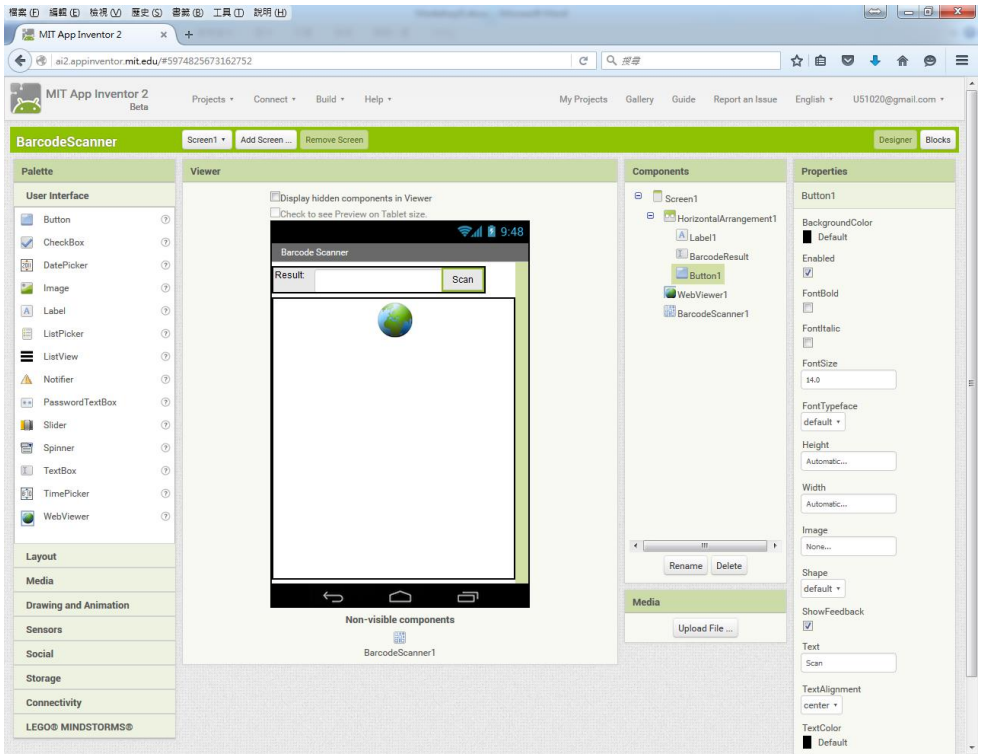
1. Barcode Scanner

1.1 Barcode Scanner

App Inventor supports scanning of product identification “bar codes” and QR codes. Bar codes are typically found on consumer product packaging. QR codes are used for web addresses (so you can scan the QR code and go directly to the web address), part and product inventory tracking, and to deliver short messages.

1.2 Exercise: QR Code Scanner

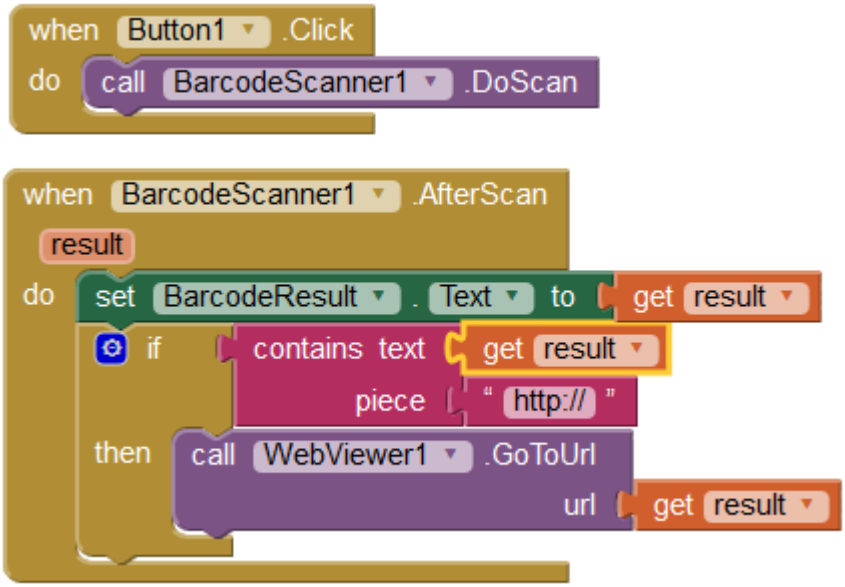
1.2.1 Designer View



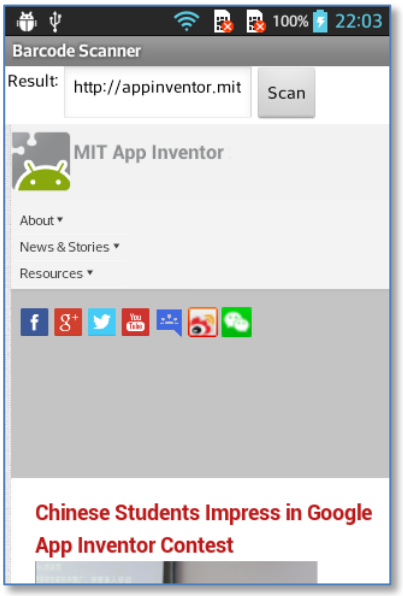
1.2.2 Components

Component	Name	Properties	Remark
Screen	Screen1	Text = “**Barcode Scanner**”	
HorizontalArrangement	HorizontalArrangement1		
Label	Label1	Text = “**Result:**”	Inside HorizontalArrangement1
Textbox	BarcodeResult	Text = *[Blank]*	Inside HorizontalArrangement1
Button	Button1	Text = “**Scan**”	Inside HorizontalArrangement1
WebView	WebView1	Height = “**Fill parent**” Width = “**Fill parent**”	
BarcodeScanner	BarcodeScanner1	USeExternalScanner = *[Blank]*	

1.2.3 Block Configuration



1.2.4 Sample Output



1.2.5 Sample QR Code

2. Sensor

2.1 Overview

Most Android-powered devices have built-in sensors that measure motion, orientation, and various environmental conditions.
These sensors are capable of providing raw data with high precision and accuracy, and are useful if you want to monitor three-dimensional device movement or positioning, or you want to monitor changes in the ambient environment near a device.

2.2 Proximity Sensor

A sensor component that can measure the proximity of an object (in cm) relative to the view screen of a device. This sensor is typically used to determine whether a handset is being held up to a person's ear; i.e. let you determine how far away an object is from a device. Many devices return the absolute distance, in cm, but some return only near and far values. In this case, the sensor usually reports its maximum range value in the far state and a lesser value in the near state. It reports the distance from the object to the device

2.3 Accelerometer Sensor

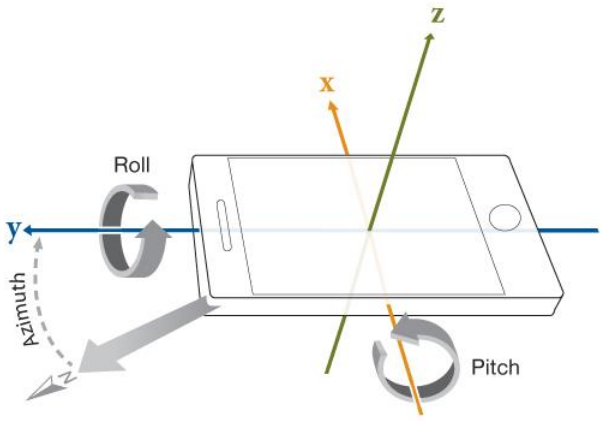
Non-visible component that can detect shaking and measure acceleration approximately in three dimensions using SI units (m/s2). The components are:

- xAccel: 0 when the phone is at rest on a flat surface, positive when the phone is tilted to the right (i.e., its left side is raised), and negative when the phone is tilted to the left (i.e., its right size is raised).
- yAccel: 0 when the phone is at rest on a flat surface, positive when its bottom is raised, and negative when its top is raised.
- zAccel: Equal to -9.8 (earth's gravity in meters per second per second when the device is at rest parallel to the ground with the display facing up, 0 when perpindicular to the ground, and +9.8 when facing down. The value can also be affected by accelerating it with or against gravity.

