TREBALL FI DE GRAU

TÍTOL: Desenvolupament d'aplicacions Android a través de AppInventor

AUTORA: Èrica Martínez Villena

TITUL·LACIÓ: Grau en Enginyeria Electrònica Industrial i Automàtica

DIRECTOR: Cristobal Raya Giner

DEPARTAMENT: 707 – Enginyeria, Sistemes, Automàtica i Informàtica Industrial

DATA: 29/01/2014

TÍTOL:
Desenvolupament d'aplicacions Android a través de AppInventor

COGNOMS: Martínez Villena **NOM:** Èrica

TITULACIÓ: Grau en Enginyeria Electrònica Industrial i Automàtica

MENCIÓ: Automàtica

DIRECTOR: Cristobal Raya Giner

DEPARTAMENT: 707 – Enginyeria, Sistemes, Automàtica i Informàtica Industrial

QUALIFICACIÓ DEL PFC

TRIBUNAL

PRESIDENT **SECRETARI** **VOCAL**

DATA DE LECTURA:

Aquest projecte té en compte aspectes mediambientals: ☐ Sí ☒ No

TREBALL FI DE GRAU

RESUM

L'objectiu principal d'aquest projecte és mostrar una de les eines que hi ha a l'hora de crear i dissenyar aplicacions pel sistema operatiu Android, tant per mòbils com per tablets.

En aquest cas, es dedicarà especialment a AppInventor, una plataforma per crear aplicacions a través de l'ús de blocs on no cal tenir uns coneixements específics sobre Java. Amb uns coneixement mínims de programació, pot funcionar qualsevol tipus d'aplicació.

Per veure els diferents tipus d'eines que hi ha disponibles i el potencial d'aquestes, s'ha dissenyat una aplicació sobre una plataforma per l'aprenentatge d'un idioma. Aquesta en concret, és sobre l'estudi del japonès. No és una plataforma només per estudiar, inclou jocs d'aprenentatge entre caràcters, lectura dels caràcters i significat dels caràcters més comuns perquè l'estudi d'aquest sigui el més atractiu possible a més de ser una primera presa de contacte tant pels principiants que comencen des de zero com els que ja han començat una mica a l'estudi del idioma i volen reforçar-n'he els caràcters més simples.

A més de la part de l'estudi de la llengua, inclou una altra part més enfocada a la geografia del país. És una altre forma d'aprendre els caràcters de totes les ciutats del país, a més de la seva localització i, a més, pots connectar-te a través de Google Maps per veure els carrers o familiaritzar-te amb aquests.

Aquesta aplicació és una de molts exemples que es poden crear aplicacions a través de AppInventor. El més destacat d'AppInventor és la seva accessibilitat: és accessible tant per la gent que mai ha programat com pels que si tenen uns coneixements mínims o amplis.

Paraules clau:

Android	**AppInventor**	**Aplicacions**	**Programació per blocs**
Programació Gràfica	**Comunicació**	**Lengua**	

ABSTRACT

The main goal of this project is to show one of the tools that we can use to create and design applications for Android Operating Systems, for mobiles and tablets.

In this case, particularly, I am going to dedicate AppInventor, a platform which you can create applications through blocks programming which you don't necessarily need advance knowledge in Java. If you have a minimum knowledge, you can create any application.

It has been designed an application about language learning platform because of that, you can see the different types of tools you can take. This particular platform is about Japanese Learning. It is not only a language learning platform to study; it includes also some games to learn the characters, read them and to know the meaning of the most commons ones. This one is the most attractive way to learn the language. Also, it is good for the people who are starting to learn the language without any knowledge and for the people that want to review the study of the characters.

In addition to language study, it includes a part about geography of the country. It is another way of learning the characters of the cities of the Japanese country. Also, you can view the localization in the map and connect connect through Google Maps to see the streets of a given region so you can get familiar with these ones.

This application is one of the examples that you can create through AppInventor. The most important part of the AppInventor it is accessible: It is for people that never had programmed or for people who have a minimum or extensive knowledge.

Key words:

Android	**AppInventor**	**Applications**	**Block programming**
Graphic Programming	**Communication**	**Language**	

SUMARI

En el primer apartat de tots, hi haurà la "Introducció", una breu explicació del projecte a més de mostrar l'objectiu principal d'aquest projecte.

En el capítol 2 "Android" s'explicarà què és Android i la seva influencia avui en dia, algunes de les alternatives per a programar aplicacions i, quines diferències hi ha en comparació amb el AppInventor i altres alternatives.

En el capítol 3 "Manual AppInventor" s'expondrà tot el manual per poder començar a utilitzar AppInventor, començant primer per com es crea un projecte, com es programa, quins elements gràfics hi ha a més de com testejar i instal·lar l'aplicació per fer utilitzar-la al dispositiu.

En el capítol 4 "Aplicació creada" s'explicarà l'aplicació creada a través de AppInventor amb les seves corresponents parts que la componen.

En el capítol 5 "Pressupost" s'explicarà les hores i diners invertits en el projecte.

En el capítol 6 "Conclusions" s'explicarà totes les conclusions finals d'aquest projecte a més de les perspectives que podria tenir aquest projecte.

Per últim, conclourem amb el capítol 7 "Bibliografia" utilitzada per a la realització d'aquest projecte a més del capítol 8 "Annexos" del projecte, com pot ser la base de dades i programació utilitzat i realitzat pel projecte.

ÍNDEX

1. Introducció 8
2. Android 10
2.1. Sistemes de programació 10
2.2. AppInventor 12
3. Manual AppInventor 16
3.1. Entorn AppInventor 16
3.1.1. Núvol i Local 16
3.1.2. Crear el projecte 16
3.2. Programació 18
3.2.1. Interfície part gràfica 18
3.2.2. Interfície part programació 34
3.2.3. Estructura Android 46
3.2.4. Estructura de pantalles de les aplicacions 46
3.3. Finalització 52
3.3.1. Descarregar còpia de seguretat del programa 52
3.3.2. Testejar l'aplicació al mòbil 52
3.3.3. Instal·lació de l'aplicació al dispositiu 55
4. Aplicació creada 57
4.1. Jocs 58
4.1.1. Creació de Test A; "Conèixer la pronunciació" 59
4.1.2. Creació de Test B; "Conèixer el caràcter" 67
4.1.3. Creació de Test C; "Conèixer el significat" 71
4.2. Geografia 75
4.2.1. Creació 75
5. Pressupost 80
6. Conclusions 84
6.1. Perspectives 84
7. Bibliografia 87
8. Annexos 89

1.INTRODUCCIÓ

En el primer apartat de tots, hi haurà una breu explicació del projecte a més de mostrar l'objectiu principal d'aquest projecte.

1. Introducció

L'objectiu principal del projecte és mostrar una de les eines necessàries per poder crear una aplicació pel dispositius que incorporen el sistema operatiu Android, compatible tant per mòbils, tablets i altres.

Avui en dia, Android ha crescut en molts aspectes i ha evolucionat molt. Ha sigut tota una revolució a nivell de mòbils i tablets. A més, és un dels sistemes operatius líders per aquests dispositius.

Gràcies a aquest èxit, hi ha moltes eines per poder crear la teva aplicació. Què s'ha de fer per crear-la? Android proporciona un entorn de desenvolupament lliure que, per a desenvolupar aplicacions, com a mínim s'ha tenir uns coneixements de nivell mig de Java. Java és el llenguatge de programació més utilitzat per crear qualsevol tipus d'aplicació, ja que els programes desenvolupats no depenen directament del sistema operatiu, sinó de l'interpret. També hi ha altres eines on no calen tenir grans coneixements d'aquestes i inclús tampoc s'ha de tenir molt de nivell de programació. Això si, es recomana que es tingui uns coneixement mínims, encara que sigui amb C per saber com funciona la metodologia de la programació.

En aquest projecte, parlaré d'una d'aquestes eines que no calen tenir un gran nivell de Java per crear i dissenyar la teva aplicació per a Android. En concret, és AppInventor.

AppInventor és una aplicació creada primer per Google i continuada per la MIT, on la seva forma de crear aplicacions és una mica diferent del que podem trobar en altres programes, ja que és un llenguatje gràfic. En aquest software, es programa gràcies al seu sistema de programació per blocs. Hi ha tot tipus de blocs: variables, booleans, etc. És molt gràfic i molt intuïtiu, gràcies a això, és accessible per a molts tipus de persones.

L'objectiu d'aquest projecte és l'estudi de la programació amb AppInventor, la estructura, el seu funcionament, i dissenyar una aplicació d'exemple amb diverses funcions per a que es pugui observar les possibilitats que ofereix l'entorn de programació de Appinventor per crear la aplicació.

L'aplicació és una plataforma per l'estudi del idioma. En aquest cas, el faré de l'idioma japonès.

2. ANDROID

En aquest capítol s'explica què és Android i la seva influencia avui en dia, algunes de les alternatives per a programar aplicacions i, quines diferències hi ha en comparació amb el AppInventor i altres alternatives

2. Android

Android és un sistema operatiu especialment creat per a dispositius amb pantalles tàctils, com poden ser avui en dia mòbils, tablets, fins i tot a televisions i rellotges, entre d'altres. Creat per Google l'any 2005, està basat en un nucli Linux, encara que no va sortir de forma oficial al primer dispositiu fins l'any 2008.

En aquest sistema operatiu hi ha una gran comunitat de desenvolupadors d'aplicacions on la poden penjar a la botiga oficial de Android: Google Play. Moltes d'aquestes són de franc. La gran majoria d'aplicacions són creats i desenvolupats a través del llenguatge de programació de Java.

Les característiques bàsiques de Android són diverses: Navegador Web, missatgeria, emmagatzematge, connectivitat, Bluetooth, reconeixedor de veu, videotrucada, tàctil, Google Play, suport de Java, streaming, entre altres. També inclou aplicacions molt utilitzades avui en dia que pertanyen a Google com pot ser Google Maps, Gmail o Youtube.

A mesura que han passat els anys, ha tingut diverses actualitzacions i versions (actualment la versió més recent és la 4.4).

Android no és l'únic sistema operatiu d'aquest tipus, avui en dia ja hi ha desenvolupats alguns com poden ser: iOS, Firefox OS, Windows, Blackberry o TIZEN. Totes són similars però cadascuna té les seves particularitats que les fan "úniques".

2.1. Sistemes de programació

A través del sistema operatiu Android es poden crear moltes aplicacions que sempre puguis tenir a sobre teu, al mateix mòbil per exemple. Gràcies als diferents sensors que poden incloure en un smartphone (Hardware), les possibilitats de comunicació (Wifi, Bluetooth, NFC, GSM, ...) i les característiques principals que conté el sistema operatiu Android (Software) es poden crear i desenvolupar moltes aplicacions de diferents tipus i per a diferents utilitats.

A més, un mateix desenvolupador/programador que domini el tema, pot crear una aplicació i pujar-la a Google Play per a que els usuaris de Android puguin baixar-la i tenir-la.

Android ha creat un gran impacte avui en dia gràcies a la tecnologia i utilitat que implica. Amb aquest impacte, avui en dia hi ha moltes alternatives a l'hora de crear aplicacions. No cal només saber Java, hi ha moltes més variables en joc. O tampoc cal saber Java del tot. Tots els entorns de programació creats per poder crear la teva aplicació són diferents i totes són valides com qualsevol altre. Només que per una sèrie de característiques, a una persona li pot anar millor un programa que un altre.

Explicarem algunes de les més utilitzades i conegudes avui en dia:

- **Android SDK:** És el programa oficial de Android i la base. Per tant, els programes estan basats en Android SDK. Això vol dir que si hi ha una actualització nova, la primera en sortir serà per a Android SDK encara que la resta la afegiran tard o d'hora.
  En aquest programa si que cal conèixer el Java, o tenir nocions més que bàsiques. Hi ha una gran llista de llibreries que pots incloure les que vulguis, depenent dels usos que vulguis fer i quines aplicacions vols crear. Inclou a més un emulador de mòbil per comprovar que funciona al dispositiu.

  + *Avantatges:* Programa oficial, sempre actualitzat, gratuït.
  - *Desavantatges*: Sense coneixements de Java no es pot fer gran cosa.

- **Titanium Studio:** És també un altre programa on pots crear i desenvolupar aplicacions per a Android. També és important tenir coneixements del llenguatge de programació de Java (per a Android) o també de C (per a iOS). La gran particularitat que té és que pot convertir les aplicacions creades per a un sistema operatiu, passar-la també per a que sigui compatible per altres sistemes operatius com són: iOS, Windows o BlackBerry.

  + *Avantatges:* Pots fer una aplicació per a un sistema operatiu i el mateix Titanium Studio pot fer la conversió per a que es pugui utilitzar en altres sistemes operatius.
  - *Desavantatges*: És de pagament, si no tens coneixements de C o Java no pots fer gran cosa.

- **Mobincube:** Un programa per crear aplicacions però aquesta té la particularitat de que es creen ja a través de la seva pàgina web online, no cal descarregar cap programa a l'ordinador com en els casos anteriors, només cal crear-se un compte i ja pots accedir al programa per crear aplicacions. Es poden fer també per diverses plataformes com Android, iOS, Windows, etc.

  + *Avantatges:* Gratuït, el programa és online, no cal tenir coneixements de programació.
  - *Desavantatges*: A les aplicacions surt publicitat, per a alguns serveis concrets sí és de pagament, les aplicacions ens poden fer fins a una certa limitació.

- **Nimbo:** Un altre programa per crear aplicacions molt semblant a Mobincube. Funciona a través de la seva pàgina web online i no cal descarregar cap programa. Amb la seva interfície creada, tampoc cal tenir grans coneixements de Java ni de cap llenguatge de programació.

  + *Avantatges:* Gratuït, el programa és online, no cal tenir coneixements de programació.
  - *Desavantatges*: Surt publicitat, per a alguns serveis concrets sí és de pagament, les aplicacions ens poden fer fins a una certa limitació.

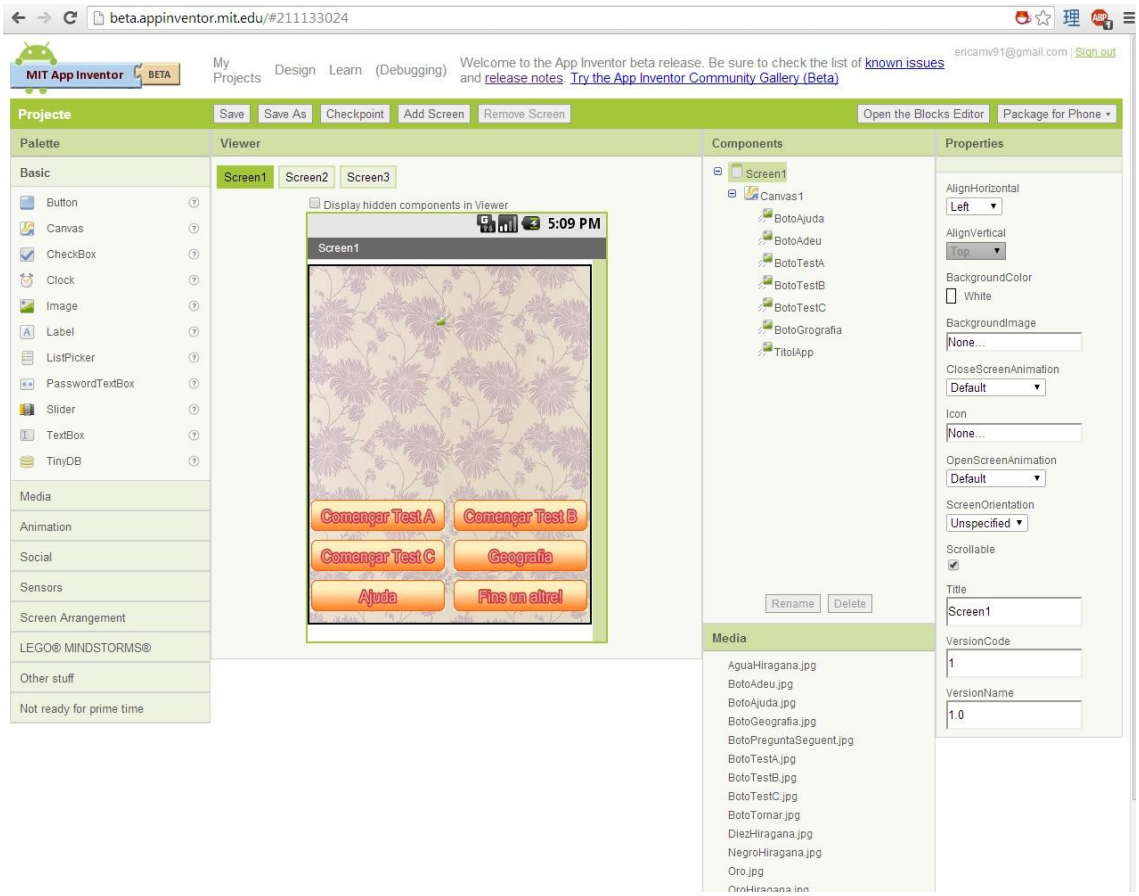
2.2. AppInventor

AppInventor és una aplicació per a la programació d'aplicacions per a Android, creada primer per Google, i desenvolupada actualment per la MIT, on la seva forma de crear aplicacions és una mica diferent del que podem trobar en altres programes.

En aquest software es programa gràcies al seu sistema de programació per blocs. Hi ha tot tipus de blocs: variables, booleans, etc. És molt gràfic i molt intuïtiu i, gràcies a això, és accessible per a molts tipus de persones.

+ *Avantatges:* Gratuït, el programa és online, programació gràfica, no porta publicitat.
- *Desavantatges*: Les aplicacions ens poden fer fins a una certa limitació, encara està en fase Beta.

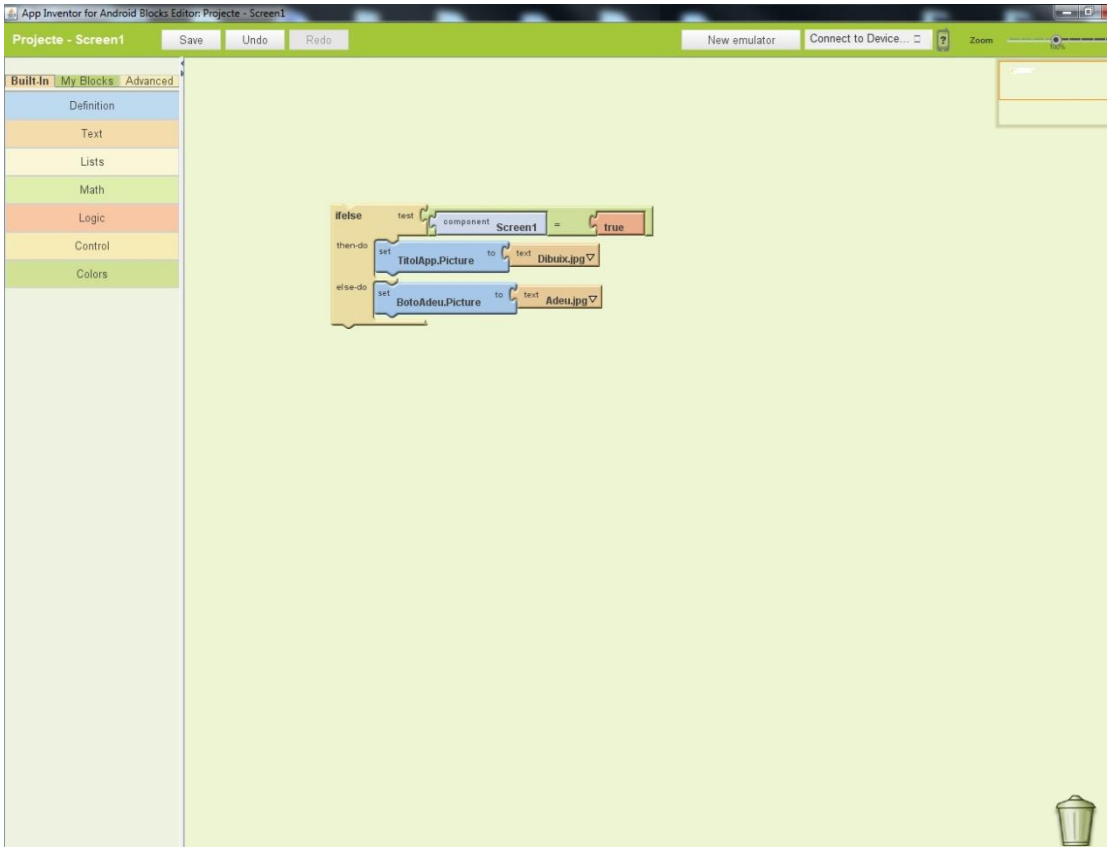
Veient els avantatges i desavantatges, he escollit aquest programa ja que comparat amb els altres programes explicats, aquest és gratuït i online, un punt molt positiu. La programació gràfica fa que sigui accessible per a moltes persones i, a més, no porta publicitat.



