



**The Abdus Salam  
International Centre for Theoretical Physics**



**2342-4**

**Scientific m-Learning**

***4 - 7 June 2012***

**Mobile Application Development using App Inventor for Android Devices**

TRIVEDI Kirankumar Rajnikant  
*Shantilal Shah Engineering College New Sidsar Campu,  
PO Vartej Bhavnagar 364001 Gujarat  
INDIA*



# Mobile Application Development using App Inventor for Android Devices

Kiran Trivedi, India

“Scientific m-learning Workshop”

# Why App Inventor?

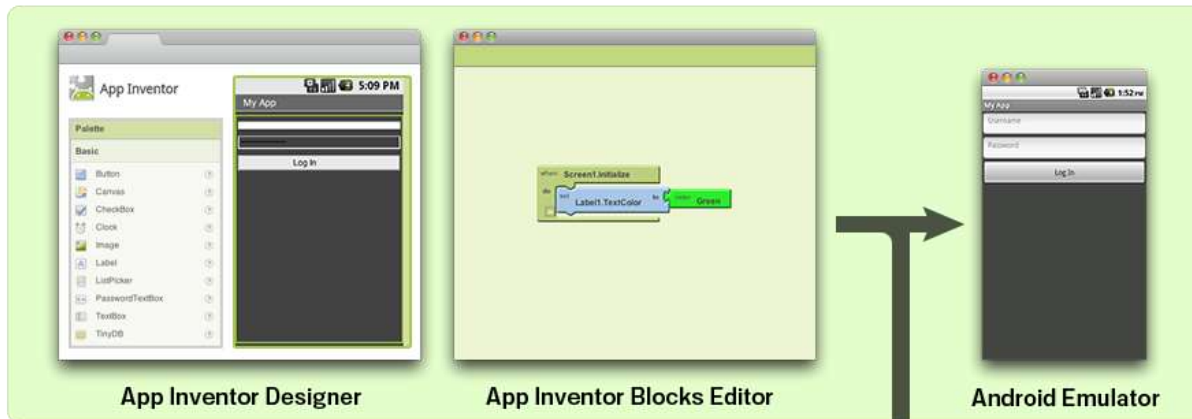
- **No syntax**
  - The blocks language eliminates the need to remember and type code
- **Everything is right in front of you**
  - The components and functions are organized into drawers. Just find, drag, and drop.
- **Events at top level**
  - "When this happens, the app does this" is the correct conceptual model.  
Down with Listeners!
- **High-level components**
  - The app inventor team has built a great library with simplicity the main goal.
- **Only some blocks plug-in**
  - You can't do things that don't make sense.
- **Concreteness**
  - You program components, not abstractions

# What is the App Inventor?

- App Inventor lets you develop applications for Android phones using a web browser and either a connected phone or emulator.
- The App Inventor servers store your work and help you keep track of your projects



Google App Inventor Servers



App Inventor Designer

App Inventor Blocks Editor

Android Emulator



Android Phone

# What you need?

- The *App Inventor Designer*, where you select the components for your app.
- The *App Inventor Blocks Editor*, where you assemble program blocks that specify how the components should behave.
- You assemble programs visually, fitting pieces together like pieces of a puzzle.

# Procedure

- Your app appears on the phone step-by-step as you add pieces to it, so you can test your work as you build.
- When you're done, you can package your app and produce a stand-alone application to install.
- If you don't have an Android phone, you can build your apps using the *Android emulator* , software that runs on your computer and behaves just like the phone.

# On the website

- The App Inventor development environment is supported for Mac OS X, GNU/Linux, and Windows operating systems, and several popular Android phone models. Applications created with App Inventor can be installed on any Android phone. (See [system requirements](#).)
- Before you can use App Inventor, you need to [set up your computer](#) and install the *App Inventor Setup* package on your computer.



# System requirements

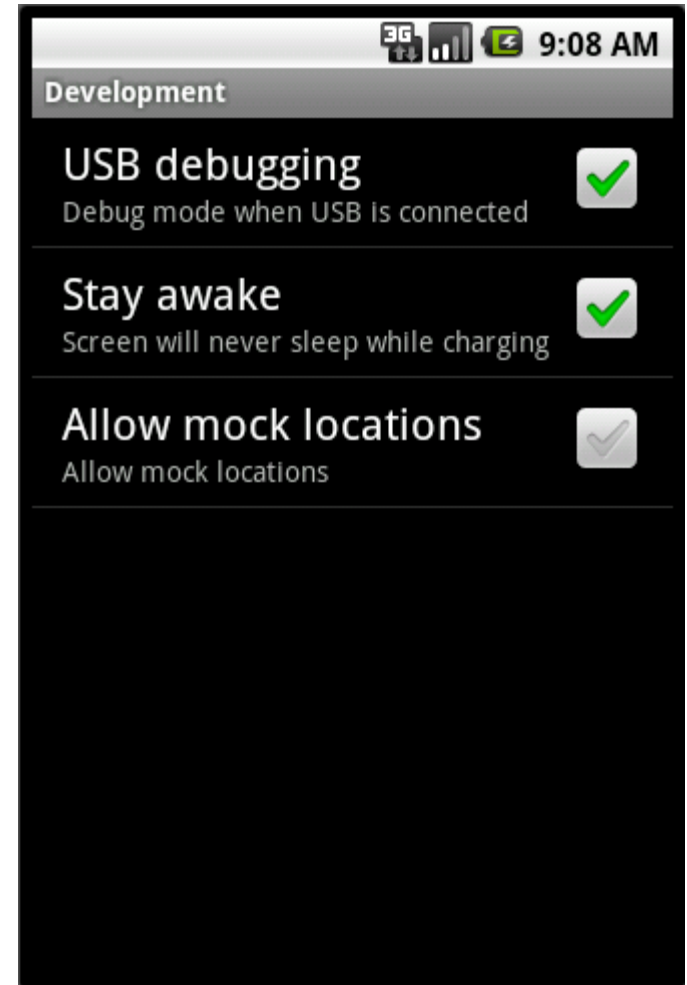
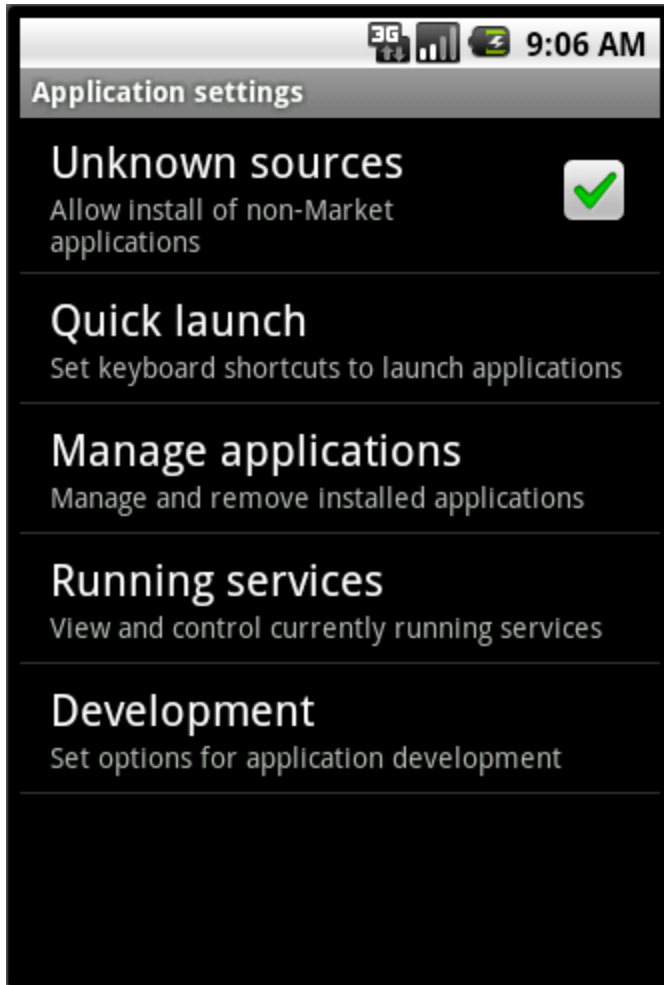
- To use App Inventor, your computer must meet the following system requirements:
- Computer and operating system
- Macintosh (with Intel processor): Mac OS X 10.5, 10.6
- Windows: Windows XP, Windows Vista, Windows 7
- GNU/Linux: Ubuntu 8+, Debian 5+
  
- Browser
- Mozilla Firefox 3.6 or higher
- Apple Safari 5.0 or higher
- Google Chrome 4.0 or higher
- Microsoft Internet Explorer 7 or higher

# Phone

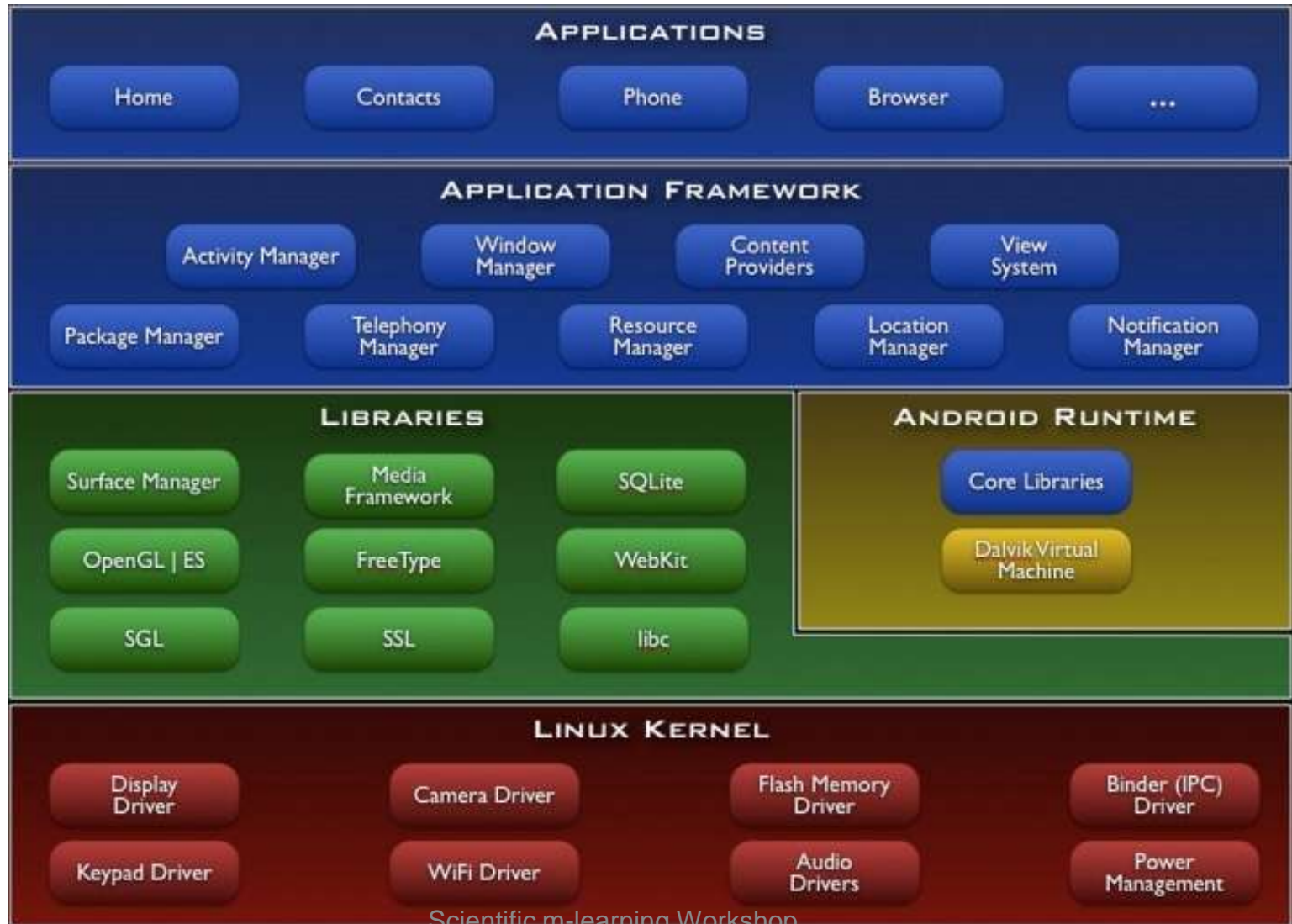
- Applications created with App Inventor can run on any Android Phone. The development environment and Setup software itself directly supports the following phones:
- Your phone must have an SD card installed, or else it won't work with App Inventor.
- App Inventor also works with many other Android phones, including models from HTC, Samsung, and Dell, but in many cases you will need to download and install additional software from the manufacturer if needed

# Set up your Android phone

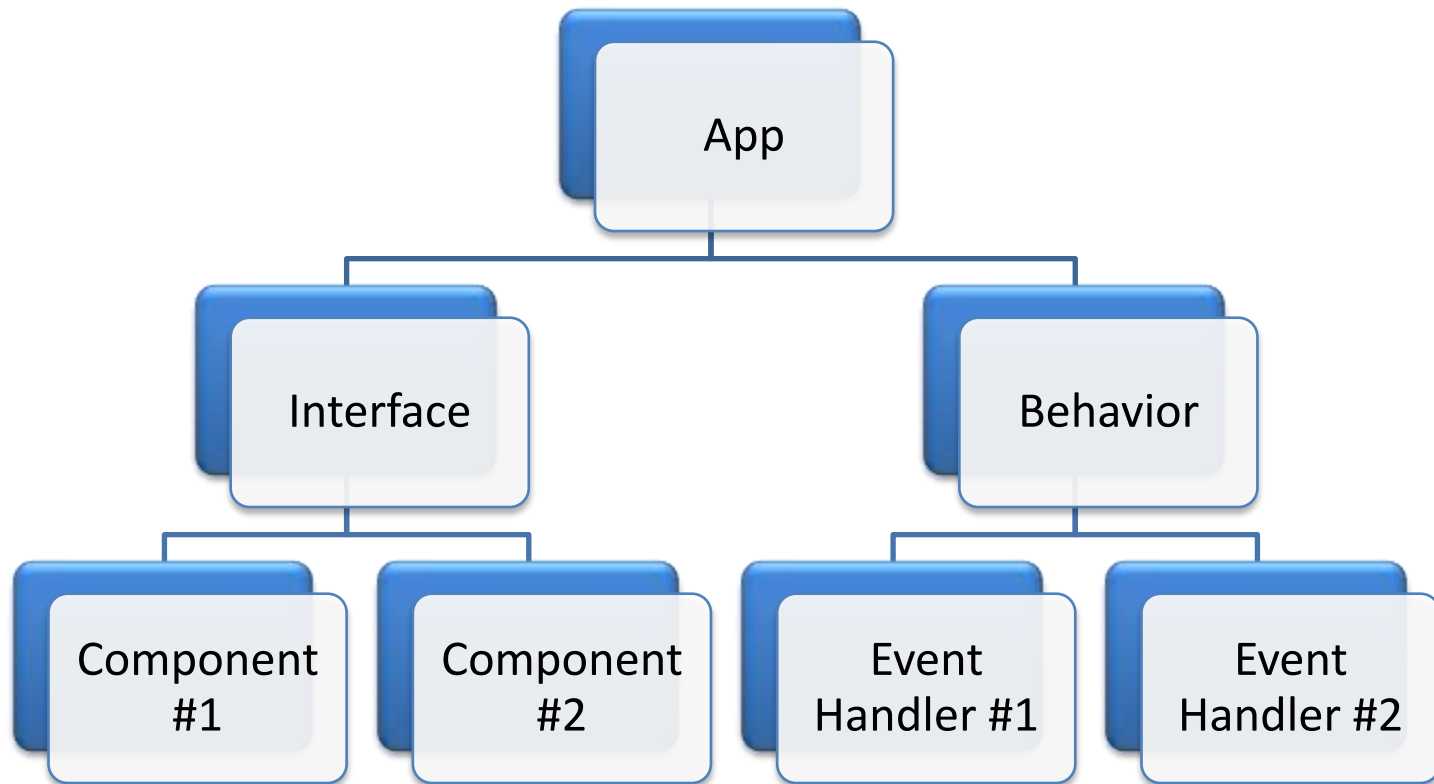
- To get your phone ready to work with App Inventor, follow these steps:
- Tap the Home button to go to your phone's Home screen.
- Tap the Menu button, then Settings, then Applications.
- If your phone has an Unknown sources setting, make sure it is checked.



