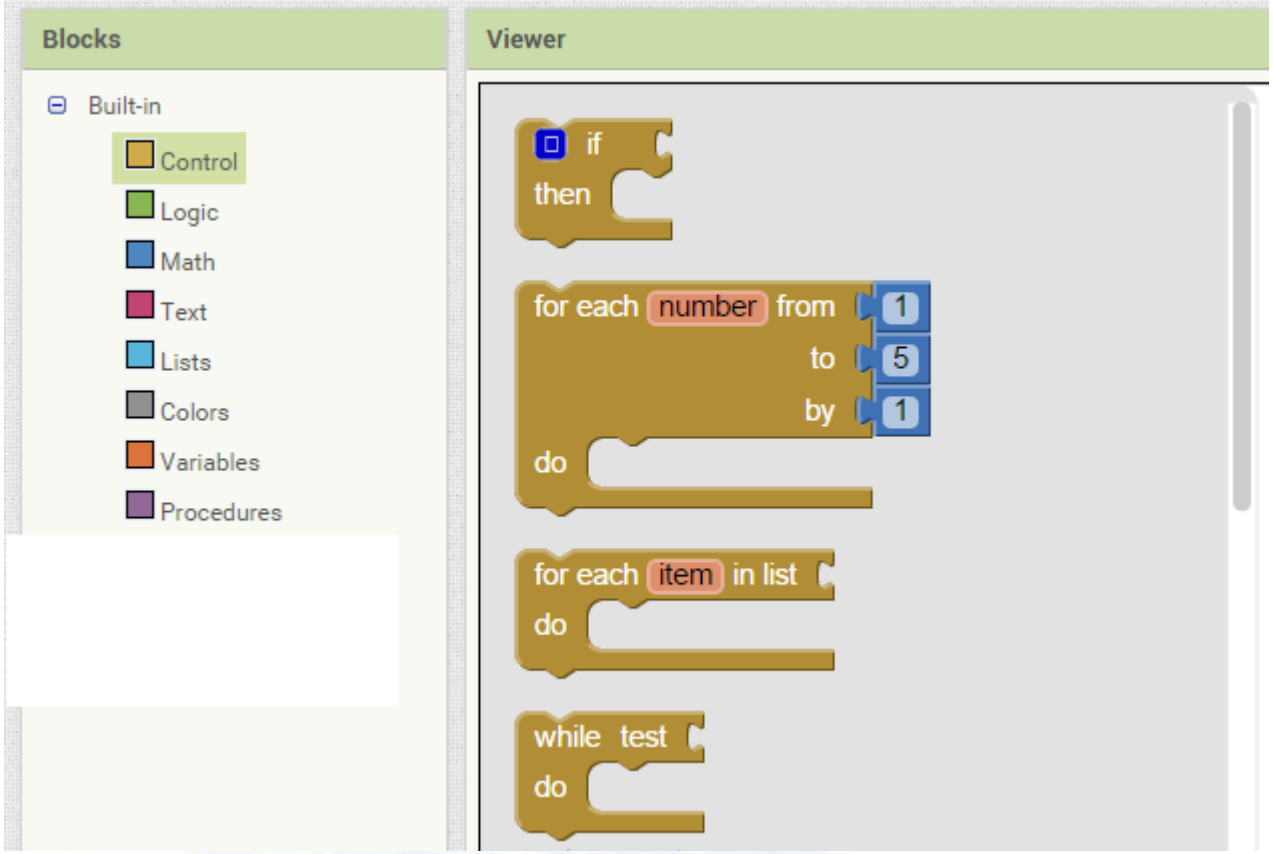
Coding with Blocks (1)

- The Blocks editor has different components that can be used to build our application logic.
- The components follow the drag and drop paradigm.



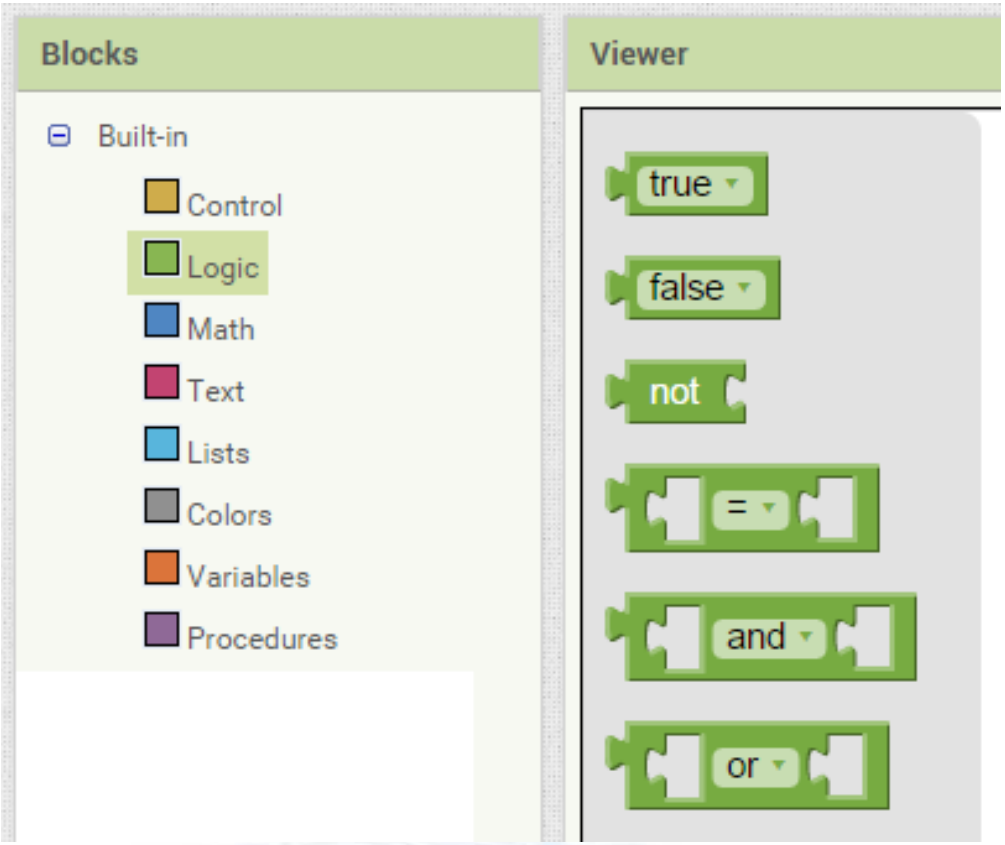
Coding with Blocks (2)