Imatge1

L'aplicació compta amb 2 parts ben diferenciades:

- ***Part gràfica:*** Aquí tenim el que seria la part del disseny de l'aplicació. A la mateixa pantalla que ens hi surt, tal i com podem veure a la *Imatge1*, tot el que s'afegeix sortirà igual al móbil/tablet. Hi ha moltes possibilitats. Com podeu veure, tenim la part esquerra on hi ha les diferents opcions de què pots ficar: Imatges, botons, textos, rellotge, ... Un cop posat allò que desitgem a la pantalla, si clickem allò que hem posat, la part dreta es modificarà depenent de l'objecte posat en qüestió. Aquesta part de la dreta serveix per modificar els paràmetres que hem possat. Per exemple, si posem una imatge, a la part dreta podem modificar la mida d'aquesta, posar una imatge en qüestió (la puges des de l'ordinador), posar un color en concret o les coordenades d'on vols que estigui la imatge.



Imatge2

- ***Part programació:*** Aquesta part és on es fa tota la programació per blocs, com veiem a la *Imatge2*. Depenent quins paràmetres es posin a la part anterior, pots tenir variables noves o no. Aquesta pantalla s'inicialitza a partir de l'anterior, clickat a la part de dalt a la dreta, al botó de "Open the Blocks Editor". Hi ha 3 pestanyes: Built-in, My Blocks, Advanced. En Built-in, és on està tota la programació bàsica: afegeixes textos, booleans, símbols matemàtics, etc. A la part de My Blocks, estàn totes les variables ja creades a partir de la part de Disseny com pot ser, per exemple, un botó. Quan poses un botó a la pantalla,

directament en My Blocks d'afegeix el nom de la variable que li has donat al botó. A partir d'aquí, pots modificar les seves variables per a que faci x acció.

NOTA: Pots utilitzar ambdues pantalles a l'hora. És més, es recomana fer-ho ja que així ho tens tot a mà. A més, qualsevol modificació que facis a la part de Disseny, directament es modificarà a la part de Programació.

Una altre particularitat del AppInventor és que pot funcionar de 2 formes: pujar-ho al núvol (tens sempre aquesta informació a mà, per quan canviïs d'ordinador) o fer-ho a través del Local (la informació es queda a l'ordinador d'on has treballat).

3. MANUAL APPINVENTOR

En aquest capítol s'exposa tot el manual per poder començar a utilitzar AppInventor, començant primer per com es crea un projecte, com es programa, quins elements gràfics hi ha a més de com testejar i instal·lar l'aplicació per fer utilitzar-la al dispositiu.

3. Manual AppInventor

Abans de començar amb el AppInventor, s'ha de tenir en compte una sèrie de temes que són principals pel perfecte funcionament de l'aplicació:

3.1. Entorn AppInventor

3.1.1. Núvol i Local

Com hem comentat anteriorment, una de les grans particularitats del AppInventor són els 2 entorns de treball diferents: núvol i local. Depenent de quines són les teves preferències, pots escollir una o una altre.

- Núvol: Com hem dit anteriorment, fa que tot el que facis, incloguis, modifiquis, etc..., pugi directament al núvol. Això fa que sigui molt útil ja que com es guarda tot automàticament a través del núvol, quan entris directament a la web (http://beta.appinventor.mit.edu/) des de qualsevol ordinador, ja pots entrar amb el teu correu electrònic i tot estarà allà. L'únic problema que hi ha fer-ho a partir d'aquí és que es limita a 2Mb.

- Local: Aquesta és una altre manera de fer el projecte amb el AppInventor. L'aplicació es troba en l'ordinador propi o "*Local*". La part positiva que té fer-ho a través del local és que no té límits de capacitat de Mb. La part negativa és, com hem explicat abans, no puja res al núvol i, per tant, si volem continuar treballant en aquest projecte amb un altre ordinador, s'ha de fer una còpia de seguretat per poder-ho pujar al "*Local"* de l'altre ordinador i poder continuar.

  En aquest cas abans, però, hem d'instal·lar una sèrie d'arxius que ens ho donen en un .rar descarregable que podem trobar a través de la pàgina web oficial de AppInventor a més d'un petit tutorial.
  Un cop fer això i ja hem instal·lat, ja podem anar a "http://localhost:8888/" i continuar! Només cal posar el correu electrònic Gmail i ja pots començar de nou o continuar el projecte ja fet anteriorment, depenent del que vulguis fer.

Un cop escollit quin és l'entorn que ens aniria millor, agafem qualsevol de les 2 i podem començar a crear el nostre projecte.

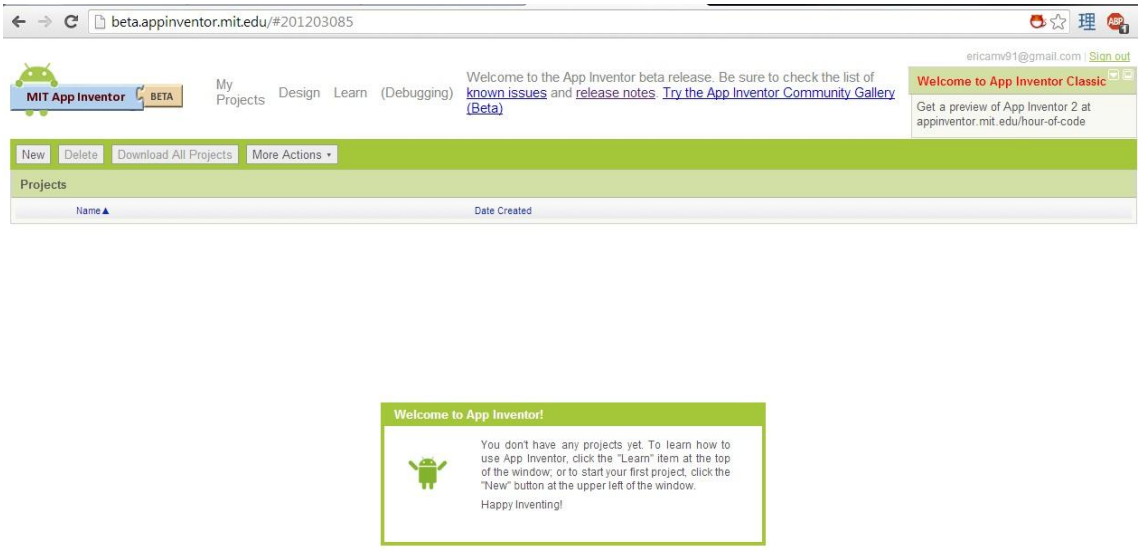
3.1.2. Crear el projecte

Ja hem decidit el nostre lloc de treball (núvol o local) i ara comença la creació del projecte.

Per començar a utilitzar AppInventor, el primer que hem de fer és entrar a la pàgina web on està l'aplicació (http://beta.appinventor.mit.edu/) i ens portarà al que mostra a la *Imatge3.*

Per poder entrar a l'aplicació, s'ha de tenir un compte Gmail. En el cas de que no tingueu, ens facilitarà un enllaç fàcil per crear-ne. Si en tenim, entrem amb el nostre correu i contrasenya.

Un cop que entrem, ens sortirà una pantalla com a *Imatge3*:



Imatge3

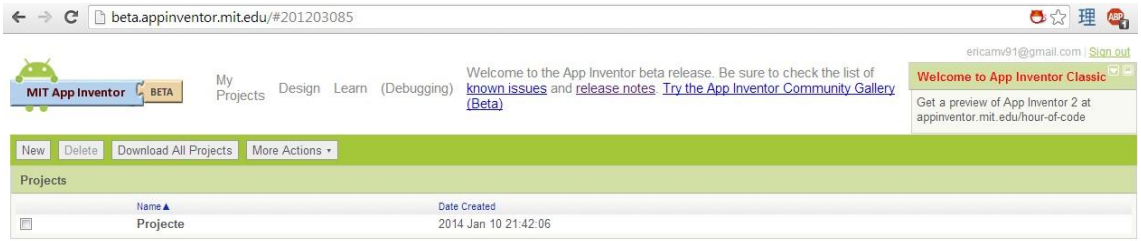
En aquesta pantalla, sortiran tots els projectes creats i on crees un nou projecte.

Per començar el nou projecte, només cal clickar a "New", ens demanaran un nom per nou projecte com veiem a la *Imatge4*.

Imatge4

Un cop donat el nom del projecte amb el botó "OK" ens sortirà per fi el que seria l'aplicació.

En aquest cas, he creat un nou projecte anomenat "Projecte". Veure a la *Imatge5*.



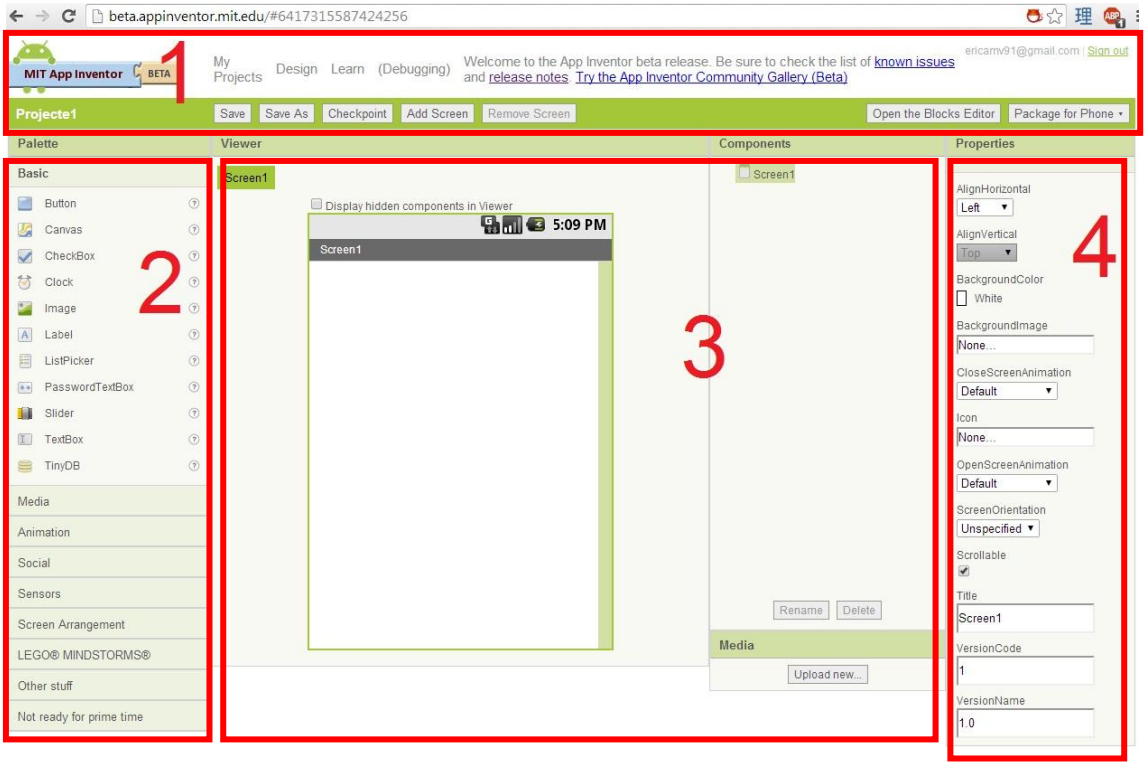
Imatge5

3.2. Programació

Ja tenim el projecte creat i tenim el nostre espai de treball, la part gràfica a més de la part de programació. El primer que hem de fer és començar per la part gràfica ja que hem d'afegir tots els elements necessaris per l'aplicació per així, després, poder-ho programar a la part de programació.

3.2.1. Interfície part gràfica

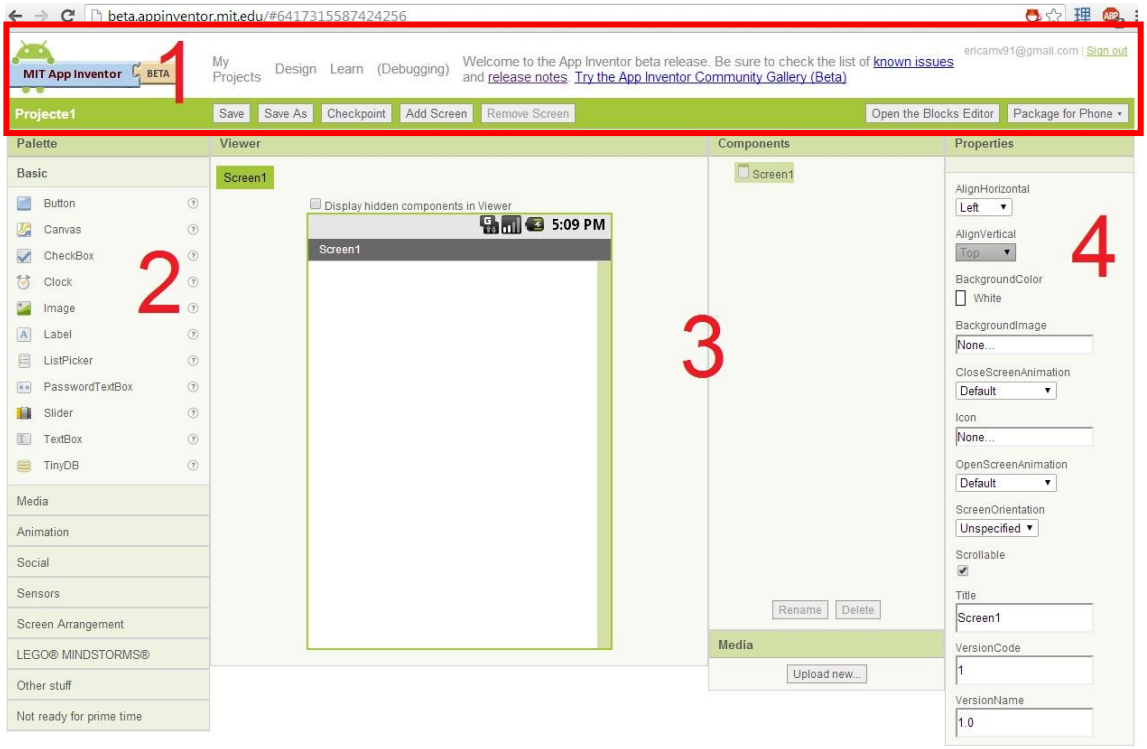
En aquest apartat és on posem totes les variables i elements que ha de contenir l'aplicació per al seu bon ús i funcionament que volem donar. Comencem per veure quines possibilitats i elements ens donen per la part gràfica (*Imatge6*).



Imatge6

Part 1:

Comencem per definir les característiques i utilitats que té la part 1 de la interfície gràfica, anomenat la barra pel mateix nom del projecte que li donem (en aquest cas: Projecte1):



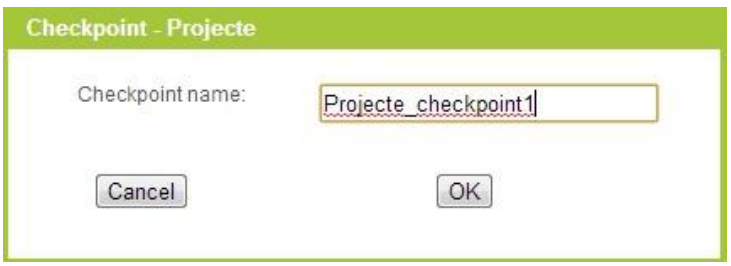
Per començar a fer el projecte, primer que hem de saber fer unes utilitats importants.

Primer, hem d'observar la barra que hi ha dalt de la part on estan les Screen, que són les diferents pantalles de l'aplicació, a *Imatge7*:

Imatge7

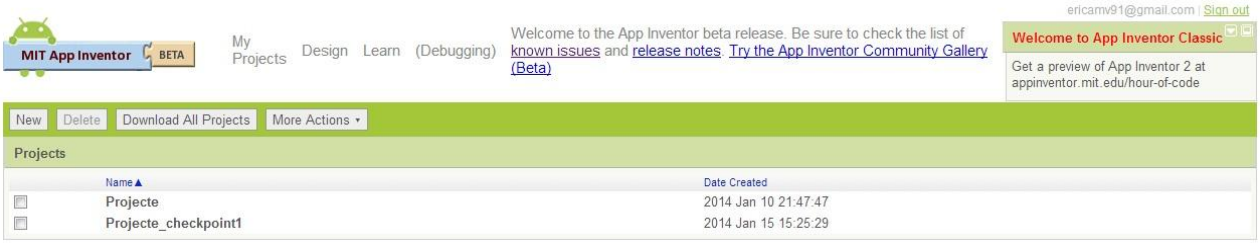
En aquesta barra amb el "Save" i "Save As" podrem guardar el nostre procediment fet.

El "Checkpoint" serveix per guardar també però en aquest cas fa com una còpia de seguretat i es guarda a la pantalla principal de AppInventor. Quan el polsem, el que ens farà serà sortir una petita pantalla com *Imatge8*:



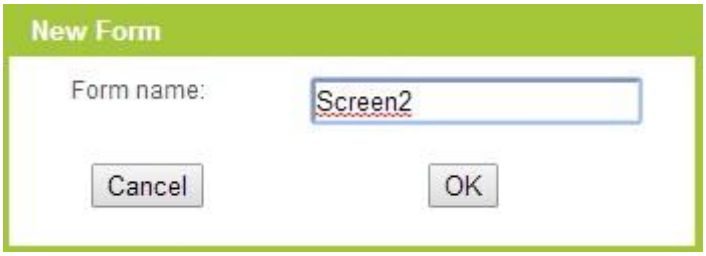
Imatge8

Ens indicarà un nom nou per no confondre-ho amb el Projecte en si. En aquest cas, ens dóna per determinat "Projecte_checkpoint1". Ho acceptem i a la pantalla principal ens sortiria una cosa com la *Imatge9*:



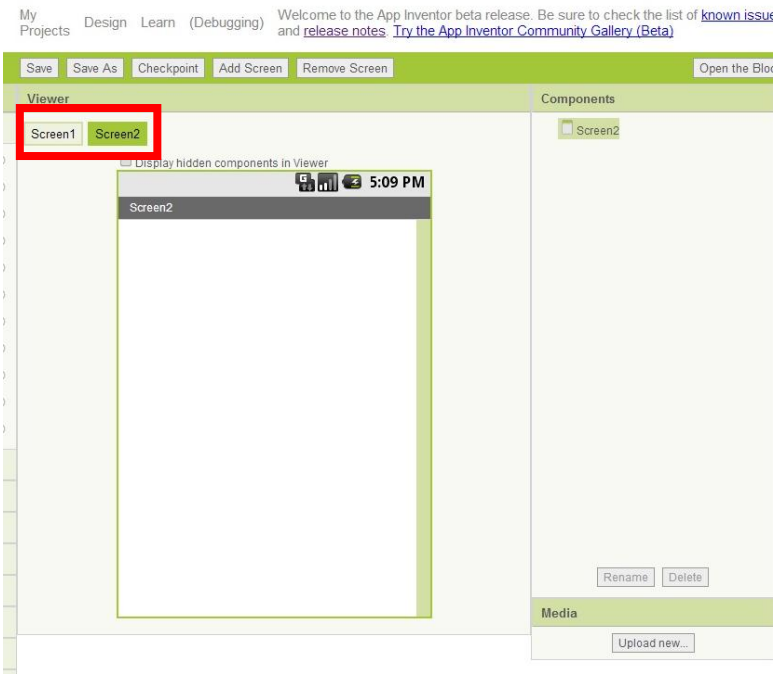
Imatge9

Per últim, tenim l'opció de "Add Screen". Amb aquesta opció, podem crear tantes Screens com volguem. Quan clickem, ens sortirà l'opció de donar un nom a la Screen (Imatge10) que, en aquest cas, la deixaré amb al mateix nom.