# Android architecture



# App Inventor Architecture

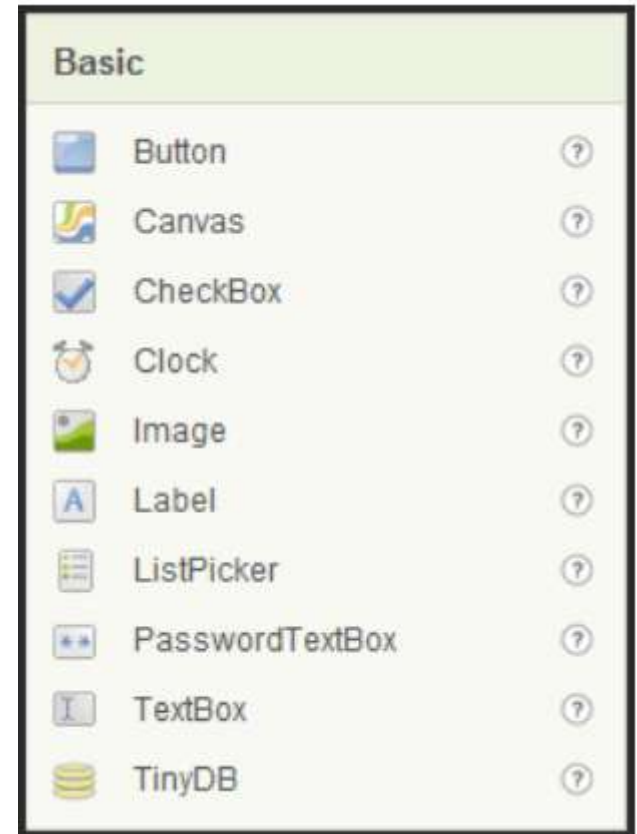


# USER INTERFACE

- Consists of Components
  - Components are same as Tools in windows forms or Web forms.
- Two Types of Components
  - Visible Components
  - Non-Visible Components

# Visible component

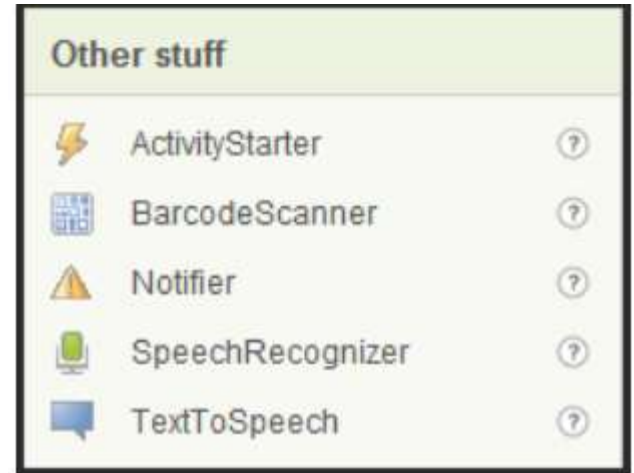
- Widely used components
- These are common components
  - Button
  - TextBox
  - Label
  - CheckBox
  - Etc.





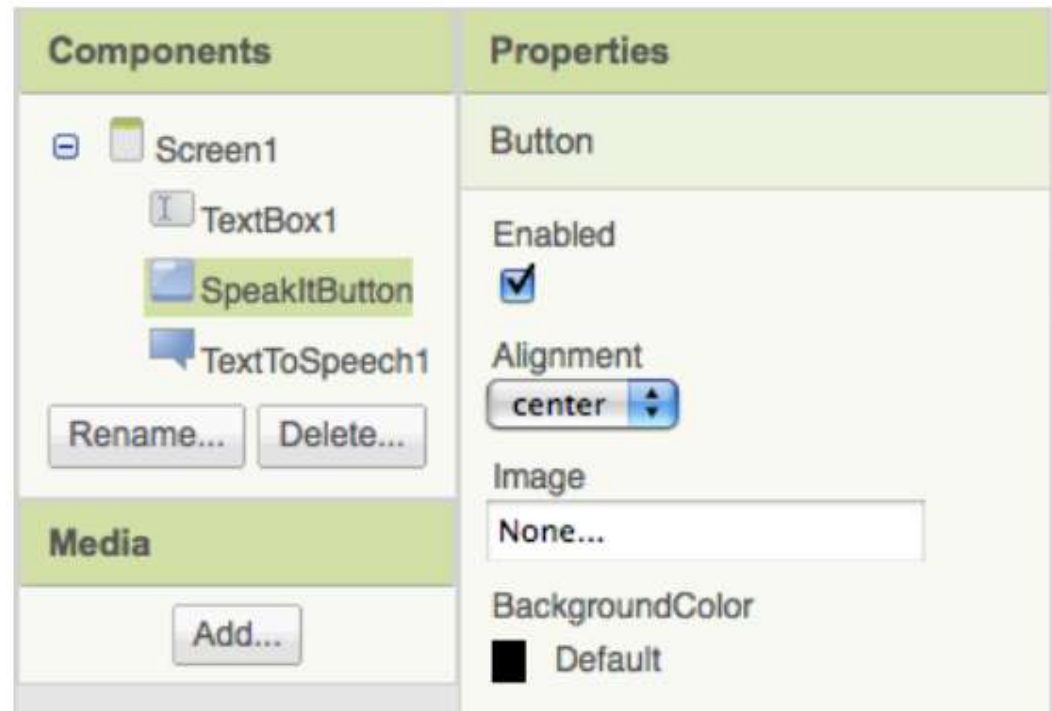
# Non visible Components

- We won't be dealing much with these controls today.
- These are controls such as timer and DataSource.
- These are not Visible on the screen but have their own functionality.
- Demo



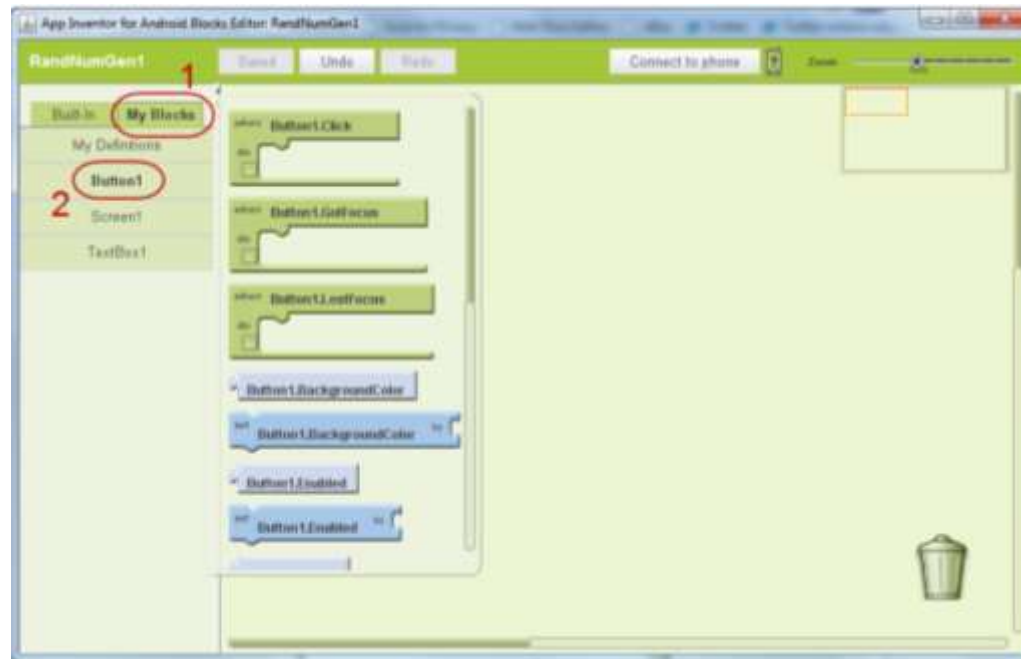
# Properties of the Components

- Select any Visible or non Visible Component.
- In the right corner we can see the properties of the item.



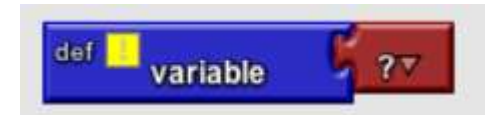
# Behaviour

- Behavior is same as the code part in windows forms forms.
- Behavior is defined using Block Editor.



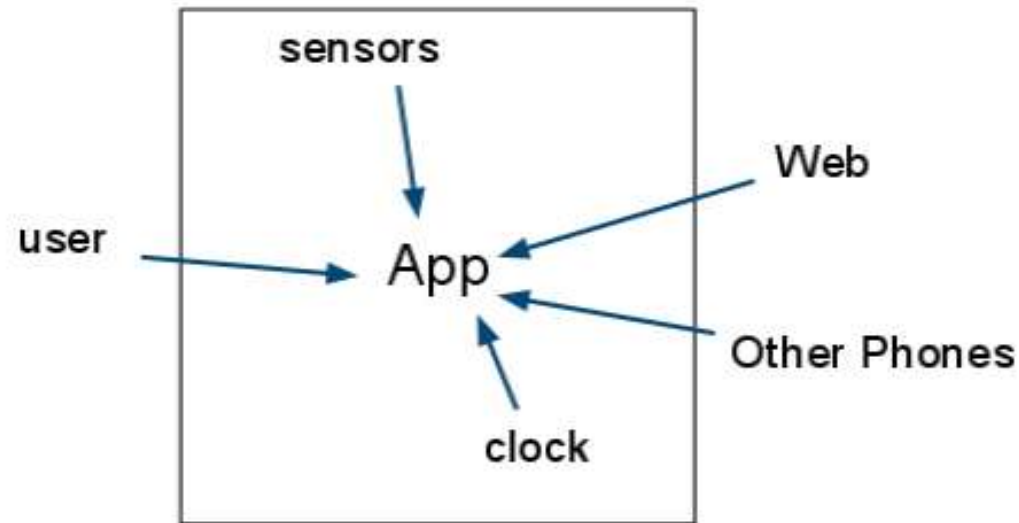
# Define a variable

- Defining:
  - Go to Built in Tab Drag Drop the Define Variable
  - Set its name by clicking on 'variable'.
  - Set its Datatype by clicking the 'Down arrow' next to '?'.
  - Assign it the value by clicking the newly appeared value and then typing in the new value.
  - Demo.



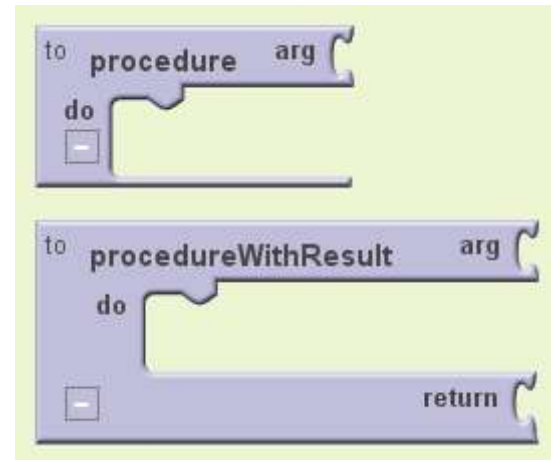
# Possible Events

Event Types	Example
User-initiated Events	Button click
Initialization Events	At App launch
Timer Events	After 1 sec do something
External Events	Receive a text



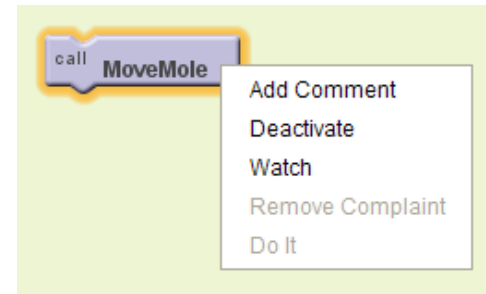
# Define a function

- Set of activities grouped together.
- Can send in Multiple Arguments
- Two Types of functions
  - With return values.
  - With out return values.



# Test and Debug

- **Deactivate**
  - Right Click on a block and choose Deactivate.
  - Choosing *Deactivate* from the block menu will keep the block from becoming part of the app when you package it.
  - Selecting *Activate* restores the deactivated block.
- **Collapsing blocks**
  - If your app has many blocks, they won't all fit on the screen at once.
  - Use *block collapsing*.
  - Click on the minus sign at the lower left of any block.
  - Only the title will be visible.
  - Click on the plus sign to restore the block to full visibility.



# Emulator

- **Starting the Emulator**
  - You don't need to download any additional software to use the emulator.
  - It was included with the software you already downloaded as part of the App Inventor Extras Package.
  - Navigate to the directory where the App Inventor Extras software was installed, locate the folder called `commands-for-appinventor`
  - Run the command `'run-emulator'`.