2.4 Orientation Sensor

Use an orientation sensor component to determine the phone's spatial orientation. An orientation sensor is a non-visible component that reports the following three values, in degrees. These measurements assume that the device itself is not moving.

- Roll: 0 degree when the device is level, increasing to 90 degrees as the device is tilted up onto its left side, and decreasing to −90 degrees when the device is tilted up onto its right side.
- Pitch: 0 degree when the device is level, increasing to 90 degrees as the device is tilted so its top is pointing down, then decreasing to 0 degree as it gets turned over. Similarly, as the device is tilted so its bottom points down, pitch decreases to −90 degrees, then increases to 0 degree as it gets turned all the way over.
- Azimuth: 0 degree when the top of the device is pointing north, 90 degrees when it is pointing east, 180 degrees when it is pointing south, 270 degrees when it is pointing west, etc.



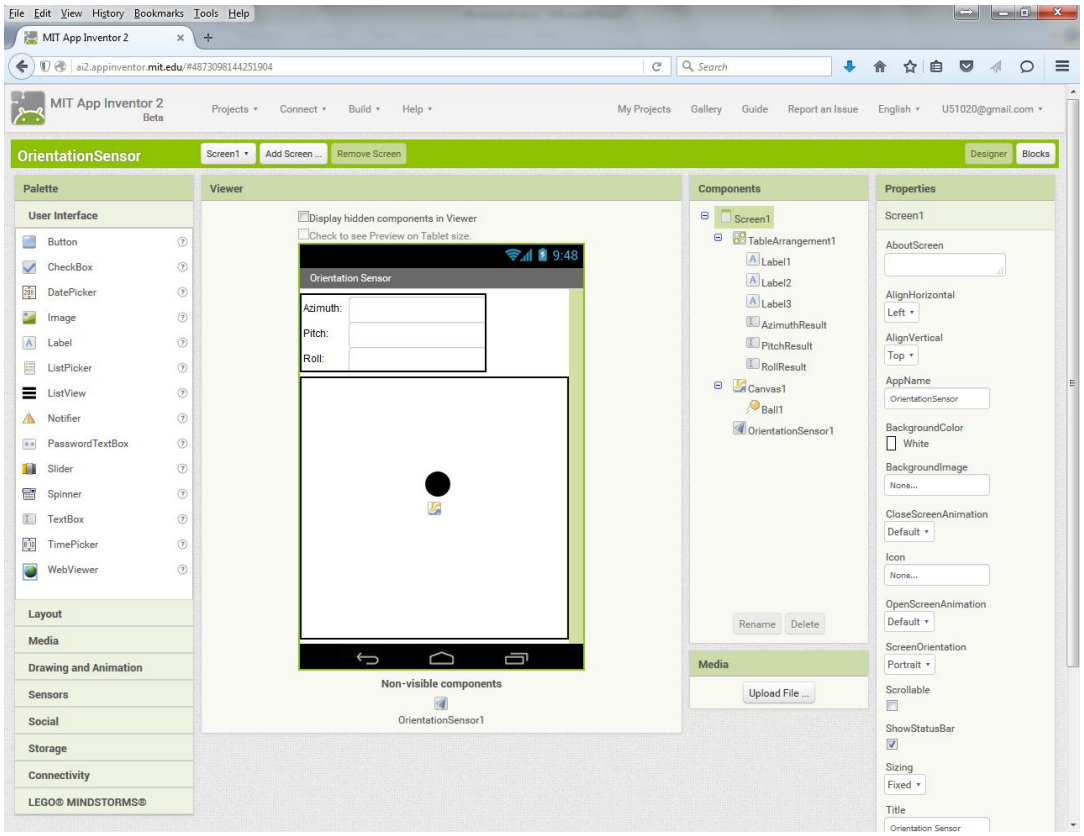
2.5 Near Field Communication (NFC)

Near field communication (NFC) is a set of communication protocols that enable two electronic devices, one of which is usually a portable device such as a smartphone, to establish communication by bringing them within 10 cm (4 in) of each other.

It is a non-visible component to provide NFC capabilities. For now this component supports the reading and writing of text tags only (if supported by the device). In order to read and write text tags, the component must have its ReadMode property set to True or False respectively.

