App Inventor + IoT: Starter Tutorial

30 min

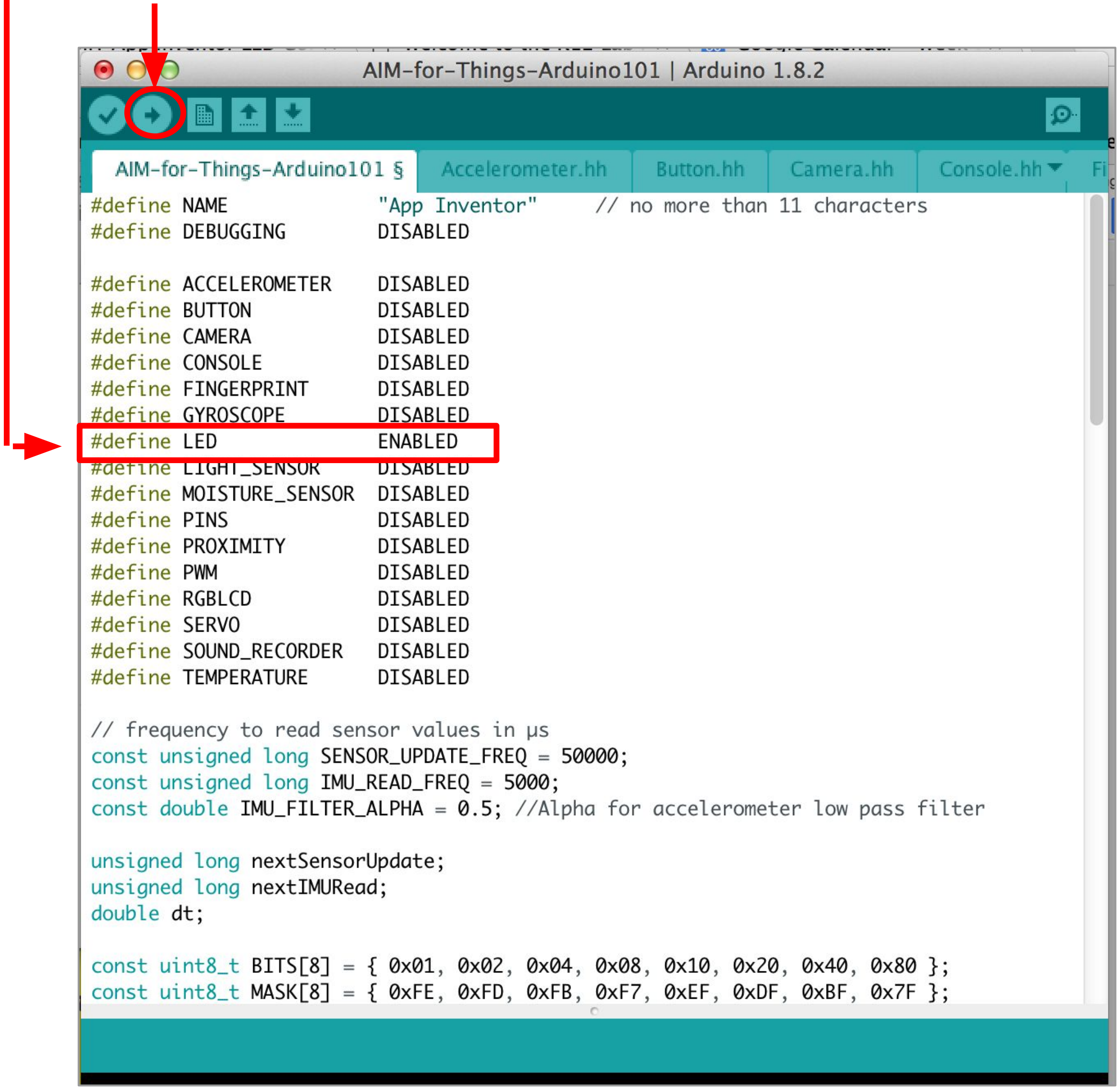
(with IoT Setup and Basic Connection tutorials completed)

This tutorial will help you get started with App Inventor + IoT and a LED light (light emitting diode … basically a small light) on an Arduino 101 controller.

Before you start, please complete the App Inventor + IoT Setup tutorial to set up your Arduino device.

