



Taller 01

Desarrollo de aplicaciones para dispositivos móviles con Android

Herramienta App Inventor-MIT

José Luis Morón Valdivia jmoron@pucp.pe

Luis Felix lfelix@pucp.pe



Objetivos

- Introducción básica de nuevas tendencias en construcción de aplicaciones.
- Definir las ventajas y limitaciones de la programación a través de bloques funcionales.
- Analizar las Herramientas Kodu de Microsoft, Scratch y Appinventor desarrolladas por el MIT



ESCUELA DE
POSGRADO

Maestría en Integración e
innovación educativa de las TIC



PRONABEC



PUCP

Cloud Computing

1- CONTEXTO



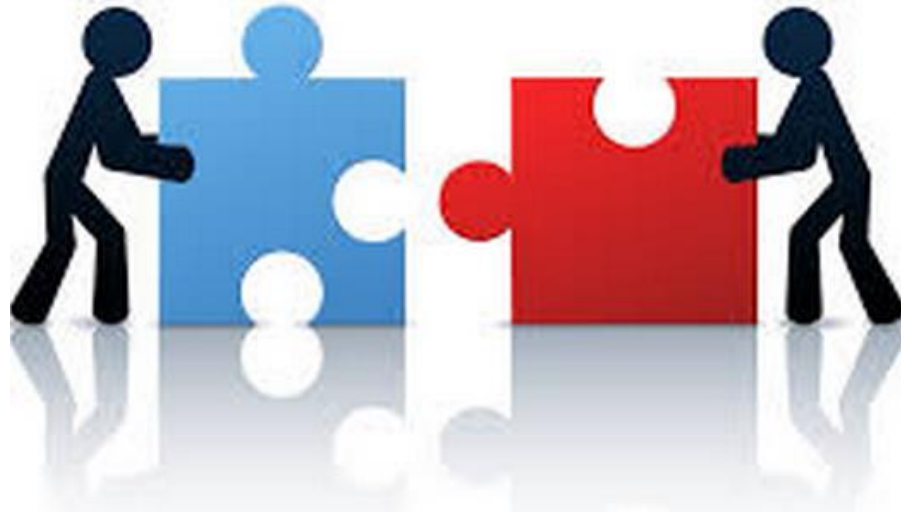
Contexto





Contexto Traiga su Propio Dispositivo (Bring Your Own Device - BYOD)





Herramientas

2- HERRAMIENTAS



2.1 Herramientas

Son nuevos lenguajes de programación diseñados para apoyar el desarrollo de la fluidez tecnológica. (Kodu, Scratch, Appinventor)

Building-Block Programming





2.2 Herramienta- App Inventor

App Inventor es un entorno de desarrollo visual de bloques, para la programación de mobile apps.

<http://www.appinventor.org/>

<http://appinventor.mit.edu/>

Apps Android

Weekly Active Users: **87K** Total Registered Users: **1.9M** Countries: **195** Apps Built: **4.7M**

Logics
Math
Text
Lists
Colors
Variables
Procedures
Sprites
Textfield
Button
TextToSpeech
AccelerometerSensor
Component

when Button1 Click
do call TextToSpeech1 Speak
message [Textfield1 Text]

when Accelerometer1 Acceleration
do call TextToSpeech1 Speak

Your ideas.
Your designs.
Your apps.

Invent Now

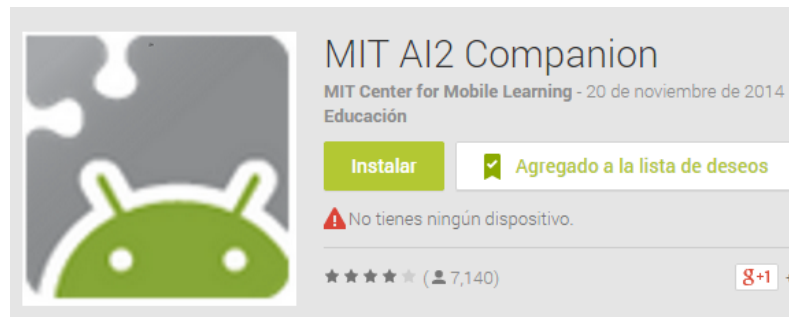
App Inventor code is [open source](#). Help us improve App Inventor!