- Control: If, for, while, open screen, close screen, close application, etc.



Coding with Blocks (3)

- Logic: true, false, and, or, not, equals.

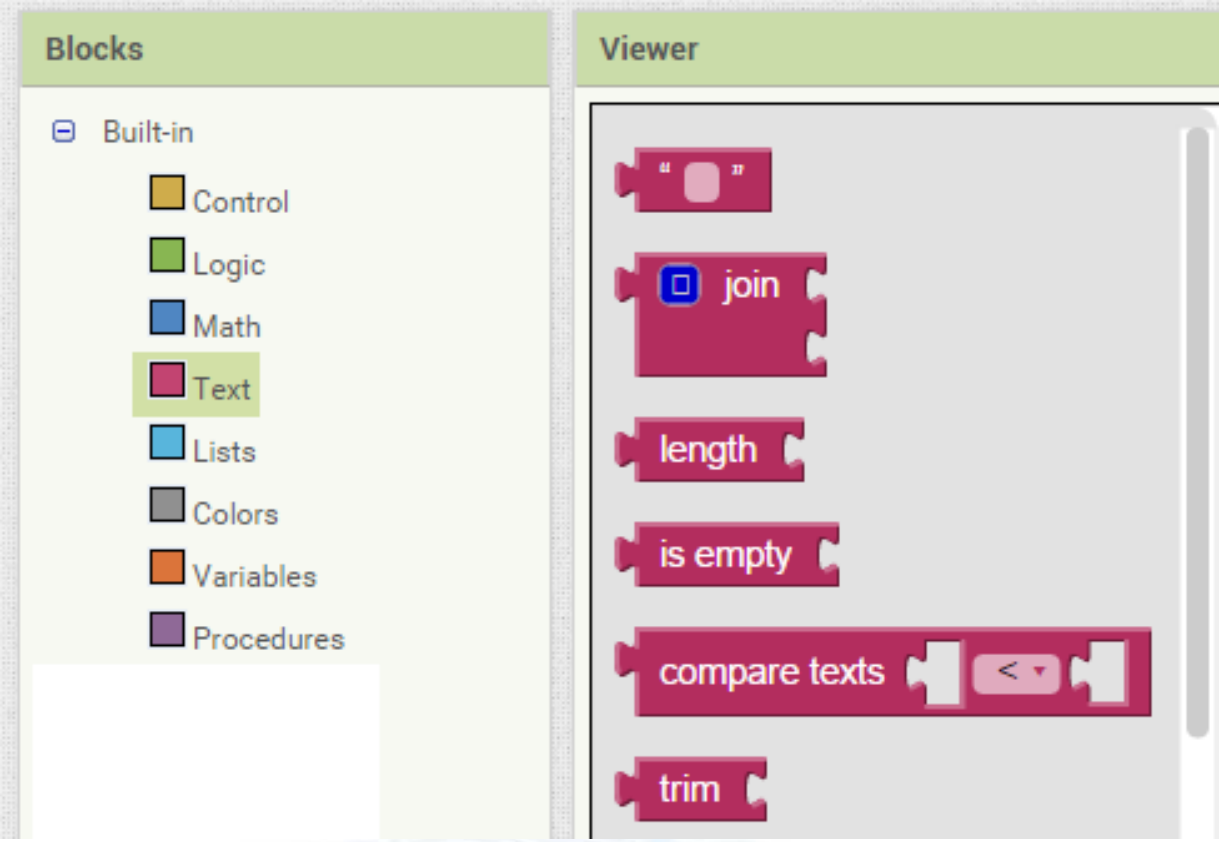


Coding with Blocks (4)

- Math: numbers, numerical operators, random number, abs, sin, cos, etc.