First, we need to make sure we have the correct Arduino code running. Plug in your Arduino to your computer and open the AIM-for-Things-Arduino101.ino file (from the Setup tutorial above).

- For this tutorial make sure **LED** is set to **ENABLED** and all others are set to **DISABLED**
- You should also click the arrow button in the top left to upload the code.


AIM-for-Things-Arduino101 | Arduino 1.8.2
AIM-for-Things-Arduino101 §
Accelerometer.hh
Button.hh
Camera.hh
Console.hh

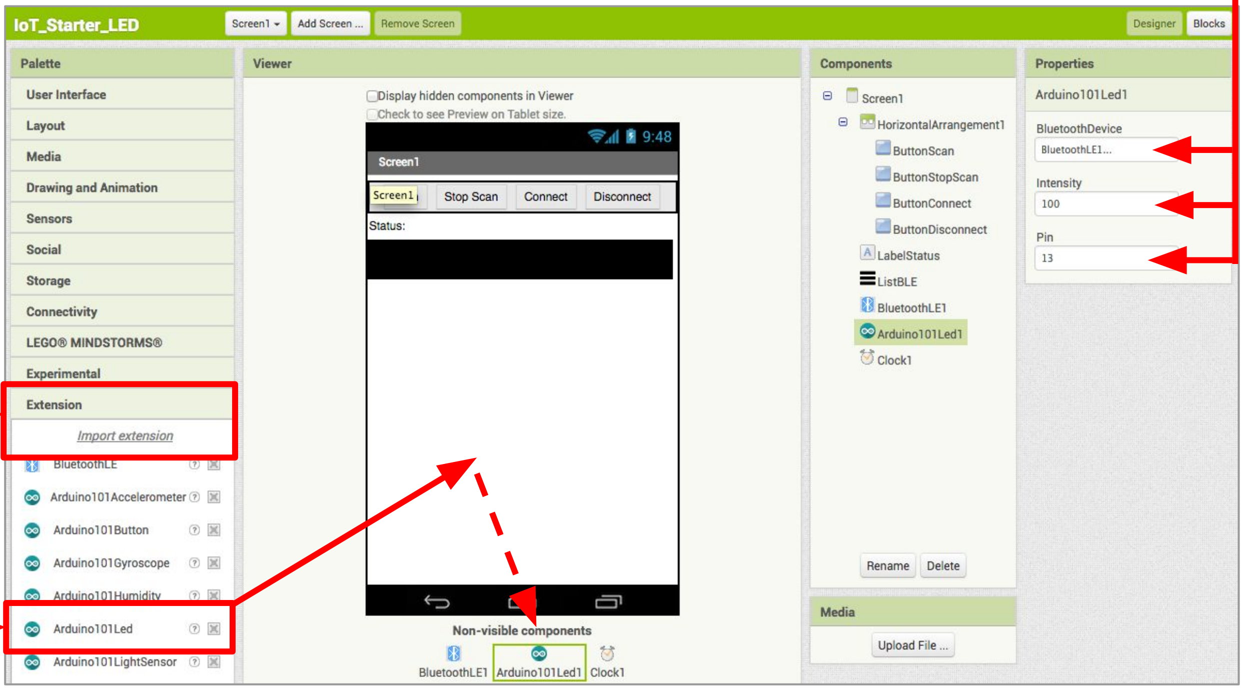
```
#define NAME             "App Inventor"     // no more than 11 characters
#define DEBUGGING        DISABLED

#define ACCELEROMETER    DISABLED
#define BUTTON           DISABLED
#define CAMERA           DISABLED
#define CONSOLE          DISABLED
#define FINGERPRINT      DISABLED
#define GYROSCOPE        DISABLED
#define LED              ENABLED
#define LIGHT_SENSOR     DISABLED
#define MOISTURE_SENSOR  DISABLED
#define PINS             DISABLED
#define PROXIMITY        DISABLED
#define PWM              DISABLED
#define RGBLCD           DISABLED
#define SERVO            DISABLED
#define SOUND_RECORDER   DISABLED
#define TEMPERATURE      DISABLED

// frequency to read sensor values in µs
const unsigned long SENSOR_UPDATE_FREQ = 50000;
const unsigned long IMU_READ_FREQ = 5000;
const double IMU_FILTER_ALPHA = 0.5; //Alpha for accelerometer low pass filter

unsigned long nextSensorUpdate;
unsigned long nextIMURead;
double dt;

const uint8_t BITS[8] = { 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80 };
const uint8_t MASK[8] = { 0xFE, 0xFD, 0xFB, 0xF7, 0xEF, 0xDF, 0xBF, 0x7F };
```

Next, you should complete the App Inventor + IoT Basic Connection tutorial to make a basic connection to the Arduino device. If you prefer, you can download the completed .aia file here.