Imatge10

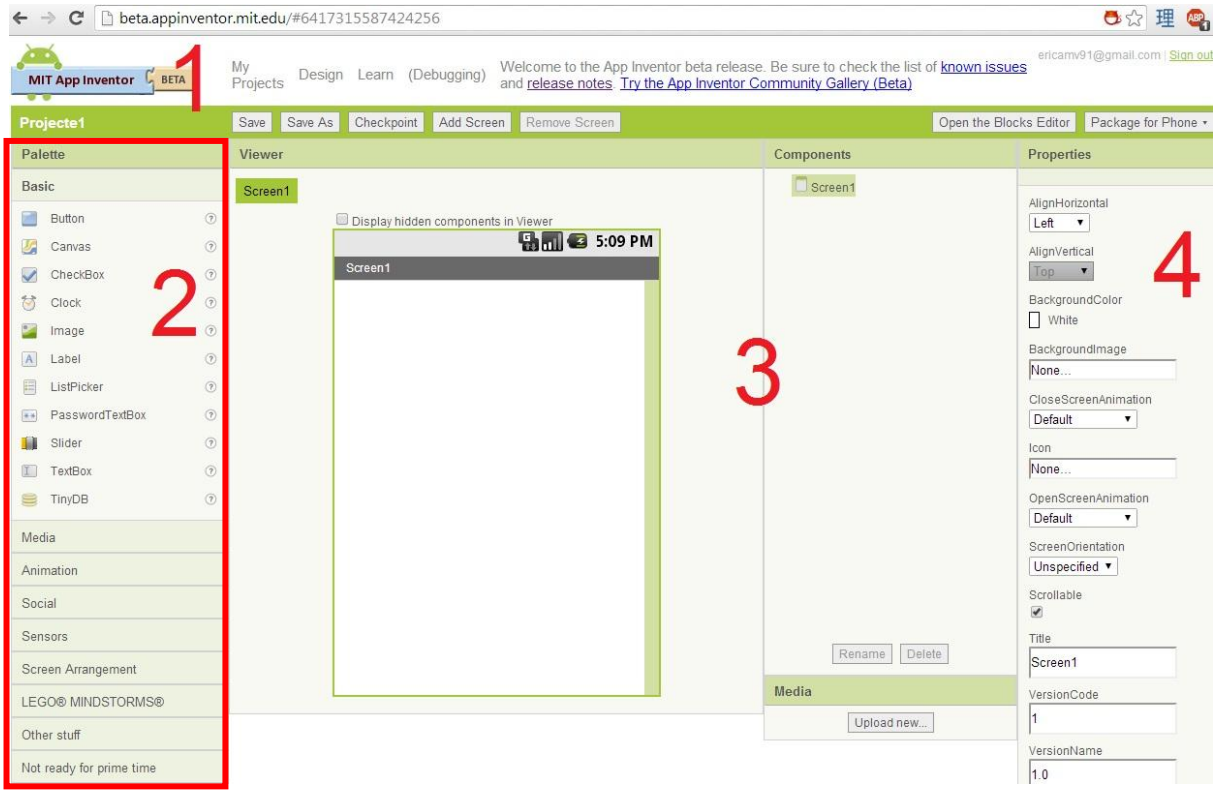
Un cop acceptem, veiem a la pàgina principal que ens ha afegit la Screen2 (*Imatge 11*):



Imatge11

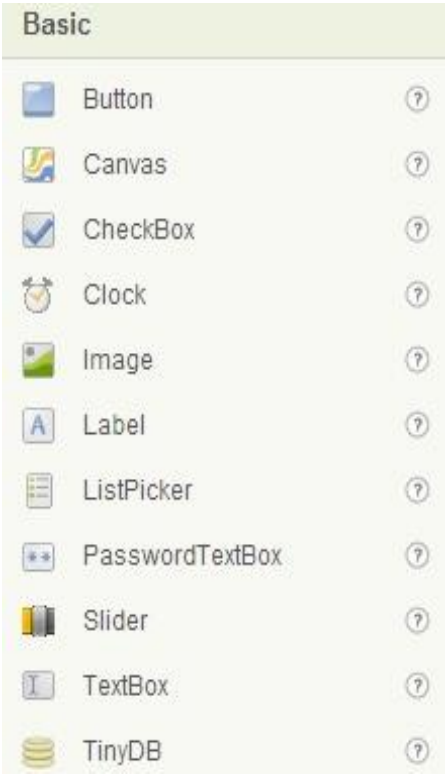
Part 2:

Ara passem a l'explicació dels elements que podem trobar en AppInventor, a l'apartat de Palette (*Imatge12*).



Imatge12

El Palette consta de:



Basic

Aquesta és la secció on hi han els elements més utilitzats de forma general ja que són de gran importància a més de poder tenir un gran pes a l'aplicació en si. Aquí podem col·locar botons, imatges, etiquetes, documents per a texts, entre altres de gran utilitat.

Aniré comentant principalment cadascuna d'elles i amb quina icona ens apareixeria en la pantalla del dipositiu que ens donen per a fer el disseny de l'aplicació. S'ha de tenir en conte, que hi ha elements que tenen part visible o que no en tenen. En AppInventor, com ho sabem? Ho sabrem perquè quan afegim l'element, no va a la pantalla, estarà a sota de la pantalla.

- *Button:* El Button és per col·locar un botó. Aquest botó pot modificar-se després gràcies a altres eines que en donen que explicarem més endavant.

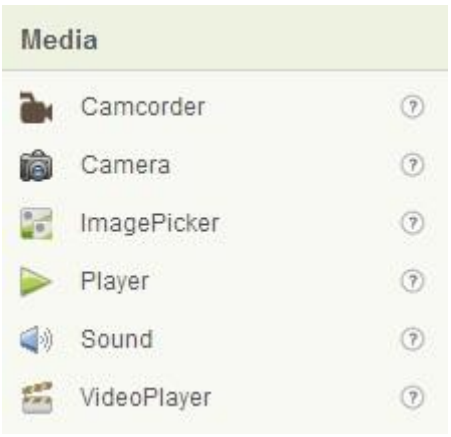
- *Canvas:* Aquest element serveix per posar després diversos elements a sobre al teu, no cal que sigui tan estricte com passa amb altres elements que exigeixen estar en un lloc específic i no el pots moure tant.

- *CheckBox:* Per col·locar un títol amb un quadrat a l'esquerra on pots clickar-hi i que una creu com que està marcat.

- *Image:* Amb aquest podem posar imatges

- *Label:* Aquí podem ficar etiquetes i títols específics.

- *PasswordTextBox:* Per poder escriure qualsevol cosa i surten els estèrics com si fos una contrasenya.



Media

Aquests elements són els necessaris per ficar arxius o fer aplicacions relacionades amb la multimedia, com pot ser so i imatge.

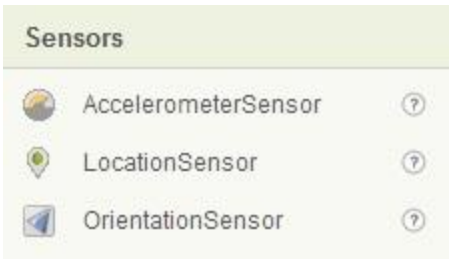
Animation

Elements necessaris per fer qualsevol tipus d'animació, com pot ser canviar una imatge de lloc.

Per aquests elements, s'ha d'especificar que ha d'haver el Canvas

Social

Aquesta part inclou elements per poder col·locar el correu electrònic, contactes de telèfon i altre com poden ser les xarxes socials (en aquest cas, Twitter).

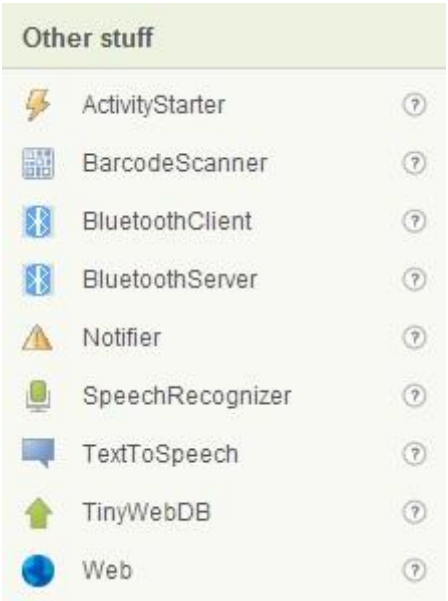


Sensors

Elements necessaris per aprofitar i amortitzar el hardware que ve inclós a la gran majoria de Smartphones.

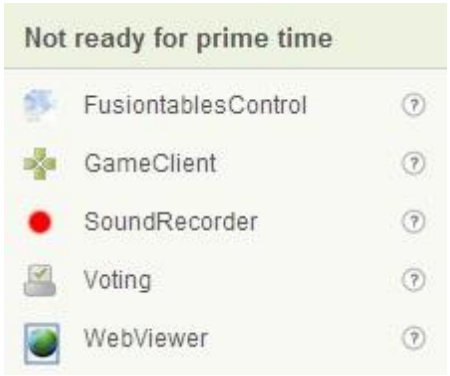
Screen Arrangement

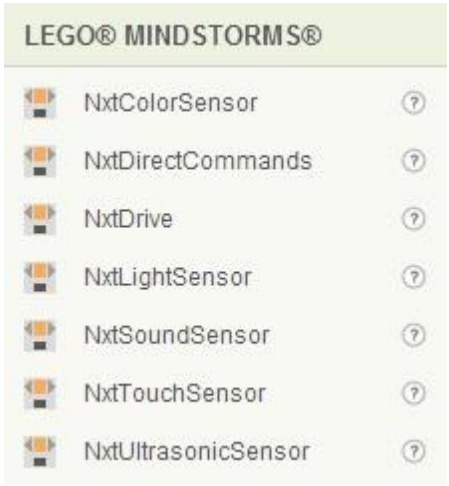
Aquesta part és per tenir en conte l'espai utilitzat a l'aplicació ja que la seva funció és organitzar i adaptar l'aplicació a les mesures que tingui la pantalla del dispositiu on executis l'aplicació.



Other stuff

Aquí podem trobar altres elements importants però per fer servir ja temes més concrets com pot ser Google Maps (gràcies al ActivityStarter), Bluetooth, reconeixedor de veu, entre d'altres.



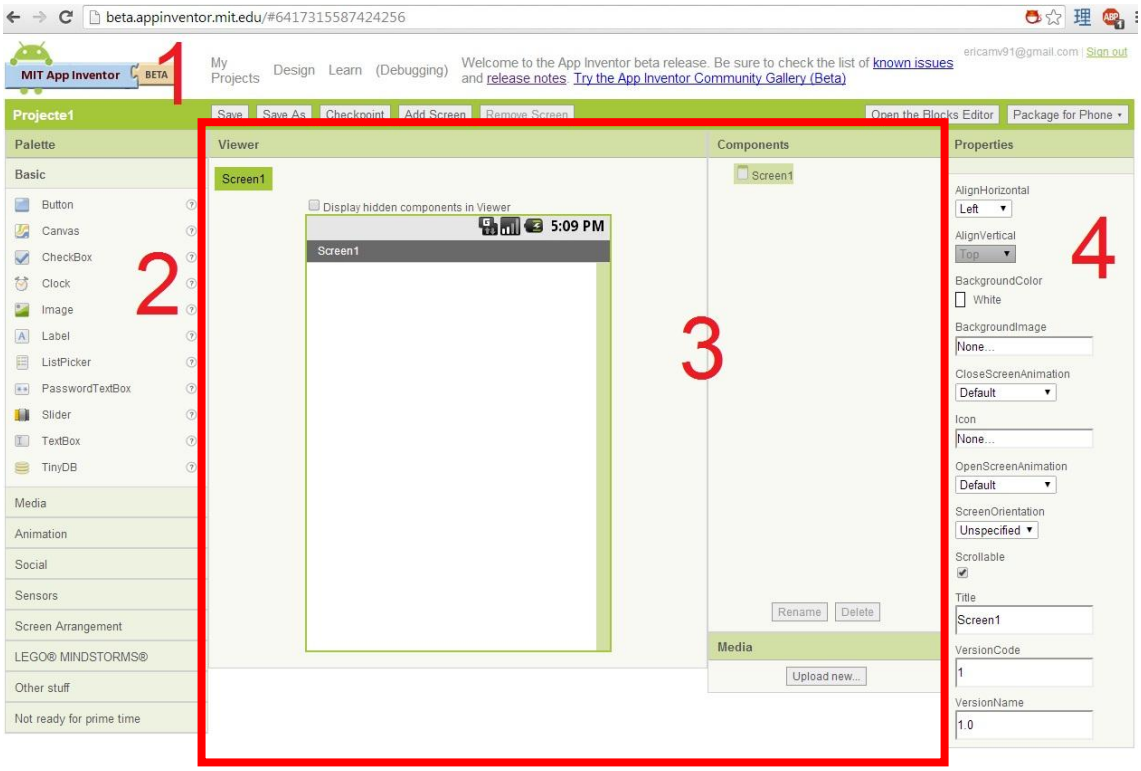


LEGO MINDSTORMS

Un altre punt a destacar entre els elements, i molt favorable per programar a través d'AppInventor, és la seva opció de poder fer programes amb els robots Lego MINDSTORMS. A través d'aquí podem utilitzar i aprofitar les diferents característiques que ofereix com pot ser el sensor de color, de llum, de so o de tacte entre altres.

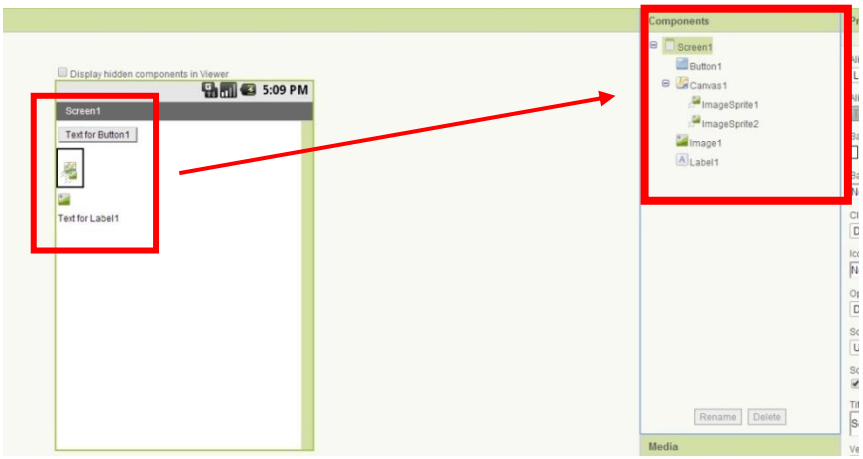
Part 3:

La Part 3 consta de 2 parts: Viewer i Components. Aquí és on podem veure una vista (Viewer) de com es mostraria la Screen corresponent per la pantalla del propi dispositiu, a més de veure tots els elements col·locats de la Palette en aquesta pantalla (Components).

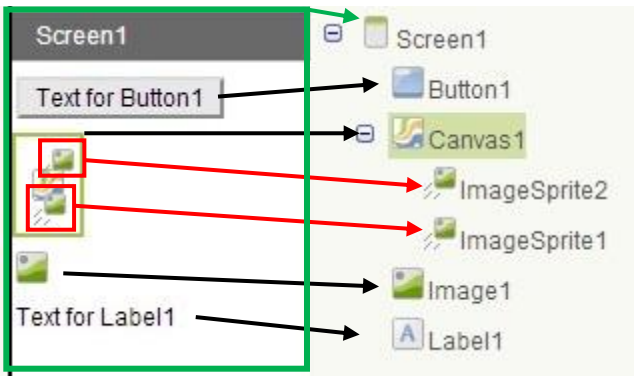


Imatge13

És la part de disseny de la creació de l'aplicació. A la part més dreta podem veure els diferents element que col·loquem com podem observar a l'exemple de la *Imatge14* en el quadrat vermell.

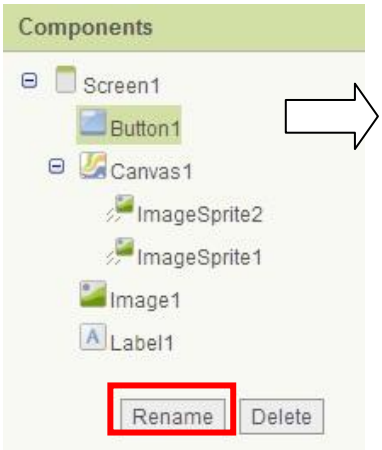


Imatge14



Imatge15

En la següent gràfica podem veure els elements que hem ficat de la Palette a Viewer i en que es transformen a la part de Components. El nom que ve per defecte és el mateix nom que surt a la Palette a més d'un número. Per exemple, si col·loquem 2 canvas, sortirà a Components Canvas1 i Canvas2. Tot i així, podem canviar el nom si clickem al component que volem canviar el nom i a sota et surt l'opció de "Rename".



Clickar al Component que volem canviar de nom i després clickar a "Rename".

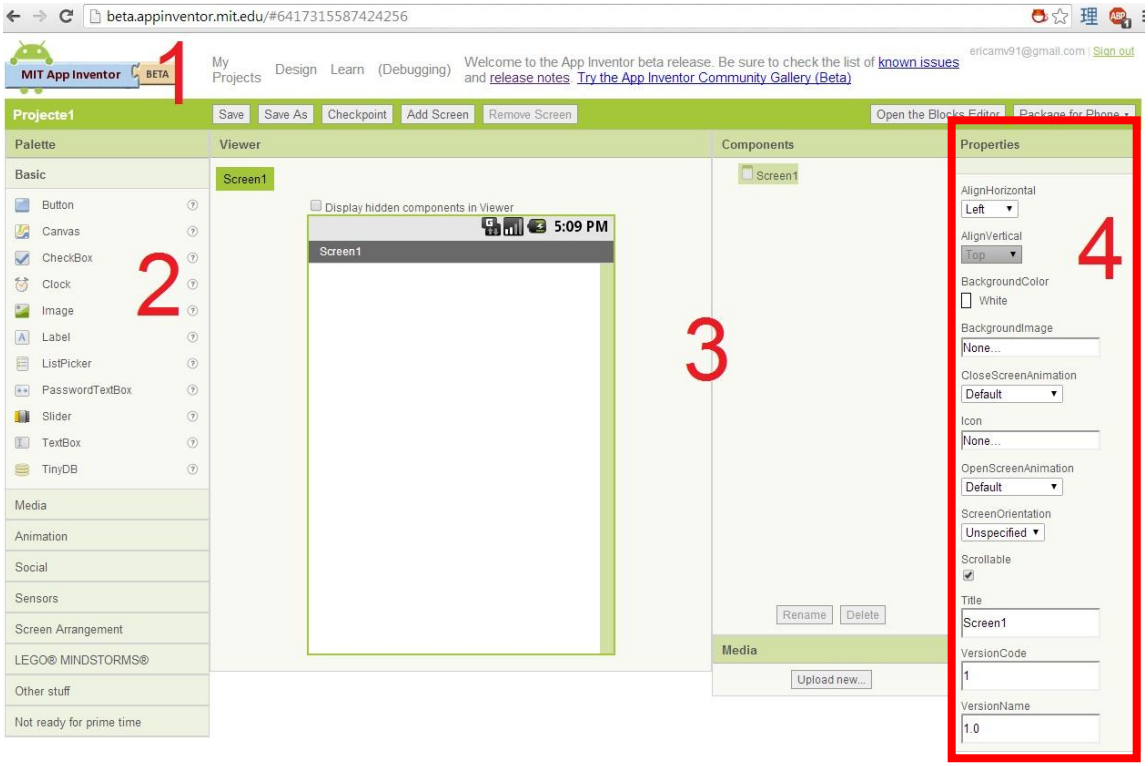
Posar el nom que volem. En aquest cas serà "BotoOK".