# Project upload/download

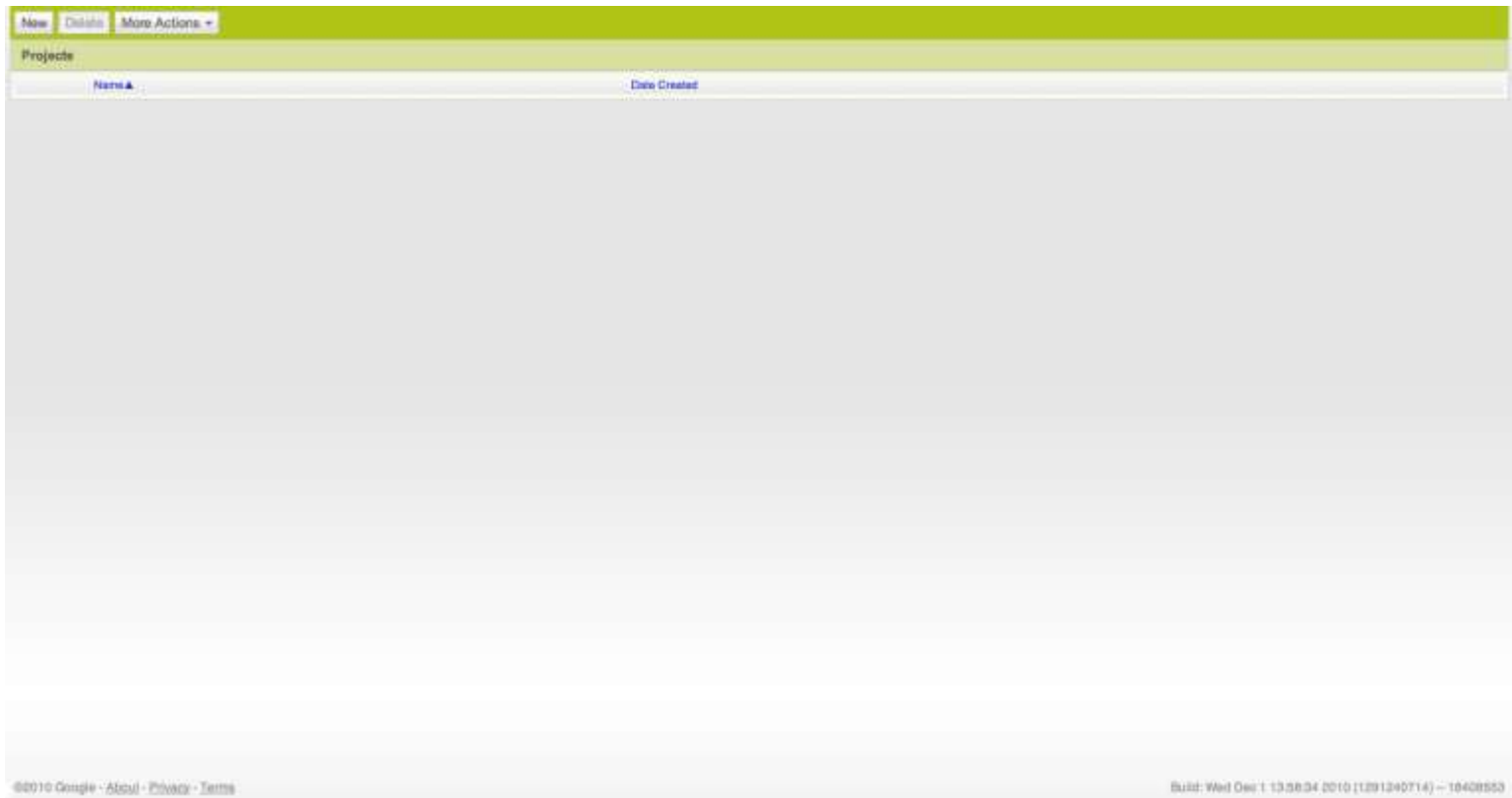
- **Download Source**

- Go to the *My Projects* page,
- Select a project,
- Then choose *More Actions | Download Source*.
- This will create a zip file which you can share with others.











- **Upload Source**






- To upload a project, go to *My Projects*,
- Choose *More Actions | Upload Source*,
- Choose the zip file previously downloaded from App Inventor.



# First Time Login









# Palette




ICTP		
Palette		
Basic		
	Button	<a href="#">?</a>
	Canvas	<a href="#">?</a>
	CheckBox	<a href="#">?</a>
	Clock	<a href="#">?</a>
	Image	<a href="#">?</a>
	Label	<a href="#">?</a>
	ListPicker	<a href="#">?</a>
	PasswordTextBox	<a href="#">?</a>
	TextBox	<a href="#">?</a>
	TinyDB	<a href="#">?</a>




ICTP		
Palette		
Basic		
Media		
	Camera	<a href="#">?</a>
	ImagePicker	<a href="#">?</a>
	Player	<a href="#">?</a>
	Sound	<a href="#">?</a>
	VideoPlayer	<a href="#">?</a>

ICTP		
Palette		
Basic		
Media		
Animation		
	Ball	<a href="#">?</a>
	ImageSprite	<a href="#">?</a>










ICTP		
Palette		
Basic		
Media		
Animation		
Social		
	ContactPicker	<a href="#">?</a>
	EmailPicker	<a href="#">?</a>
	PhoneCall	<a href="#">?</a>
	PhoneNumberPicker	<a href="#">?</a>
	Texting	<a href="#">?</a>
	Twitter	<a href="#">?</a>

# Pallatte

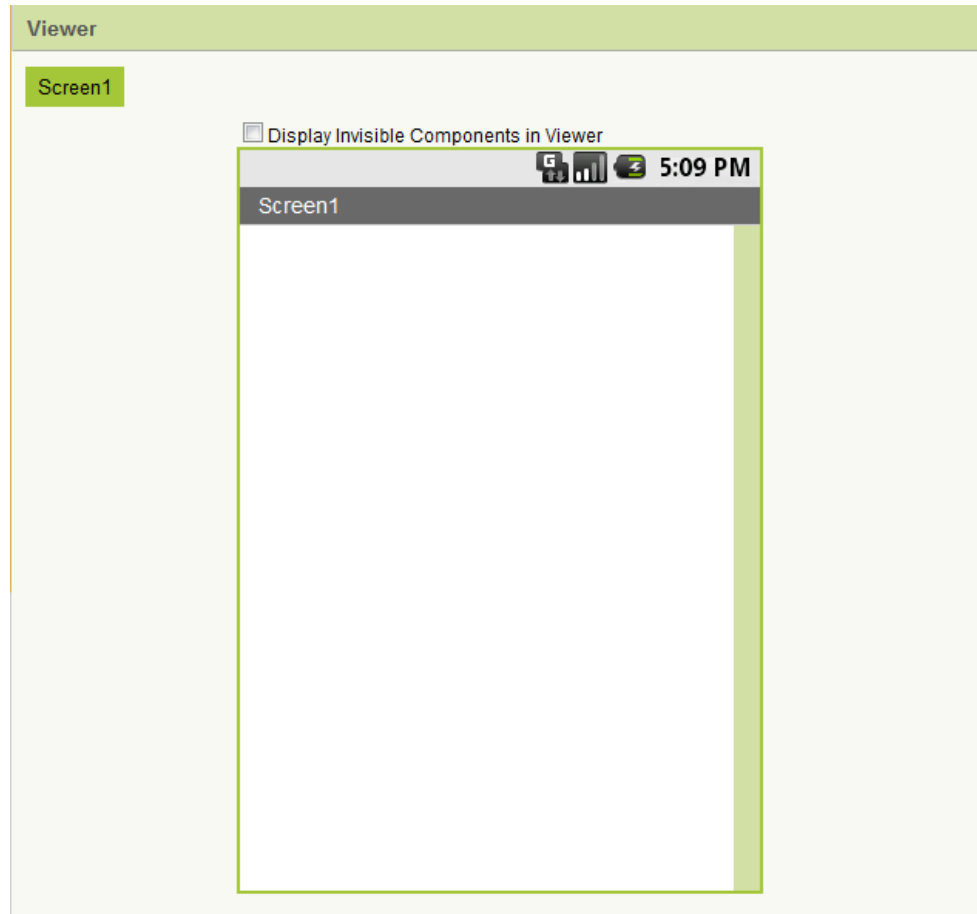
ICTP		
Palette		
Basic		
Media		
Animation		
Social		
Sensors		
 AccelerometerSensor		<a href="#">?</a>
 LocationSensor		<a href="#">?</a>
 OrientationSensor		<a href="#">?</a>

ICTP		
Palette		
Basic		
Media		
Animation		
Social		
Sensors		
Screen Arrangement		
 HorizontalArrangement		<a href="#">?</a>
 TableArrangement		<a href="#">?</a>
 VerticalArrangement		<a href="#">?</a>

Not ready for prime time		
 FusiontablesControl		<a href="#">?</a>
 GameClient		<a href="#">?</a>
 SoundRecorder		<a href="#">?</a>
 Voting		<a href="#">?</a>
 WebView		<a href="#">?</a>

Other stuff		
 ActivityStarter		<a href="#">?</a>
 BarcodeScanner		<a href="#">?</a>
 BluetoothClient		<a href="#">?</a>
 BluetoothServer		<a href="#">?</a>
 Notifier		<a href="#">?</a>
 SpeechRecognizer		<a href="#">?</a>
 TextToSpeech		<a href="#">?</a>
 TinyWebDB		<a href="#">?</a>
 Web		<a href="#">?</a>

# Viewer / Workspace



# Component Properties

krtrivedi@gmail.com | [Sign out](#)

My Projects Design Learn (Debugging) Welcome to the App Inventor beta preview release. Be sure to check the list of [known issues](#).

Save

Save As

Checkpoint

Add Screen

Remove Screen

Open the Blocks Editor

Package for Phone ▾

The screenshot displays the App Inventor interface. On the left, the 'Components' panel shows a single component named 'Screen1'. Below it are 'Rename' and 'Delete' buttons. At the bottom of the components panel is a 'Media' section with an 'Add...' button. On the right, the 'Properties' panel is open, showing the following settings:

- BackgroundColor:  White
- BackgroundImage:
- Icon:
- ScreenOrientation:  ▾
- Scrollable:
- Title:

# First Example !

The screenshot displays the MIT App Inventor web interface. At the top left, the MIT App Inventor logo is visible with a 'BETA' badge. The top navigation bar includes 'My Projects', 'Design', 'Learn', and '(Debugging)'. A welcome message states: 'Welcome to the App Inventor beta preview release. Be sure to check the list of [known issues](#)'. The user's email 'krtrivedi@gmail.com' and a 'Sign out' link are in the top right.

The main workspace is titled 'Hello\_Lion' and contains a toolbar with 'Save', 'Save As', 'Checkpoint', 'Add Screen', and 'Remove Screen' buttons. On the right side of the toolbar, it indicates 'Blocks Editor is open' and 'Package for Phone'.

The interface is divided into four main panels:

- Palette:** A list of components categorized into Basic, Media, Animation, Social, Sensors, Screen Arrangement, LEGO® MINDSTORMS®, Other stuff, Not ready for prime time, and Old stuff. The 'Basic' category is expanded, showing components like Button, Canvas, CheckBox, Clock, Image, Label, ListPicker, PasswordTextBox, TextBox, and TinyDB.
- Viewer:** A central area showing a mobile phone simulation. The screen displays a status bar with signal strength, battery, and time (5:09 PM). Below the status bar is a label 'm-learning@ ICTP @@ Lion of Gujarat' and a large image of a lion's face. Underneath the image is a blue button labeled 'Touch the Lion'. Below the button is another label 'Shake the Phone to Roar Lion'. At the bottom of the screen, there is a paragraph of text: 'This application is for demo at "Scientific m-learning" workshop @ ICTP, Trieste Italy. Kiran Trivedi, Associate Professor, India. When a user touches the lion or shakes the phone, the lion will roar and the phone will vibrate'. Below the viewer, there is a 'Non-visible components' section showing 'Sound1' and 'AccelerometerSensor1'.
- Components:** A tree view showing the hierarchy of components on the screen. It starts with 'Screen1', which contains 'HorizontalArrangement1' (containing 'Button1'), 'HorizontalArrangement4', 'HorizontalArrangement2' (containing 'Label1'), 'HorizontalArrangement5', 'HorizontalArrangement3' (containing 'Label2'), 'Label3', 'Sound1', and 'AccelerometerSensor1'. 'Rename' and 'Delete' buttons are visible at the bottom of this panel.
- Properties:** A panel on the right showing the properties for the selected component, 'Screen1'. Properties include 'BackgroundColor' (set to 'White'), 'BackgroundImage' (set to 'None...'), 'Icon' (set to 'None...'), 'ScreenOrientation' (set to 'Unspecified'), 'Scrollable' (checked), and 'Title' (set to 'm-learning@ ICTP @@ L').

At the bottom of the interface, there is a 'Media' panel showing 'images.jpg' and 'lion.wav' with an 'Add...' button.

# My Blocks

The image shows a screenshot of the 'My Blocks' interface in an IDE. On the left, there is a sidebar with a tabbed interface. The 'My Blocks' tab is selected, showing a list of blocks under the heading 'My Definitions'. The blocks listed are: AccelerometerSensor1, Button1, HorizontalArrangement1, HorizontalArrangement2, HorizontalArrangement3, HorizontalArrangement4, HorizontalArrangement5, Label1, Label2, Label3, Screen1, and Sound1. The 'Button1' block is highlighted. On the right, the workspace shows a vertical stack of blocks. The top four blocks are 'when' blocks: 'when Button1.Click', 'when Button1.GotFocus', 'when Button1.LongClick', and 'when Button1.LostFocus'. Each 'when' block has a 'do' slot. Below these are several property blocks: 'Button1.BackgroundColor', 'set Button1.BackgroundColor to', 'Button1.Enabled', 'set Button1.Enabled to', and 'Button1.Height'.