2.6 Exercise: Orientation Sensor

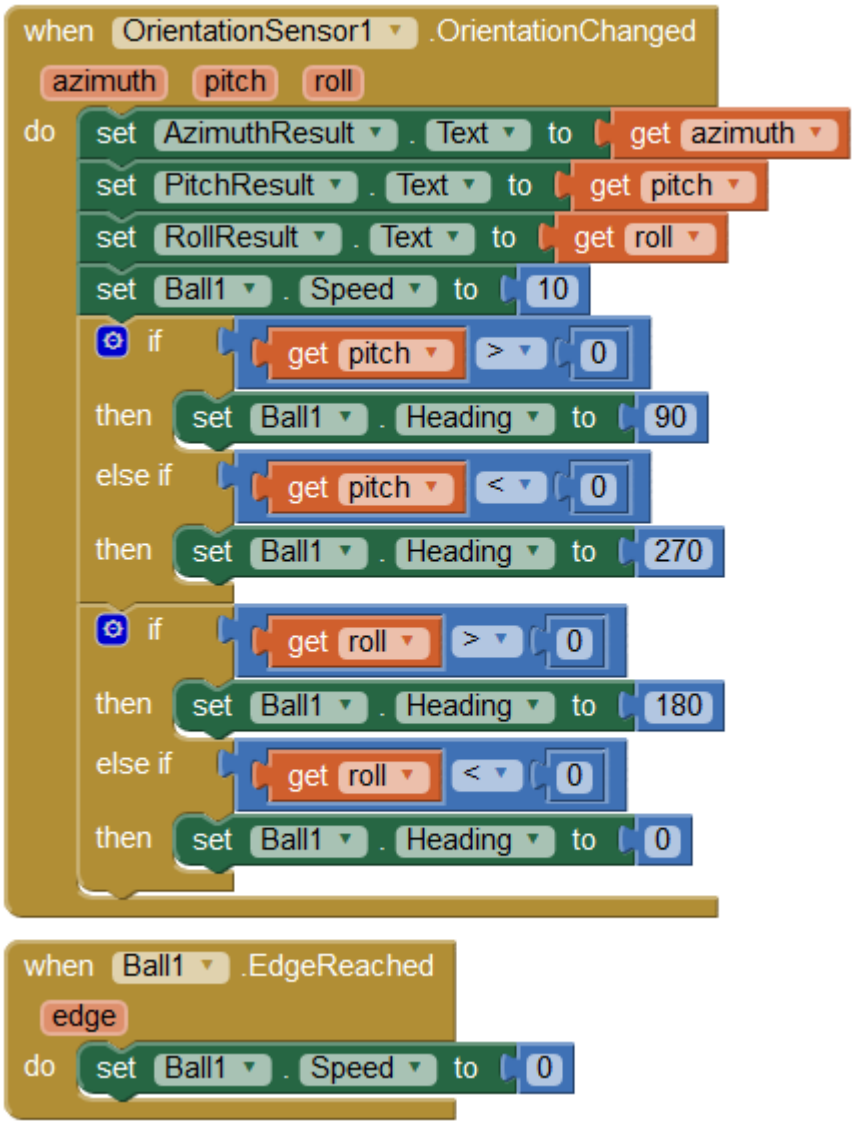
2.6.1 Designer View



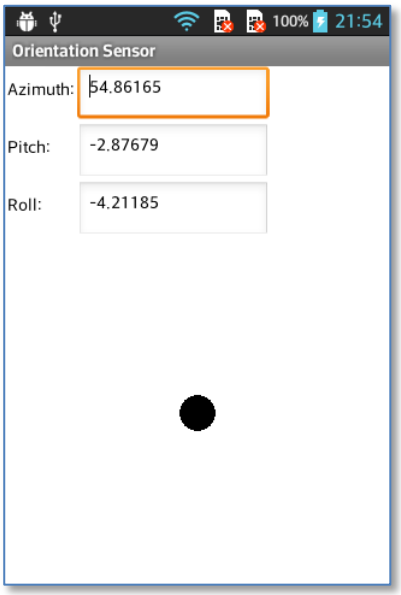
2.6.2 Components

Component	Name	Properties	Remark
Screen	Screen1	Text = "**Orientation Sensor**" ScreenOrientation = "**Portrait**"	
TableArrangement	TableArrangement1	Columns = "**2**" Rows = "**3**"	
Label	Label1	Text = "**Azimuth:**"	Inside TableArrangement1
Label	Label2	Text = "**Pitch:**"	Inside TableArrangement1
Label	Label3	Text = "**Roll:**"	Inside TableArrangement1
Textbox	AzimuthResult	Hint = "**Azimuth**"	Inside TableArrangement1
Textbox	PitchResult	Hint = "**Pitch**"	Inside TableArrangement1
Textbox	RollResult	Hint = "**Roll**"	Inside TableArrangement1
Canvas	Canvas1	Height = "**Fill parent**" Width = "**Fill parent**"	
Ball	Ball1	Radius = "**15**"	Inside Canvas1
OrientationSensor	OrientationSensor1		

2.6.3 Block Configuration

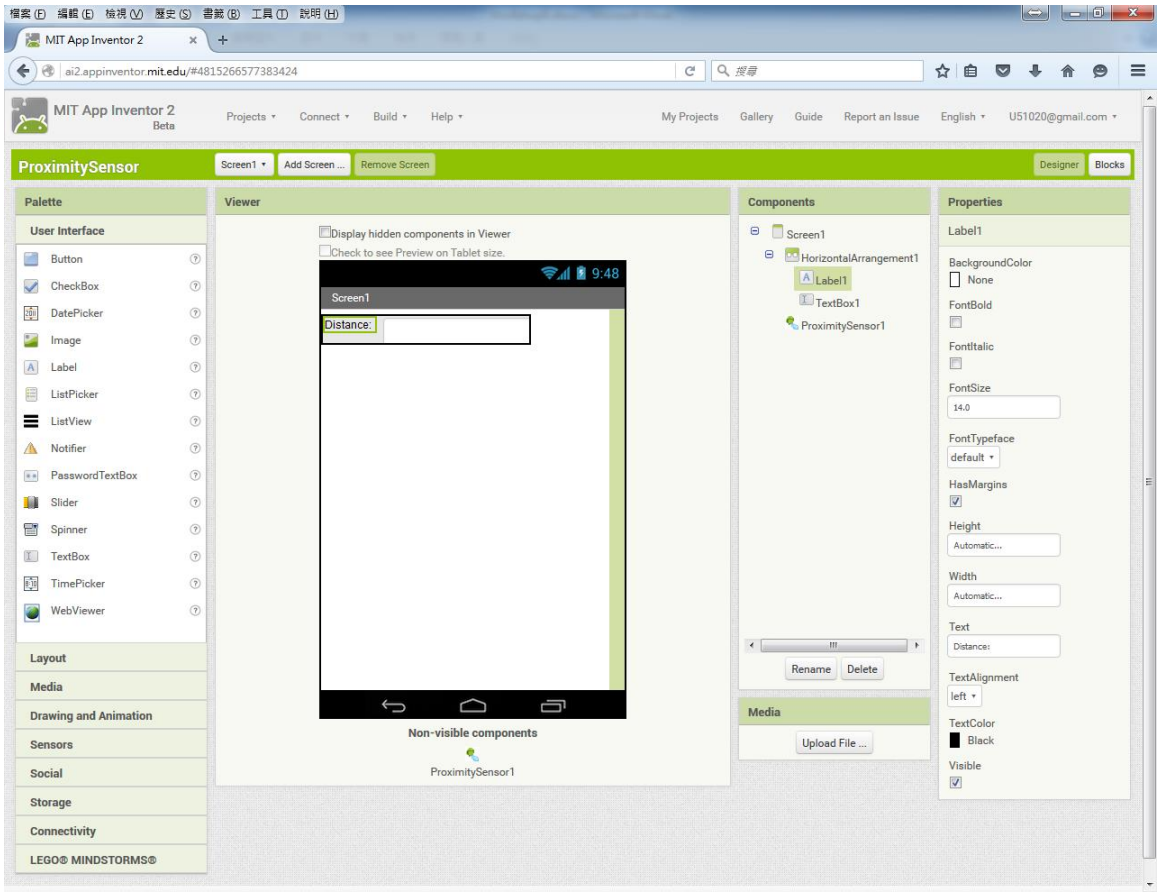


2.6.4 Sample Output



2.7 Exercise: Proximity Sensor

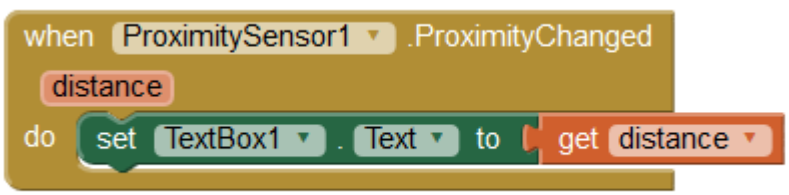
2.7.1 Designer View



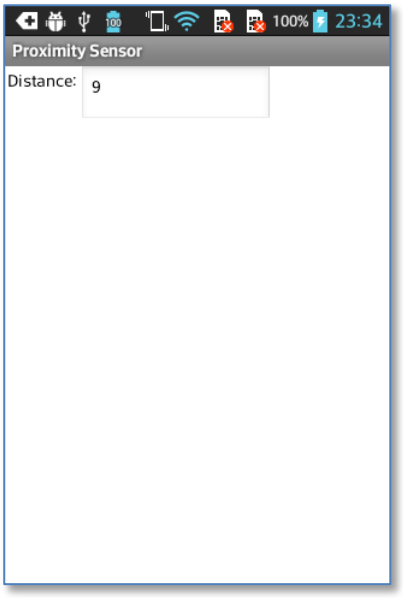
2.7.2 Components

Component	Name	Properties	Remark
Screen	Screen1	Text = "**Proximity Sensor**"	
HorizontalArrangement	HorizontalArrangement1		
Label	Label1	Text = "**Distance:**"	Inside HorizontalArrangement1
Textbox	BarcodeResult	Text = *[Blank]*	Inside HorizontalArrangement1
ProximitySensor	ProximitySensor1		