Ja podem veure el nom canviat a Components.

Pels noms escrits no s'ha d'escriure cap accent ni cap símbol ortogràfic ja que el nou nom que posem és el nom de la nova *variable* del Component. Molt important saber-ho ja que a la següent part, a programació gràfica, ens sortiran aquestes variables (amb el nom que hem anomenat) i hem de saber quins noms són els que posem.

Observem que a la pantalla del dispositiu hi ha diferents elements i que a la part més dreta es defineixen tots. També podem observar a la *Imatge16* que estan ordenats i que, per exemple, dins del Canvas1 hi ha una ImageSprite1 i una ImageSprite2 perquè, com sabem, aquest tipus d'element concretament, només es col·loca si hi ha un Canvas.

Imatge16



Imatge17

Part 4:

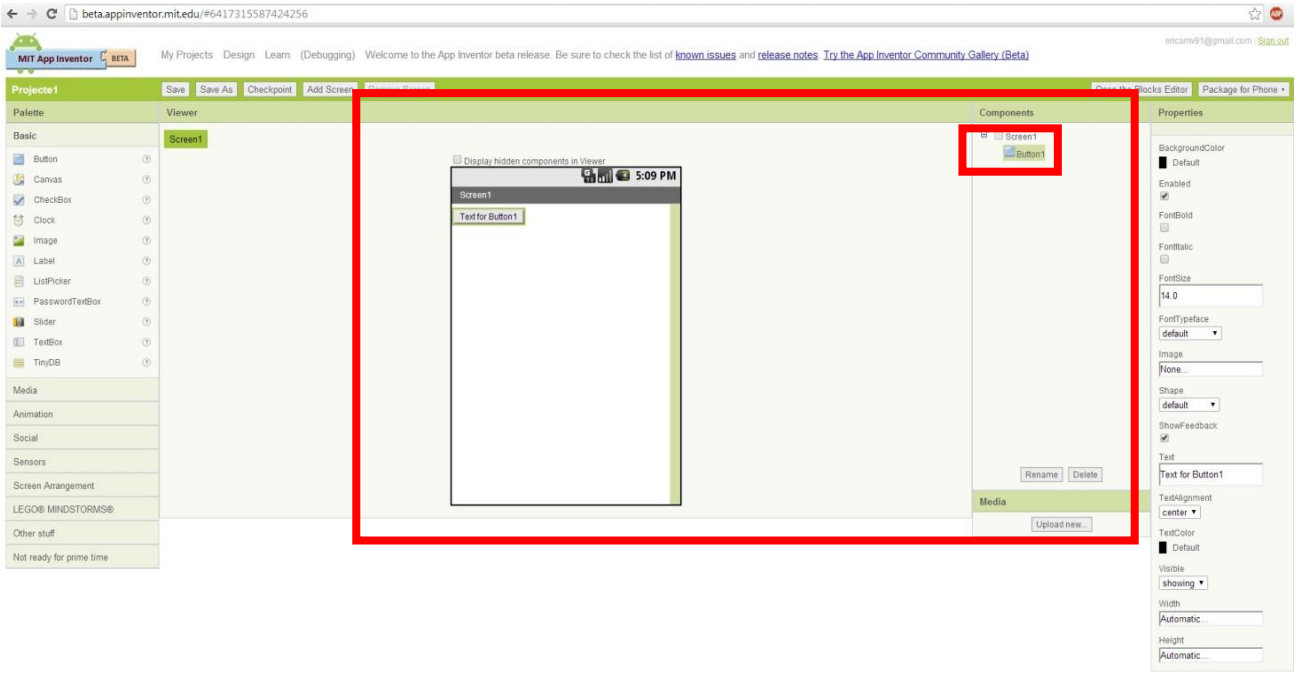
Aquesta és la part on hi ha la taula de Properties (*Imatge17*), on es modifiquen les característiques dels elements col·locats al Viewer a través de la Palette, les que hauríem d'haver afegit anteriorment.

Per cada element hi ha diverses característiques a modificar però hi han algunes que molts vegades es repeteixen en la majoria d'elements, com poden ser:

- Picture
- BackgroundImage
- Text
- PaintColor
- Rotates/visible
- X/Y/Z
- Width/Height

Mostrem una llista del que podem modificar amb diferents elements ja col·locats gràcies a la Palette vista a la Part 2.

Button: Per poder modificar les seves característiques, hem de clickar a la part on posa Button de la part de Components com veiem a la *Imatge18* i *Imatge19* i, a la part dreta de la part gràfica, podem veure que las Properties han canviat i ens donen les següent opcions de la *Imatge20.*



Imatge18

Imatge19

Veiem que està marcat el Button i, a la part de Properties veiem:

Imatge20

Qualsevol de les modificacions que es realitzin a Properties s'aniran actualitzant i canviarà la seva forma al Viewer.

- Background Color: Canvia el color interior del botó.

- Enabled: Habilita la interacció amb el botó.

- FontBold: Per a que la lletra estigui en negreta.

- FontItalic: Per a que la lletra estigui en cursiva.

- FrontSize: Per definir l'amplada del botó.

- FontTypepeface: Per canviar el tipus de lletra.

- Image: Per col·locar una imatge que vulguis a dins del botó.

- Shape: Per canviar la forma del botó.

- ShowFeedback: Per a mostrar un missatge si el botó conté una imatge.

- Text: Per canviar el text que hi ha a l'interior del botó.

- TextAlignment: Per col·locat el text a la dreta, esquerra o centre del botó.

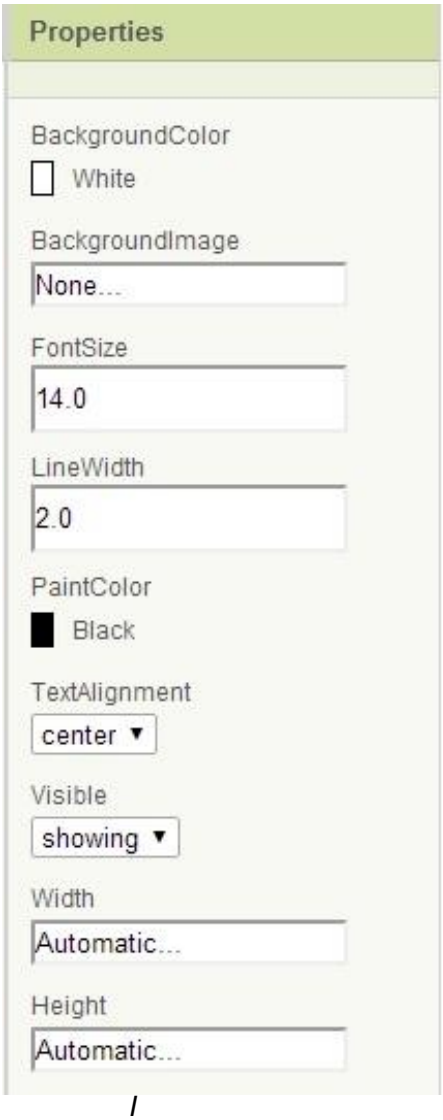
- TextColor: Per definir un color al text de dins del botó.

- Visible: Per amagar el botó de la pantalla Viewer.

- Width/Height: Per col·locar una mida d'alçada i d'amplada. Es pot posar valors en pixels.

Canvas: En aquest cas, si clickem a Canvas1, les característiques que es poden canviar no coincideixen amb els de Button, són més diferents.

Imatge21



I

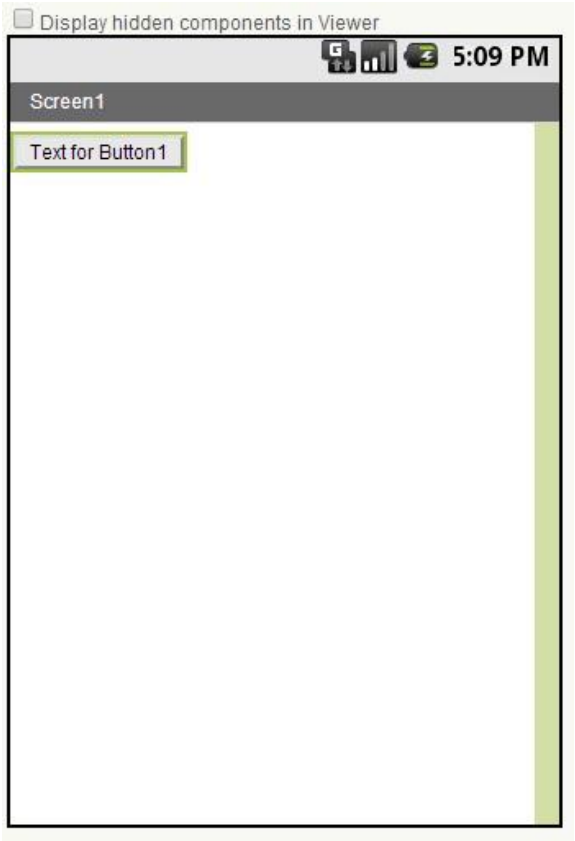
Imatge22

- Background Color: Canvia el color interior del Canvas.
- Background Image: Per col·locar una imatge que vulguis a dins del Canvas.
- FontSize: Mida del text.
- LineWidth: Ample de la línia externa del Canvas.
- PaintColor: Color del Canvas.
- TextAlignment: Alineament del text.
- Visible: Per amagar el botó de la pantalla Viewer.
- Width/Height: Per col·locar una mida d'alçada i d'amplada. Es pot posar valors en pixels.

Passem a fer un exemple, un pel Button, per veure com es poden modificar les seves característiques.

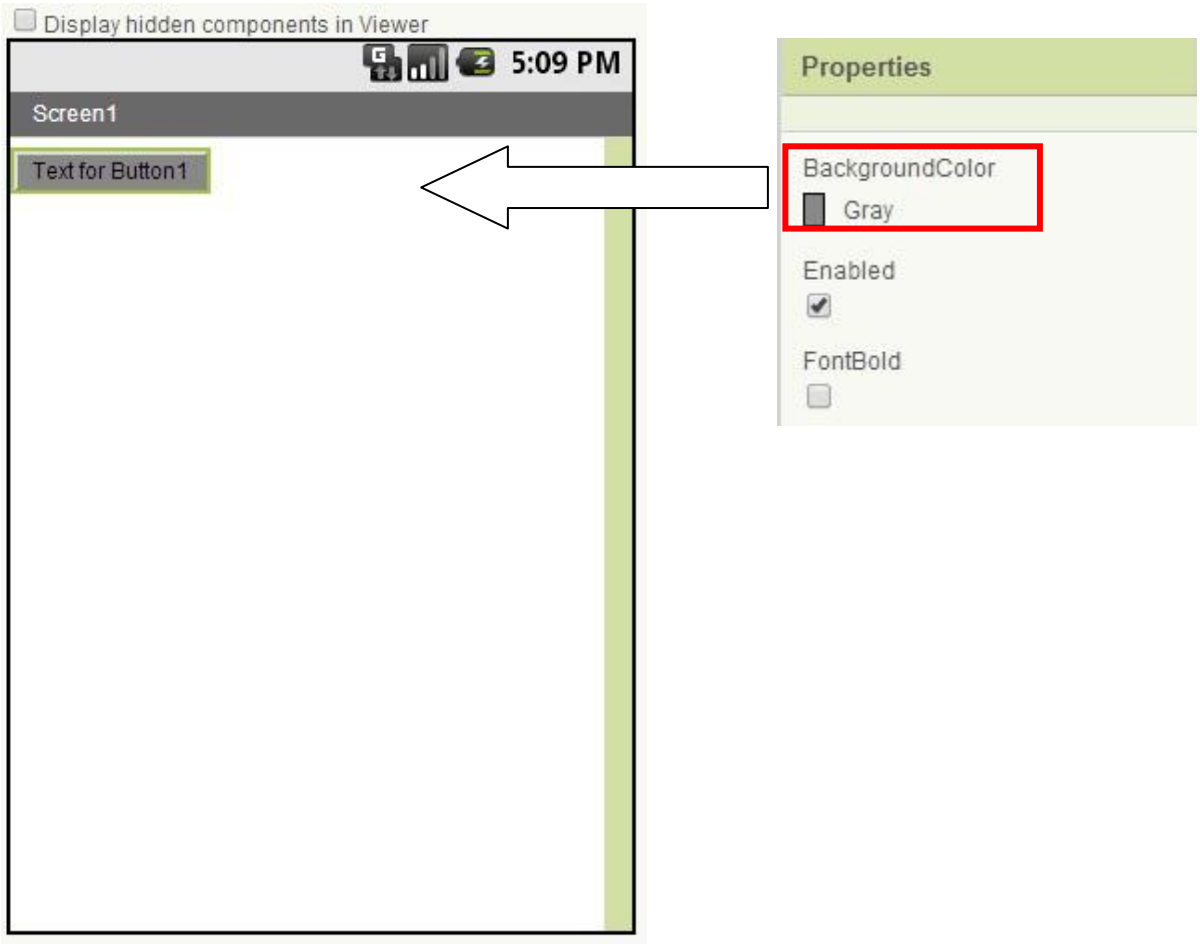
Volem aconseguir que quedi com aquest resultat:

Primerament, si arrastrem el Button de la Palette a la pantalla View, tindrem la *Imatge23*:



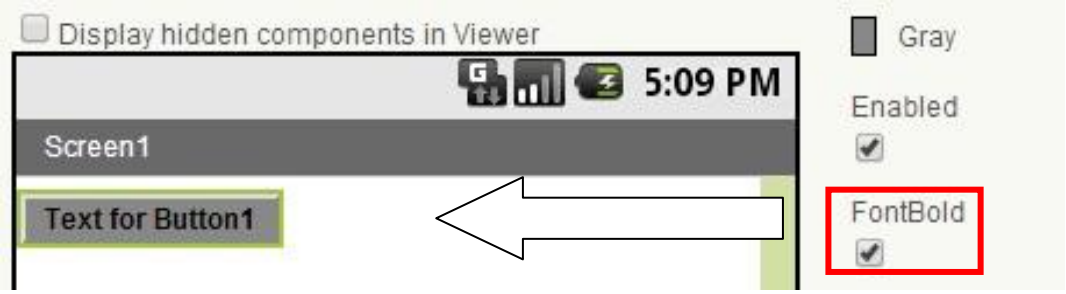
Imatge23

Comencem a modificar les Properties del Button per canviar-la al nostre gust. Per exemple, seleccionarem a BackgroundColor el color desitjat. En el cas dels botons, agafaré un color gris. Un cop es selecciona, ja ens modifica a la pantalla el que hem escollit, com podem veure a la *Imatge24*:



Imatge24

A continuació, ens podria interesar posar les lletres del text de dins del botó en forma cursiva o en negreta. En aquest cas, la deixaré en negreta



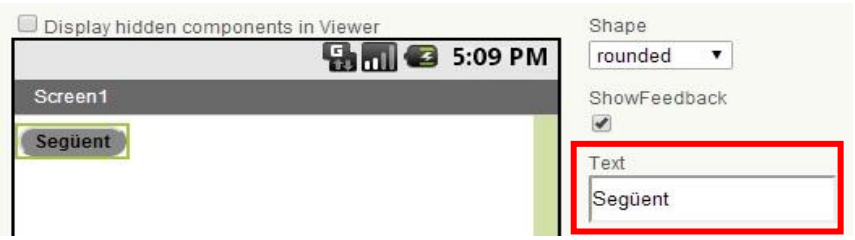
Imatge25

Continuem amb les Properties que podem encara modificar. Podem canviar el tipus de lletra del text que hi ha a dins del botó. En aquest cas el deixarem tal i com està.

Imatge26

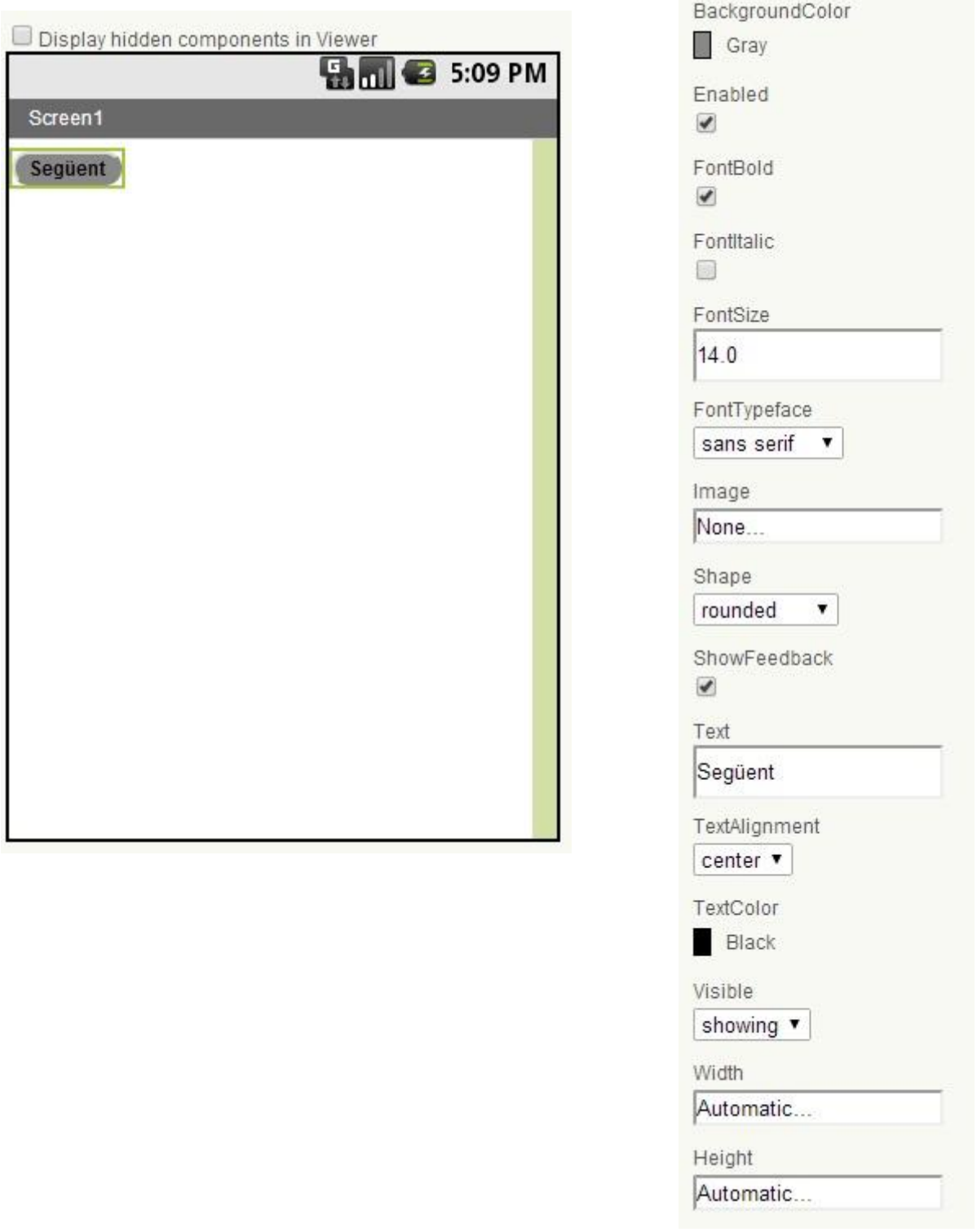
Podem modificar la forma del botó, en aquest cas he optat per modificar-la i que tingui forma ovalada. Ho canviem en la part corresponent, a Shape, i decidim la forma.