2.3 Herramientas- App Inventor

Requisitos

1. Cuenta de Google. Por ejm Gmail.
2. Descargar MIT AI2 Companion desde Google Play.
3. Contar con un dispositivo (Smartphone, Tablet)



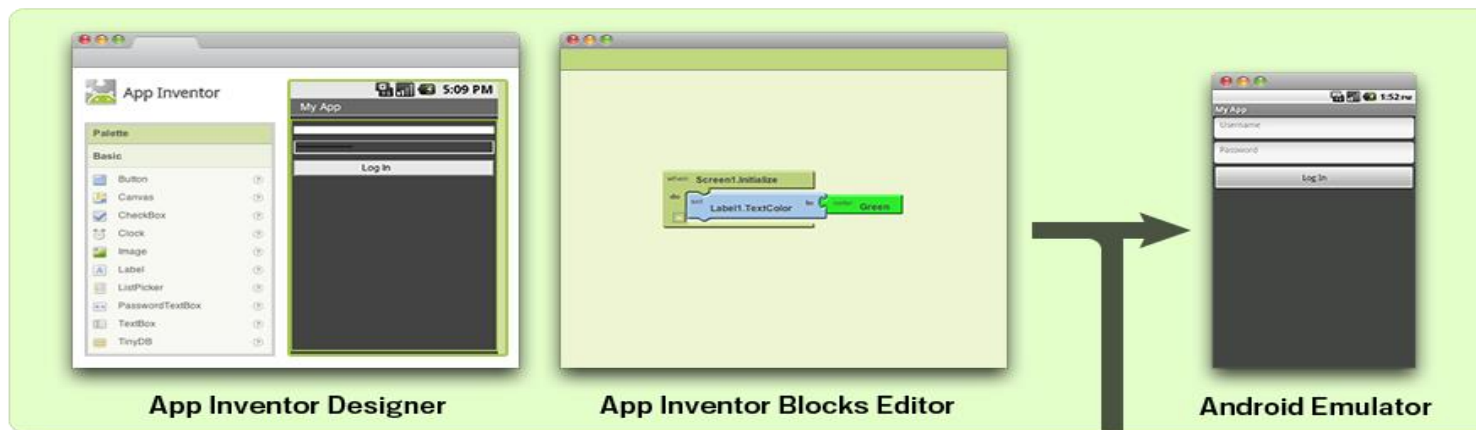
<http://appinventor.mit.edu/>



2.4 Esquema Trabajo- App Inventor



Google App Inventor Servers



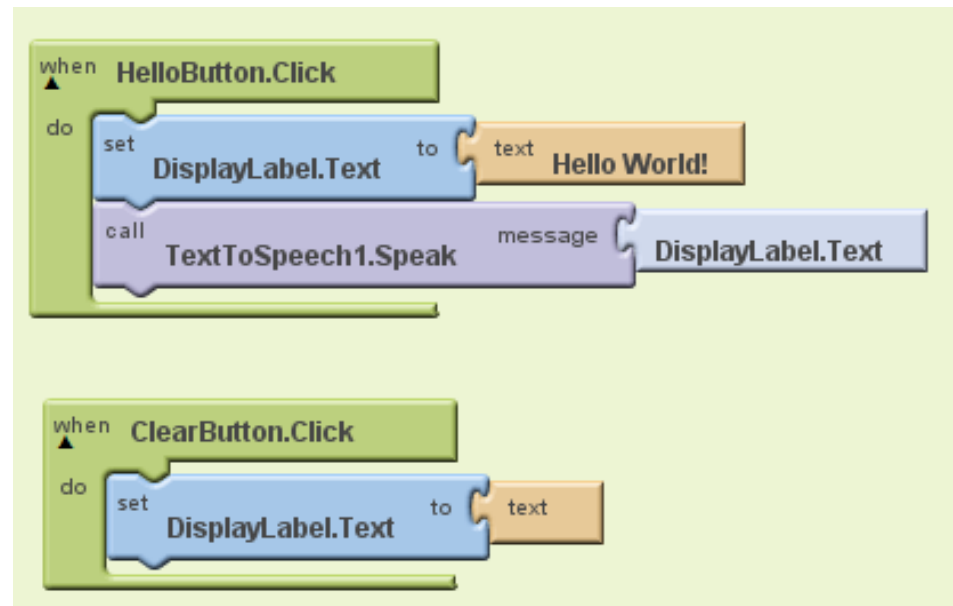


3. Código vs App Inventor

- Java Code

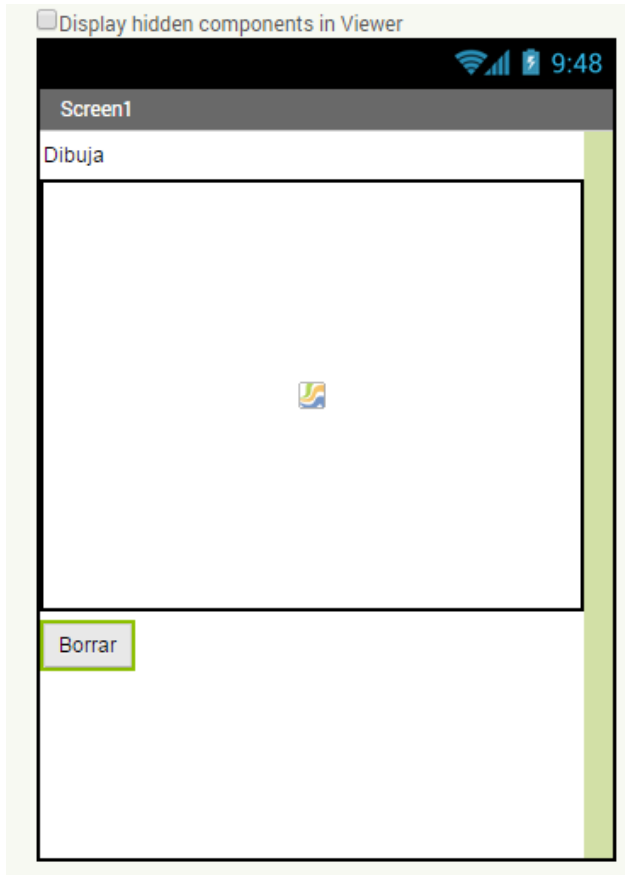
```
public class HelloWorldApp {  
    public static void main(String[] args) {  
        System.out.println("Hello World!");  
    }  
}
```