The remaining steps all build off of the the starter code for Basic Connection tutorial and .aia.

First, we need to add the necessary extension.

- In the Palette window, click on Extension at the bottom and then on "Import extension" and click on "URL".
  - Paste in this URL:
    http://iot.appinventor.mit.edu/assets/resources/edu.mit.appinventor.iot.arduino101.aix
- Add the **Arduino101Led** extension to your app by dragging it onto the Viewer.
- In the Properties tab for the **Arduino101Led1**
  - Set *BluetoothDevice* to "BluetoothLE1".
  - Set *Intensity* to "100" (should already be set).
  - Set the *Pin* to 13. This is the pin number of the built-in LED on the Arduino board.



Arduino101Led1

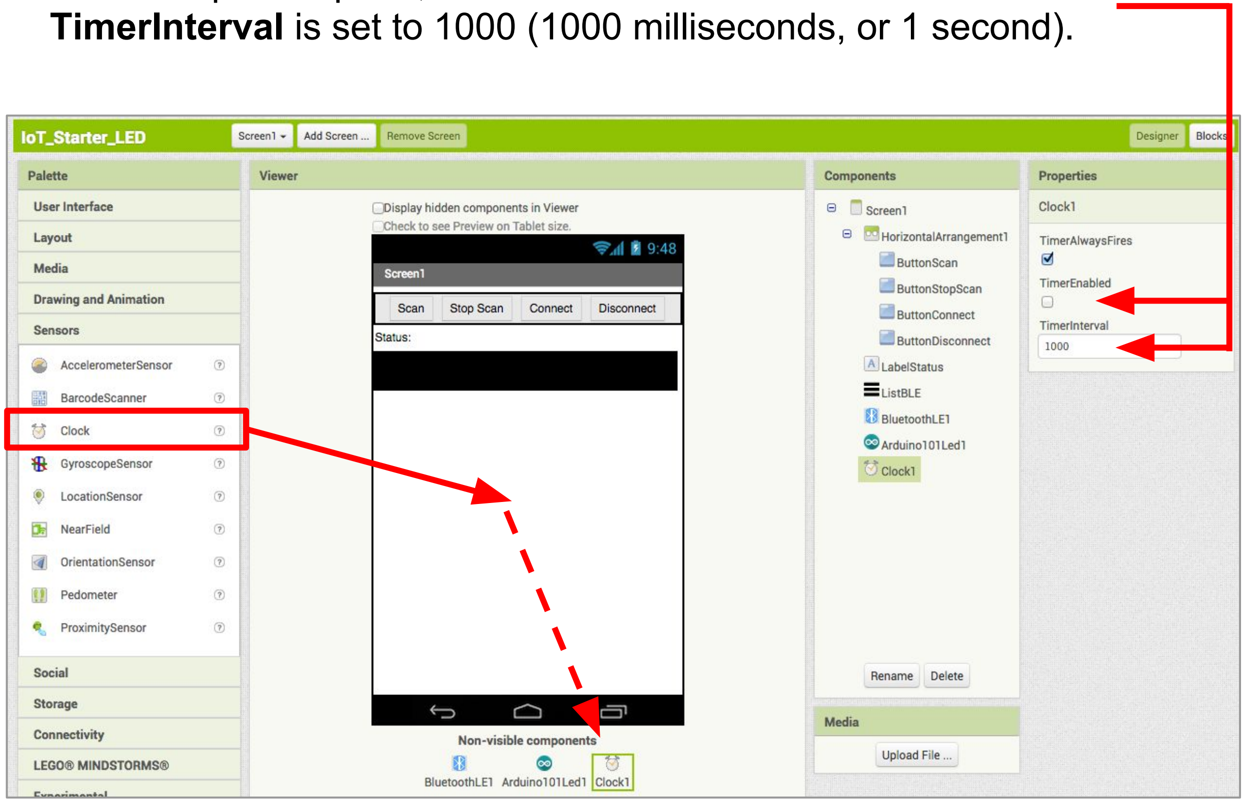
BluetoothDevice: BluetoothLE1...