Ara podem canviar també el text que volem que hi hagi a dins del Button. En aquest cas, hem decidit escriure "Següent". El resultat quedaria com a la *Imatge27.*



Imatge27

El resultat final amb les Properties finals serien iguals que els de la *Imatge28.*

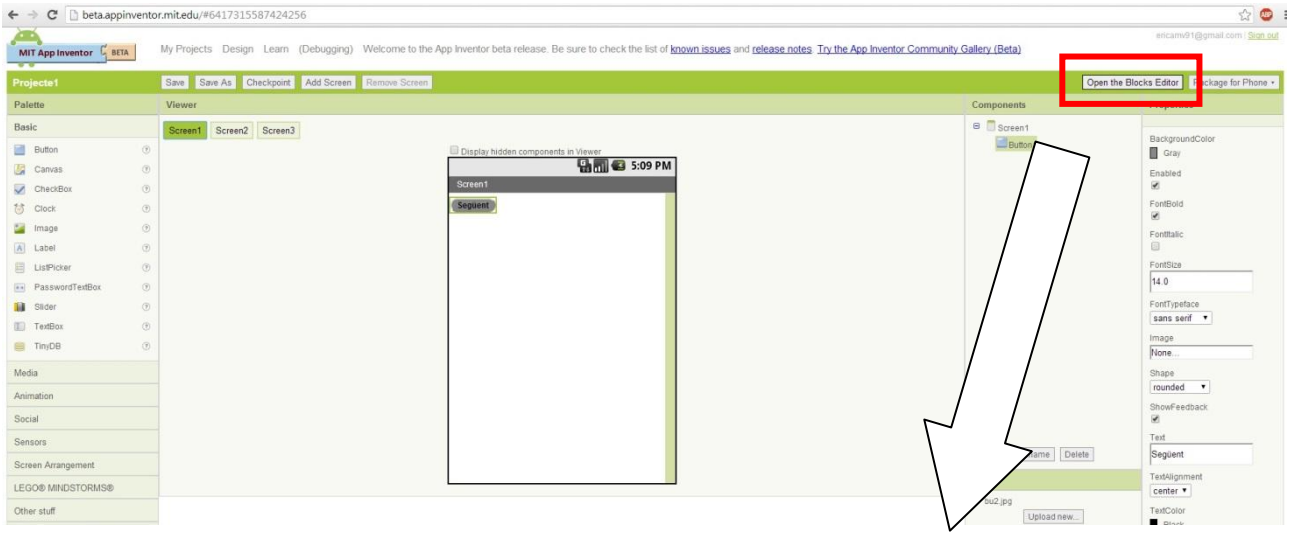


Imatge28

Un cop tot seleccionat i tot modificat tal i com volem, ja podem fer el programa específic per a cadascun dels elements escollits i col·locats.

3.2.2. Interfície part programació

Per accedir a la part de programació, primerament s'ha de tenir instal·lat el Java. Un cop ja ho tenim preparat, anem a la interfície gràfica i ens fixem a la part on ens indica la *Imatge29.*

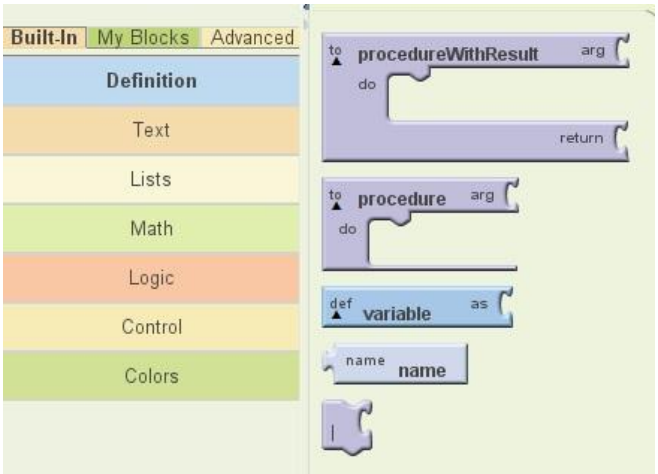


Imatge29

Clickem a la part "Open the Blocks Editor" per accedir a la interfície de la programació de blocs.

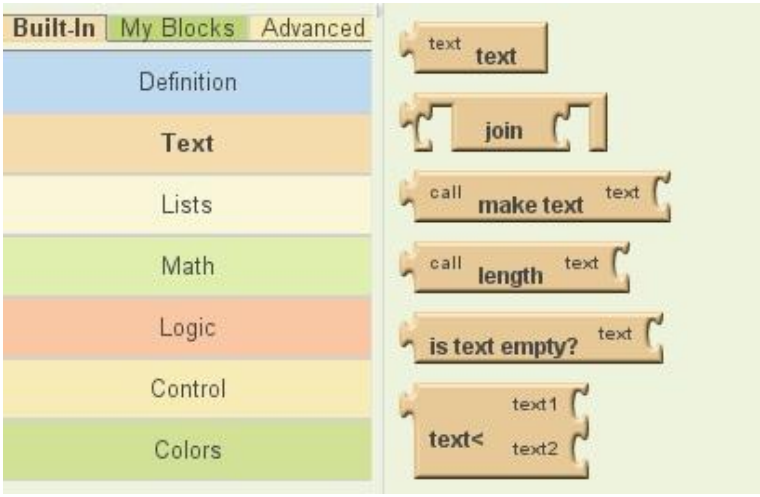
La part de programació té diversos blocs dividits en diversos sectors. Principalment estan dividits en 3 grans:

-Built-In: En aquest mai canvia, sempre hi ha els mateixos blocs per a totes les Screen.



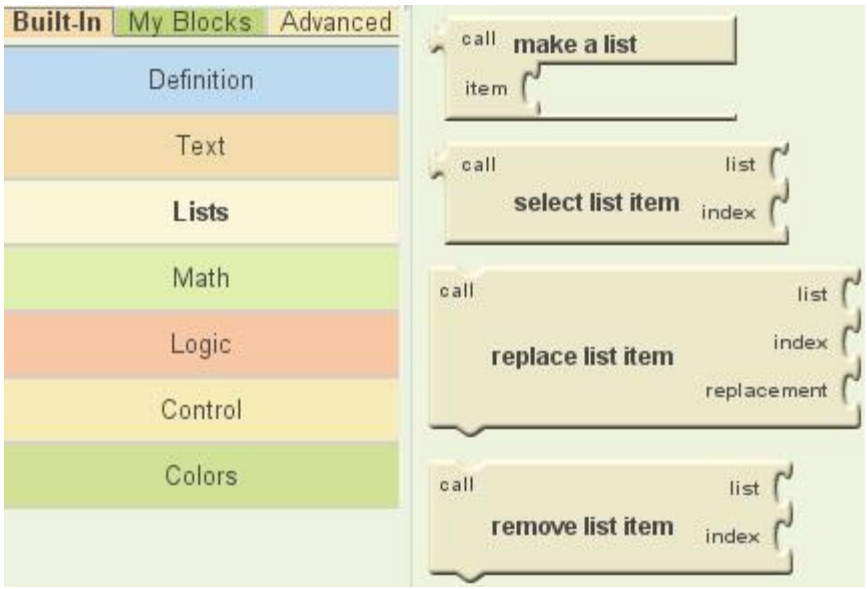
Definition

En aquest apartat de blocs podem definir variables i constants.



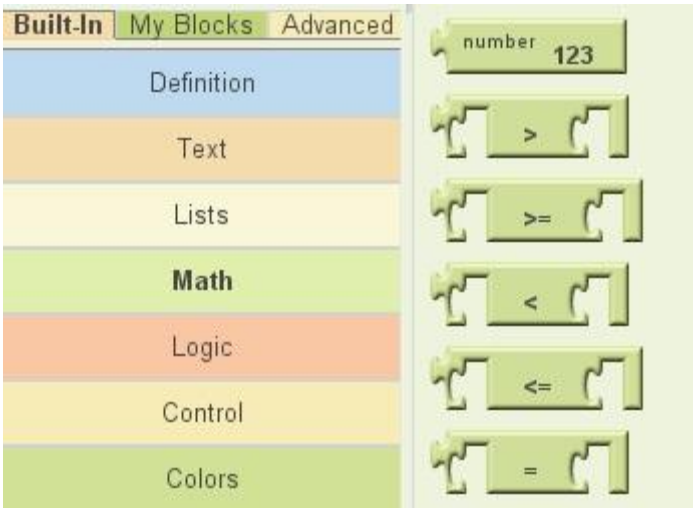
Text

En aquesta part podem fer diverses accions, com per exemple, posar diferents tipus de textos, com poden ser el nom d'alguna imatge exacte (.jpg) o alguna pàgina web i que aquesta s'executi per quan un element concret faci alguna acció, depenent de que faci el programa.



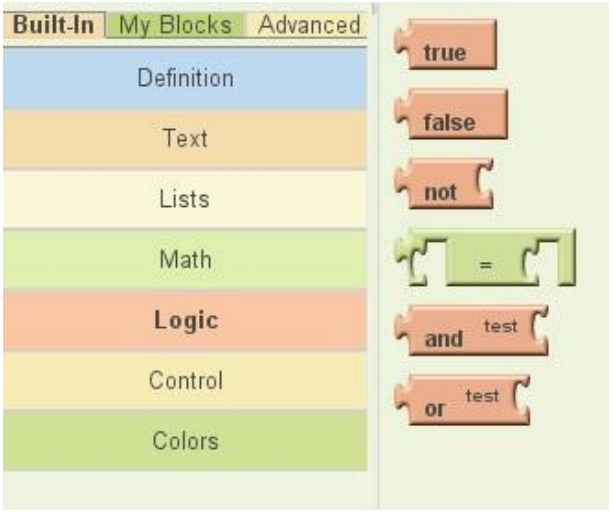
Lists

Blocs que s'utilitzen a l'hora de crear llistes o seleccionar-les.



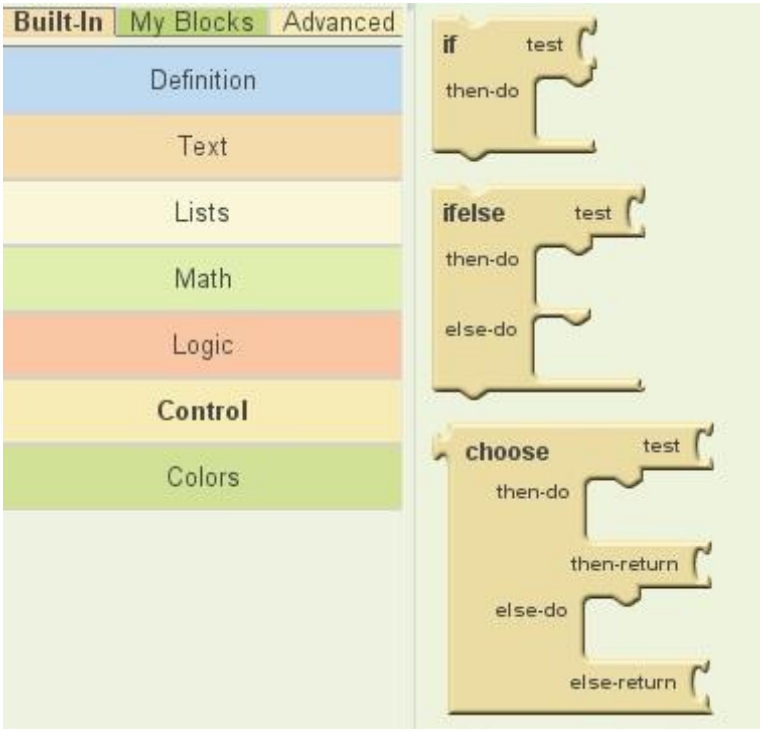
Math

En aquest apartata, inclouen tots els blocs relacionats amb les matemàtiques: posar un número en concret, major o igual que, menor o igual que, igual, etc.



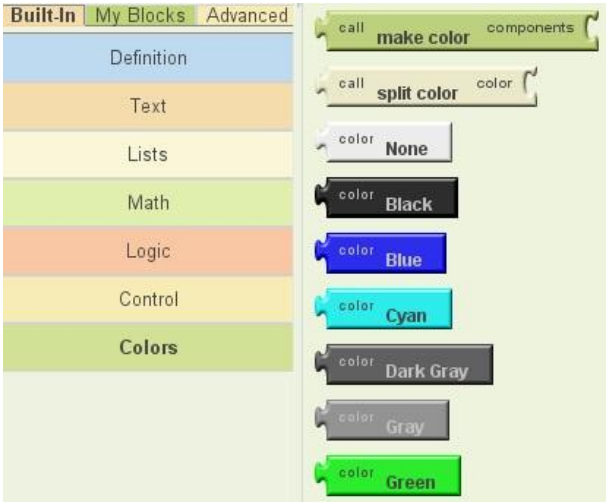
Logic

Blocs de lògica, per a booleans: false, true, not, and i or.



Control

La part on estan els blocs per posar condicions, fer bucles, entre altres també molt diferents com pot ser per canviar d'una Screen a una altre.



Colors

Per poder canviar el color a un element exacte a través d'una funció, booleà, condició, etc.

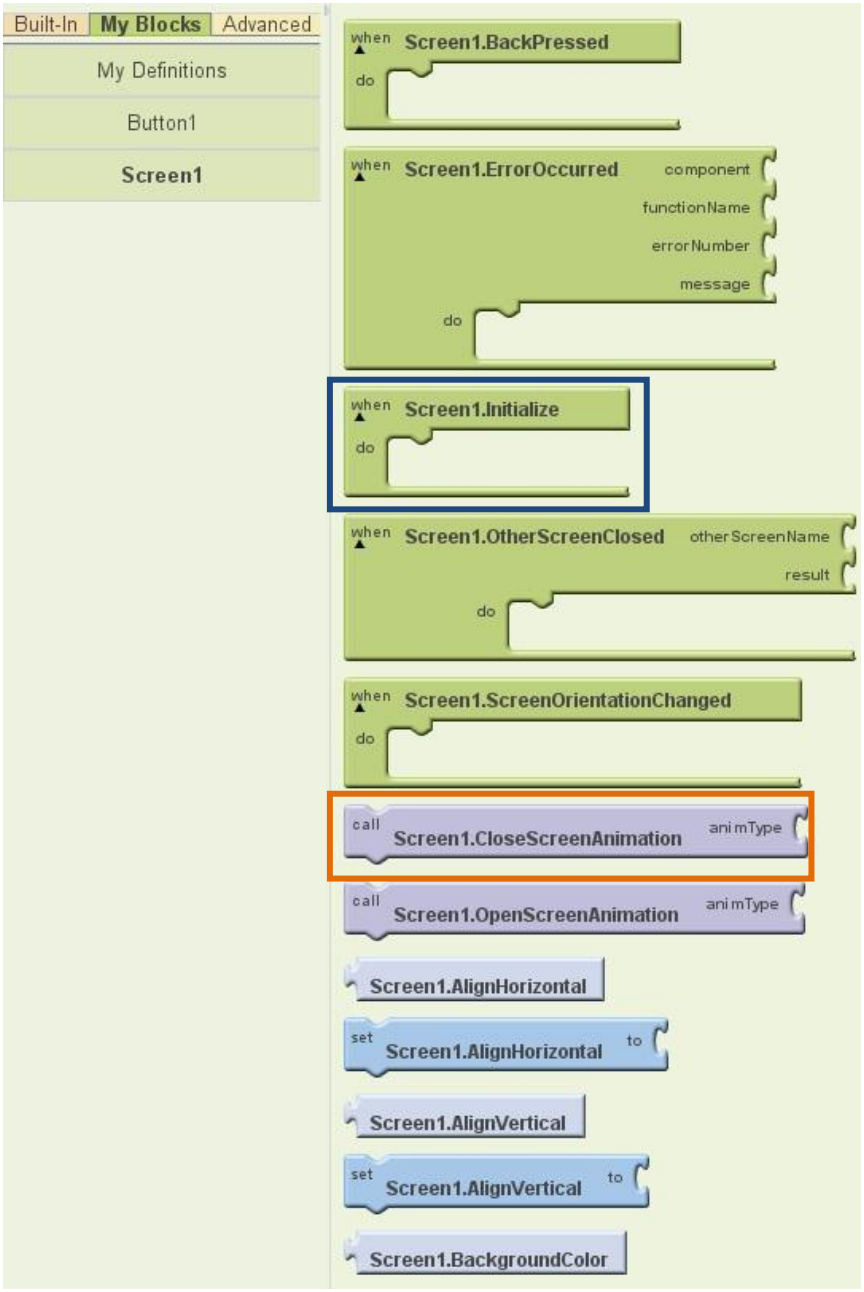
-My Blocks: Aquesta part ja canvia depenent dels elements que hem ficat anteriorment a la part gràfica a través de la Palette.

Per exemple, en aquest cas, hem afegit com a elements Button. Per tant, sortiran, a més de My Definitions (aquest ja ve predefinit) i Screen (vindrà amb el nom corresponent que s'hagi posat. En aquest cas, es va quedar com a Screen1), la part de Button1 (nom del Button que haurem posat). I quedaria:

Button1 és l'element que hem col·locat a la pantalla View.

Screen1 i My Definitions ja surten com a prefefinits.

My definitions



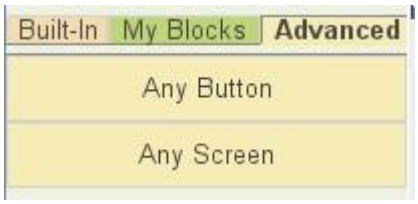
Sobre els blocs, destacar que, com es pot veure, hi ha blocs d'un color o d'un altre. Per aquesta part, sempre sortiran 4 tipus de colors: verd, lila, blau cel, blau més fosc. Cada color té un significat. En el cas del verd, són aquells blocs que fan una acció a partir d'una altre acció feta per part de l'usuari. En el cas del lila, a partir d'una acció,

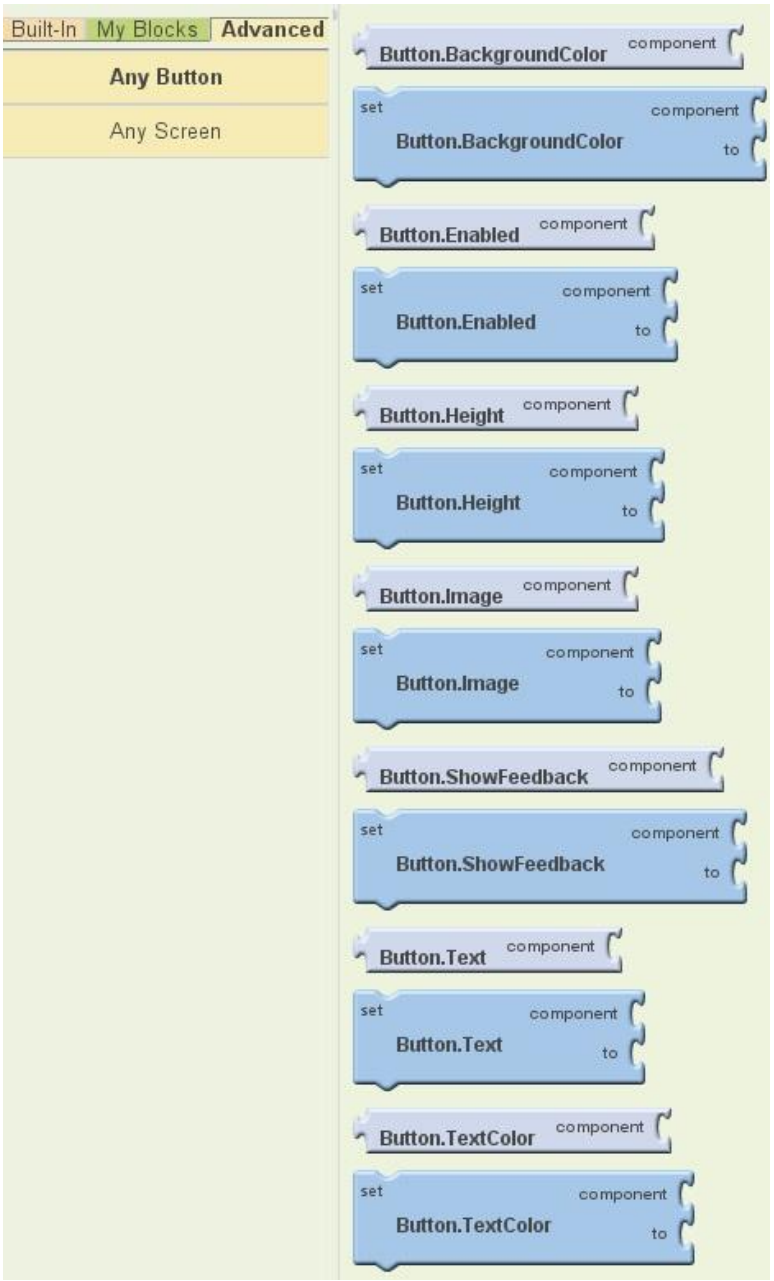
es crida a una altre acció. En el cas del blau cel, són constants mentre que el blau més fosc són varibles.