- AppInventor





Ejm 1 Lienzo Dibujo . Designer



Se implementará

Label: Etiqueta

Canvas: Lienzo

Button: Botón

Se colocarán las propiedades y el
nombre del objeto



Ejm 1 Blocks - Lienzo

```
when Canvas1 .Dragged
  startX  startY  prevX  prevY  currentX  currentY  draggedSprite
do
  call Canvas1 .DrawLine
    x1  get prevX
    y1  get prevY
    x2  get currentX
    y2  get currentY

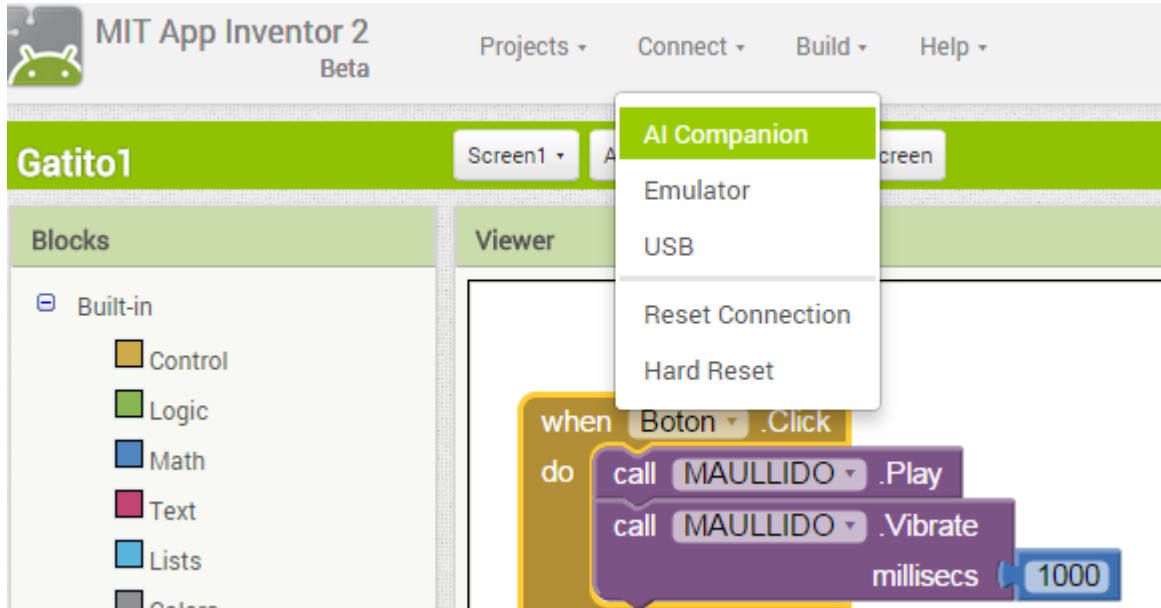
when ClearButton .Click
do
  call Canvas1 .Clear
```

Para el lienzo y el botón

Se colocarán los bloques apropiados en la programación



Comunicación con tu Dispositivo



- Conectamos el proyecto con AI Companion.
- Iniciamos AI Companion en el smartphone.