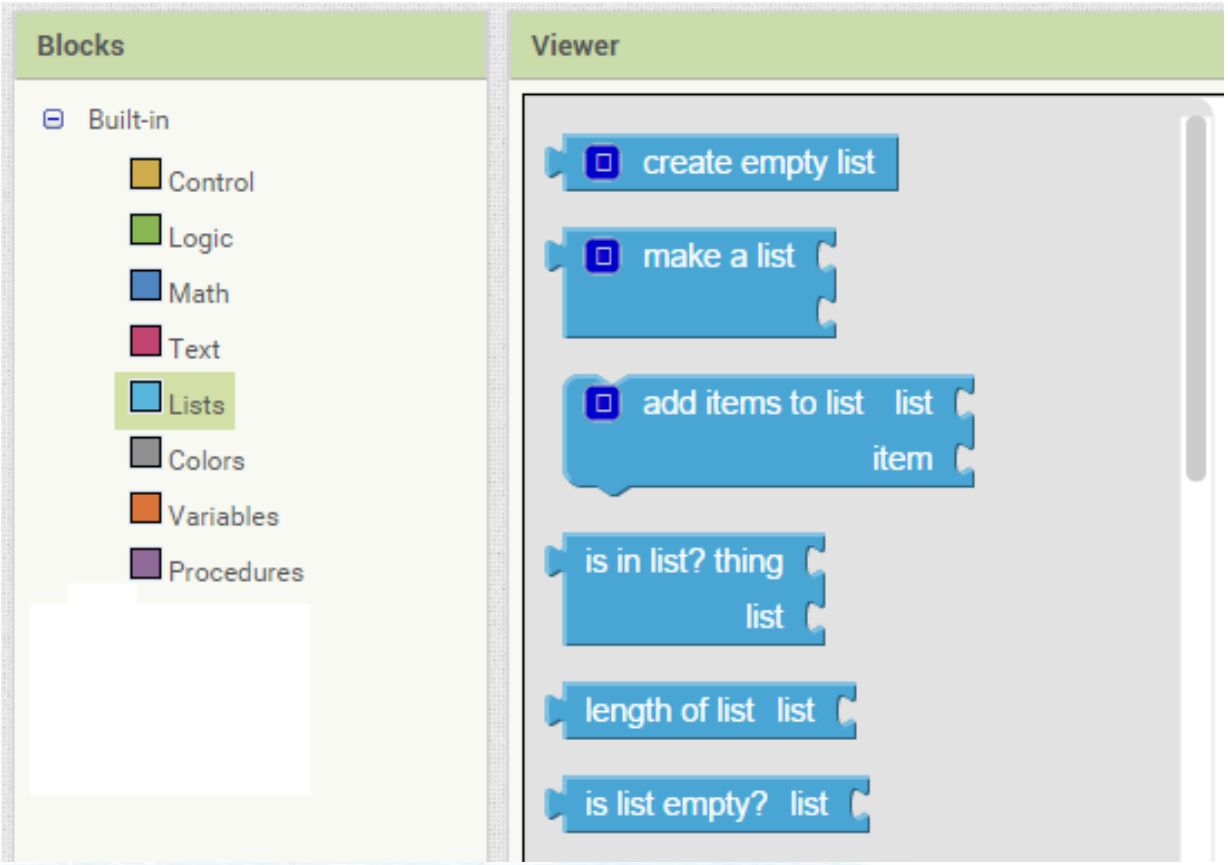
Coding with Blocks (5)

- Text: input user text, join strings, trim, split, replace, etc.



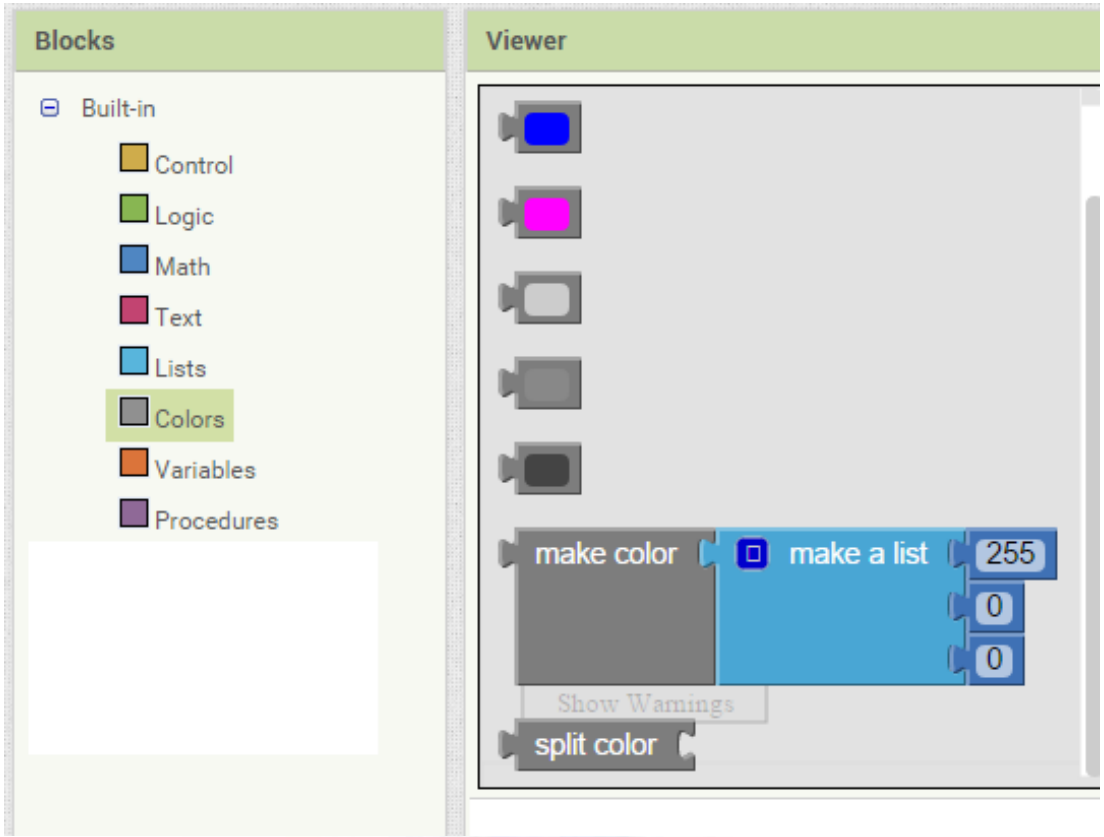
Coding with Blocks (6)

- Lists: functions for managing list of objects.