2.7.3 Block Configuration

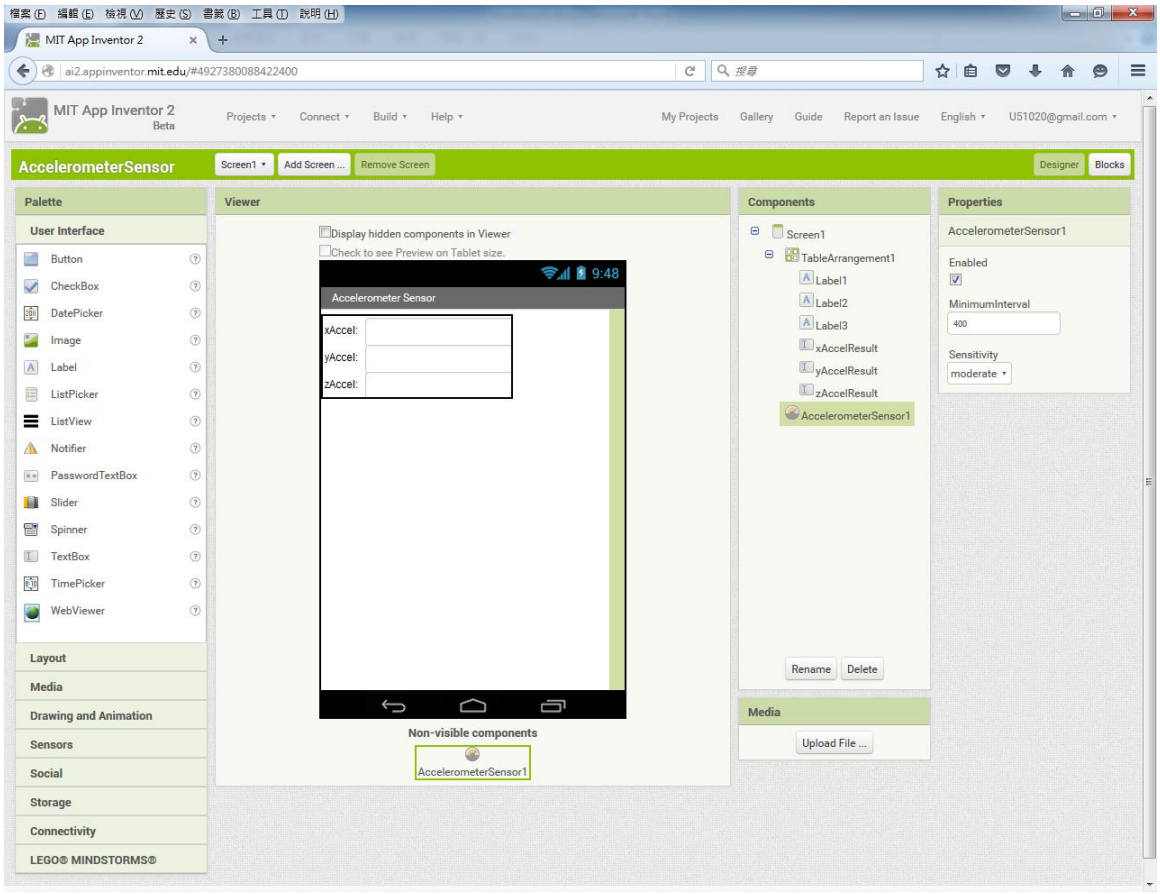


2.7.4 Sample Output



2.8 Exercise: Accelerometer Sensor

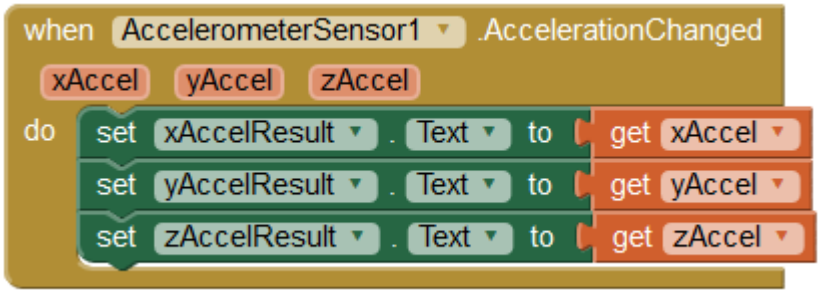
2.8.1 Designer View



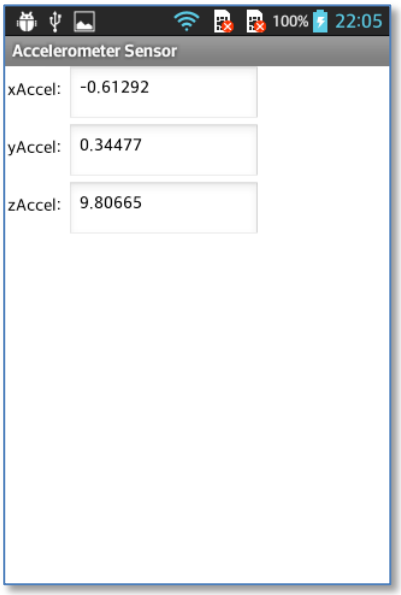
2.8.2 Components

Component	Name	Properties	Remark
Screen	Screen1	Text = "**Accelerometer Sensor**" ScreenOrientation = "**Portrait**"	
TableArrangement	TableArrangement1	Columns = "**2**" Rows = "**3**"	
Label	Label1	Text = "**xAccel:**"	Inside TableArrangement1
Label	Label2	Text = "**yAccel:**"	Inside TableArrangement1
Label	Label3	Text = "**zAccel:**"	Inside TableArrangement1
Textbox	xAccelResult	Hint = "**xAccel**"	Inside TableArrangement1
Textbox	yAccelResult	Hint = "**yAccel**"	Inside TableArrangement1
Textbox	zAccelResult	Hint = "z**Accel**"	Inside TableArrangement1
AccelerometerSensor	AccelerometerSensor1		