En aquesta part de blocs, si clickem a Screen1, ens sortirà tots els blocs disponibles que tindran relació amb la mateixa Screen1. Per exemple, tenim el bloc seleccionat en blau que significa que quan s'inicialitza, farà l'acció que posem entre l'espai que ens donen.

Per la resta de blocs, ja son accions que es farien a partir d'una funció, condició, etc. Per exemple, tenim el bloc seleccionat en taronja que indica que si hi ha condició o funció o qualsevol acció, s'obrirà una animació. A més, pots afegir després de l'animació quina seria a través d'un bloc text amb el seu nom corresponent (nom.gif).

-Advanced: Aquí és igual que My Blocks, també canvia depenent dels elements que hi hem ficat a través de la Palette.

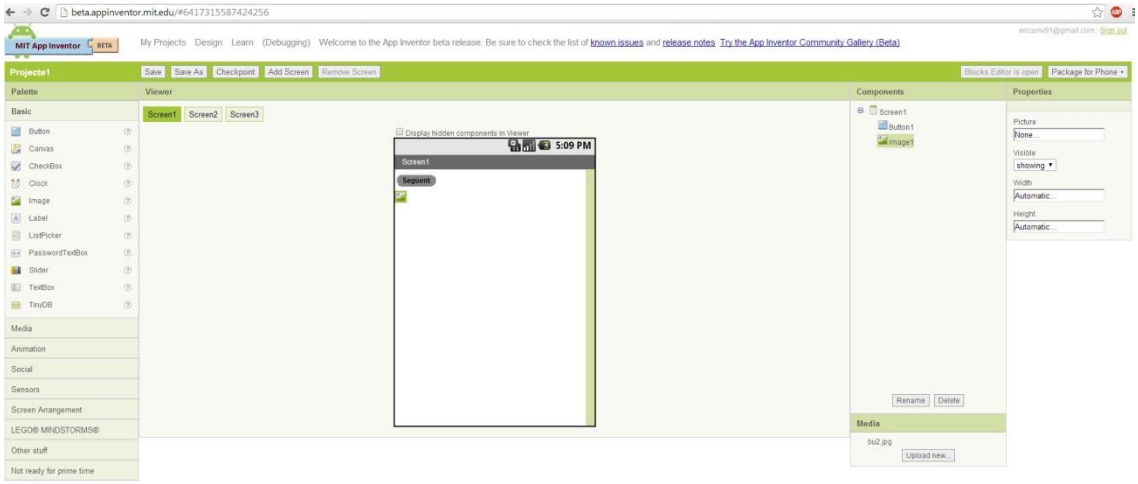




Aquí podem trobar blocs també relacionats amb els elements que hem ficat (en aquest cas, Screen i Button, com hem indicat anteriorment). Per exemple, hi ha blocs que dependran de l'acció que es vulgui fer, el color de l'element canviaria o podria canviar l'alçada o amplada de l'element corresponent.

Fins aquí, una mica la funcionalitat d'una part bàsica dels blocs. Ja tenim tots els coneixements bàsics per crear una aplicació a través del blocs mostrats. Ara hem de veure la seva funcionalitat i com ho podem fer servir.

Comencem amb un exemple sobre el mateix element, el Button i la Screen1 ja definida desde el començament i a més afegim una Image (*Imatge30*).



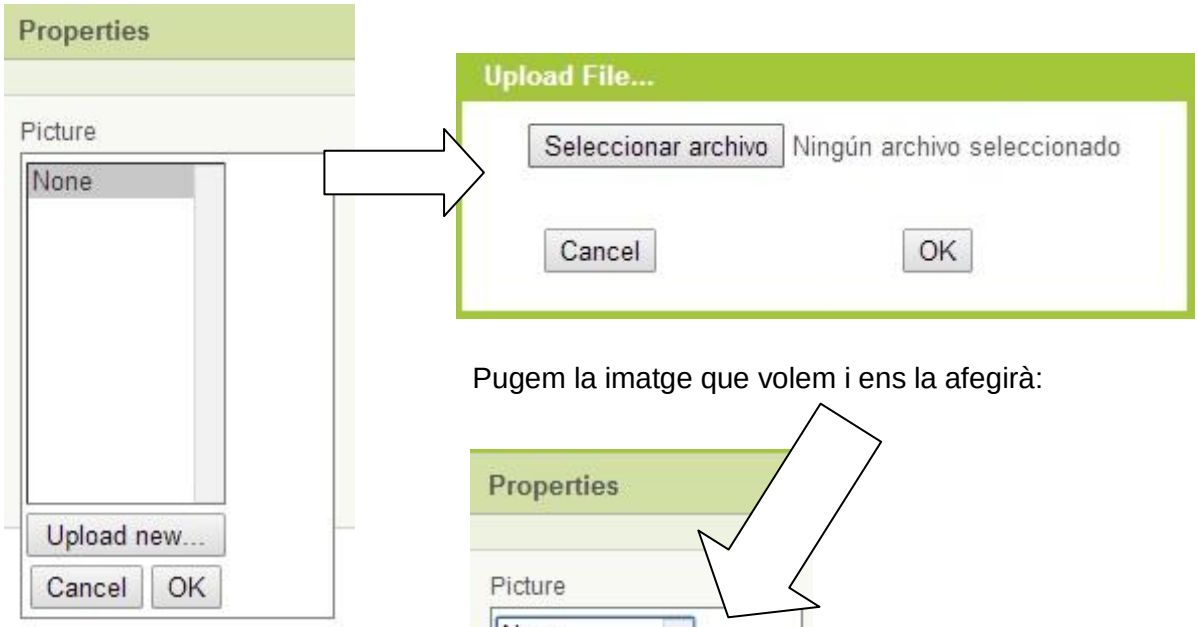
Imatge30

En aquest exemple volem que quan s'inicialitzi l'apliació al dispositiu només es vegi el Button. Si clickem el Button, haurà de sortir una imatge.

Per començár a fer l'aplicació, hem de ficar primer els elements necessaris (Button i Image) i pujar a través de les Properties de Image, la imatge que volem que ens mostri quan clickem al Button. Per tant, primer clickem a Image i ens sortirà les seves properties (*Imatge31*):

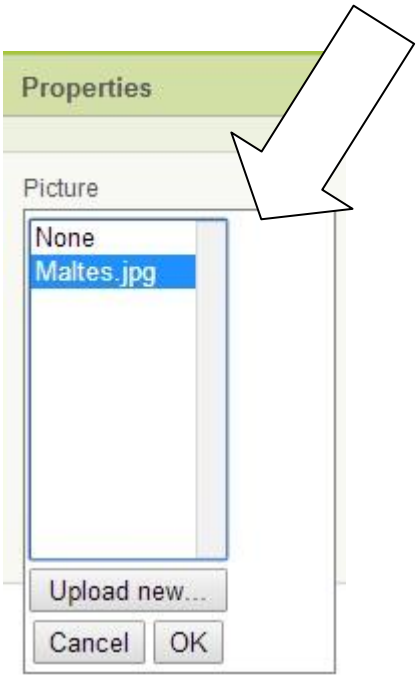
Imatge31

Hem de pujar la imatge que volem a través de l'opció Picture de Properties. Si clickem, ens donarà l'opció *(Imatge32*):

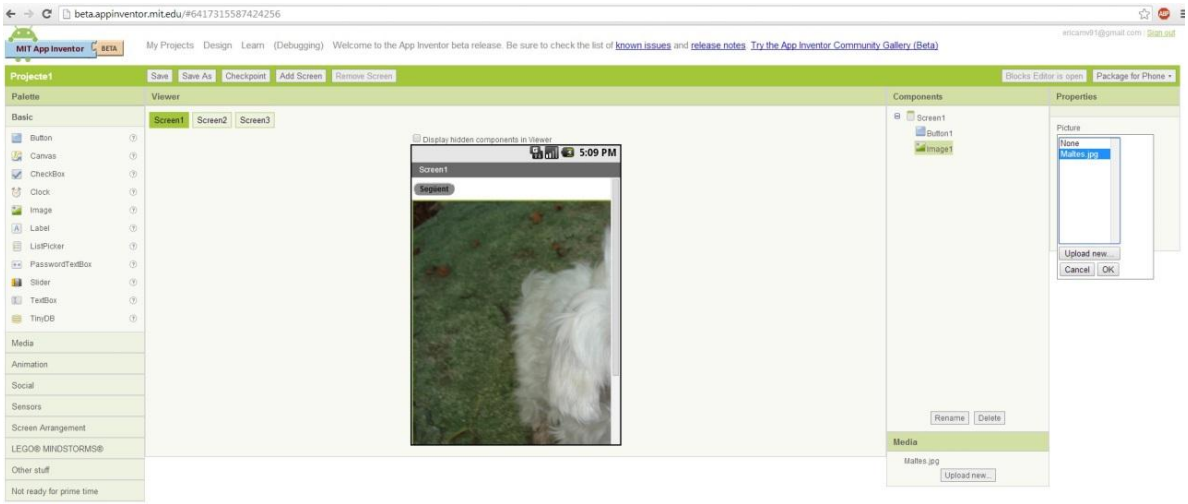


Imatge32

Pugem la imatge que volem i ens la afegirà:



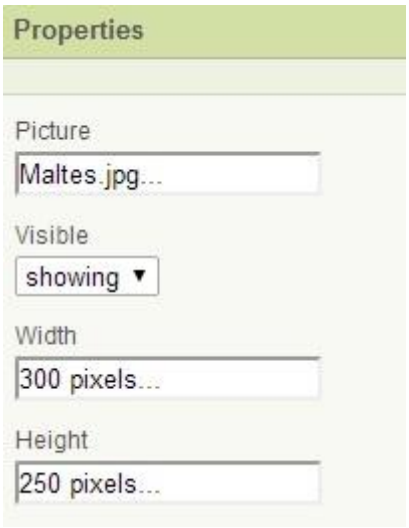
I en sortirà *Imatge33*:



Imatge33

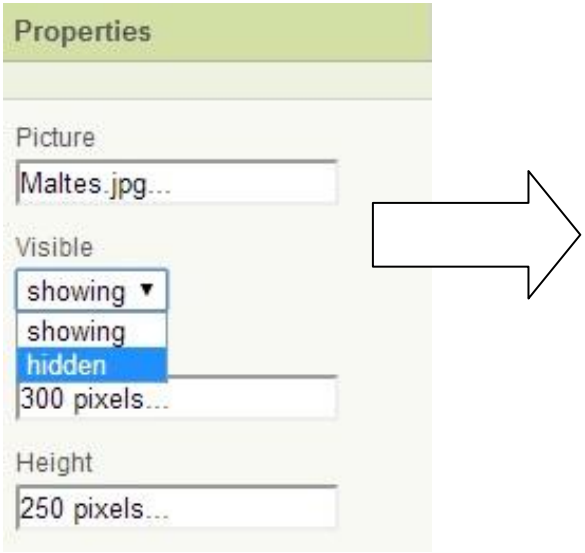
Però hem de seleccionar que la foto no es vegi a la pantalla (recordem que tot el que surti a la pantalla Viewer, ho visualitzarà un cop s'inicialitzi l'aplicació).

Abans de deixar amagada la imatge, a més, hem de definir la mida exacte que volem que tingui la imatge quan surti per pantalla. Si tornem a mirar les Properties, tenim les opcions de Width i Height (Amplada i alçada) per seleccionar la mida en píxels. En aquest cas, ho faré de 300x250 px *(Imatge34*).

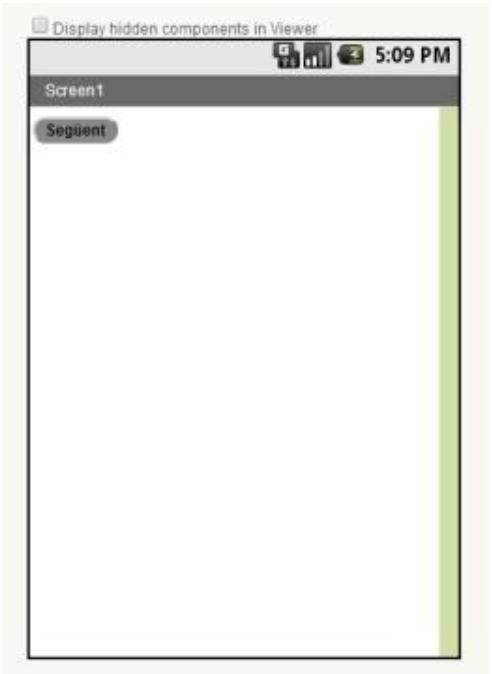


Imatge34

Per seleccionar l'opció, també hem de mirar a Properties i seleccionar l'opció Visible i marcar que s'amagui (*Imatge35*) i ens quedaria com la *Imatge36*.



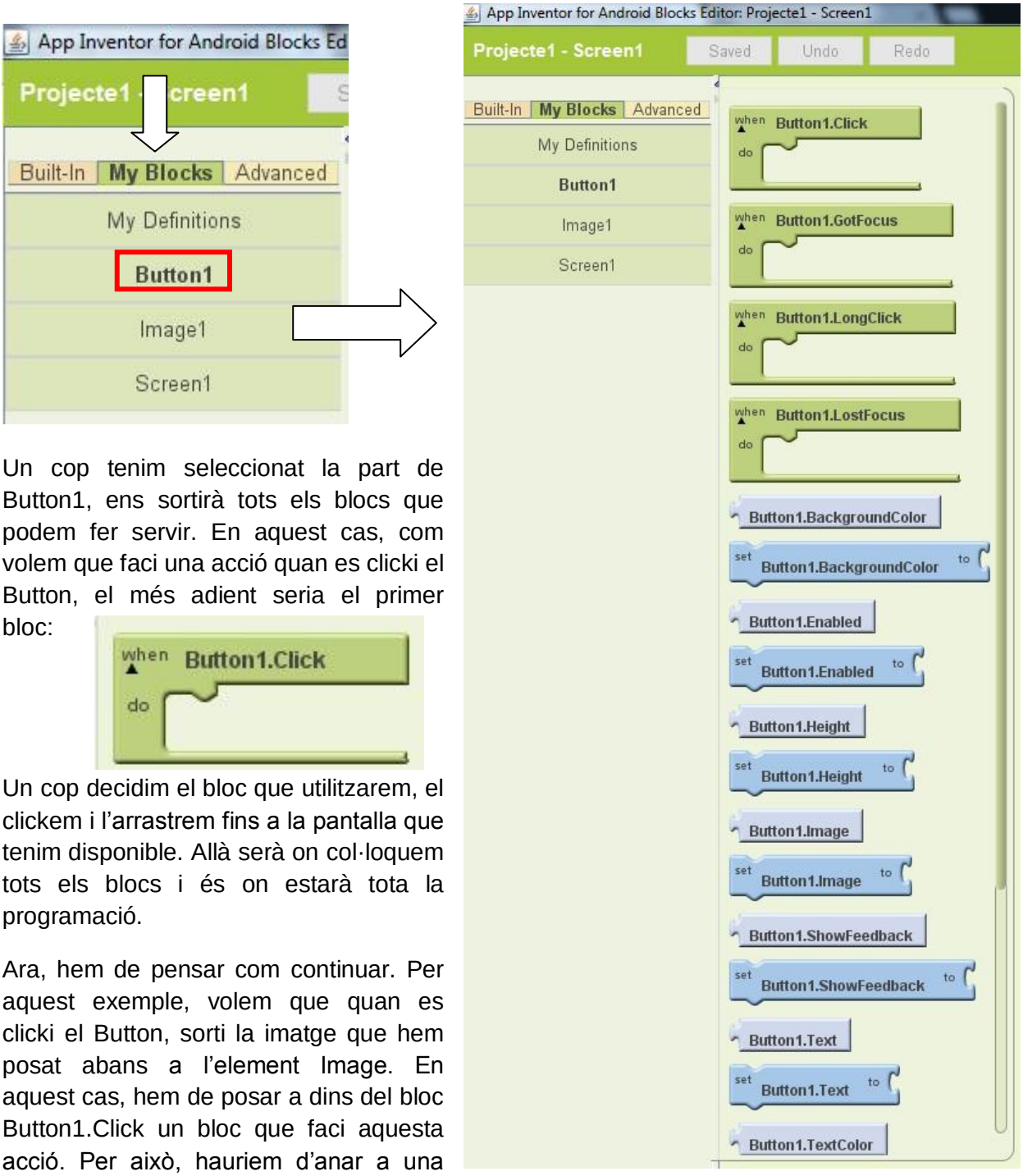
Imatge35



Imatge36

Ara ja hem deixat els elements tal i com voliem. Ara passem a la part de programació. Per això, hem d'obrir la interfície de programació de blocs.

Comencen a programar per blocs. Primer, hem de tenir clar el que hem de fer. En aquest cas, fins que no es clicka el botó, no hauria de pasar res a la pantalla. Per tant, hem d'anar primer als blocs relacionats amb el Button, a través de la pestanya My Blocks i després a Button1 (*Imatge37*).



Un cop tenim seleccionat la part de Button1, ens sortirà tots els blocs que podem fer servir. En aquest cas, com volem que faci una acció quan es clicki el Button, el més adient seria el primer bloc:

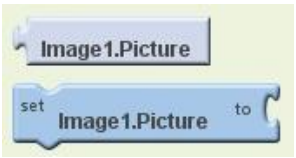
Un cop decidim el bloc que utilitzarem, el clickem i l'arrastrem fins a la pantalla que tenim disponible. Allà serà on col·loquem tots els blocs i és on estarà tota la programació.

Ara, hem de pensar com continuar. Per aquest exemple, volem que quan es clicki el Button, sorti la imatge que hem posat abans a l'element Image. En aquest cas, hem de posar a dins del bloc Button1.Click un bloc que faci aquesta acció. Per això, hauriem d'anar a una altre banda on hi hagi blocs relacionats amb l'element Image.

Imatge37

Anem a la part de l'element Image, Image1 concretament (*Imatge38*).

Aquí ens surt tots els blocs que vagi relacionat amb l'element Image. En aquest exemple, hem de veure quin podria ser el bloc més adient per l'acció que volem fer, que un cop es clicki el Button, la imatge de Image es mostri per pantalla. En aquest cas, podem veure que hi ha dos blocs diferentes relacionats amb imatges:



Imatge38

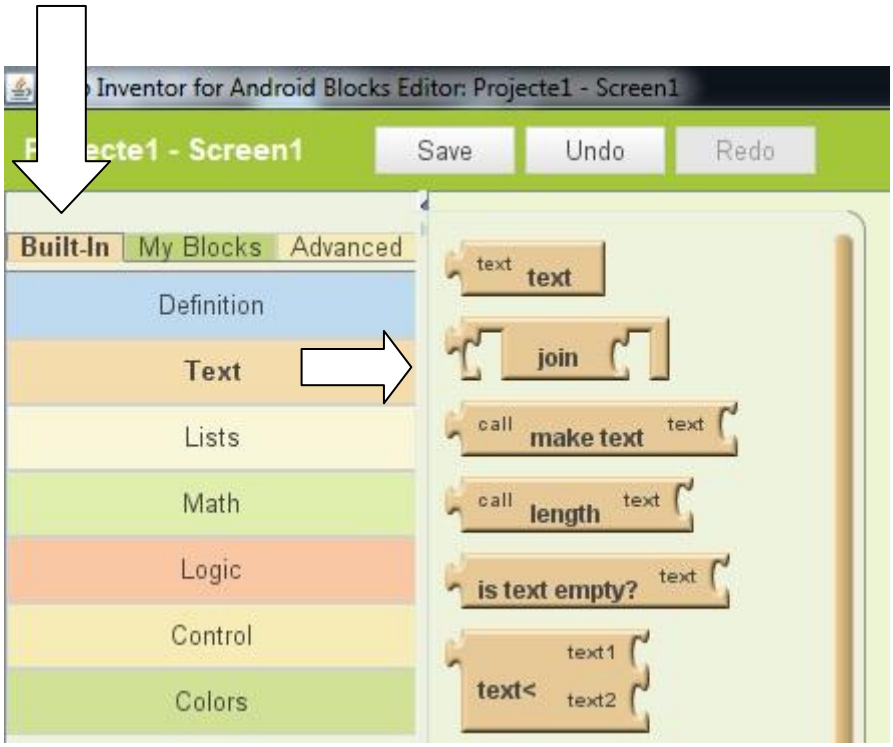
En aquest cas, quin hauriem d'escollir? Per aquest cas, seria la segona opció perquè la diferència que hi ha entre aquests dos blocs és una ben clara: la primera és una constant i la segona és una varible. Com volem que surti una imatge especifica, hem de ficar la variable per així donar-li el valor necessari (en aquest cas seria la imatge que hem pujat abans al mateix programa).