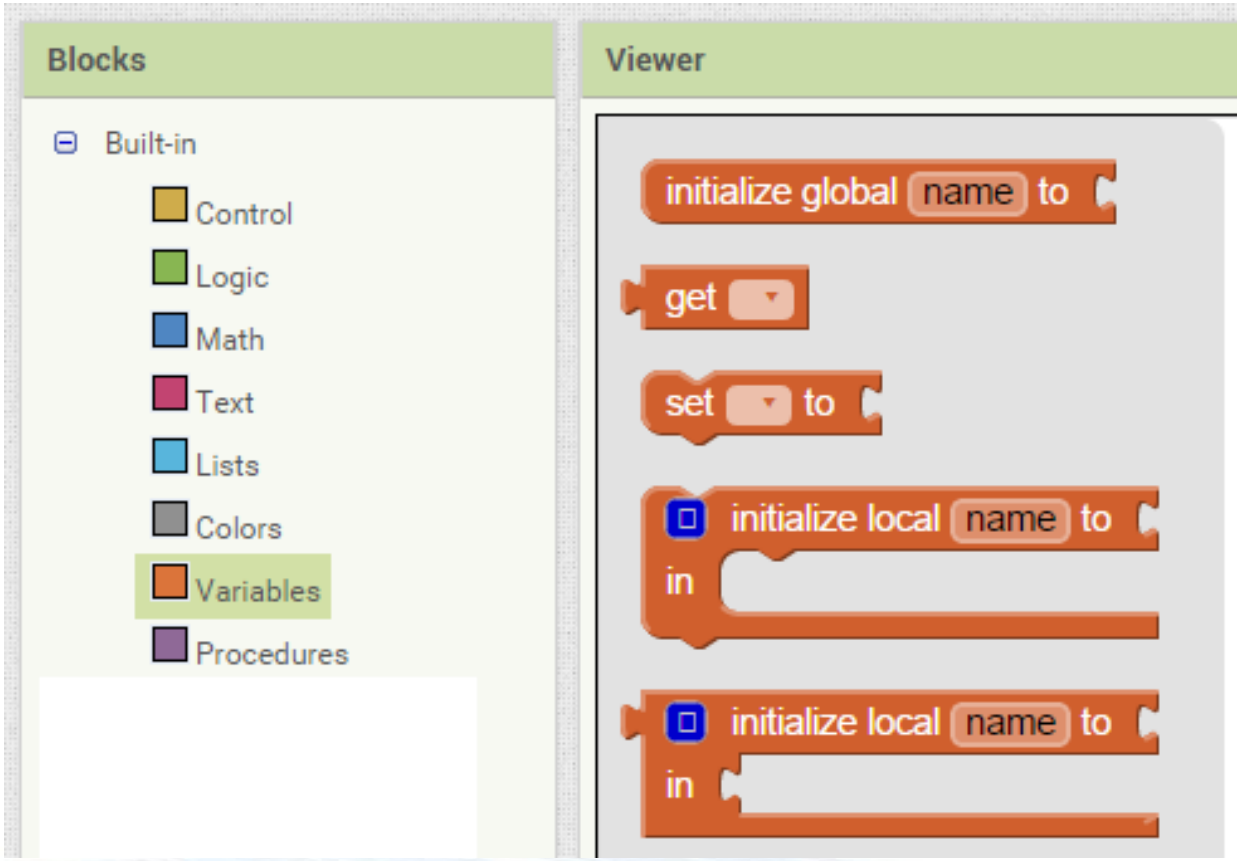
Coding with Blocks (7)

- Colors: common colors, create your own color based on r, g, b.