2.8.3 Block Configuration

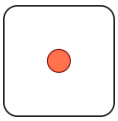


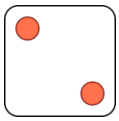
2.8.4 Sample Output

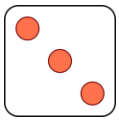


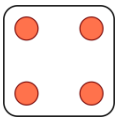
2.9 Exercise: Roll Dice

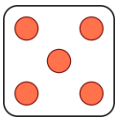
2.9.1 Media Files

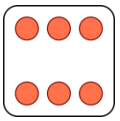

Dice1.png


Dice2.png


Dice3.png

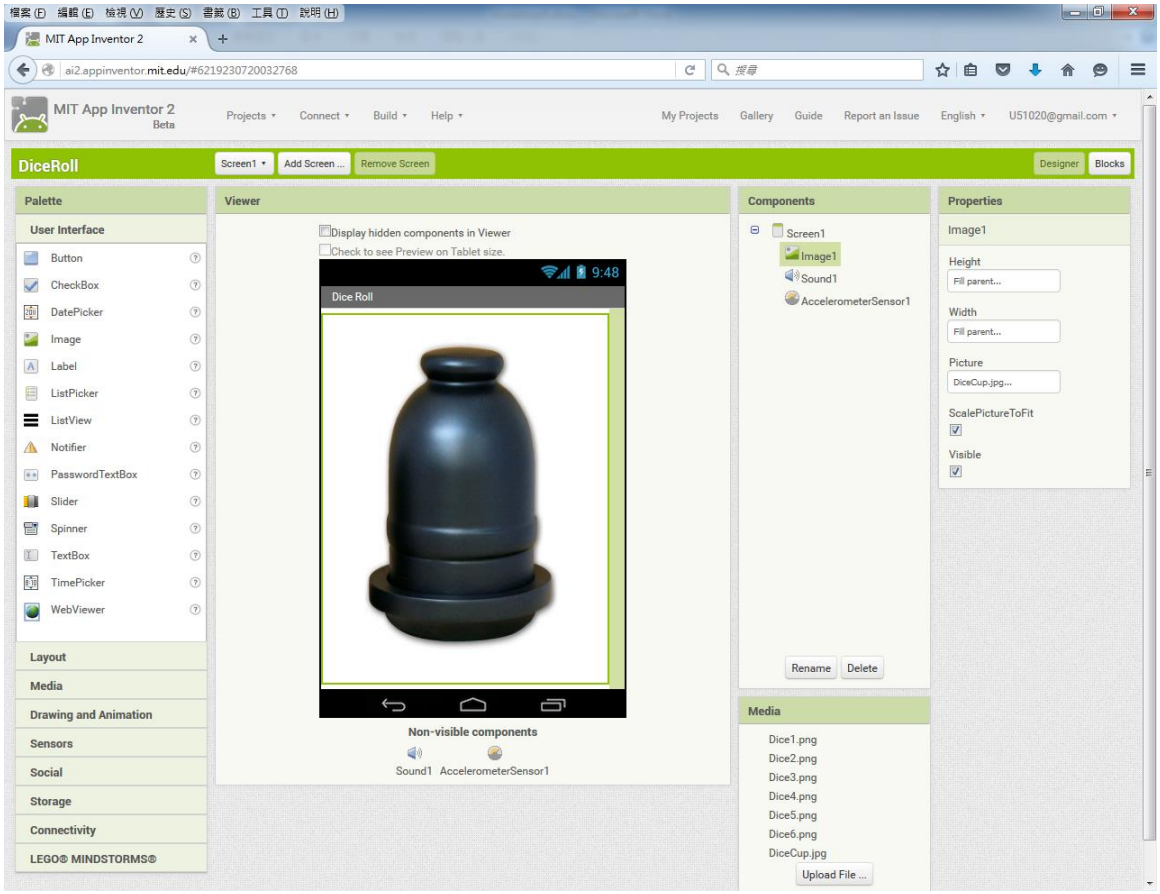

Dice4.png


Dice5.png


Dice6.png


DiceCup.jpg

2.9.2 Designer View

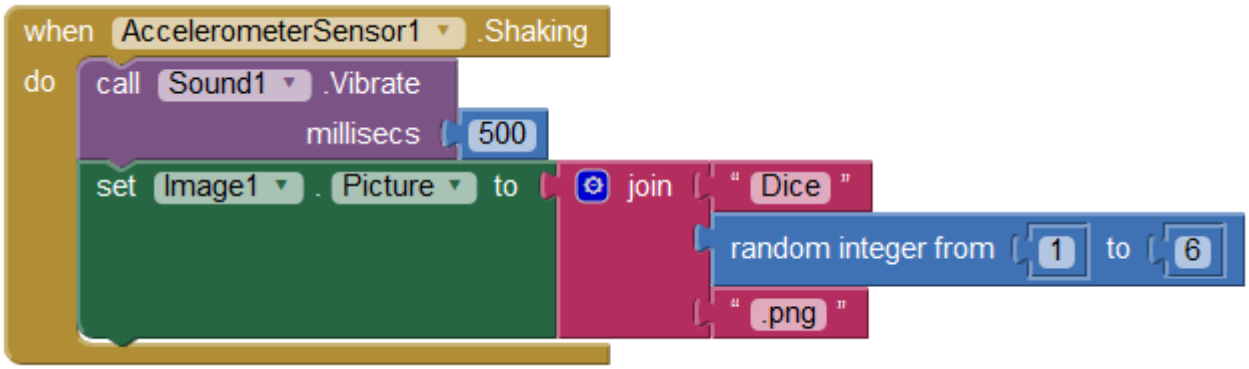


2.9.3 Components

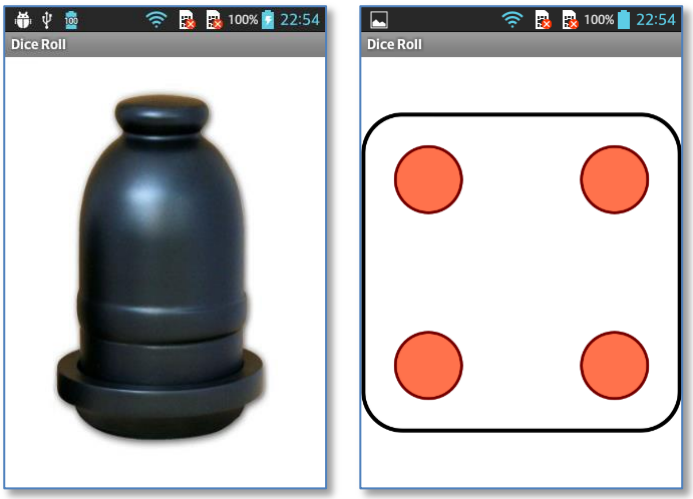
Component	Name	Properties	Remark
Screen	Screen1	Text = "**Roll Dice**" ScreenOrientation = "**Portrait**"	
Image	Image1	Height = "**Fill parent**" Width = "**Fill parent**" Picture = "**DiceCup.jpg**" ScalePictureToFit = "**X**"	
Sound	Sound1		

AccelerometerSensor	AccelerometerSensor1	MinimumInterval = "**2000**" Sensitivity = "**weak**"	

2.9.4 Block Configuration



2.9.5 Sample Output

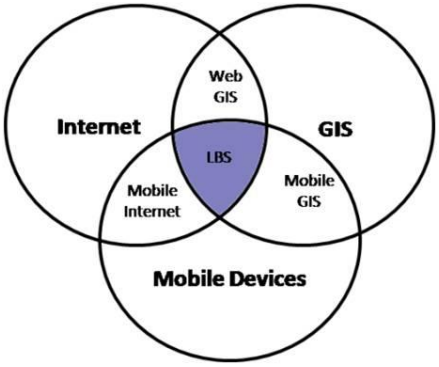


3. Location Based Service

3.1 Overview

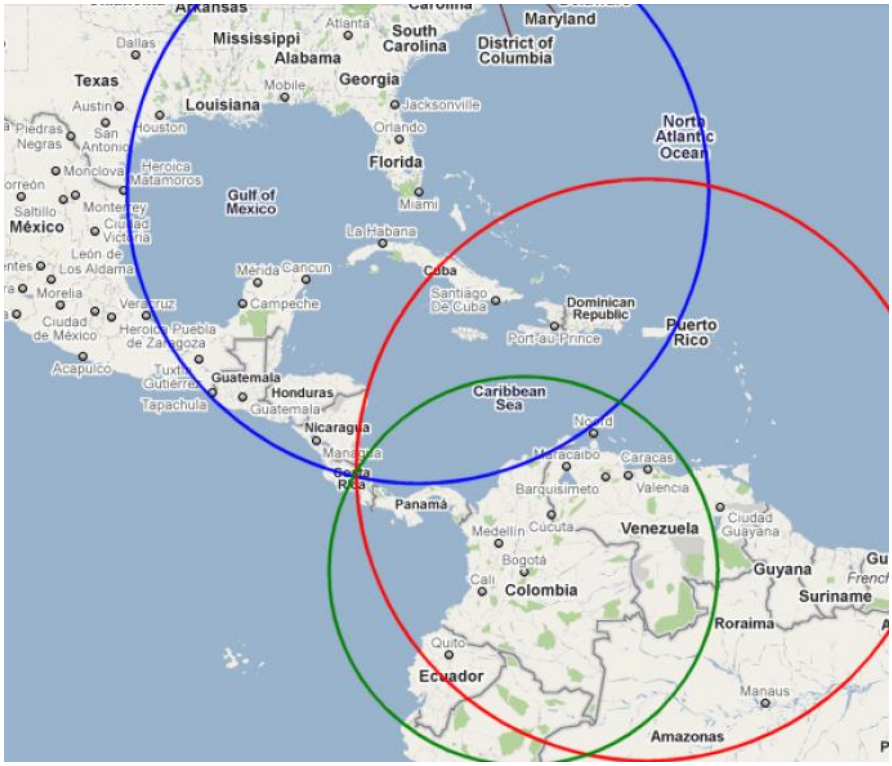
Location Based Service (LBS) is an information system driven by the ability of the central system to detect the geographical position of the mobile devices. For example:

- Locate the nearest bank, restaurant, gas station, hotel, golf course, hospital, police station, etc.
- Provide transportation information on how to go from 'here' to 'there'.
- Social networking is used to locate and reach events, friends and family members.



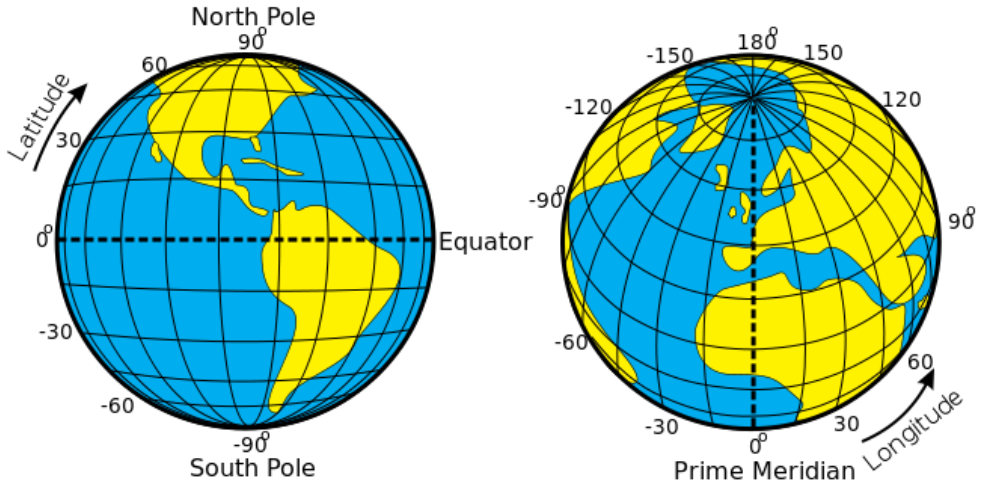
3.2 How the GPS Works?