Built-In My Blocks Advanced

My Definitions

**AccelerometerSensor1**

Button1

HorizontalArrangement1

HorizontalArrangement2

HorizontalArrangement3

HorizontalArrangement4

HorizontalArrangement5

Label1

Label2

Label3

Screen1

Sound1

when AccelerometerSensor1.AccelerationChanged xAccel yAccel zAccel

do

when AccelerometerSensor1.Shaking

do

AccelerometerSensor1.Available

AccelerometerSensor1.Enabled

set AccelerometerSensor1.Enabled to

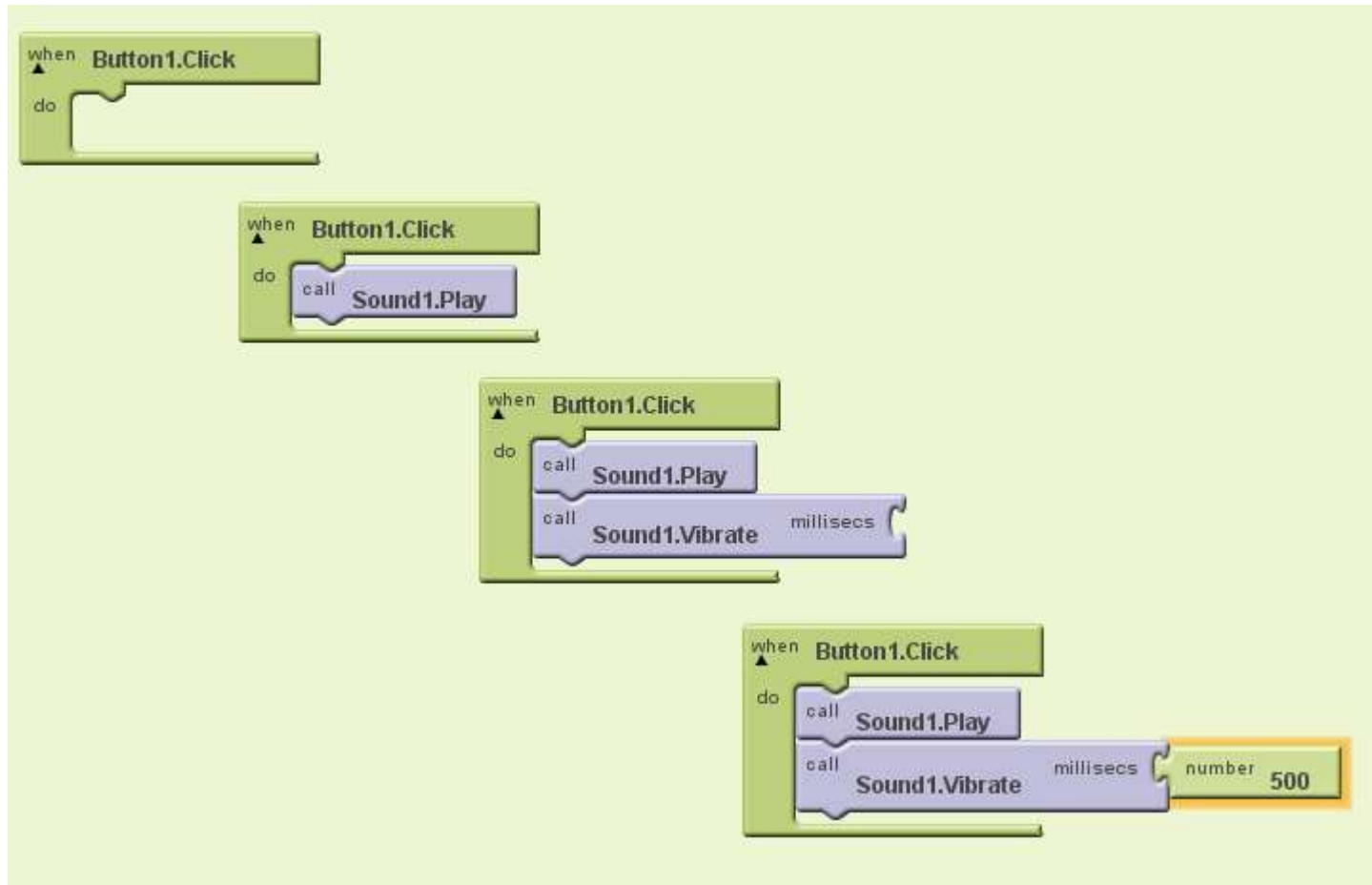
AccelerometerSensor1.XAccel

AccelerometerSensor1.YAccel

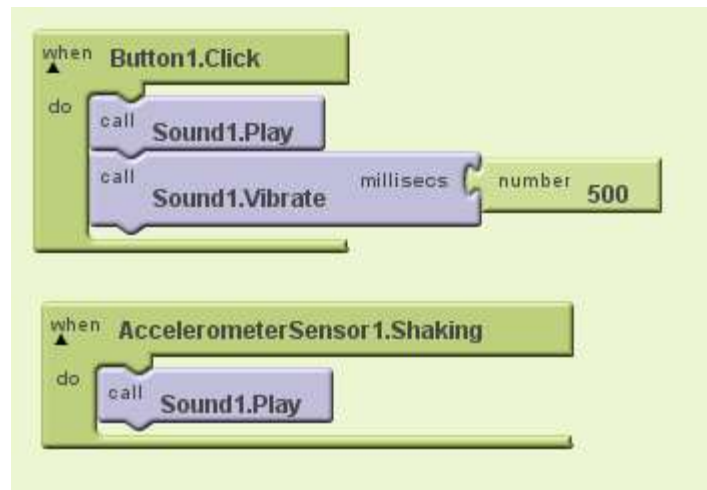
AccelerometerSensor1.ZAccel

component AccelerometerSensor1

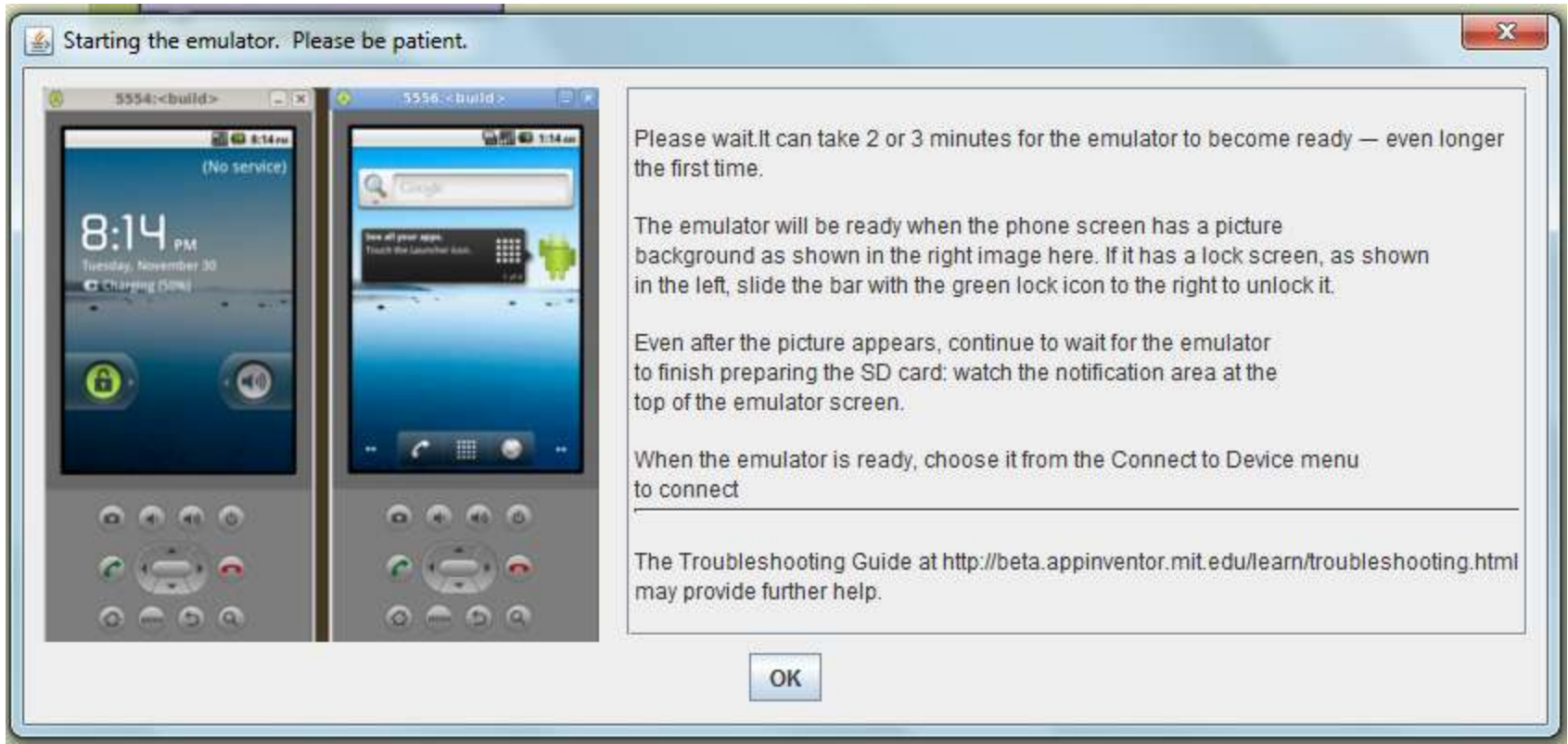
# Step by Step Block Creation



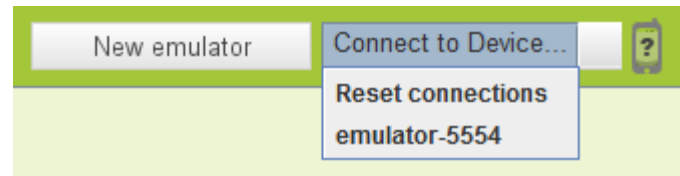
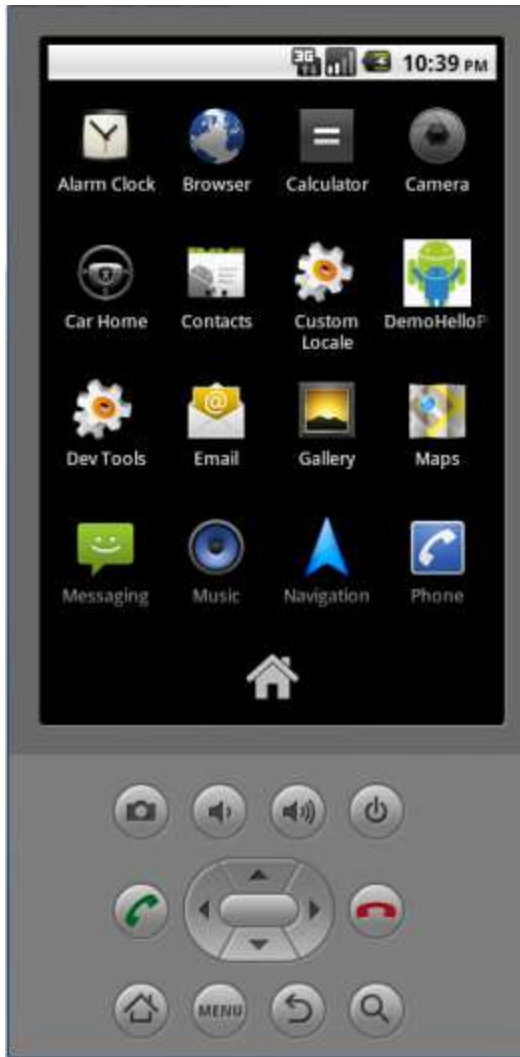
# Setting up Blocks for Hello\_Lion



# Running an Emulator

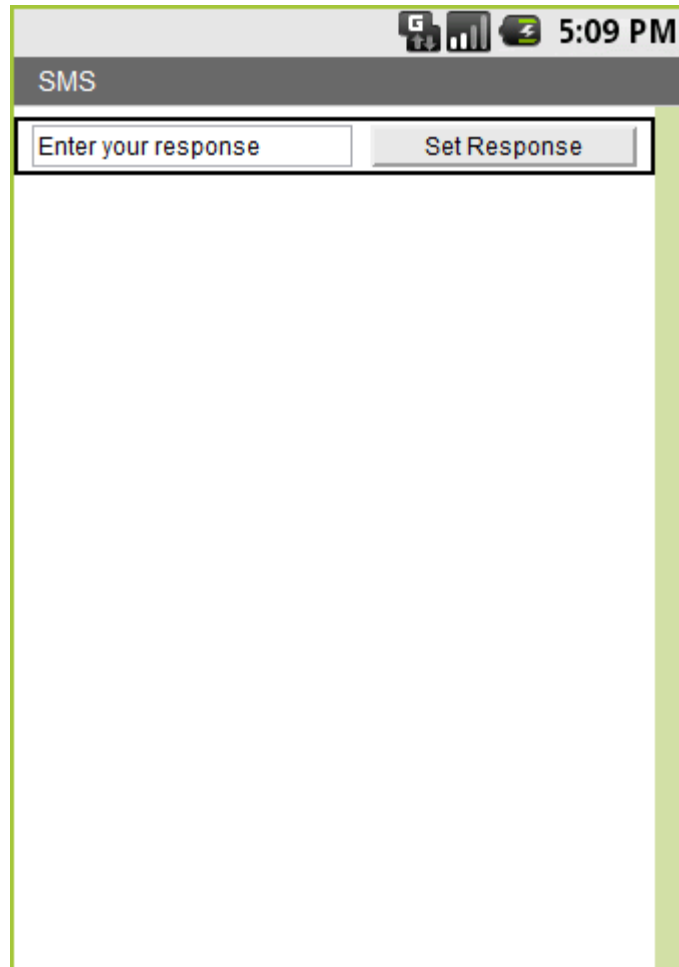


# Connect to Emulator



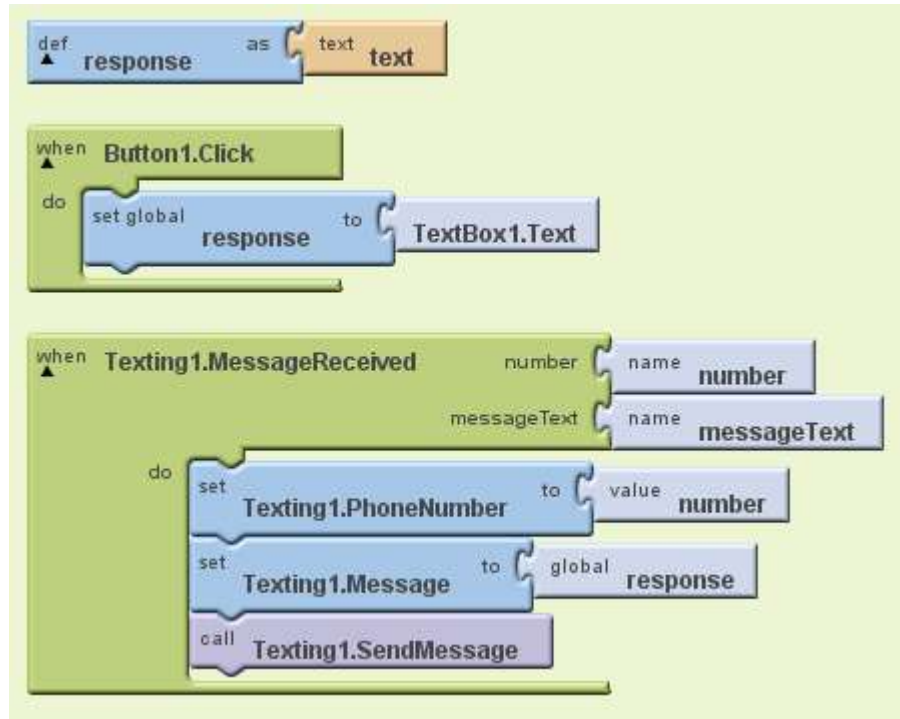
# Example: SMS Auto Responder

# UI





# Basic Auto Responder



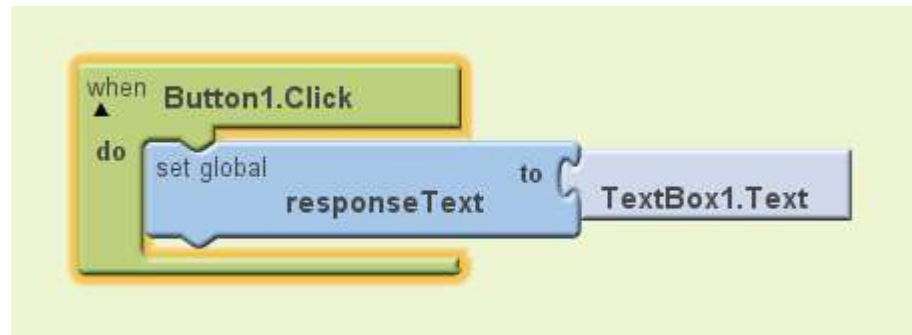
# Define the Variable



A Scratch code block for defining a variable. The block is blue and contains the text "def", a small black triangle icon, and "responseText". It is connected to an orange text block containing "text" and "Hi, I m Busy.".

```
def ▲ responseText as text Hi, I m Busy.
```