Intensity: 100

Pin: 13

We are going to have our onboard LED blink, so we need a Clock component as a trigger to turn the LED on and off every second.

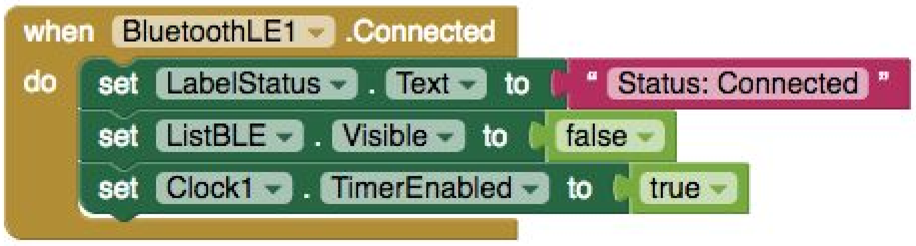
- From the Sensors drawer in the Palette, drag a Clock component onto the Viewer.
- In the Properties pane, *uncheck* **TimerEnabled** and make sure **TimerInterval** is set to 1000 (1000 milliseconds, or 1 second).



Now switch to the Blocks Editor view

We want to set the blinking to start once the user connects the Arduino in the app.

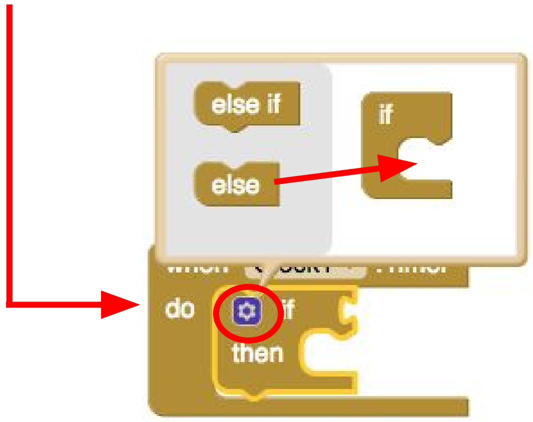
- From **Clock1** in the Blocks pane, add **set Clock1.TimerEnabled** to the existing **when BluetoothLE1.Connected** block from the Basic Connection tutorial.
  - From the Logic drawer in the Blocks pane, add a **true** block and snap to **set Clock1.TimerEnabled.**



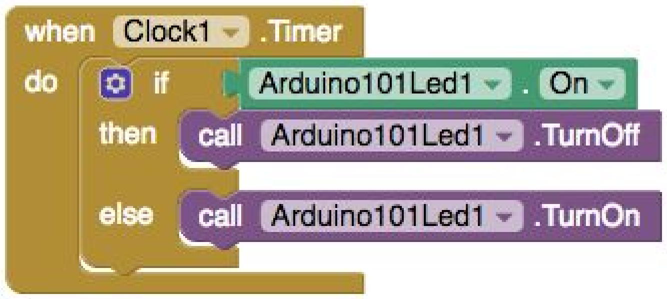
```
when BluetoothLE1 .Connected
do  set LabelStatus . Text to " Status: Connected "
    set ListBLE . Visible to false
    set Clock1 . TimerEnabled to true
```

Next we want to turn the LED on and off each second, when the Timer is triggered.

- From **Clock1** in the Blocks pane, drag out **when Clock1.Timer.**
    - from the Control drawer, drag out an **if-then** block.
    - Click on the blue gear icon and drag an else block into the **if-then** to make it an **if-then-else** block.



-
    - From **Arduino101Led1** in the Blocks pane, drag out a **Arduino101LED1.On** block and snap to **if**.
    - From **Arduino101Led1** in the Blocks pane, drag out a **call Arduino101Led1.TurnOff** and snap it to **then.**
    - From **Arduino101Led1** in the Blocks pane, drag out a **call Arduino101Led1.TurnOn** and snap it to **else.**



```
when Clock1 .Timer
do  if   Arduino101Led1 . On
    then call Arduino101Led1 .TurnOff
    else call Arduino101Led1 .TurnOn
```

Your app should now be working! Test it out by connecting your Arduino device using the MIT AI2 Companion (if you haven't already). Once you press “Connect”, you should see the LED on the Arduino board blink. Disconnecting should stop it blinking.