Un cop sabem el bloc, ja els podem unir i quedaria com a la *Imatge39.*

Imatge39

Però ara ens queda una última cosa, ja que el mateix bloc blau Image1.Picture ens "exigeix" un bloc més per unir. En aquest cas, ens queda una última acció per posar: quina imatge ha de sortir per pantalla.

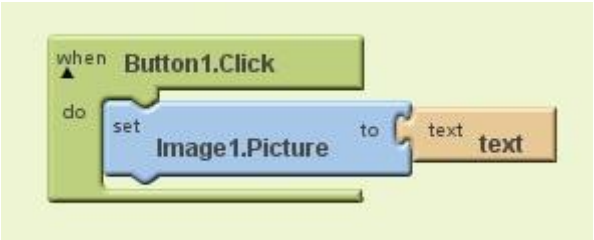
Per solucionar aquesta última part, hem d'anar a un altre apartat de blocs que es relacioni amb posar imatges. A AppInventor hi ha diverses formes de col·locar imatges però la més sencilla és indicar el nom exacte de la imatge que hem pujat al programa. Per col·locar textos s'ha de ficar en un bloc de Text. Per tant, anem a la part de blocs de Text (*Imatge40*).



Imatge40

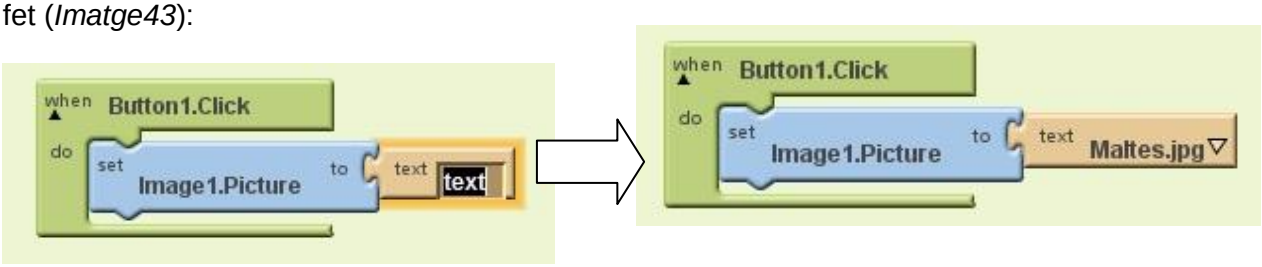
Hi ha molts tipus de blocs a l'apartat de Text però només ens interesa posar el nom de la imatge corresponent. Per tant, amb el primer bloc de tots ja en tindriem suficient.

Traiem el bloc de text a la part on es col·loquem els blocs i l'unim amb els blocs que ja teniem, l'unim amb el bloc Image1.Picture (*Imatge41*).



Imatge41

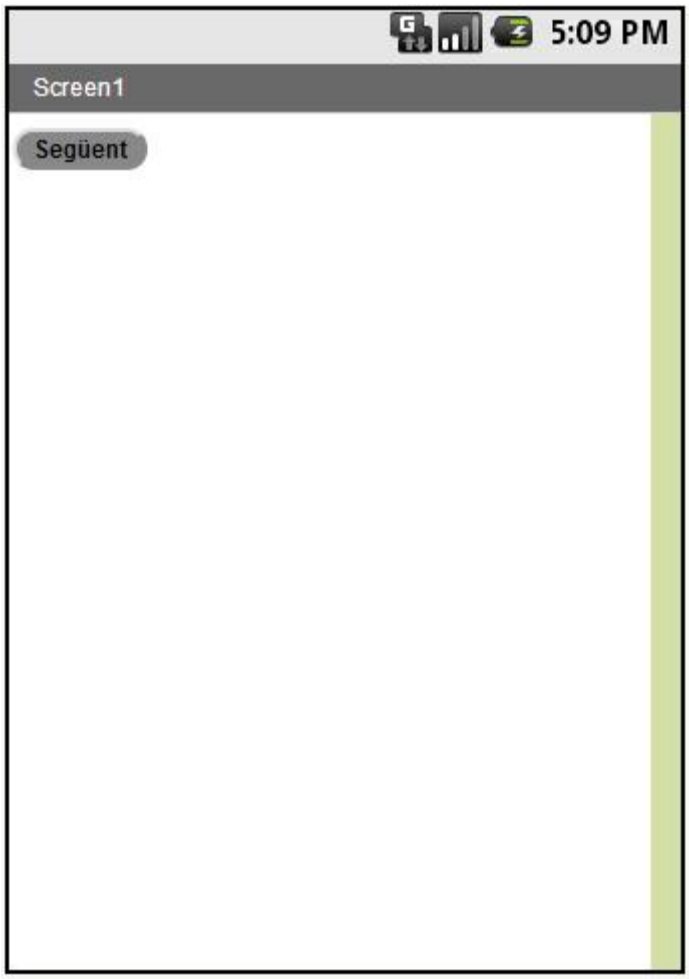
Per últim, hem de modificar el text del mateix bloc text i, per això, hem d'escriure el nom de la imatge que volem que ens mostri per pantalla (en aquest cas: Maltes.jpg). Per canviar el text, haurem de clickar al mateix bloc text i ens sortirà per escriure (*Imatge42*). I per fi, un cop escrit el nom complet de la imatge, ja tindriem el programa fet (*Imatge43*):



Imatge42 *Imatge43*

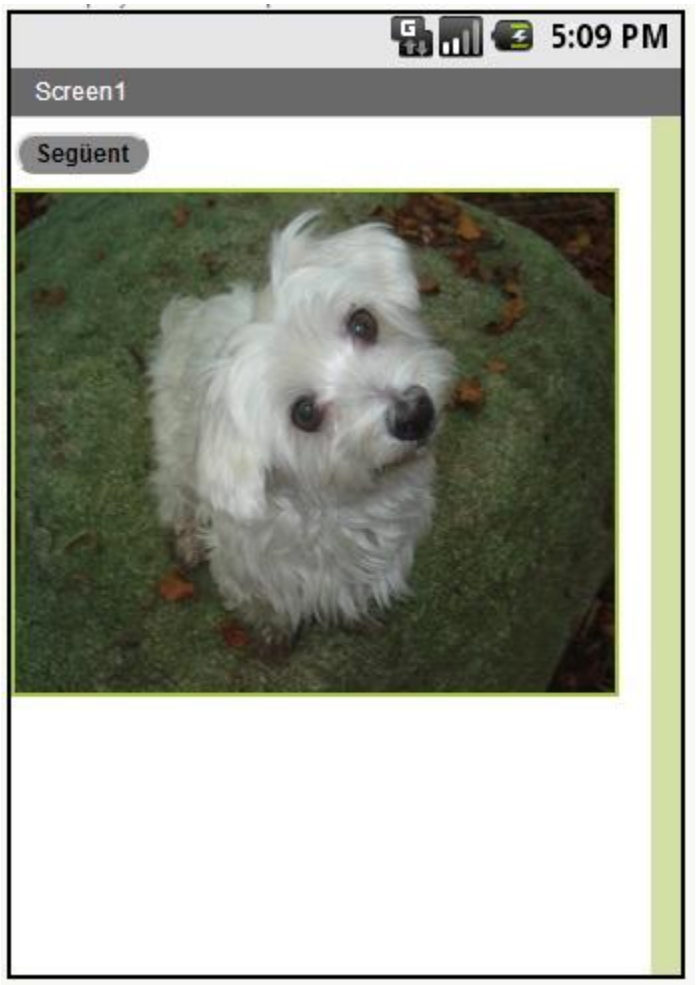
L'aplicació amb aquest programa farà el següent:

Res més inicialitzar-se l'aplicació, sortirà la pantalla de la *Imatge44.*



Imatge44

Un cop clickem al Button "Següent", haurà de sortir la imatge seleccionada, que seria el resultat final (*Imatge45*).



Imatge45

3.2.3. Estructura Android

A Android, en general, l'aplicació funciona d'una forma. Primer es comença amb una pantalla (Screen) però després es poden anar afegint més per a que vagi d'una a una altre depenent de les funcions que es vulguin.

3.2.4. Estructura de pantalles de les aplicacions

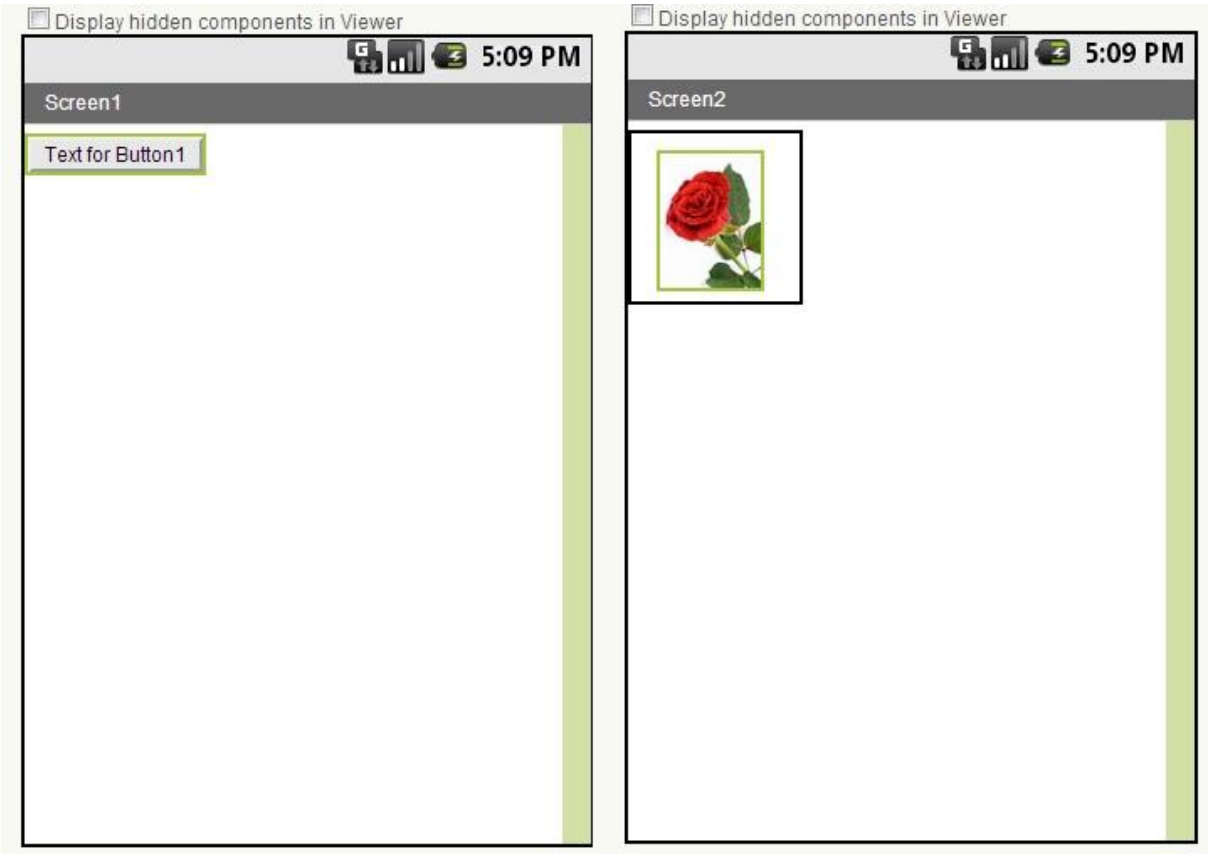
Un dels primers exemples explicatius i demostratius de veure com funcionen els blocs i la programació en si de AppInventor, és passar d'una Screen a una altre. És un tema important ja que si volem una aplicació amb un cert pes, hem de saber com funciona perquè necessitaríem utilitzar més d'una Screen.

Per començar, amb AppInventor, quan obres un projecte nou, ja et dona per defecte la "Screen1". Aquesta serà sempre la primera pantalla que sortirà quan executis l'aplicació.

Llavors, per poder passar d'una pantalla a una altre, s'ha d'afegir un element de la part esquerra Palette – Basic, com per exemple: Button, Canvas, Label, ... Hi ha moltes possibilitats i depenen com vulguis fer-ho, et vindrà millor un que altre. A més d'això, s'ha de crear una nova screen clickant la part de dalt a "Add Screen". Et dóna l'opció de canviar el nom (surt Screen2 per defecte).

En aquest cas, volem pasar de la Screen1 a la Screen2 i viceversa també. Per fer-ho, a la Screen1 ficarem un "Button" i quan clickem el Button, volem que ens faci passar a Screen2, on hi haurà una imatge qualsevol. Si després volem passar a la Screen1, s'haurà de clickar a la imatge que hi haurà.

En resum, tindrem (*Imatge44*):



Imatge44

Com hem explicat anteriorment, podem "costumitzar" el botó posant un altre text que no sigui el "Text for Button1" per defecte, un altre color, etc. Tot es modifica a la part dreta del programa, a Properties. Per exemple, a "BlackgroundColor" pots canviar el color de fons del botó, a "Text" pots canviar el text que vols que hi hagi a l'interior del

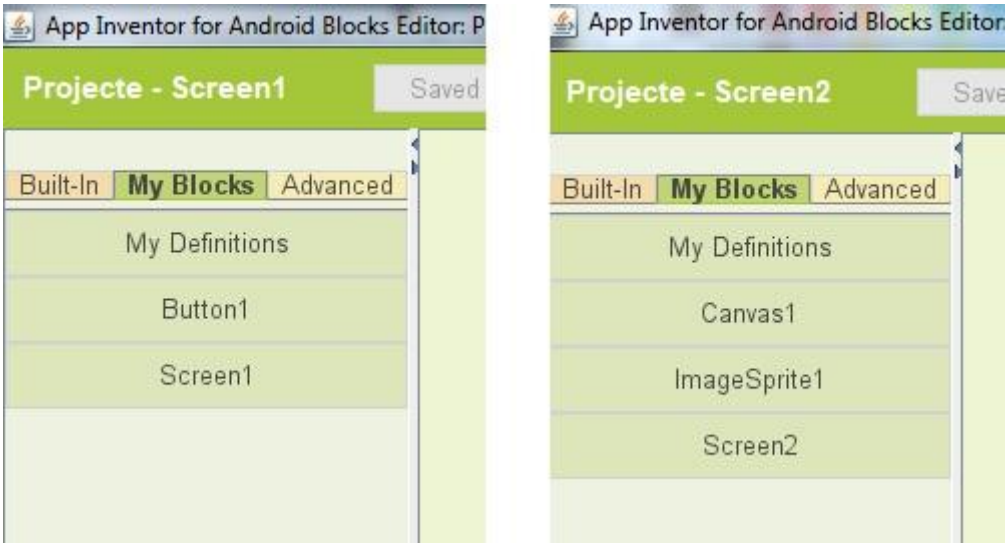
botó, canviar la mida amb Width/Height (ample/altura), etc. En el cas de la imatge, és el mateix.

Un cop tenim escollit i ben col·locat el que volem que tingui a cada pantalla per passar d'una a una ltre, ja hem de passar per la part de programació.

Obrim l'editor de blocs clickant "Open the Blocks Editor".

NOTA: Depenent des de quina Screen s'obre l'editor de blocks, sortiran les variables de la Screen1 o de la Screen2.

Posem la comparació de les dues Screen (I*matge45*).



Imatge45

Com es pot veure, veiem els diferents elements col·locats que hi ha a cada Screen. En aquest cas, a la Screen1 només comptem amb la mateixa Screen1, My Definitions i l'element col·locat expressament a la Screen1 que és el Button1.

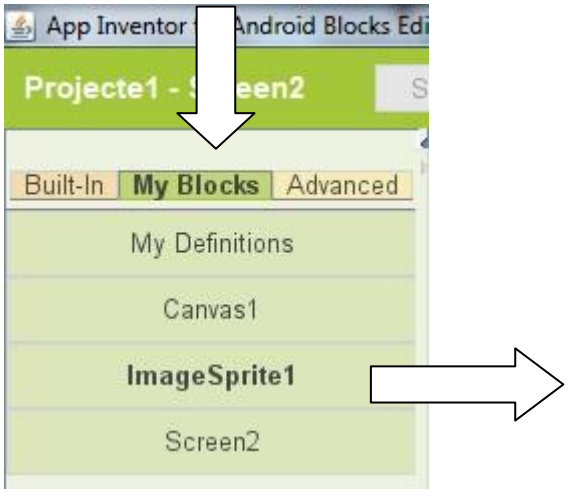
Per la Screen2, també hi ha la mateixa Screen2, My Definitions i les que hem afegit com a elements a la interfície gràfica: Canvas1 i ImageSprite1.

Per poder passar d'una pantalla a una altre tocant un o l'altre, s'ha d'agafar els blocs de la corresponent variable "tocada", en aquest cas, per Screen1 seria Button1 i per Screen2 seria ImageSprite1.

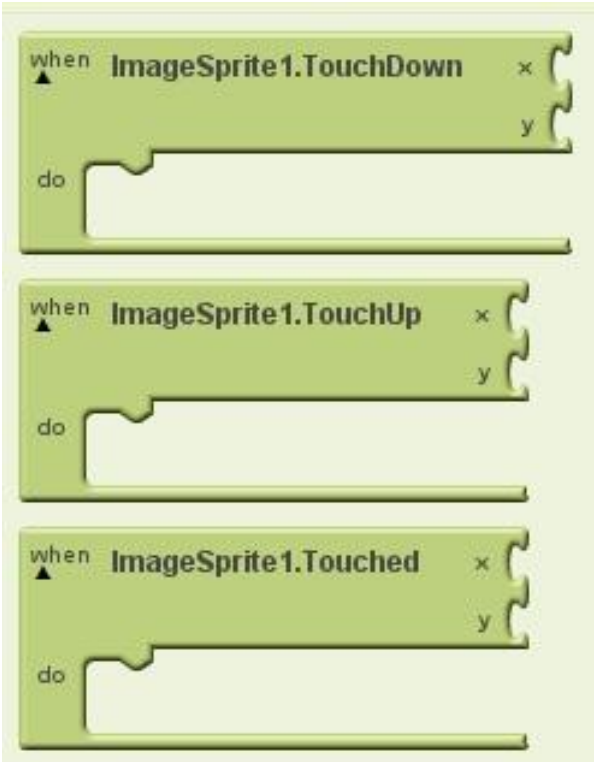
Per tant, hem d'anar als blocs corresponents de Button1 (per la Screen1) (*Imatge46*) i ImageSprite1 (per la Screen2) (I*matge47*).

El bloc corresponent per a Button1, seria el primer ja que quan el botó es clicki, es farà l'acció.