Created by DOD-USA under the name NAVSTAR (Navigation System for Timing and Ranging) but it is commonly known as Global Positioning System (GPS). The system's backbone consists of 27 Earth-orbiting satellites (24 in operation and 3 in stand-by mode)



3.3 Latitude and Longitude

A geographic coordinate system is a coordinate system that enables every location on the Earth to be specified by a set of numbers or letters, or symbols. The coordinates are often chosen such that one of the numbers represents vertical position, and two or three of the numbers represent horizontal position. A common choice of coordinates is latitude, longitude and elevation.



3.4 Location Sensor

Non-visible component providing location information, including longitude, latitude, altitude (if supported by the device), and address. This can also perform "geocoding", converting a given address (not necessarily the current one) to a latitude (with the LatitudeFromAddress method) and a longitude (with the LongitudeFromAddress method).

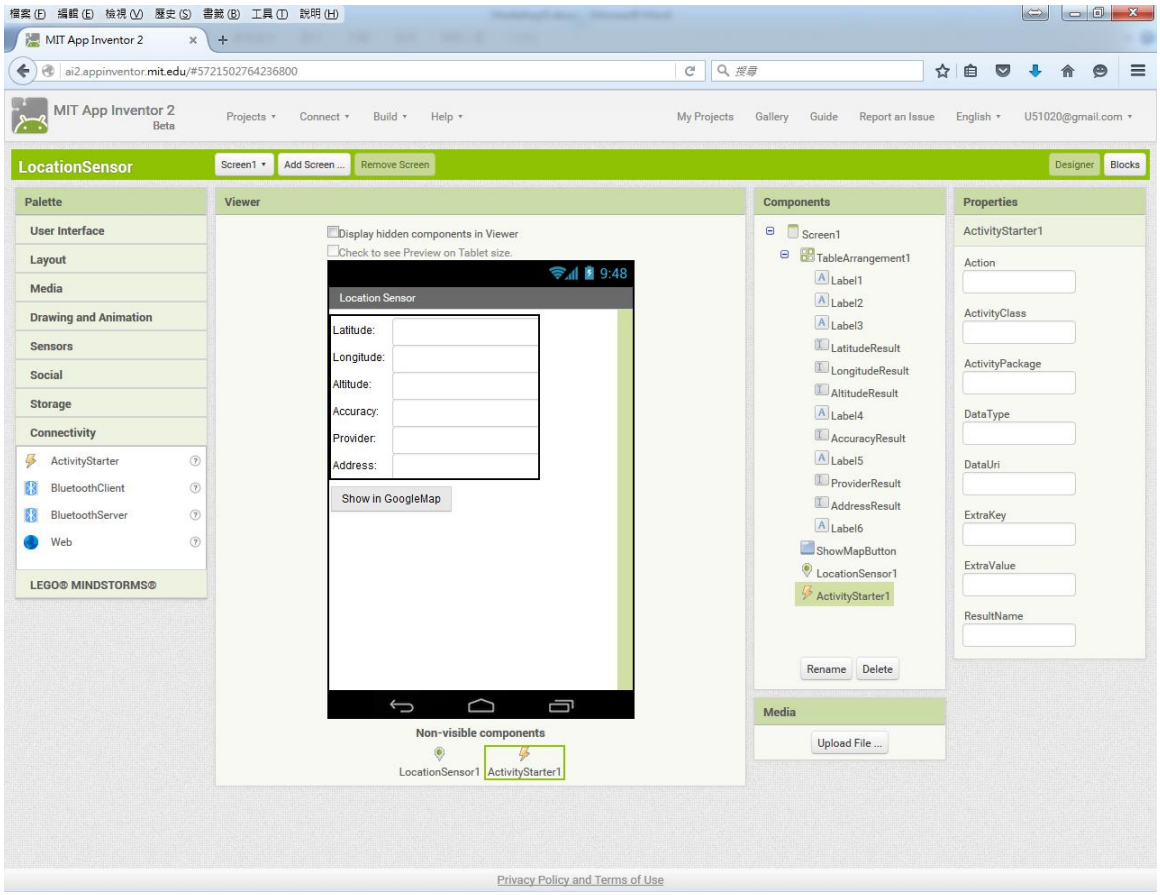
In order to function, the component must have its Enabled property set to True, and the device must have location sensing enabled through wireless networks or GPS satellites (if outdoors).

Location information might not be immediately available when an app starts. You'll have to wait a short time for a location provider to be found and used, or wait for the OnLocationChanged event

DistanceInterval is used to determines the minimum distance interval, in meters, that the sensor will try to use for sending out location updates. For example, if this is set to 5, then the sensor will fire a LocationChanged event only after 5 meters have been traversed. However, the sensor does not guarantee that an update will be received at exactly the distance interval. It may take more than 5 meters to fire an event, for instance.