# Add Click Event to Button for changing the response text

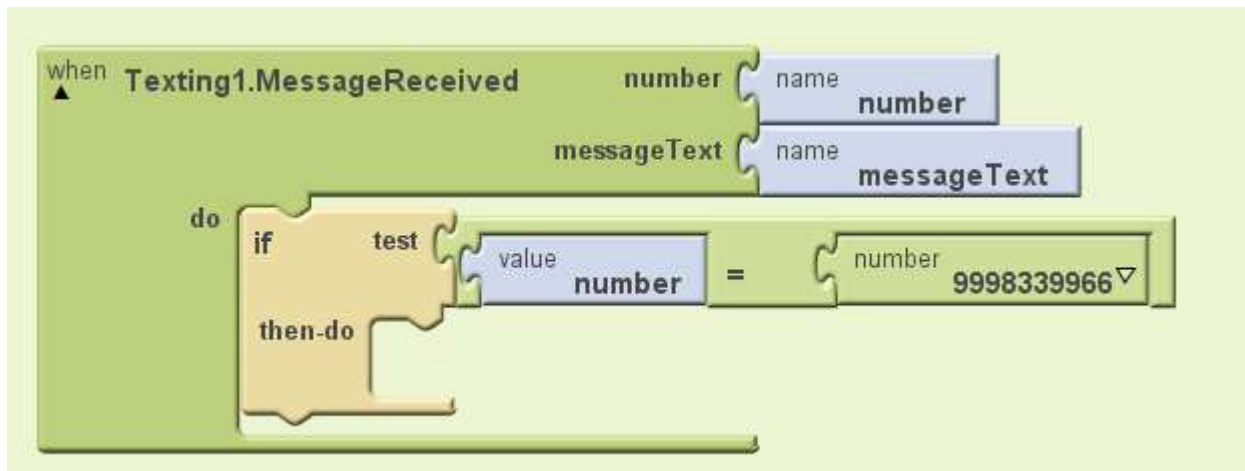


# Add Message Received Event

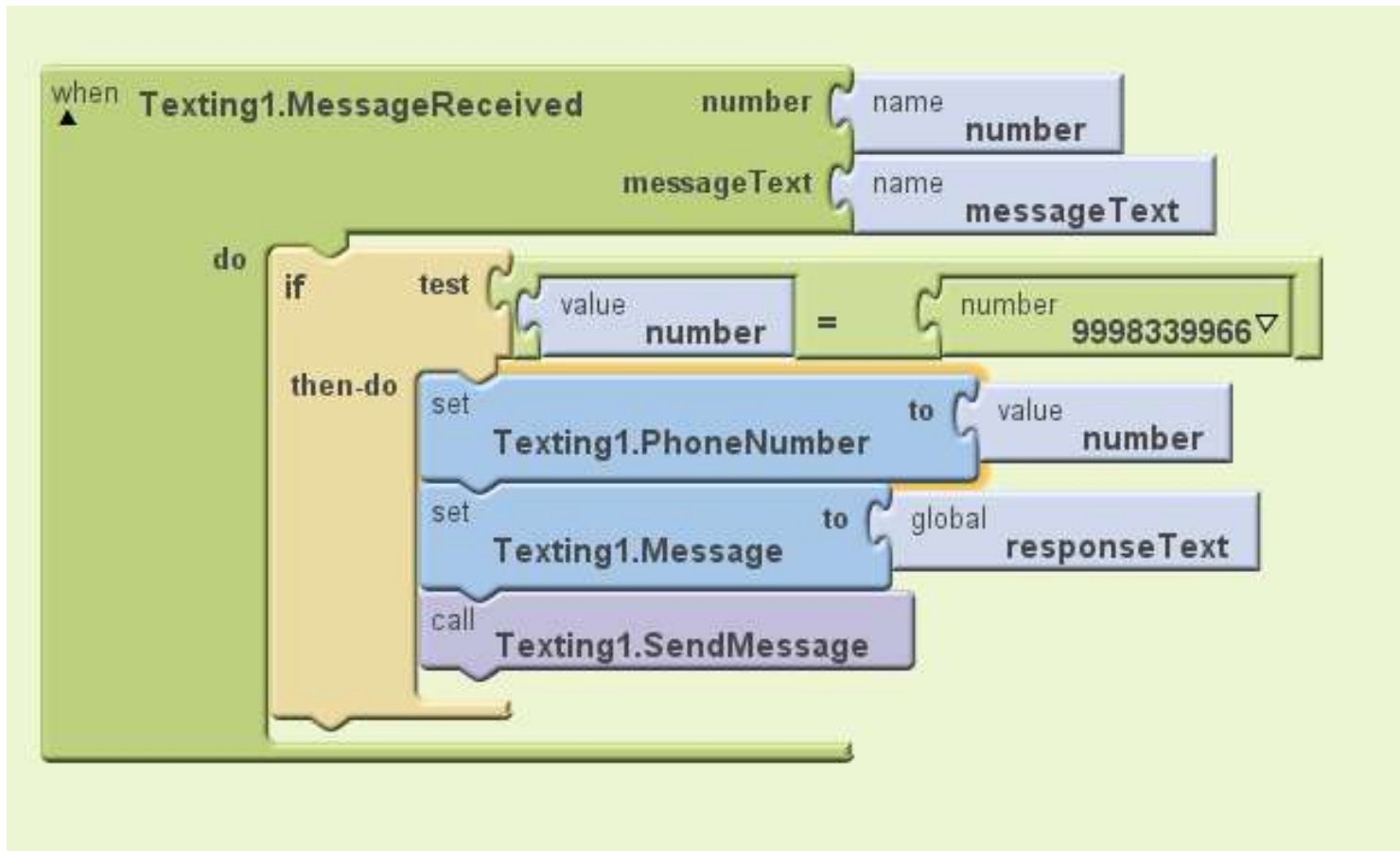


# Add If Else from built in control block

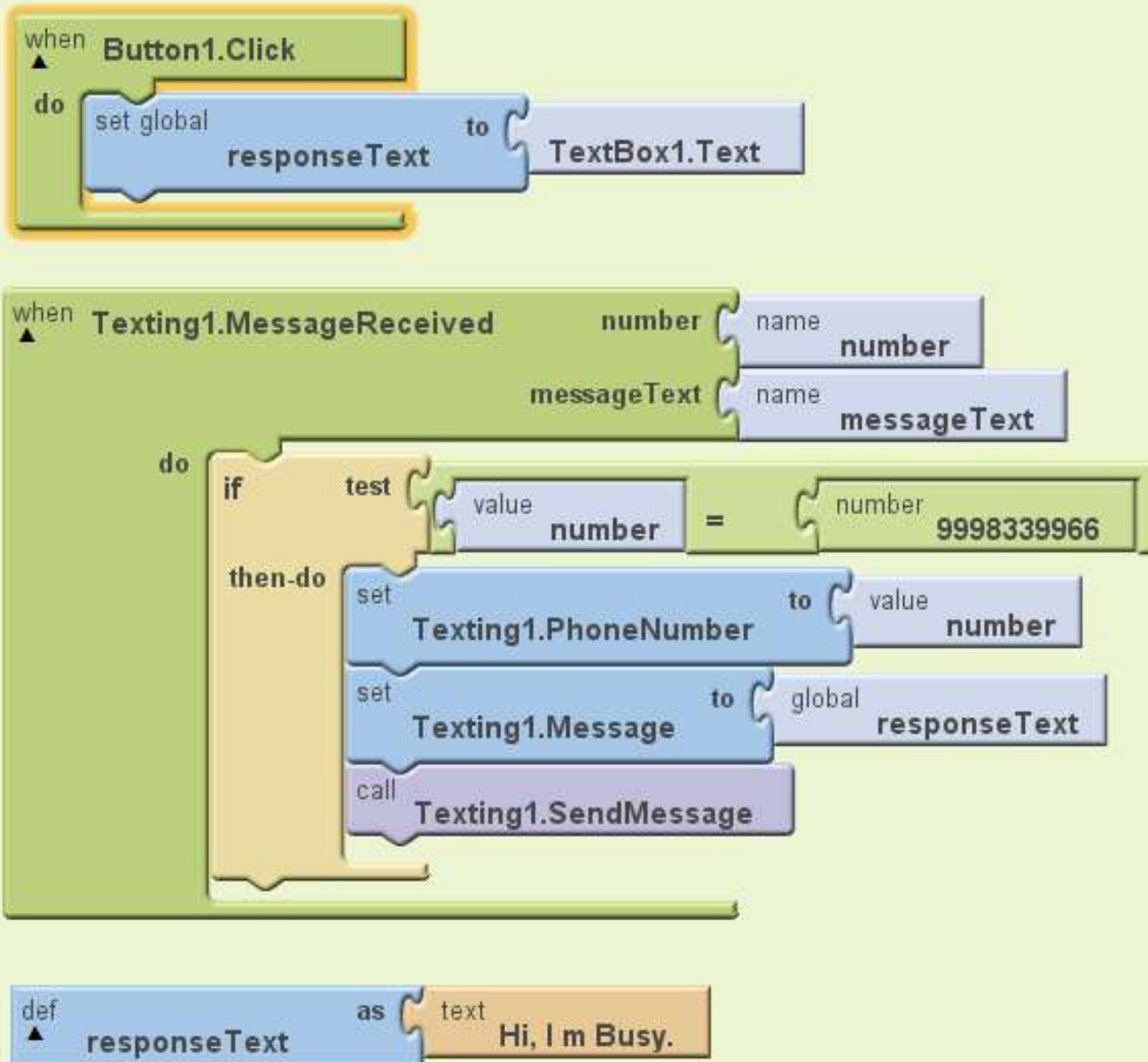
- This app will response to a specific number



# Set the SMS text and phone number for sending the SMS



# Complete Application

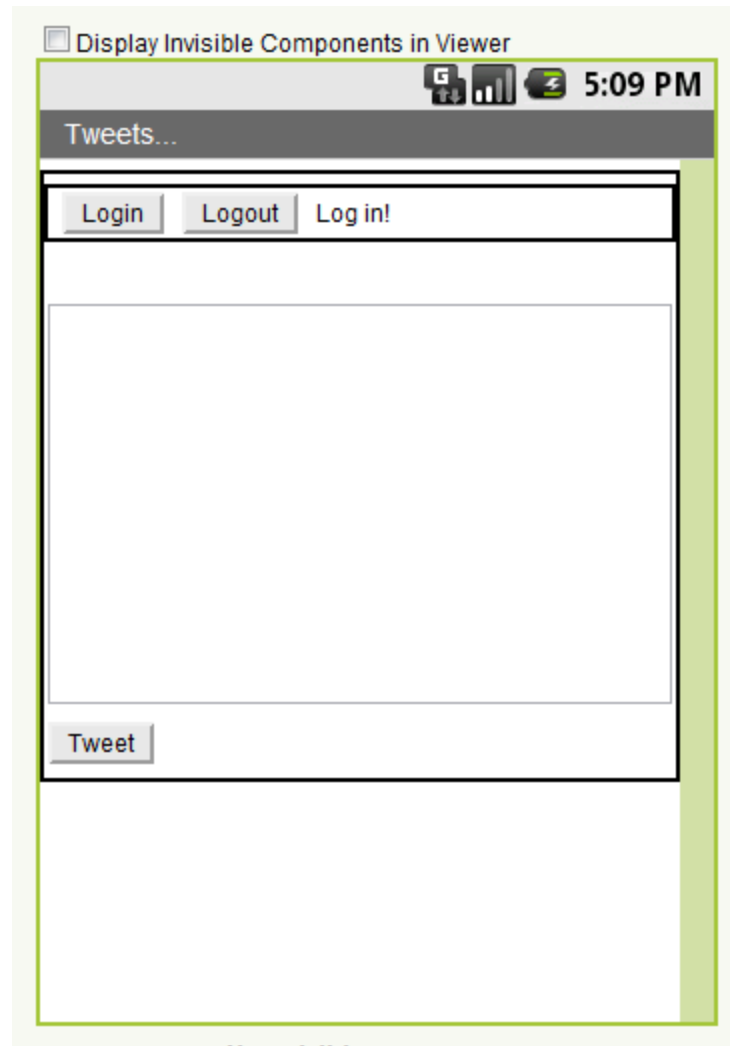


# Now Let us make a Workshop Twitter





# The UI for the Twitter App



# Login Button Definition

The screenshot displays a software development environment with two main panes. The left pane, titled "My Blocks", lists various components: My Definitions, ButtonLogin, ButtonLogout, HorizontalArrangement1, Label1 username, LabelStatus, Screen1, Tweet\_Button, Tweet\_Text, Twitter1, and VerticalArrangement1. The right pane shows the definition for the "ButtonLogin" component, including event handlers and property settings.