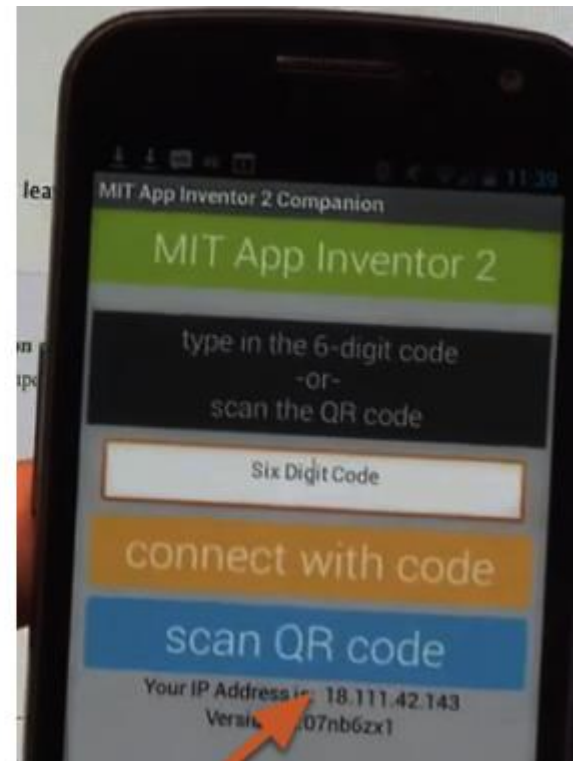
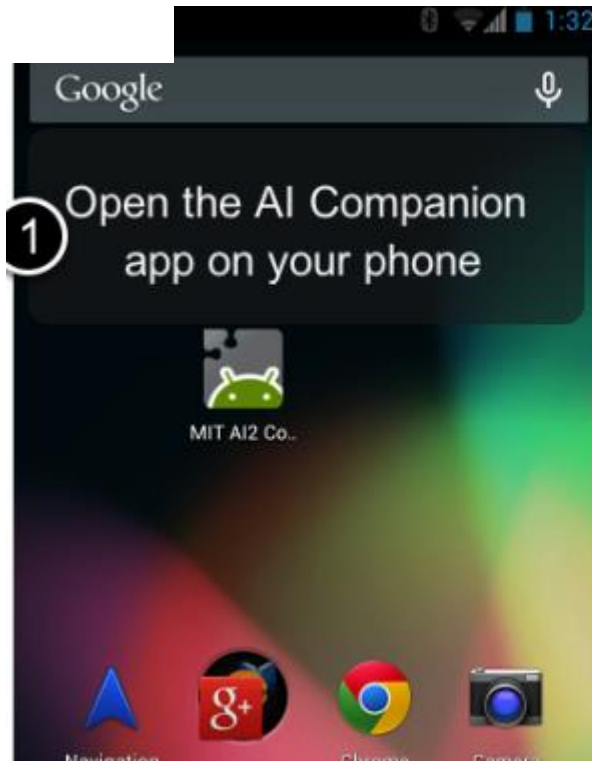


Ejm 1 Comunicación con tu Dispositivo



Your code is:

sftddr






Comunicación con tu Dispositivo

Viewer

when Boton .Click
do
call MAULLIDO
call MAULLIDO

Connect to Companion

Launch the MIT AI2 Companion on your device and then scan the barcode or type in the code to connect for live testing of your app.
[Need help finding the Companion App?](#)



Your code is:
sftddr

Cancel

- Conectamos el proyecto con AI Companion.
- Iniciamos AI Companion en el smartphone.
- Lee el QR
- Escribe el código que permitirá enlaza



Ejm 1 Blocks – Lienzo Adicional

The screenshot displays the PaintPotV2 software interface. At the top, there is a green header bar with the text "PaintPotV2" and buttons for "Screen1", "Add Screen ...", and "Remove Screen". On the right side of the header, there are buttons for "Designer" and "Blocks".

The interface is divided into several panels:

- Palette:** Located on the left, it contains categories like "User Interface" (with items like Button, CheckBox, Clock, Image, Label, ListPicker, Notifier, PasswordTextBox, Slider, TextBox, and WebViewer), "Layout", "Media", "Drawing and Animation", "Sensors", and "Social".
- Viewer:** The central area shows a mobile app preview. It includes a status bar at the top with the time "9:48" and signal strength. Below that, the app title "PaintPot" is visible. The main content area features three colored buttons (Red, Blue, Green) and a large image of an orange cat. At the bottom, there are three buttons labeled "Wipe", "Big dots", and "Small dots". A checkbox labeled "Display hidden components in Viewer" is located above the preview.
- Components:** On the right, this panel shows a tree view of the app's components. It includes "Screen1", "ThreeButtons" (containing ButtonRed, ButtonBlue, and ButtonGreen), "DrawingCanvas", and "BottomButtons" (containing ButtonWipe, ButtonBig, and ButtonSmall). "Rename" and "Delete" buttons are at the bottom of this panel.
- Properties:** The rightmost panel shows the properties for the selected "BottomButtons" component. It includes settings for "AlignHorizontal" (set to "Left"), "AlignVertical" (set to "Top"), "Visible" (set to "showing"), "Width" (set to "Automatic..."), and "Height" (set to "Automatic...").



Ejm 1 Blocks – Lienzo Adicional

The screenshot shows the Scratch IDE interface for a project named "PaintPotV2". The interface is divided into several sections:

- Top Bar:** Contains "Screen1", "Add Screen ...", "Remove Screen", "Designer", and "Blocks" buttons.
- Left Panel (Blocks):** Lists built-in blocks categorized by Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures. It also shows a tree view of the current project's objects: "Screen1" containing "ThreeButtons" (ButtonRed, ButtonBlue, ButtonGreen), "DrawingCanvas", "BottomButtons" (ButtonWipe, ButtonBig, ButtonSmall), and "Media".
- Viewer:** Displays the Scratch script for the "DrawingCanvas" object. The script includes:
 - When ButtonRed is clicked: set DrawingCanvas PaintColor to red.
 - When ButtonBlue is clicked: set DrawingCanvas PaintColor to blue.
 - When ButtonGreen is clicked: set DrawingCanvas PaintColor to green.
 - When ButtonWipe is clicked: call DrawingCanvas Clear.
 - When DrawingCanvas is touched: call DrawingCanvas DrawCircle with x, y, and r (set to global dotsize).
 - When DrawingCanvas is dragged: call DrawingCanvas DrawLine with startX, startY, prevX, prevY, currentX, currentY, and draggedSprite.
 - Initialize global variables: small to 2, big to 8, and dotsize to 2.
 - When ButtonBig is clicked: set global dotsize to get global big.
 - When ButtonSmall is clicked: set global dotsize to get global small.
- Bottom Left:** Shows "0" warnings and a "Show Warnings" button.
- Bottom Right:** Contains a trash can icon.



Taller 02

Sound, Button

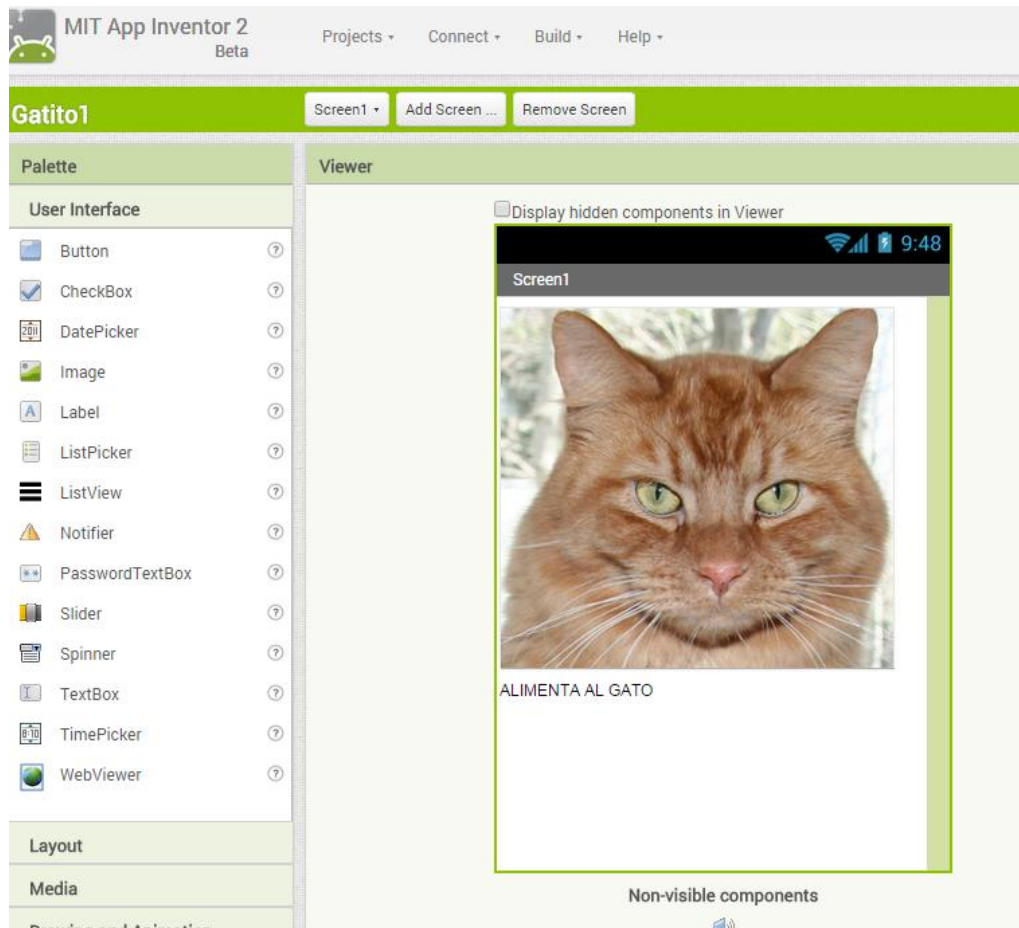
Herramienta App Inventor-MIT

José Luis Morón Valdivia

jmoron@pucp.pe



Designer



Agregar:

Button

Sound:

Meow.mp3

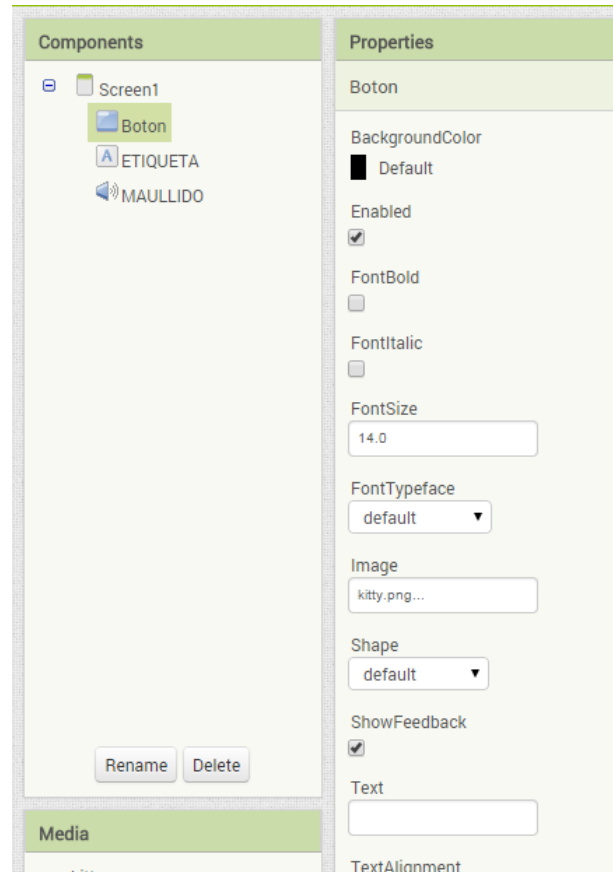
Player:Sonido de
fondo

[http://www.lastfm.es/music/
+free-music-
downloads/background](http://www.lastfm.es/music/+free-music-downloads/background)

<http://appinventor.mit.edu/explore/ai2/hellopurrr.html>



Ej1. Designer Paleta - Properties





Ej2. BLOCKS - Editor de Bloques

MIT App Inventor 2 Beta

Projects ▾ Connect ▾ Build ▾ Help ▾

Gatito1

Screen1 ▾ Add Screen ... Remove Screen

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Viewer

when AccelerometerSensor1 .Shaking

do

- call Meow .Play
- call Meow .Vibrate
- milliseconds 20

when Boton .Click

do

- call MAULLIDO .Play
- call MAULLIDO .Vibrate
- milliseconds 1000

when Button1 .Click

do

- call Meow .Play

when Screen1 .Initialize

do

- call BackgroundMusic .Start

Definimos los bloques o fichas para el evento Click del botón.

Programación Orientada a Eventos. Minuto 16:30
<https://www.youtube.com/watch?v=EEKxqyrtAwQ>



Ej2. BLOCKS - Editor de Bloques

```
when Button1 .Click  
do call Meow .Play
```

```
when Screen1 .Initialize  
do call BackgroundMusic .Start
```

```
when AccelerometerSensor1 .Shaking  
do call Meow .Play  
call Meow .Vibrate  
millisecs 20
```

Adicionales



Taller 03

Location Sensor, Camera, SpeechRecognizer

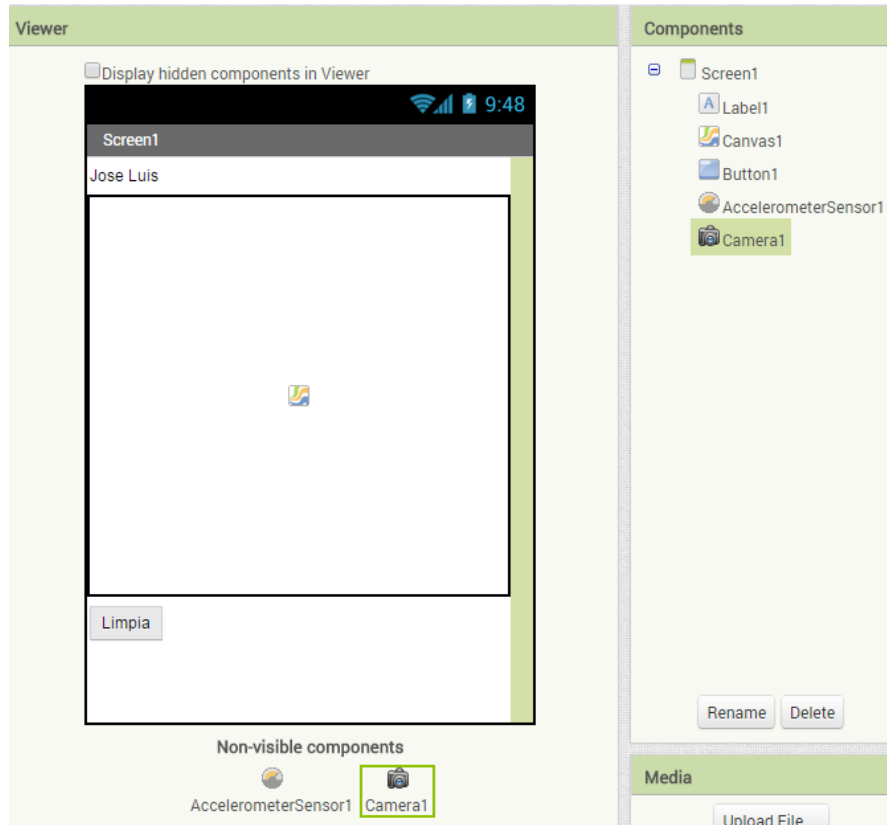
Herramienta App Inventor-MIT

José Luis Morón Valdivia

jmoron@pucp.pe



Ejm 1 Blocks – Lienzo Adicional-Cámara





Ejm 1 Blocks – Lienzo Adicional-Cámara

The image shows a visual programming interface with two main panels: 'Blocks' on the left and 'Viewer' on the right.