3.5 Exercise: Location Sensor

3.5.1 Designer View

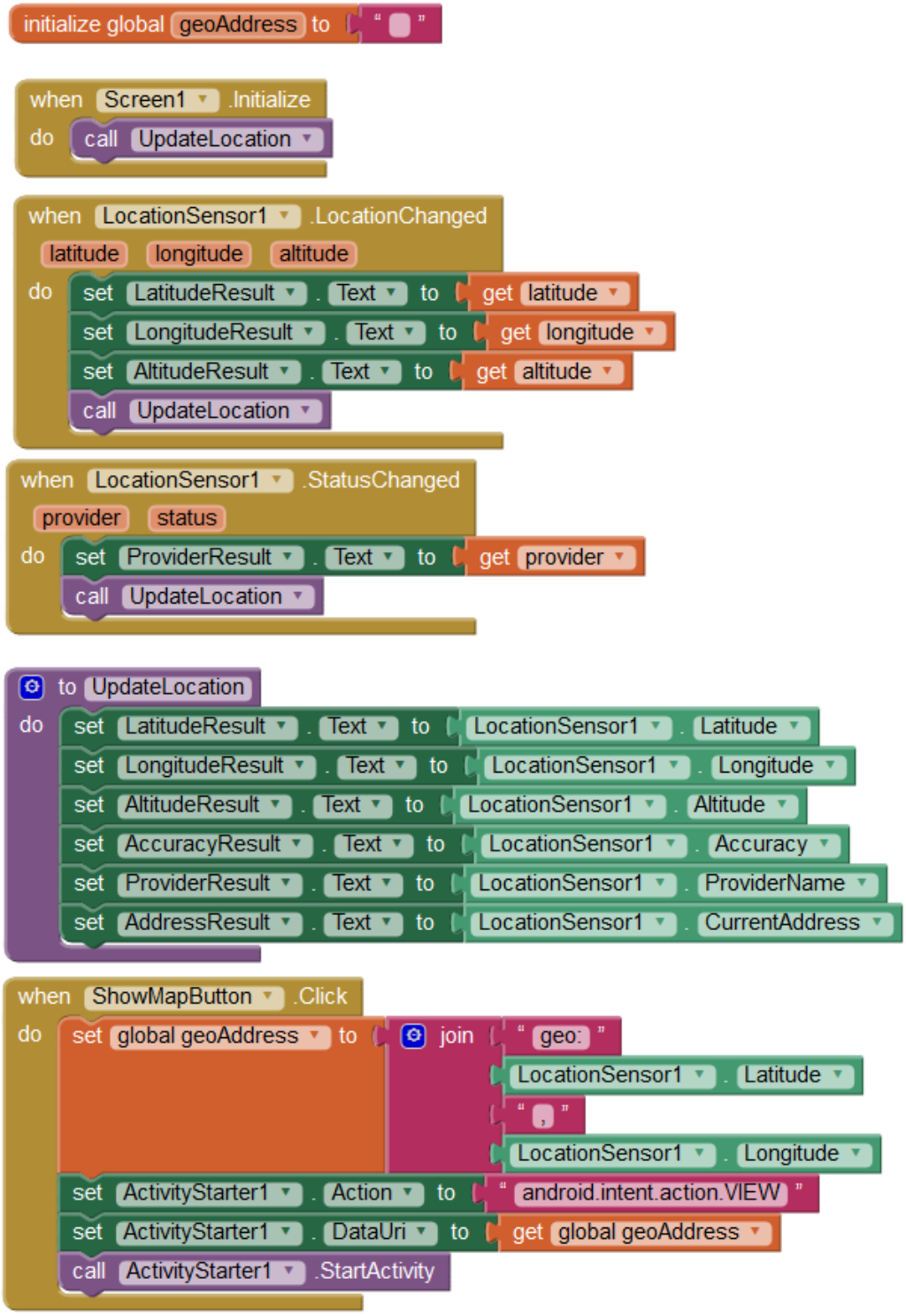


3.5.2 Components

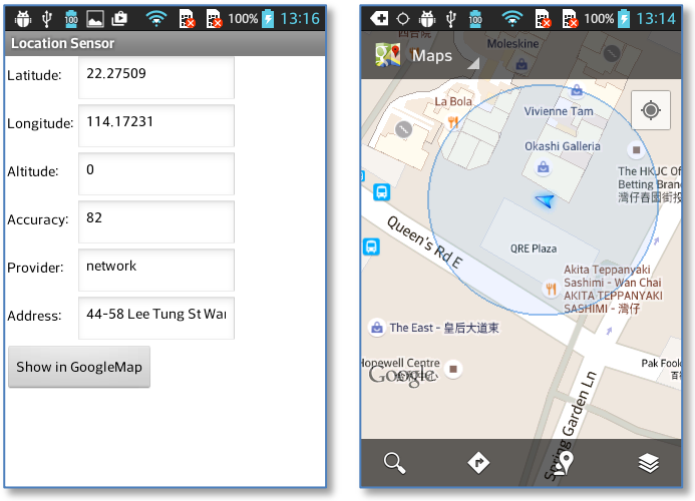
Component	Name	Properties	Remark
Screen	Screen1	Text = "**Location Sensor**"	
TableArrangement	TableArrangement1	Columns = "**2**" Rows = "**6**"	
Label	Label1	Text = "**Latitude:**"	Inside TableArrangement1
Label	Label2	Text = "**Longitude:**"	Inside TableArrangement1
Label	Label3	Text = "**Altitude:**"	Inside TableArrangement1
Label	Label4	Text = "**Accuracy:**"	Inside TableArrangement1
Label	Label5	Text = "**Provider:**"	Inside TableArrangement1
Label	Label6	Text = "**Address:**"	Inside TableArrangement1
Textbox	LatitudeResult	Hint = "**Latitude**"	Inside TableArrangement1
Textbox	LongitudeResult	Hint = "**Longitude**"	Inside TableArrangement1
Textbox	AltitudeResult	Hint = "**Altitude**"	Inside TableArrangement1
Textbox	AccuracyResult	Hint = "**Accuracy**"	Inside TableArrangement1
Textbox	ProviderResult	Hint = "**Provider**"	Inside TableArrangement1

Textbox	AddressResult	Hint = "**Address**"	Inside TableArrangement1
Button	ShowMapButton	Text = "**Show in Google Map**"	
LocationSensor	LocationSensor1		
ActivityStarter	ActivityStarter1		