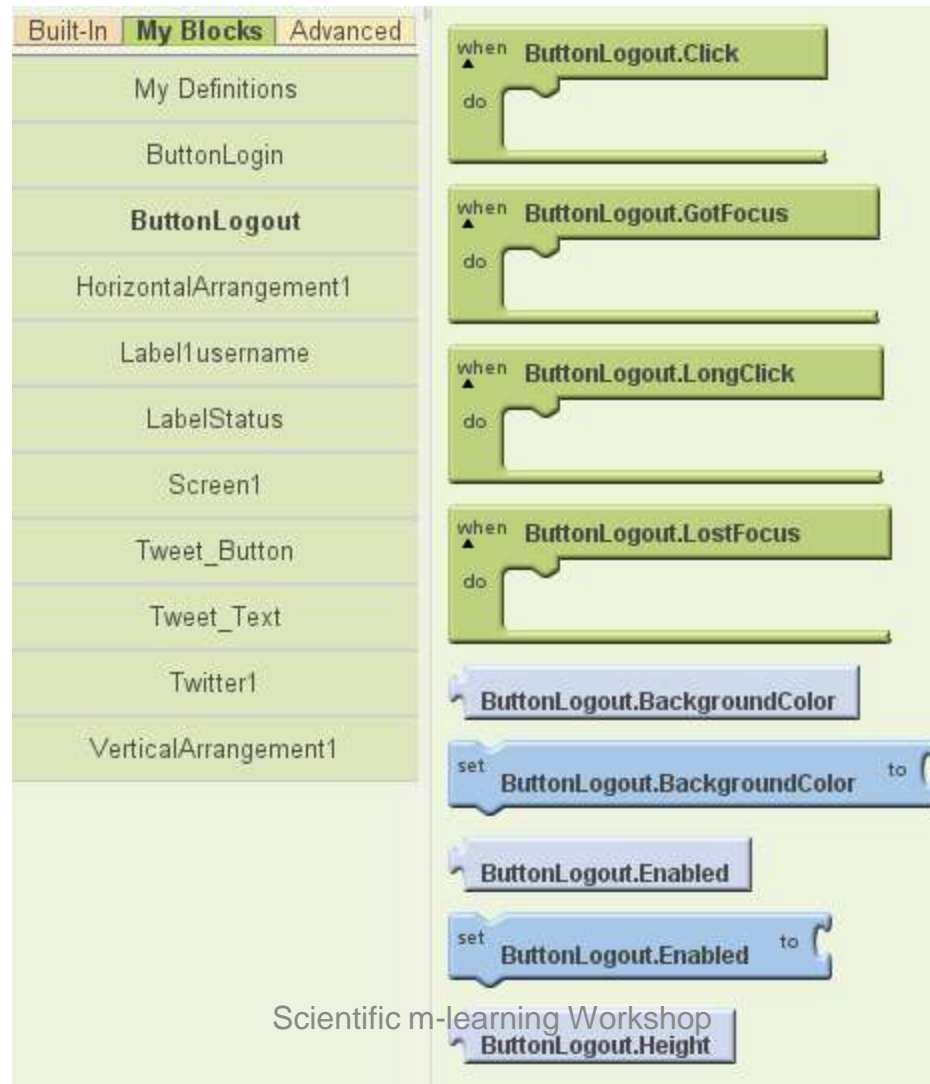
**Event Handlers:**

- when ButtonLogin.Click**: A "do" block is present but empty.
- when ButtonLogin.GotFocus**: A "do" block is present but empty.
- when ButtonLogin.LongClick**: A "do" block is present but empty.
- when ButtonLogin.LostFocus**: A "do" block is present but empty.

**Property Settings:**

- ButtonLogin.BackgroundColor**: A "set" block with "to" connector.
- ButtonLogin.Enabled**: A "set" block with "to" connector.
- ButtonLogin.Height**: A property setting block.

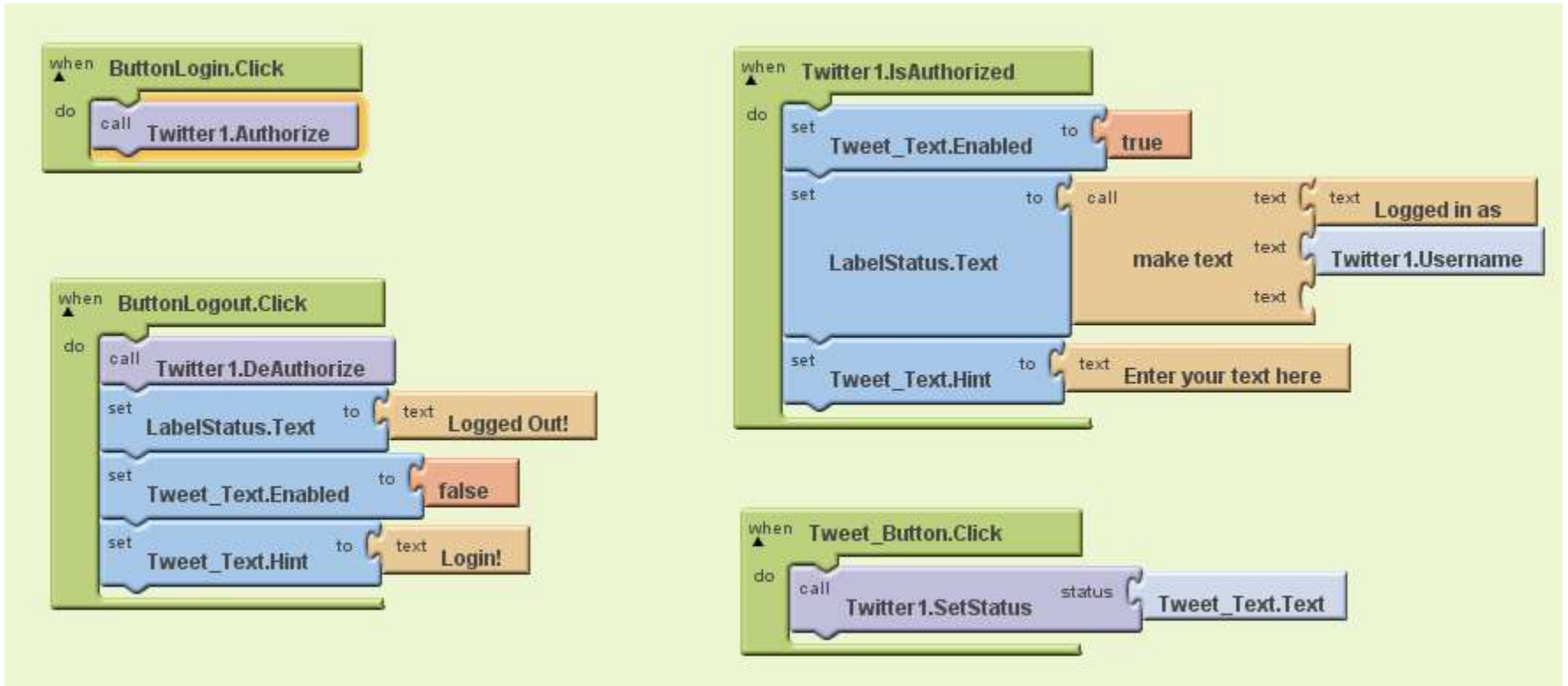
# Logout Button Defination



The image displays a visual programming interface with a component palette on the left and a workspace on the right. The component palette is divided into three tabs: 'Built-In', 'My Blocks', and 'Advanced'. Under the 'My Blocks' tab, a list of components is shown, with 'ButtonLogout' highlighted in bold. The workspace on the right contains the following blocks:

- Four event listener blocks, each starting with a 'when' block and followed by a 'do' block:
  - when ButtonLogout.Click
  - when ButtonLogout.GotFocus
  - when ButtonLogout.LongClick
  - when ButtonLogout.LostFocus
- Three property setting blocks, each starting with a 'set' block and followed by a 'to' block:
  - ButtonLogout.BackgroundColor
  - set ButtonLogout.BackgroundColor to
  - ButtonLogout.Enabled
  - set ButtonLogout.Enabled to
  - ButtonLogout.Height

# Ingredients



# LabelStatus Defination

The image displays a software development environment interface. On the left, a sidebar lists various components under the 'My Blocks' tab. The 'LabelStatus' component is highlighted. On the right, a list of properties for 'LabelStatus' is shown, each with a corresponding 'set' block. The properties and their corresponding blocks are:

- LabelStatus.BackgroundColor
- set LabelStatus.BackgroundColor to
- LabelStatus.FontSize
- set LabelStatus.FontSize to
- LabelStatus.Height
- set LabelStatus.Height to
- LabelStatus.Text
- set LabelStatus.Text to
- LabelStatus.TextColor
- set LabelStatus.TextColor to
- LabelStatus.Visible
- set LabelStatus.Visible to
- LabelStatus.Width

# Twitt Button Definition

The image shows a visual programming environment with a 'My Blocks' palette on the left and a workspace on the right. The palette lists various blocks, with 'Tweet\_Button' selected. The workspace contains the following blocks:

- when **Tweet\_Button.Click** (do loop)
- when **Tweet\_Button.GotFocus** (do loop)
- when **Tweet\_Button.LongClick** (do loop)
- when **Tweet\_Button.LostFocus** (do loop)
- Tweet\_Button.BackgroundColor**
- set **Tweet\_Button.BackgroundColor** to
- Tweet\_Button.Enabled**
- set **Tweet\_Button.Enabled** to
- Tweet\_Button.Height**

# Twitt Text Definition

The image shows a software development environment with a 'My Blocks' palette on the left and a script area on the right. The palette lists various blocks, with 'Tweet\_Text' highlighted. The script area contains a sequence of blocks for the 'Tweet\_Text' object:

- when **Tweet\_Text.GotFocus** (green)
- do (green)
- when **Tweet\_Text.LostFocus** (green)
- do (green)
- call **Tweet\_Text.HideKeyboard** (purple)
- Tweet\_Text.BackgroundColor** (purple)
- set **Tweet\_Text.BackgroundColor** to (blue)
- Tweet\_Text.Enabled** (purple)
- set **Tweet\_Text.Enabled** to (blue)
- Tweet\_Text.FontSize** (purple)
- set **Tweet\_Text.FontSize** to (blue)
- Tweet\_Text.Height** (purple)
- set **Tweet\_Text.Height** to (blue)

Scientific m-learning Workshop

# Twitter1 Definition

The image shows a visual programming environment with a 'My Blocks' tab selected. On the left, a list of built-in blocks includes: My Definitions, ButtonLogin, ButtonLogout, HorizontalArrangement1, Label1username, LabelStatus, Screen1, Tweet\_Button, Tweet\_Text, **Twitter1**, and VerticalArrangement1. The main workspace contains the following blocks:

- when Twitter1.DirectMessagesReceived** (messages) - do
- when Twitter1.FollowersReceived** (followers) - do
- when Twitter1.FriendTimelineReceived** (timeline) - do
- when Twitter1.IsAuthorized** - do
- when Twitter1.MentionsReceived** (mentions) - do
- when Twitter1.SearchSuccessful** (searchResults) - do
- call Twitter1.Authorize**
- call Twitter1.CheckAuthorized**



# Add Twitter to your Application

The screenshot shows the Twitter Developers website. The browser's address bar displays <https://dev.twitter.com/user/login?destination=home>. The page features a blue header with the 'twitter developers' logo, a search bar, and navigation links for 'API Health', 'Blog', 'Discussions', 'Documentation', and 'Sign in'. The main content area has the text 'Extend your reach. Multiply your audience.' and a prominent blue button labeled 'Add Twitter to your website'. To the right is a large, faint Twitter bird logo. Below the main content, there are two columns of links: 'Recent posts from Twitter Developer Blog' with dates and titles like 'Working with the Twitter Streaming APIs' and 'Community Developer Teatimes in Nairobi, Manila, Buenos Aires, and Singapore'; and 'Create applications that integrate Twitter' with links for 'Get started with the API', 'Create an app', and 'Discuss'. The Windows taskbar at the bottom shows various application icons and the system clock indicating 14:30 on 01/06/2012.

[Home](#) → [My applications](#)

# ICTP m-learning Workshop

[Details](#)[Settings](#)[OAuth tool](#)[@Anywhere domains](#)[Reset keys](#)[Delete](#)

## Application Details

**Name: \***

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

**Description: \***

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

**Web Site: \***

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.  
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

## Application Icon



Change icon:

 No file chosen

Maximum size of 700k. JPG, GIF, PNG.



## Application Icon



Change icon:

No file chosen

Maximum size of 700k. JPG, GIF, PNG.

## Application Type

Access:

- Read only
- Read and Write
- Read, Write and Access direct messages

What type of access does your application need? Note: @Anywhere applications require read & write access.

Find out more about our [Application Permission Model](#).

Callback URL:

Where should we return after successfully authenticating? For @Anywhere applications, only the domain specified in the callback will be used. OAuth 1.0a applications should explicitly specify their oauth\_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

## Organization

Organization name:

The organization or company behind this application, if any.

Organization website:

The organization or company behind this application's web page, if any.



### OAuth settings

Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

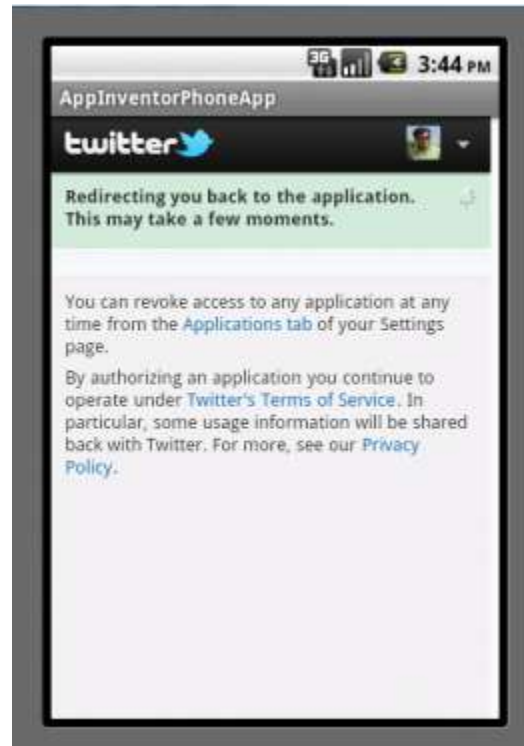
Access level	Read, write, and direct messages <a href="#">About the application permission model</a>
Consumer key	fy0CCUn3kNRixRkSmT2X91Q
Consumer secret	n2YGIKo3jUw7rh2EtZEAg4J02FEwZ9QIHwXjNxSFF90
Request token URL	https://api.twitter.com/oauth/request_token
Authorize URL	https://api.twitter.com/oauth/authorize
Access token URL	https://api.twitter.com/oauth/access_token
Callback URL	http://twitter.com

### Your access token

Use the access token string as your "oauth\_token" and the access token secret as your "oauth\_token\_secret" to sign requests with your own Twitter account. Do not share your oauth\_token\_secret with anyone.