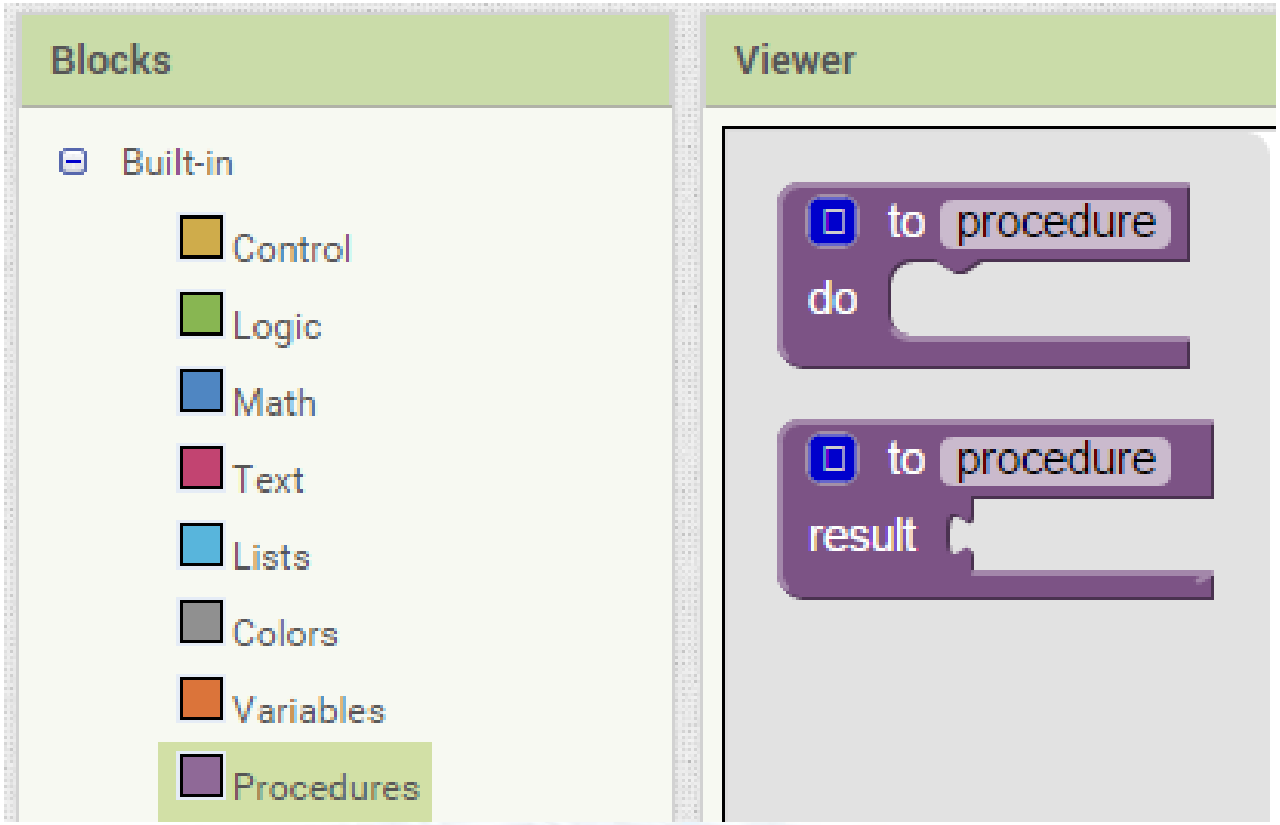
Coding with Blocks (8)

- Variables: local variables, global variables, get value, set value.



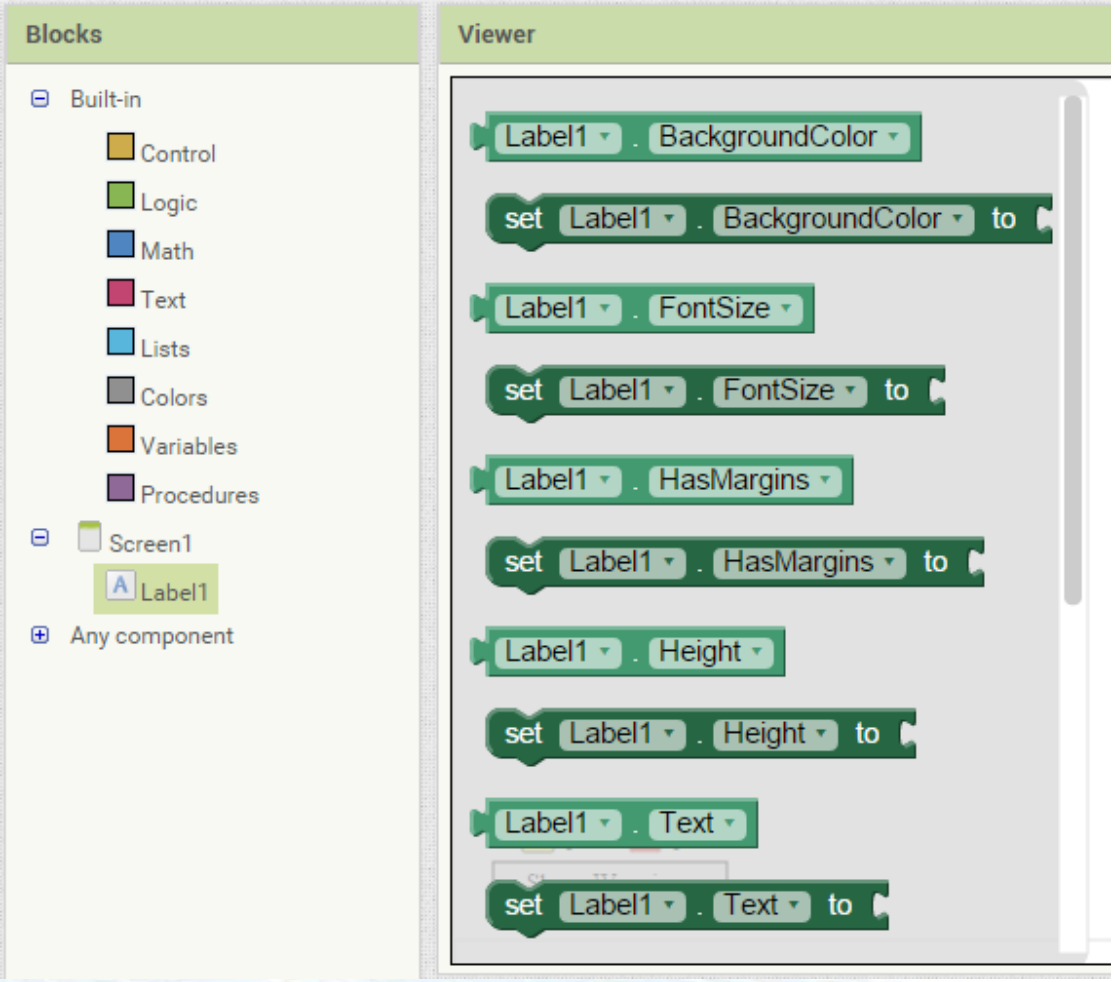
Coding with Blocks (9)

- Procedures: allows reusability of code creating functions.