3.5.3 Block Configuration



3.5.4 Sample Output



4. Interactive Sensor Game

4.1 Exercise: Lucky Monkey

4.1.1 Media Files

4.1.1.1 Image Files

banana.png

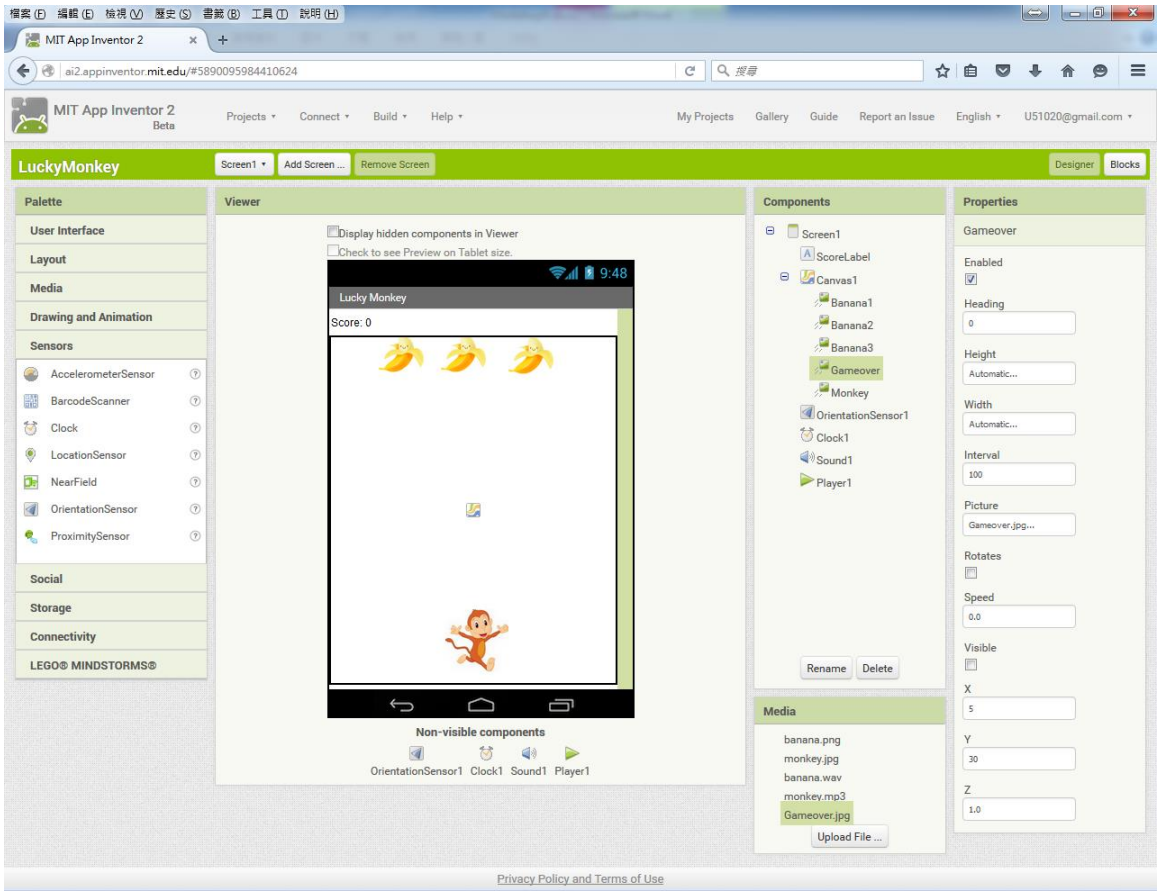
monkey.jpg

4.1.1.2 Sound Files

banana.wav

monkey.mp3

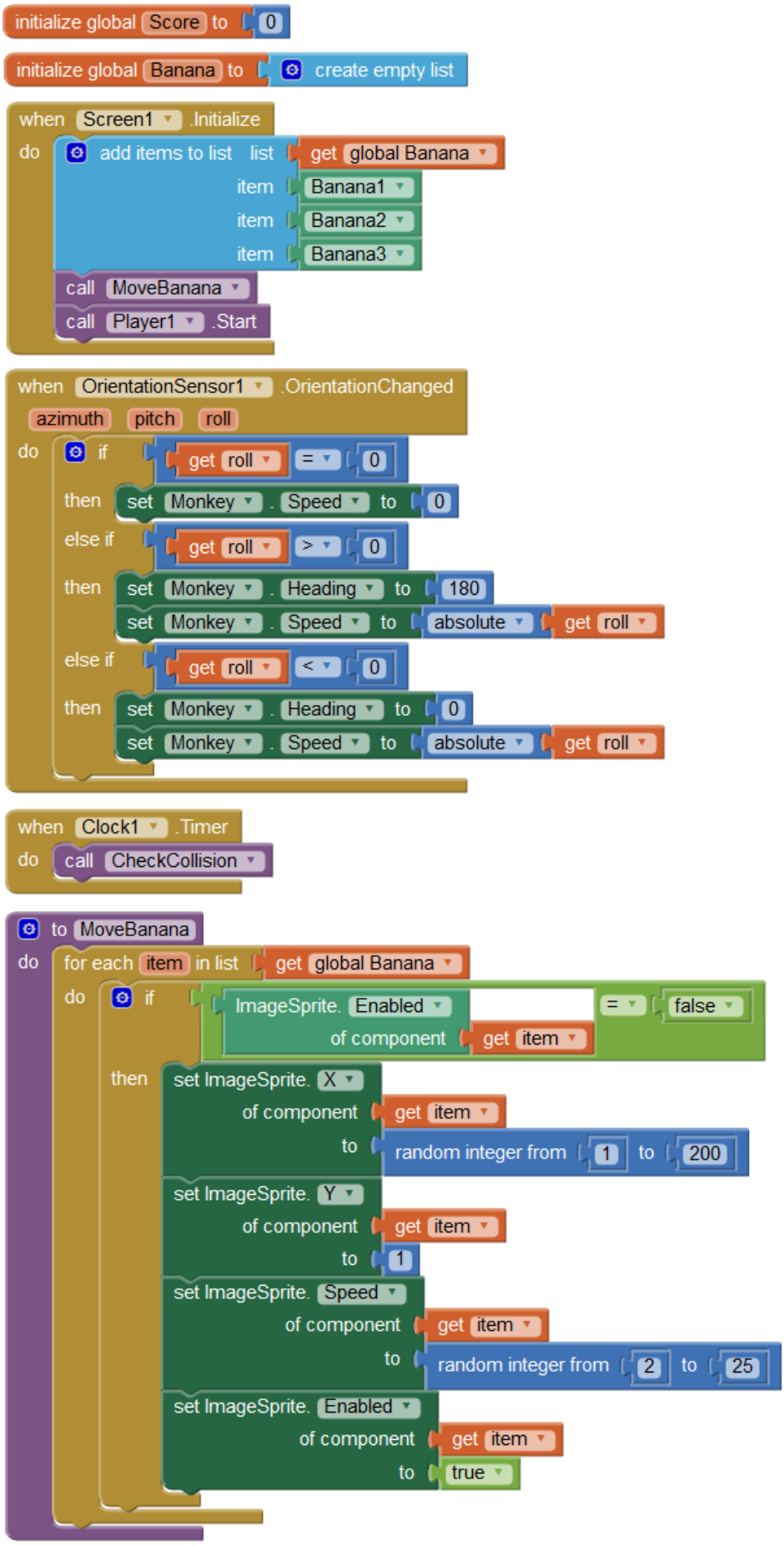
4.1.2 Designer View

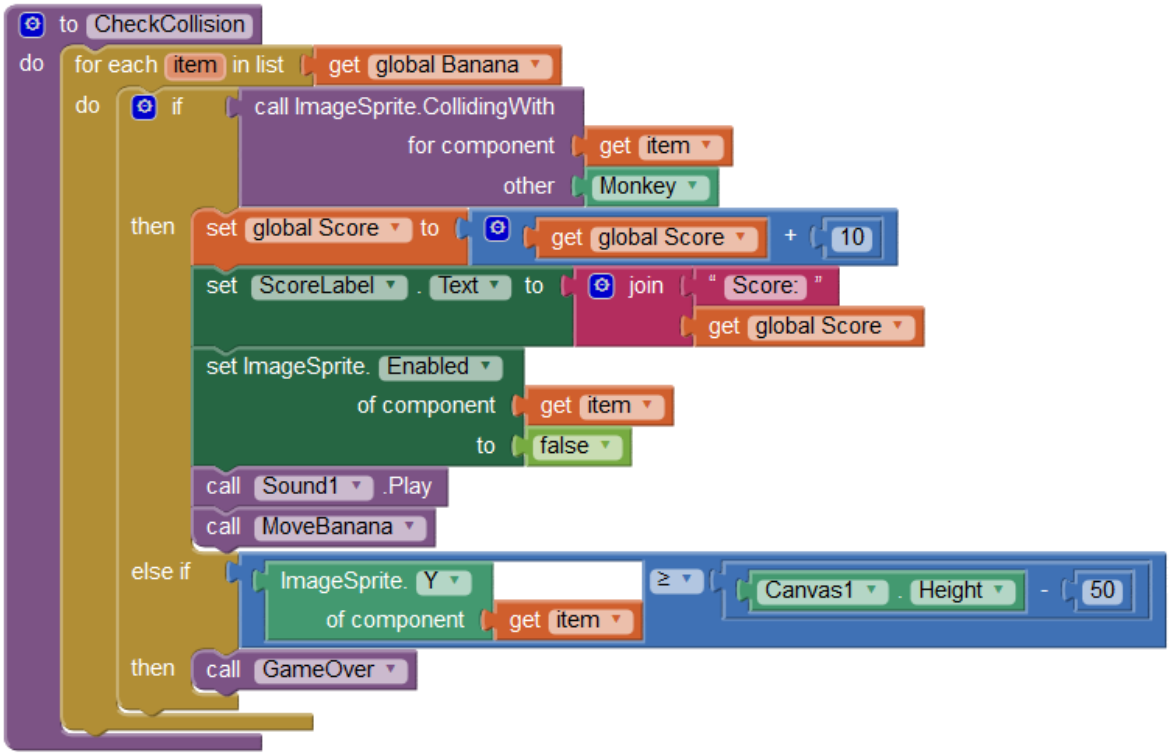


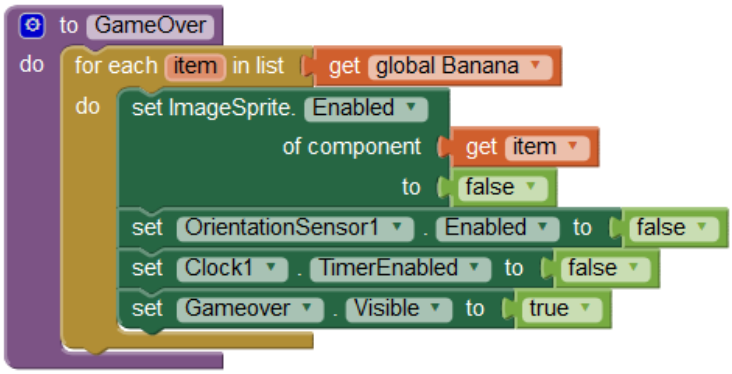
4.1.3 Components

Component	Name	Properties	Remark
Screen	Screen1	Text = "**Lucky Monkey**" ScreenOrientation = "**Portrait**"	
Label	ScoreLabel	Text = "**Score: 0**"	
Canvas	Canvas1	Height = "**Fill parent**" Width = "**Fill parent**"	
ImageSprite	Banana1	Enabled = *[Blank]* Heading = "**270**" Height = "**50 pixels**" Width = "**50 pixels**" Picture = "**banana.png**"	Inside Canvas1
ImageSprite	Banana2	Enabled = *[Blank]* Heading = "**270**" Height = "**50 pixels**" Width = "**50 pixels**" Picture = "**banana.png**"	Inside Canvas1
ImageSprite	Banana3	Enabled = *[Blank]* Heading = "**270**" Height = "**50 pixels**" Width = "**50 pixels**" Picture = "**banana.png**"	Inside Canvas1
ImageSprite	Monkey	Height = "**70 pixels**" Width = "**70 pixels**" X = "**125**" Y = "**300**"	Inside Canvas1
ImageSprite	Gameover	Picture = "**Gameover.jpg**" Visible = *[Blank]* X = "**5**" Y = "**30**"	Inside Canvas1
OrientationSensor	OrientationSensor1		
Clock	Clock1	TimeInterval = "**100**"	
Sound	Sound1	Source = "**banana.wav**"	
Player	Player1	Loop = "**X**" Source = "**monkey.mp3**"	

4.1.4 Block Configuration







4.1.5 Sample Output