Access token	26739783-olGZyl5qhKyWQvBkKtyPYShZg2MDpG6vZlNt9fMX
Access token secret	BZN0EhnbLLb3rqUa2a8ZHnV4SsZ9OknXq9exdHErup0
Access level	Read, write, and direct messages

Components	Properties
<ul style="list-style-type: none"><li>[-] Screen1<ul style="list-style-type: none"><li>[-] VerticalArrangement1<ul style="list-style-type: none"><li>[-] HorizontalArrangement1<ul style="list-style-type: none"><li>ButtonLogin</li><li>ButtonLogout</li><li>LabelStatus</li><li>Label1username</li><li>Tweet_Text</li><li>Tweet_Button</li><li>Twitter1</li></ul></li></ul></li></ul></li></ul>	<p>ConsumerKey fy0CUn3kNRixRkSmT2X9</p> <p>ConsumerSecret JO2FEwZ9QIHwXjNxSF9d</p>

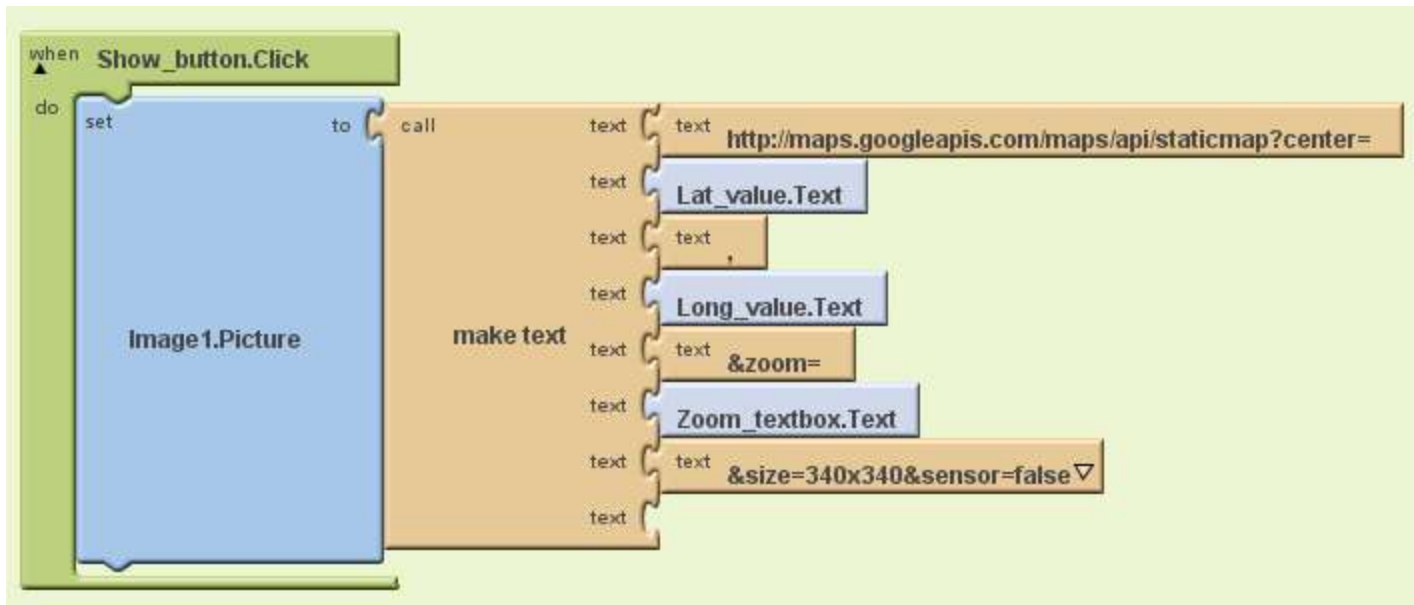


# Google Static Map App

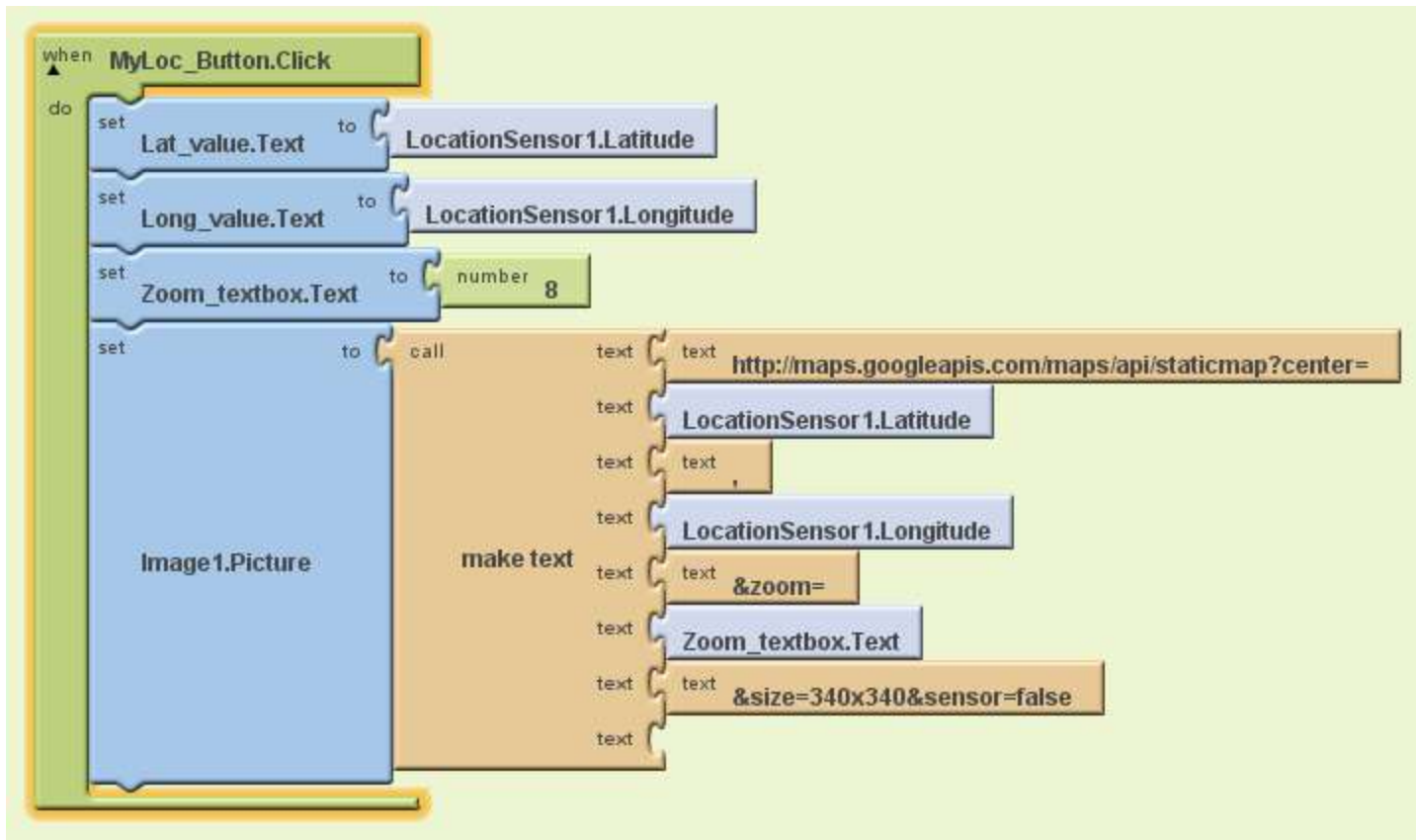




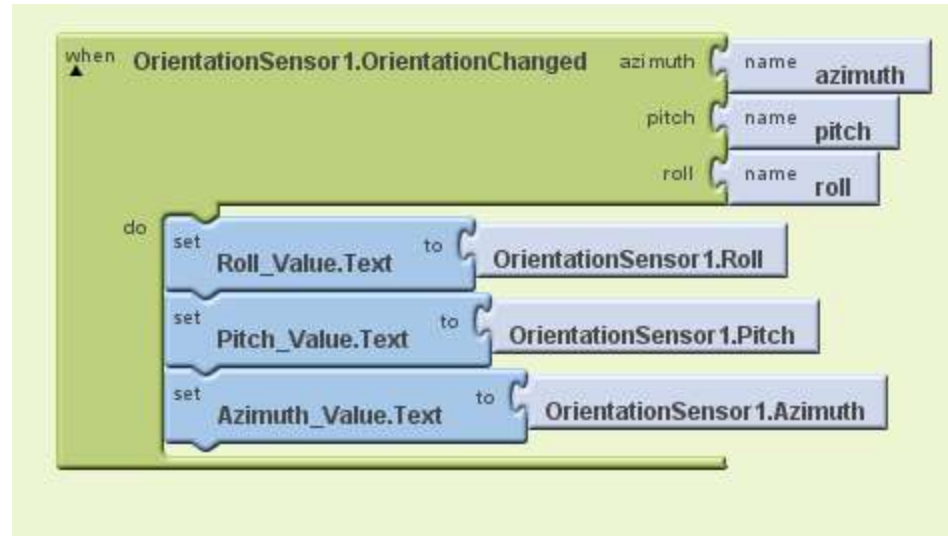
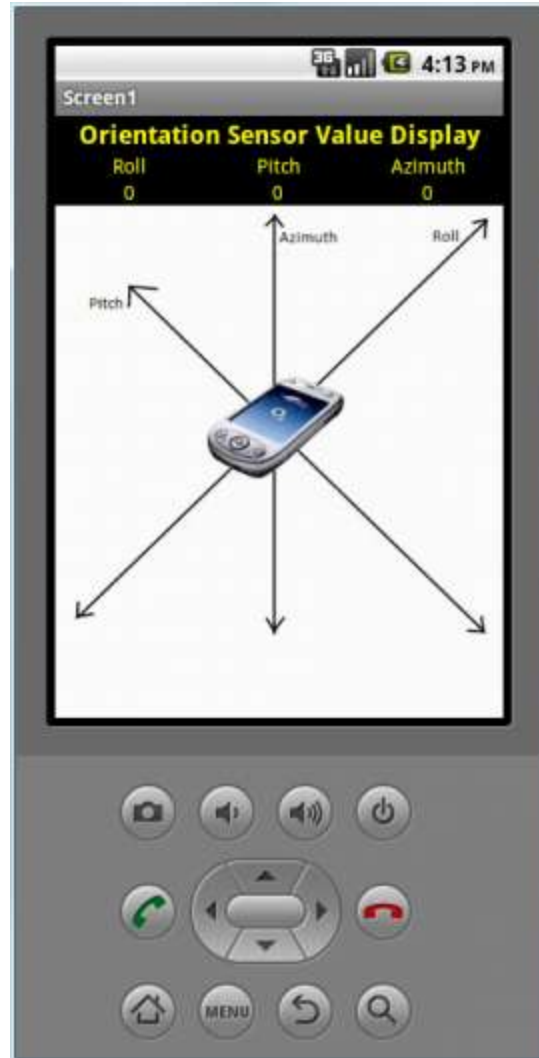
# Show Location (Manual Entry)