Blocks Panel:

- Built-in:**
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1:**
 - Label1
 - Canvas1
 - BotonTomaFoto
 - BotonLimpia
 - AccelerometerSensor1
 - Camera1
- Any component

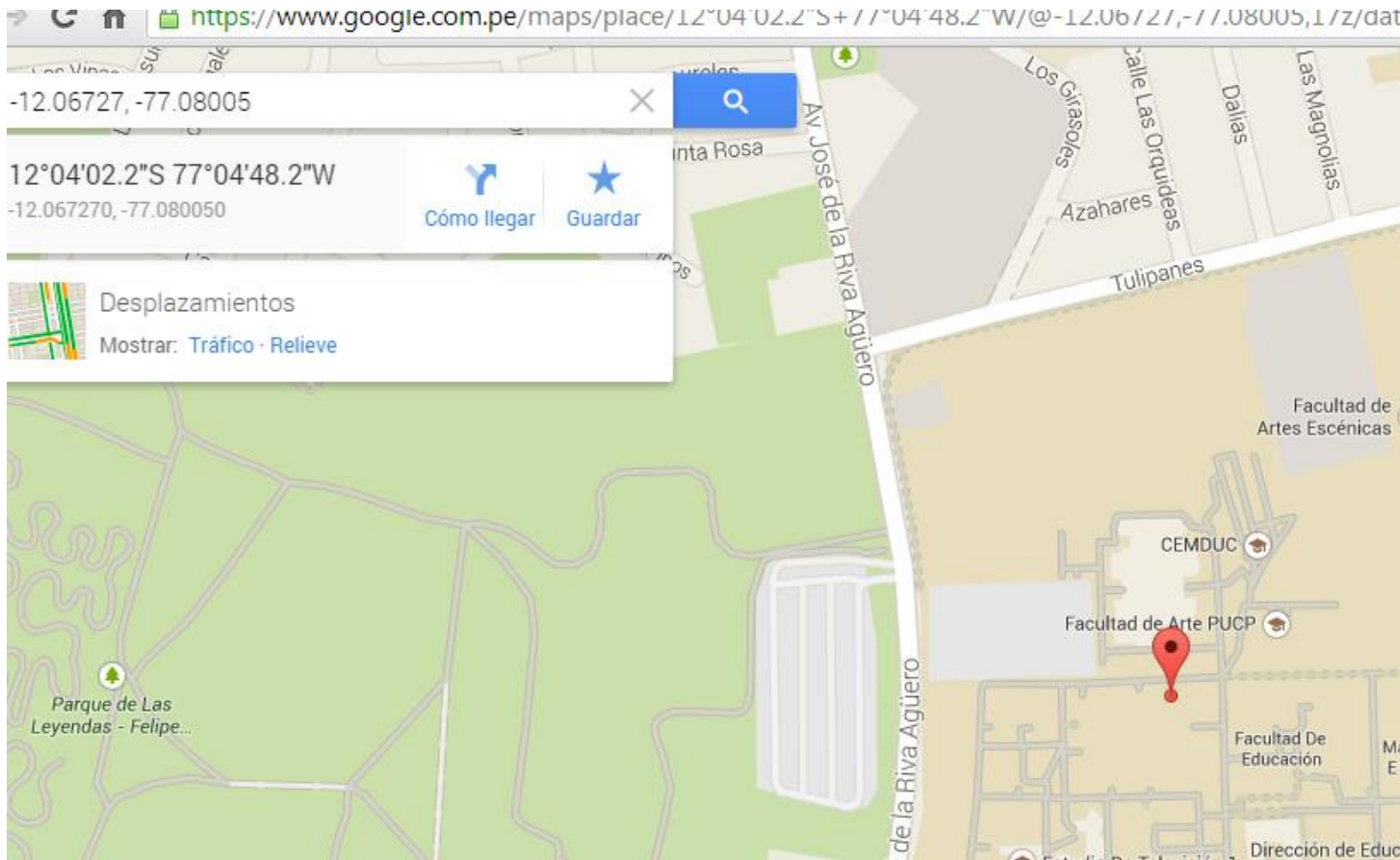
Viewer Panel:

The viewer displays several event-driven code blocks:

- when Canvas1 .Dragged:** A large block containing:
 - startX, startY, prevX, prevY, currentX, currentY, draggedSprite (input variables)
 - do: set Canvas1 . PaintColor to (red color block)
 - call Canvas1 .DrawLine
 - x1: get prevX
 - y1: get prevY
 - x2: get currentX
 - y2: get currentY
- when AccelerometerSensor1 .Shaking:** do: call Canvas1 .Clear
- when BotonLimpia .Click:** do: call Canvas1 .Clear
- when BotonTomaFoto .Click:** do: call Camera1 .TakePicture
- when Camera1 .AfterPicture:** image (input variable) do: set Canvas1 . BackgroundImage to (get image)