Coding with Blocks (10)

- Additionally, each GUI component has its own blocks. For example, a label has blocks for getting and setting its text, getting and setting its font height and size, etc.

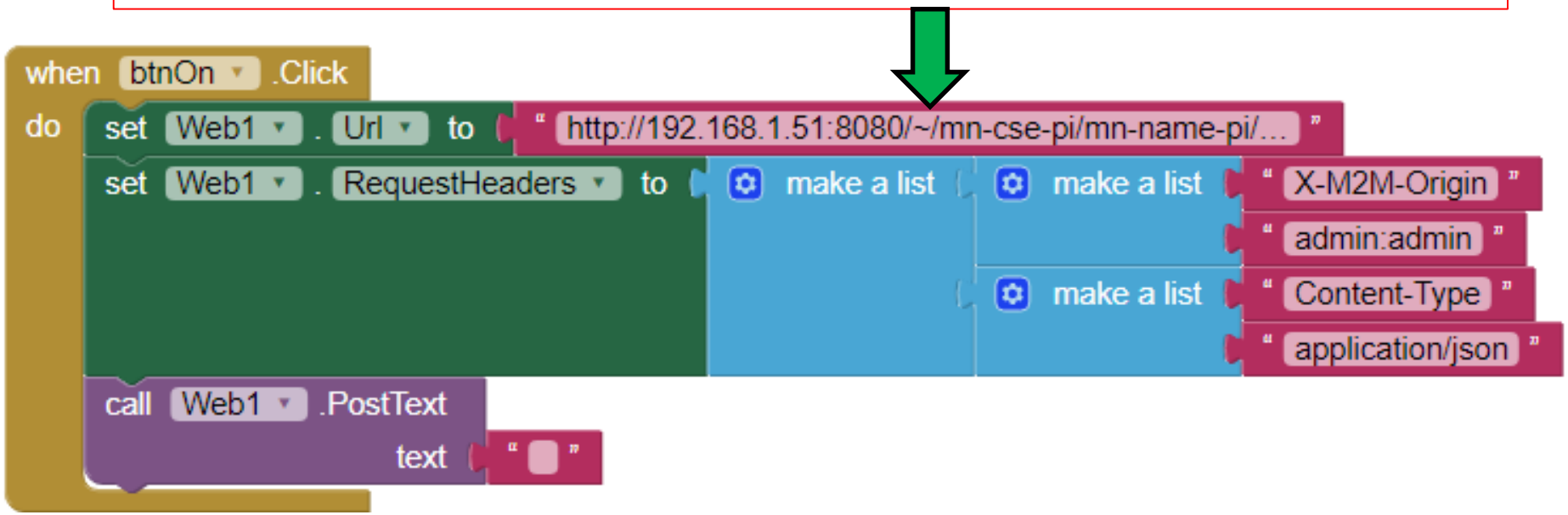


Remote Control Smart Home App

In the blocks area, draw the corresponding objects for:

a. Turn ON the Alarm

http://**YOUR.VM.IP.ADDRESS**:8080/~mn-cse-pi/mn-name-pi/ALARM_ON

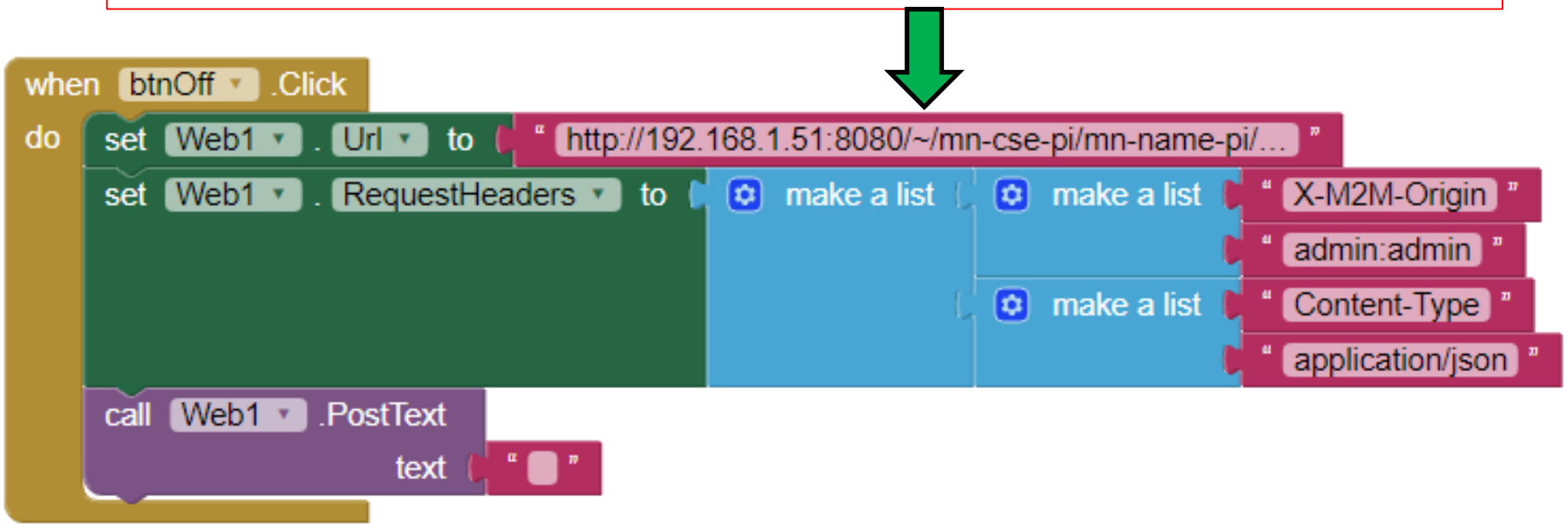


Remote Control Smart Home App

In the blocks area, draw the corresponding objects for:

b. Turn OFF the Alarm

http://**YOUR.VM.IP.ADDRESS**:8080/~mn-cse-pi/mn-name-pi/ALARM_OFF

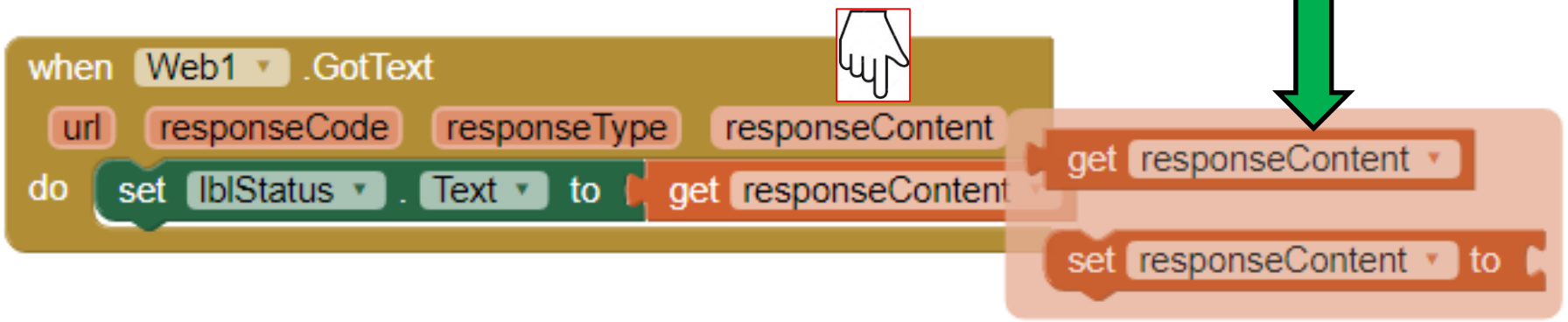


Remote Control Smart Home App

In the blocks area, draw the corresponding objects for:

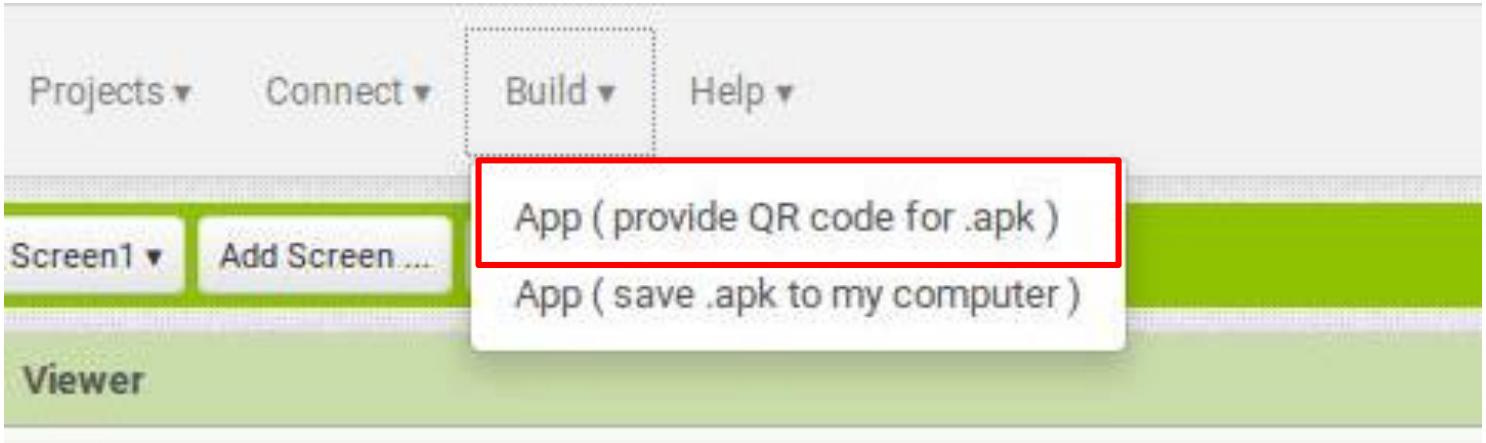
c. Get Server Response.