# Show My Location (GPS)



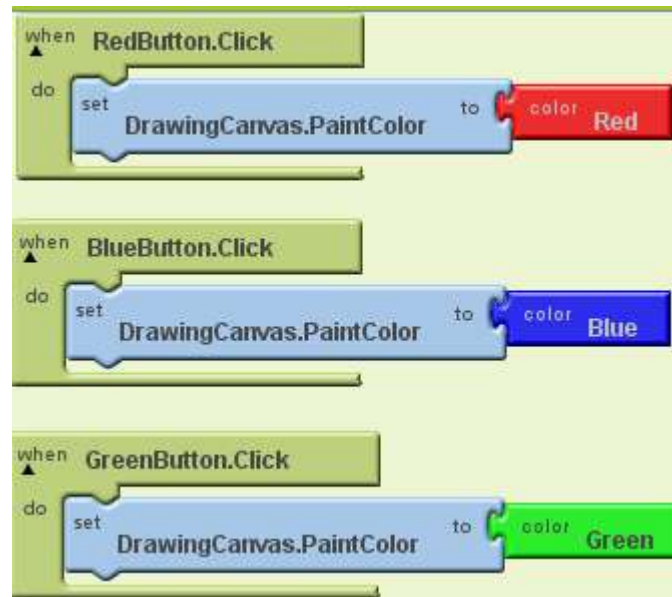
# Orientation Sensor Example



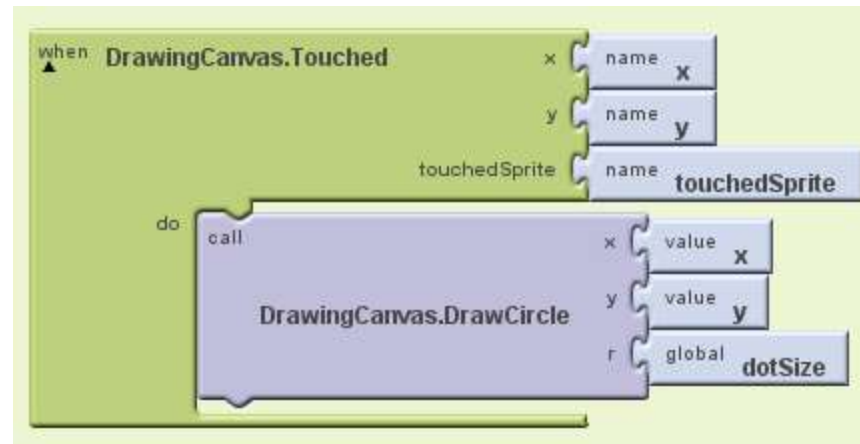
# Drawing on Canvas



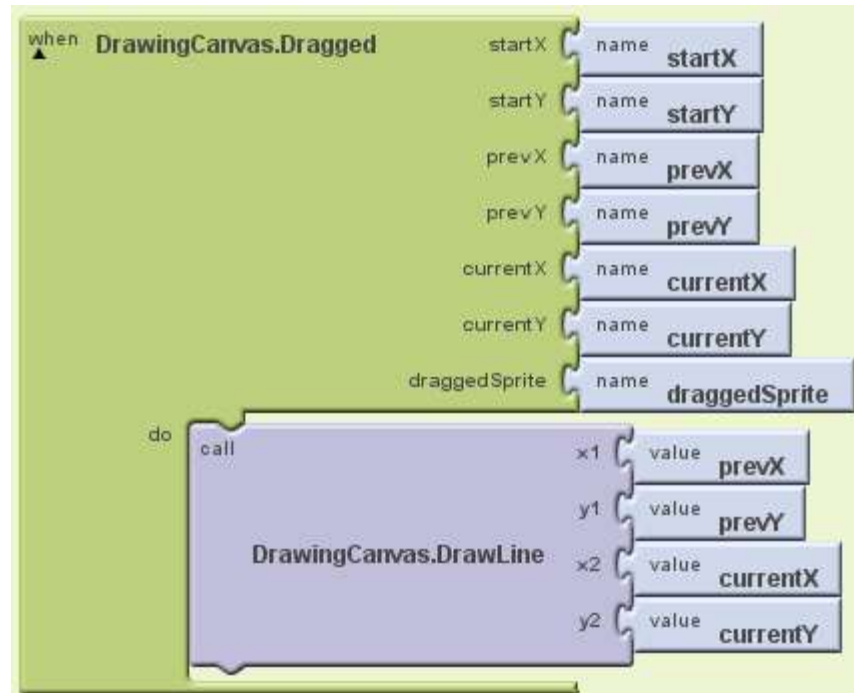
# Setting Colors



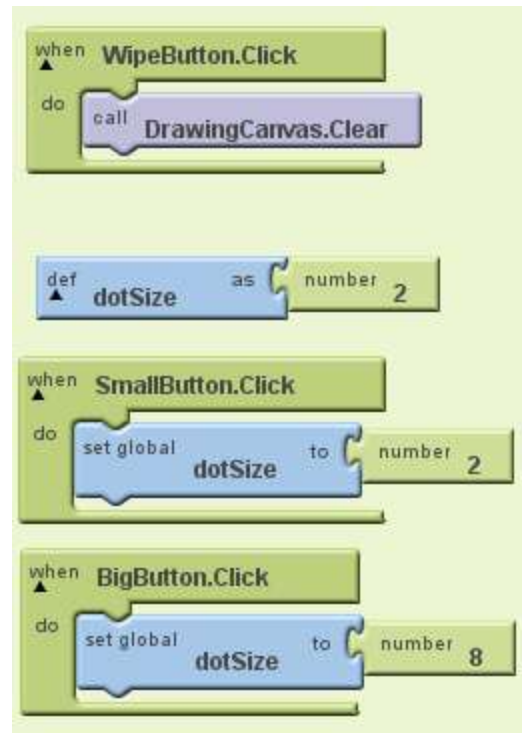
# Canvas Touched Event



# Canvas Dragged Event

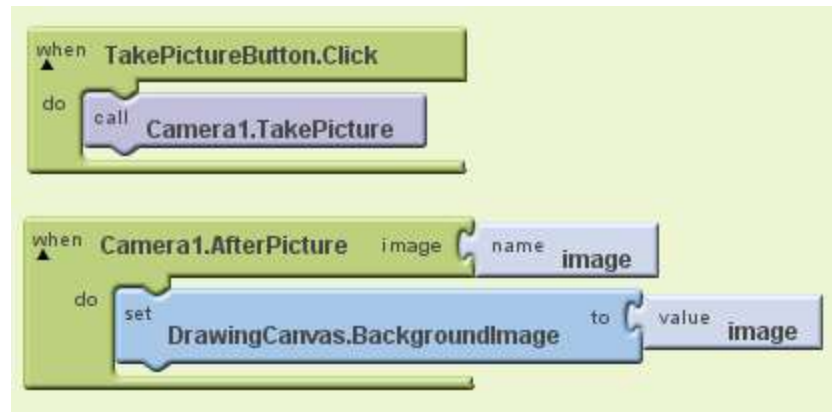


# Setting Dot Size





# Using Camera



# Trieste Tour

The screenshot displays the MIT App Inventor web-based development environment. At the top, the title bar reads "MIT App Inventor BETA" and includes navigation links for "My Projects", "Design", "Learn", and "(Debugging)". A welcome message states: "Welcome to the App Inventor beta preview release. Be sure to check the list of [known issues](#)."

The main workspace is titled "Trieste\_Places" and features a toolbar with "Save", "Save As", "Checkpoint", "Add Screen", and "Remove Screen" buttons. On the right side of the toolbar, it indicates "Blocks Editor is open" and "Package for iPhone".

The interface is divided into several panels:

- Palette:** A vertical list of UI components categorized into "Basic" (Button, Canvas, CheckBox, Clock, Image, Label, ListPicker, PasswordTextBox, TextBox, TinyDB), "Media", "Animation", "Social", "Sensors", "Screen Arrangement", "LEGO® MINDSTORMS®", "Other stuff", "Not ready for prime time", and "Old stuff".
- Viewer:** The central area showing a mobile app preview. It includes a status bar with "5:09 PM", a title bar "ICTP Trieste Tour", a large image of a coastal building, the text "Trieste Tour @ m-learning workshop", and a button labeled "Find Trieste Attractions". A checkbox "Display Invisible Components in Viewer" is checked. Below the viewer, a "Non-visible components" section shows "ActivityStarter1".
- Components:** A tree view showing the hierarchy of components on the screen: "Screen1" (containing "Image1", "Label1", "HorizontalArrangement1", "ListPicker1", and "ActivityStarter1").
- Properties:** A panel for configuring the selected component. For "Image1", the "BackgroundColor" is set to "White", "BackgroundImage" is "None", and "Icon" is "None". For "HorizontalArrangement1", "ScreenOrientation" is "Unspecified" and "Scrollable" is checked. The "Title" property is set to "ICTP Trieste Tour".
- Media:** A section at the bottom right showing a list of media files: "Trieste.jpg" and "metro.jpg", with an "Add..." button.

At the bottom left, there is a link for "Privacy Policy and Terms of Use". At the bottom right, the version number is "Version: 125 fcf\_4fbee9e4023c".

# Blocks for Trieste Tour App

App Inventor for Android Blocks Editor: Trieste\_Places - Screen1

Trieste\_Places - Scre... Saved Undo Redo New emulator Connect to Device... Zoom 100%

Built-In My Blocks Advanced

Definition  
Text  
Lists  
Math  
Logic  
Control  
Colors

def destinations as call

make a list

- item text Museo Revoltella
- item text Castello di San Giusto
- item text Grotta Gigante
- item text Acquario Marino
- item text Castello di Miramare
- item text Canal Grande
- item text ICTP

when Screen1.Initialize

do set ListPicker1.Elements to global destinations

when ListPicker1.AfterPicking

do set ? to call

Concatenate the maps URL with the search terms selected in ListPicker1, e.g., "http://maps.google.com/?q=Tour Eiffel".

ActivityStarter1.DataUri

make text

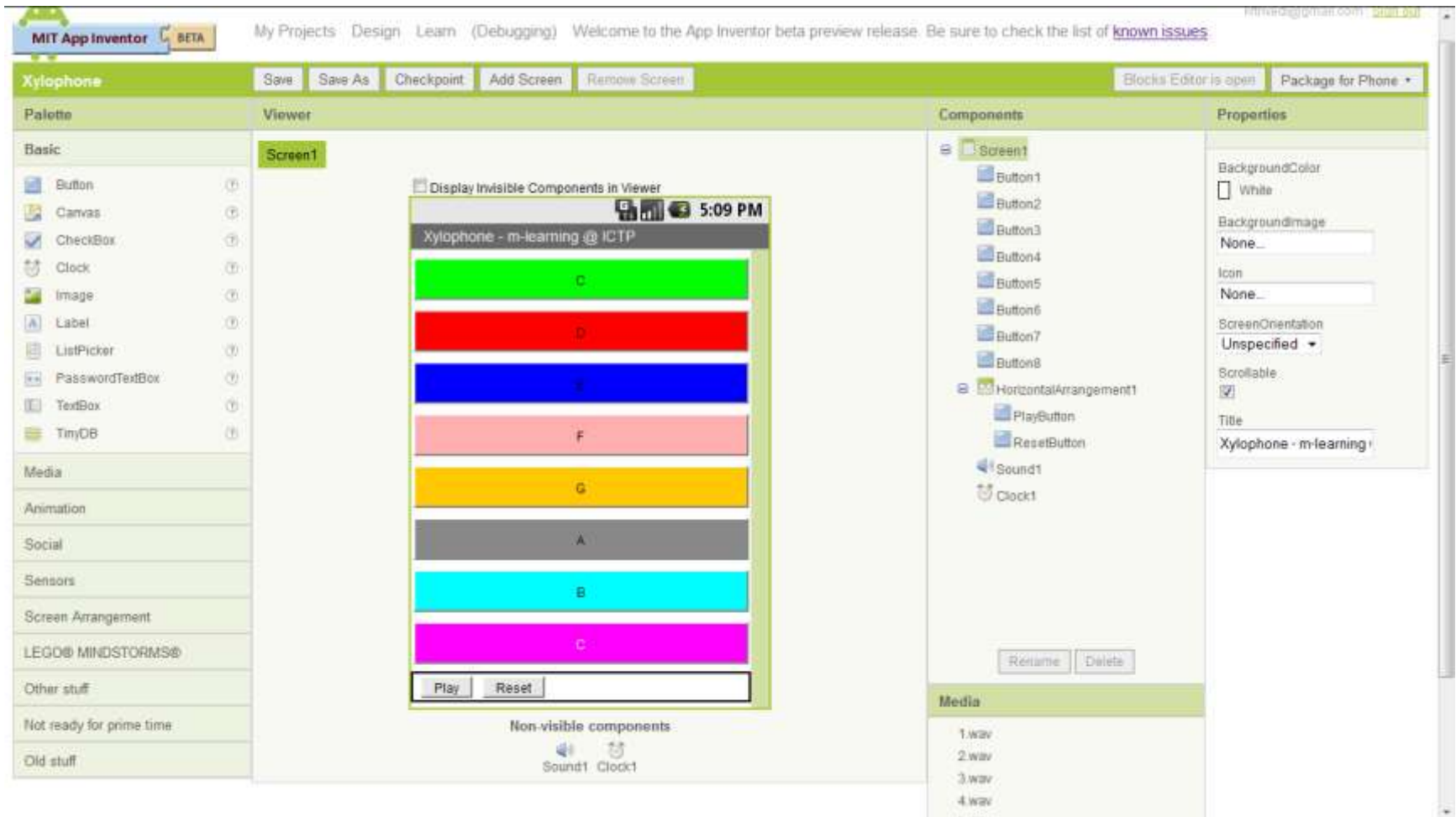
- text http://maps.google.com/?q=
- text ListPicker1.Selection

call ActivityStarter1.StartActivity

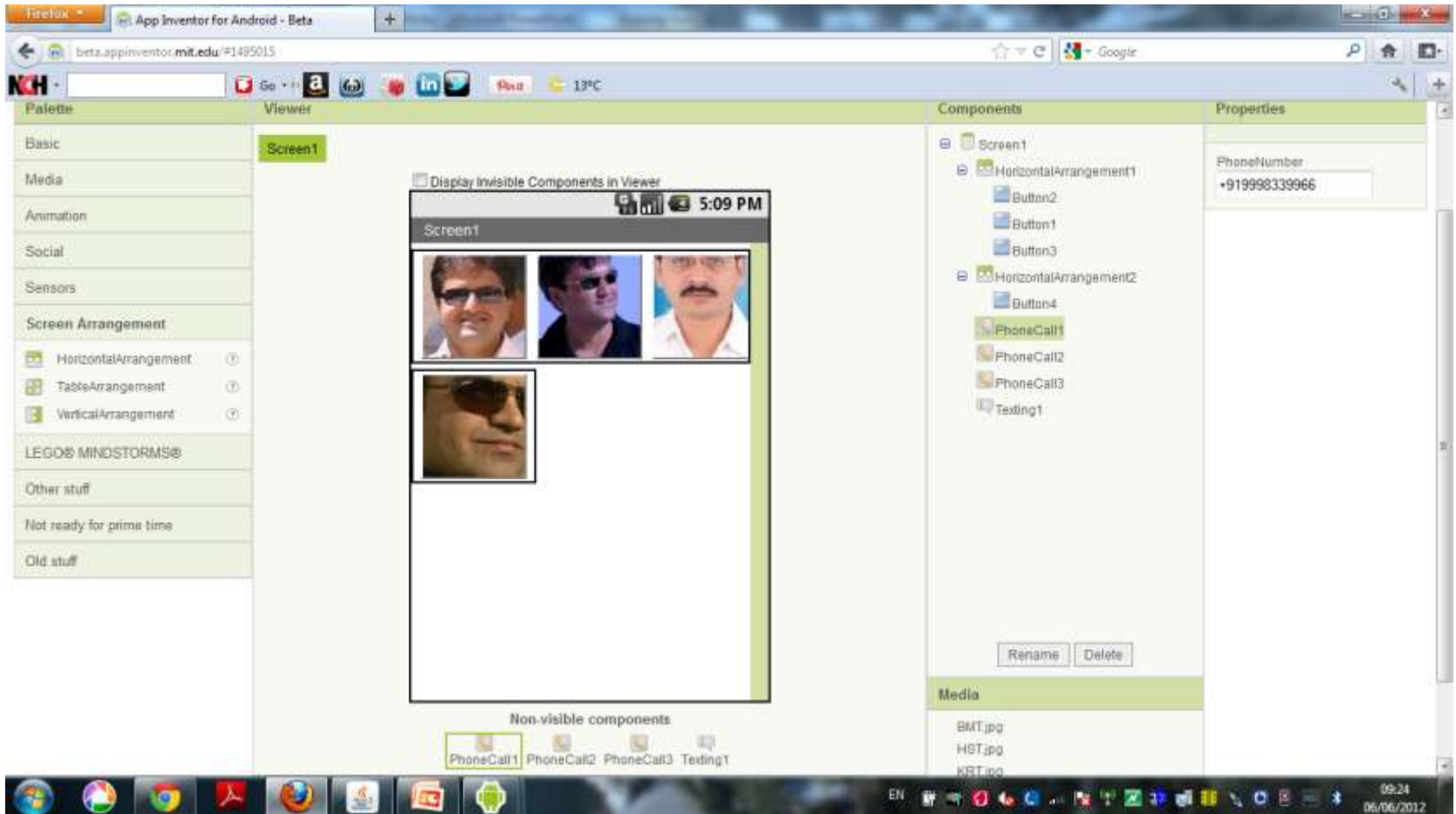
Version: 125 Id: 4fbee9e4023c

EN 07:16 06/06/2012

# Xylophone



# Call / Text App



# Blocks for the App

The screenshot displays the App Inventor for Android Blocks Editor interface for a project named "Friend Caller - Screen1". The interface includes a top toolbar with "Saved", "Undo", "Redo", "New emulator", and "Connect to Device..." buttons, along with a zoom slider. On the left, a "My Blocks" palette lists components: Button1, Button2, Button3, Button4, HorizontalArrangement1, HorizontalArrangement2, PhoneCall1, PhoneCall2, PhoneCall3, Screen1, and Texting1. The main workspace contains four event-driven logic blocks:

- when Button1.Click** (green) → **do call PhoneCall1.MakePhoneCall** (purple)
- when Button2.Click** (green) → **do call PhoneCall2.MakePhoneCall** (purple)
- when Button3.LongClick** (green) → **do call PhoneCall3.MakePhoneCall** (purple)
- when Button4.Click** (green) → **do call Texting1.SendMessage** (purple)

At the bottom left, the version and ID are shown as "Version: 125 Id: 4fbee9e4023c". The Windows taskbar at the bottom features icons for Internet Explorer, Google Chrome, and other applications, with the system clock showing "09:25 06/06/2012". A watermark "Scientific Learning Workshop" is visible at the bottom center.