Ejm 3 Location Sensor





Ejm 3 Location Sensor

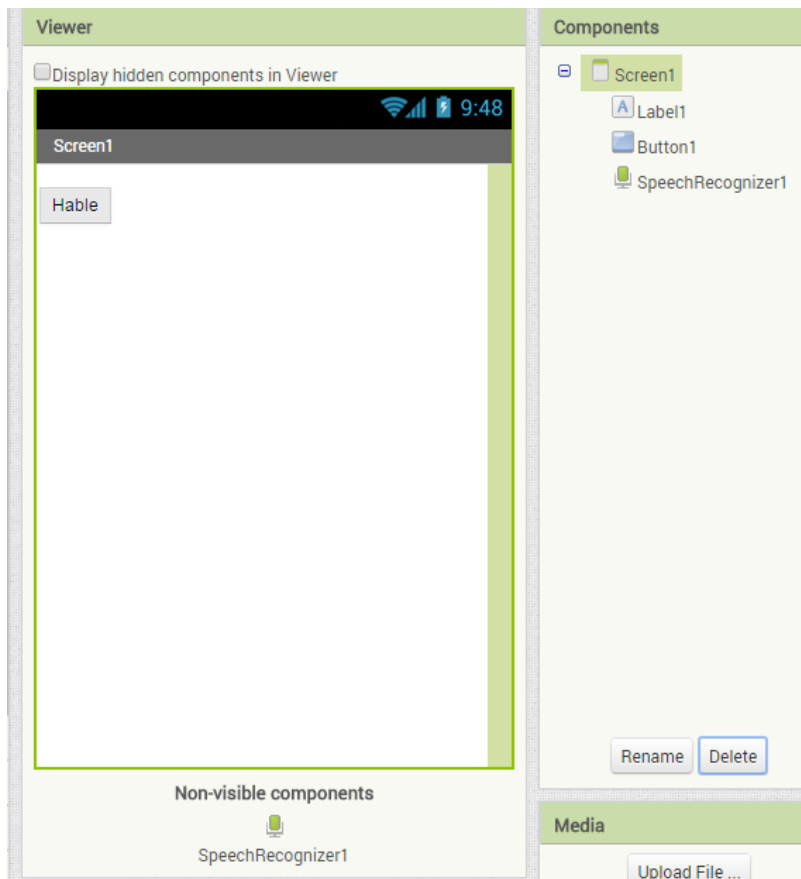
The screenshot shows the Android Studio interface. On the left is the 'Viewer' pane, which displays a mobile app preview with a status bar at the top showing signal strength, Wi-Fi, and the time 9:48. Below the status bar, the app has a list of labels: 'Latitud', 'Longitud', 'Ubicación', and 'Altitud'. The 'Altitud' label is highlighted with a yellow border. Below the labels is a button labeled 'Obtener Ubicación'. On the right is the 'Components' pane, which lists the components used in the app: 'Screen1', 'Latitud', 'Longitud', 'Direccion', 'Altitud', 'Button1', 'LocationSensor1', and 'TextToSpeech1'. At the bottom of the Components pane are 'Rename' and 'Delete' buttons. Below the Components pane is the 'Non-visible components' section, which shows a location pin icon and a speech bubble icon. At the very bottom is the 'Media' section.

```
when LocationSensor1 . LocationChanged
  latitude longitude altitude
do
  set Latitud . Text to get latitude
  set Longitud . Text to get longitude
  set Direccion . Text to LocationSensor1 . CurrentAddress
  set Altitud . Text to get altitude
  call TextToSpeech1 . Speak
  message join " Tu ubicación ha cambiado "
  LocationSensor1 . CurrentAddress
  set LocationSensor1 . Enabled to false

when Button1 . Click
do
  set LocationSensor1 . Enabled to true
```



Ejm 4 Reconocimiento Texto



```
when Button1 .Click  
do call SpeechRecognizer1 .GetText
```

```
when SpeechRecognizer1 .AfterGettingText  
result  
do set Label1 . Text to get result
```

```
when SpeechRecognizer1 .BeforeGettingText  
do set Label1 . Text to " "
```



Conclusiones

- Más Concreto, menos abstracto
- Al no escribir el código, no hay errores de sintaxis.
- Los eventos en el primer nivel
- Construcción Como armar un rompecabezas (sólo algunas piezas encajan)
- Recoge el esfuerzo y experiencias de comunidades tecnológicas de alto nivel. Microsoft, Mit y Google.



Bibliografía

- <http://appinventor.mit.edu/explore/resources/beginner-app-inventor-concept-cards.html>
- <http://appinventor.mit.edu/>
- <http://research.microsoft.com/en-us/projects/kodu/>
- <http://scratch.mit.edu>
- <http://appinventor.mit.edu>
- <http://www.appinventor.org/>
- <http://developer.android.com/sdk/index.html>