Put your mouse pointer over responseContent and then drag and drop “get ResponseContent”



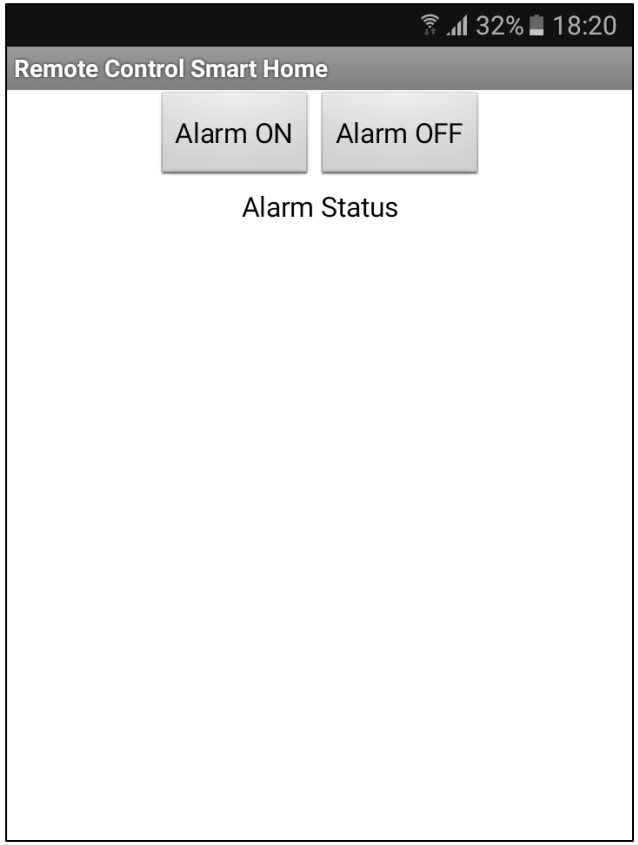
Remote Control Smart Home App

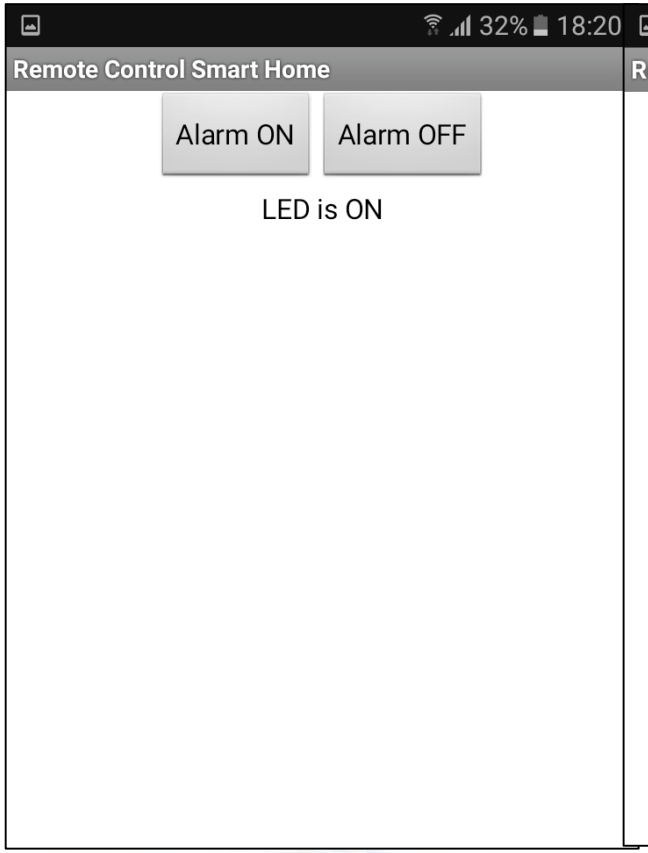
Build and Install the App in your smartphone.

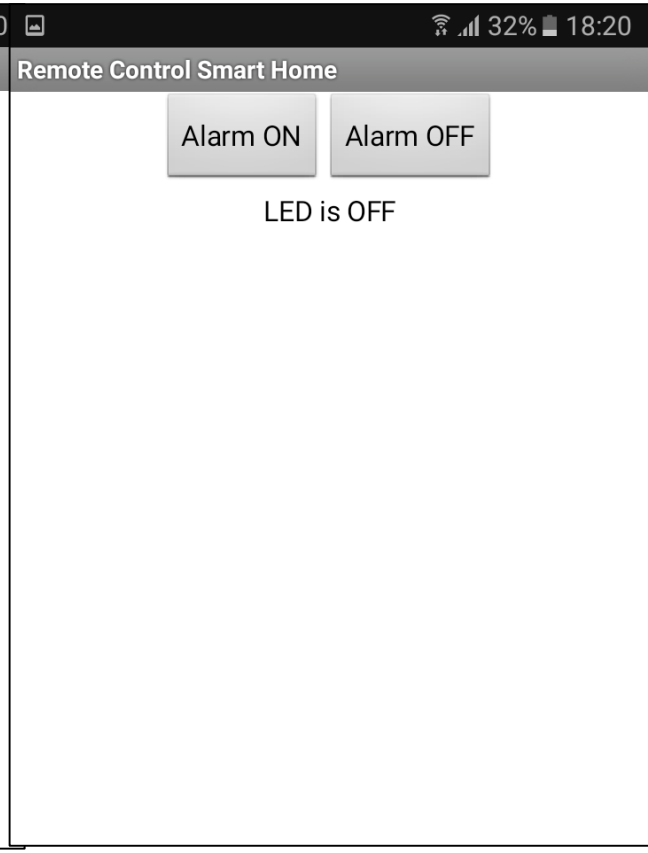


Remote Control Smart Home App

Run the app.







CHECKPOINT 6!