Imatge46



En el cas de ImageSprite1, hi ha molts tipus de blocs i hauriem d'escollir el més adient per a que quan es toqui, faci l'acció corresponent. En aquest cas, el bloc que es necessita seria el següent:



Imatge47

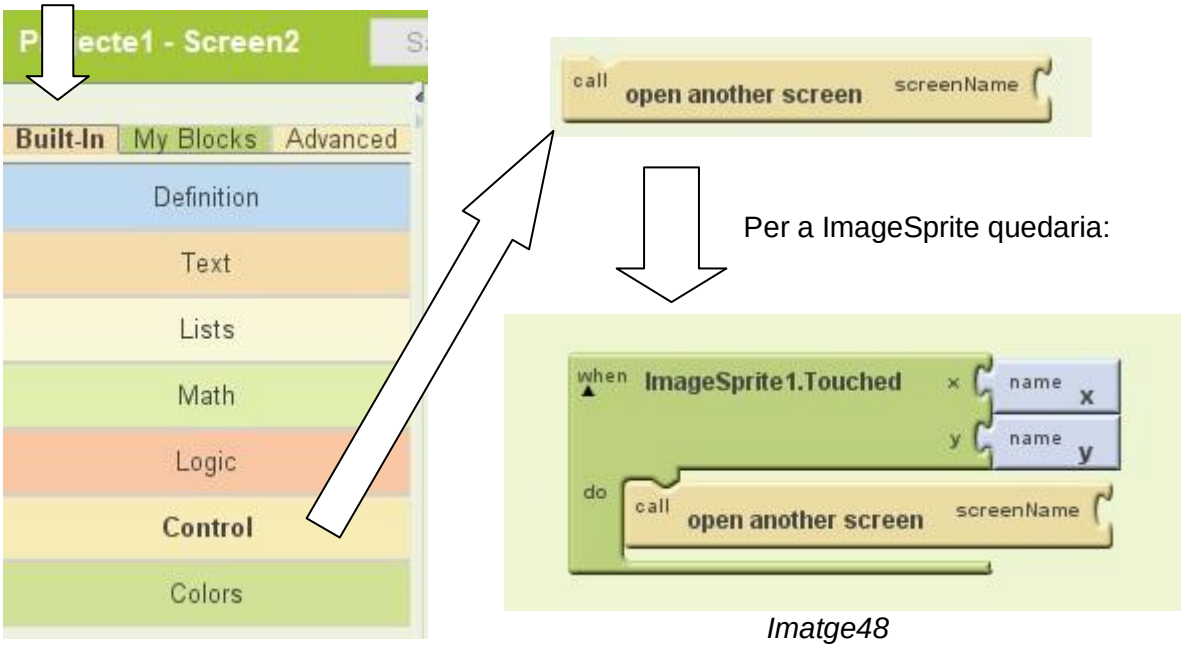
NOTA: Un cop que col·loquem el bloc de ImageSprite1.Touched a la pantalla, veurem que s'han afegit 2 blocs més a la part d'amunt (*Imatge48*).

Imatge48

Aquestes variables es col·loquen soles i són necessaries pel bon funciomanet del programa. No cal fer res ja que AppInventor les col·loca soles.

Un cop col·loquem aquests blocs a la respectiva pantalla on estarà tots els blocs i programació, ara hem de pensar quin seria el bloc adient per a que faci l'acció de passar d'una Screen a una altre.

En aquest cas, només hi ha un bloc que indica que farà una acció si aquesta variable o element es toca a la pantalla. Aquest bloc, es troba a la pestanya Built-In, a Control concretament. Aquesta seria la que posaríem (*Imatge48*).



Imatge48

Pel cas de Button quedaria:

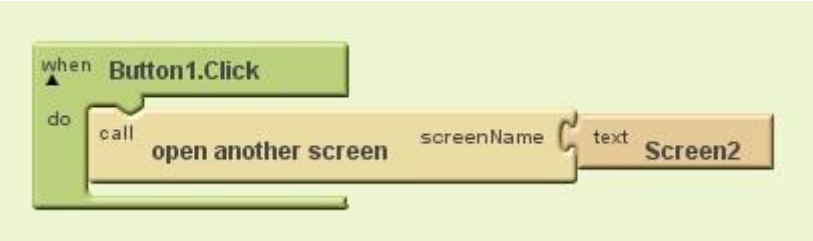
Imatge49

Ara només cal col·locarles entre elles i ja tindriem l'acció posada per quan es pulsi el Button o es toqui la ImageSprite però encara queda fer una última cosa. L'acció està posada però, encara no s'ha especificat cap a on han d'anar. L'acció està incompleta: Sap que han de moure's a una altre Screen però encara no s'ha especificat quina.

Quedaria col·locar un últim bloc, s'ha d'afegir un Text i haurem d'escriure el nom de la Screen corresponent. En aquest cas, si estem a la Screen1, hem de posar el nom de Screen2.

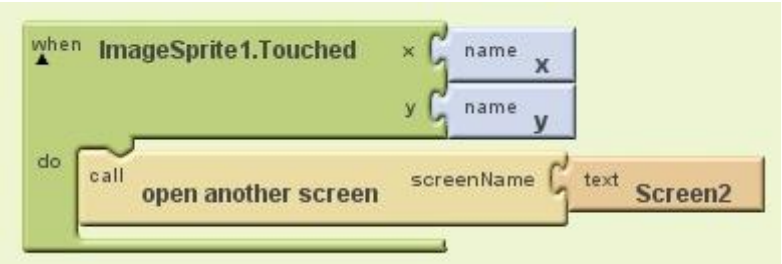
Al final, el resultat quedaria:

Per a la Screen1 (*Imatge50*):



Imatge50

Per a la Screen2 (*Imatge51*):



Imatge51

O també hi ha una altre opció pel cas de la Screen2 (*Imatge52*):

Imatge52

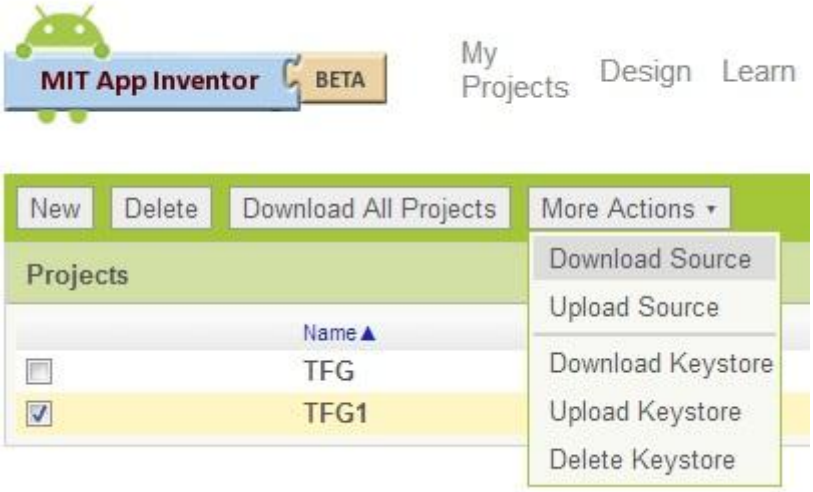
L'única diferencia entre "Open another screen" amb "close screen" és que si col·loques "Open another screen", tornaràs a obrir la Screen i es queda "allà". Això implica que si clickes amb el mateix botó del dispositiu per tornar darrere, tornarà de nou a la pantalla i no podràs sortir de l'aplicació. En canvi, amb el "close screen" ja

t'encarregues que no estigui més i si clickes al botó de darrere que té el mateix dispositiu, ja et fa fora de l'aplicació com hauria de ser.

3.3. Finalització

3.3.1. Descarregar còpia de seguretat del programa

En el cas que vulguem fer una còpia de seguretat per treballar a un altre ordinador o qualsevol altre motiu, hem de seleccionar al quadrat de l'esquerra al projecte que volem descarregar i després anar a "More Actions" → "Download Source" (*Imatge53*).



Imatge53

Una vegada fet, es descarregarà un fitxer amb l'aplicació comprimida en format rar.

Pel cas contrari, si voleu adjuntar un projecte en la mateixa pàgina, heu de seleccionar també a "More Actions" → "Upload Source" i heu de seleccionar el .rar corresponent.

3.3.2. Testejar l'aplicació al mòbil

Per testejar l'app hi ha 3 maneres de fer-ho:

- *Mitjançant Wifi:* Abans de tot, per testejar l'app amb aquest mètode, ens hem de baixar una aplicació gratuïta a partir de la Play Store anomenat "MIT AICompanion". Amb aquets aplicació baixada, ja podem testejar-ho.

  Per testejar l'app, com diu el nom principal, el dispositiu ha d'estar connectat a internet. Una vegada repassat tot, s'ha d'estar a la part de programació de blocs. A la part de dalt dreta, tenim 2 opcions: "New Emulator" i "Connect to

Device". Per poder veure com funciona la nostra app al dispositiu, hem de donar-li a "Connect to Device" → "WiFi" (*Imatge54*).

Imatge54

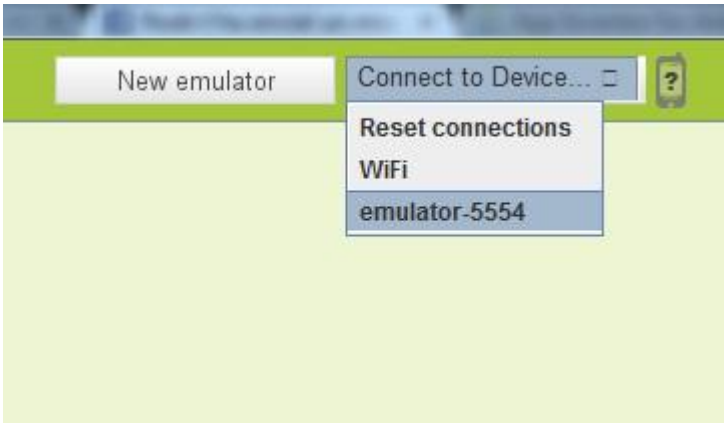
Un cop que hem donat allà, ens sortirà una finestra amb un codi de 6 lletres a més d'un codi QR. Ara amb el dispositiu, obrim l'aplicació MIT AICompanion i podem veure que tenim dues opcions: posar el codi de 6 lletres i clikar directament el "Connect to AppInventor" o escanejar el codi QR. Qualsevol de les dues coses seria correcte i ens passarà a l'aplicació feta. Si per qualsevol cosa té errors, ja ens ho indicaria.

NOTA: Un cop ja hem mirat/observat el que volíem, abans de retocar res de la part de programació, hem de clickar a "Connect to Device" → "Reset connections". Es fa per desconnectar la connexió que hi ha entre l'app feta i el dispositiu, ja que si modifiquem el programa mentre hi ha una connexió, pot donar errors i s'hauria de reiniciar la pantalla de programació. Per evitar problemes, millor desconnectar-ho fent-lo d'aquesta forma.

- *Mitjançant New Emulator:* En aquest cas, no cal que es connecti a través de nostre dispositiu. És un emulador que incorpora el mateix programa. Per activar-ho, s'ha de clickar a "New emulator". Un cop creat, apareixerà un emulador d'un mòvil com la *Imatge55.*

I un cop ha aparegut, hem de clickar a "Connect to Device" → "emulator-5554" (*Imatge56*).

Imatge55



Imatge56

- *Descarregar l'app i instal·lar-ho al dispositiu:* Una altre forma de veure-ho és instal·lar l'app mateixa al dispositiu encara que no es recomana ja que els altres dos mètodes són més ràpids. En tot cas, s'explica a l'apartat següent com es fa per quan ja tinguis l'app llesta.

3.3.3. Instal·lació de l'aplicació al dispositiu

Una vegada veiem que ja tenim l'app tal i com la volíem i ja la volem tenir, hem de passar a fer els següents pasos:

A la part gràfica, a la part amunt dreta, hem de clickar a la part següent: "Package for Phone" → "Download to this Computer"(*Imatge57*).

Imatge57

Un cop clickat, ens sortirà una finestra on s'ha de completar un percentatge fins arribar al 100%. Quan arribi, és quan se descarregarà l'aplicació en forma .apk. Aquest serà l'arxiu que hem de ficar al dispositiu.

Un cop ficat al dispositiu, s'haurà de fer com a qualsevol aplicació: anar a ella, instal·lar-la i ja funcionarà al executar-la.

4. APLICACIÓ CREADA

En aquest capítol s'explica l'aplicació d'exemple creada a través de AppInventor amb les corresponents parts que la componen.

4. Aplicació creada

En aquest apartat, s'explicarà tot el procés de creació i disseny d'una aplicació. Aquesta aplicació tracta sobre una plataforma per l'estudi del idioma japonès.

Les funcions que hi hauran a l'aplicació són 2 ben marcades:

- ***Jocs per l'aprenentatge dels caràcters:*** Al japonès hi ha molts caràcters per aprendre. Per aquest motiu, s'ha creat una sèrie de jocs per aprendre'n el caràcter en si, la seva pronunciació i el seu significat. La metodologia és molt intuïtiva, per exemple, sortirà un caràcter japonès i a sota hi haurà 4 possibles respostes. Si toques la resposta correcta, sortirà un missatge com que està correcte i pots clickar al botó "Pregunta següent" per passar a la següent. En cas de que no clickis la resposta correcta, sortirà un missatge que està malament i podràs clickar el botó "Pregunta següent" també per passar a la següent. Sempre hauràs de clickar alguna resposta, sino no et deixa passar a la següent. La base de dades del 3 jocs estan basades en 80 caràcters japonesos.

- ***Joc geogràfic:*** Per conèixer amb profunditat una llengua, és important saber sobre la geografia del país. Per això, s'ha creat una funció per poder aprendre els caràcters de totes les ciutats importants del país. En aquesta part, estarà el mapa on podràs clickar qualsevol prefectura on veuràs que es marcarà (en aquest cas, la regió marcada estarà de color verd) i sortirà el caràcter del nom de la ciutat. En cas de no saber-ho, pots clickar al nom que surt en la pantalla mateixa i et guiarà fins a Google Maps, on sortirà el nom de la ciutat en el nostre alfabet. A més, com estàs connectat a Google Maps, pots accedir a les altres funcions típiques de la mateixa aplicació Google Maps: pots mirar carrers, com accedir-hi, llocs d'interès, etc.

Comencem doncs amb els diferents apartats en que consisteix aquesta aplicació.

L'aplicació, quan s'inicialitza, consisteix en una pantalla principal amb 6 botons diferents on et poden portar a 5 pantalles diferents a més de sortir de l'aplicació.

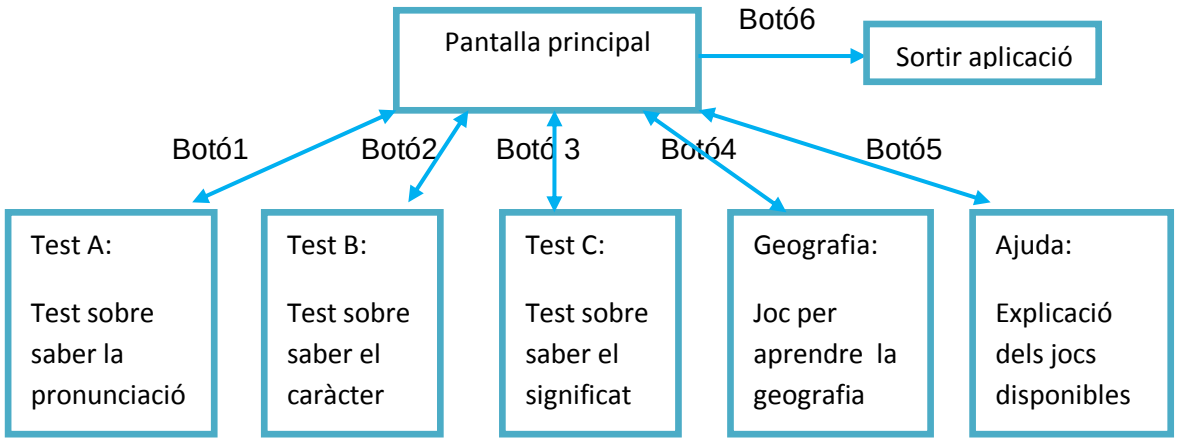


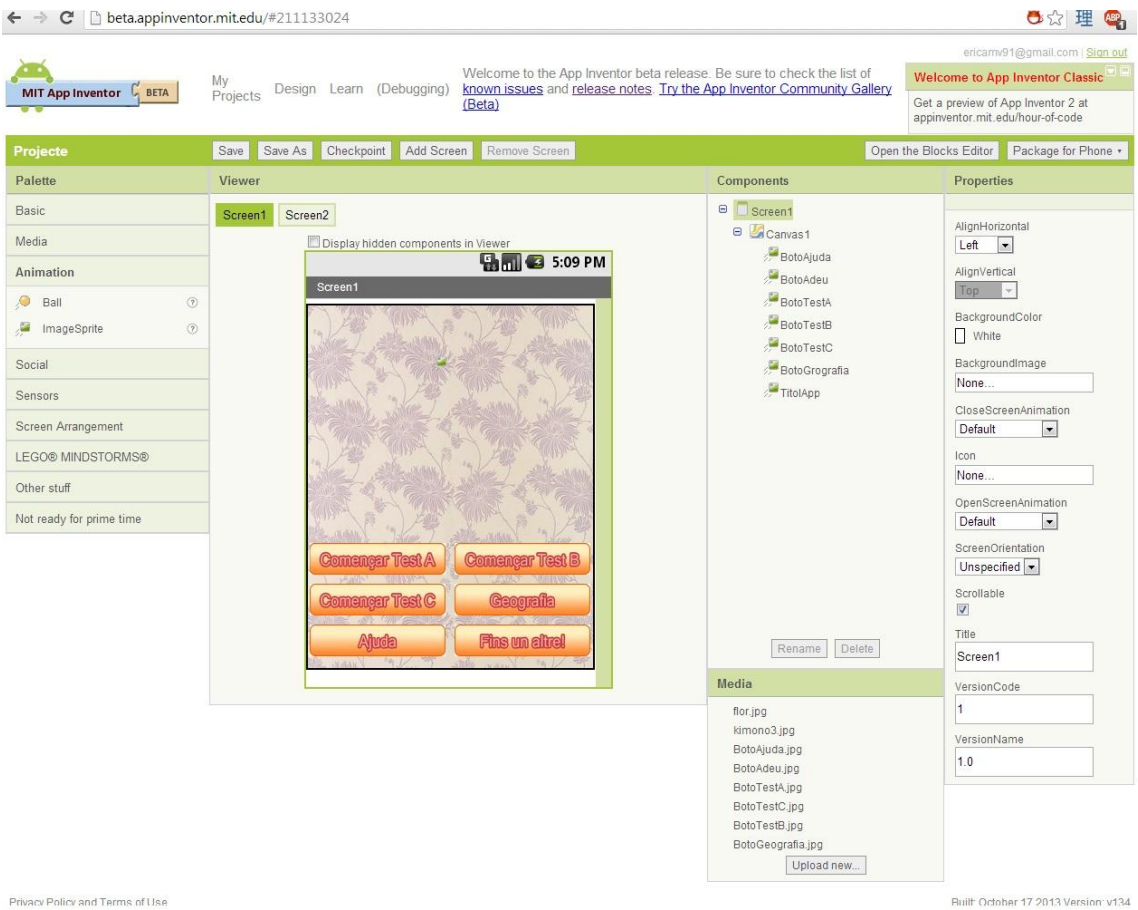
Figura1

En qualsevol de les 5 Screens, pots tornar a la Pantalla Principal amb un altre botó.

4.1. Jocs

La part de Jocs de l'aplicació tracta sobre una sèrie de tres jocs on es valorarà els coneixements dels caràcters japonesos. Concretament, cadascú valora: Coneixement del caràcter, pronunciació i significat.

Per tant, vaig crear 3 Screens diferents per a cadascuna dels jocs. Principalment, volia una Screen principal on fos el menú principal i, a partir d'allà, em pogués permetre moure a les diferentes opcions de jocs (diferentes Screens, una per cada joc) que hi ha. Com podem veure a la *Imatge58*, veiem la pantalla principal que es va crear per seleccionar les diverses opcions que tenim: Test A, Test B, Test C, Geografia, Ajuda i Sortir de l'aplicació.

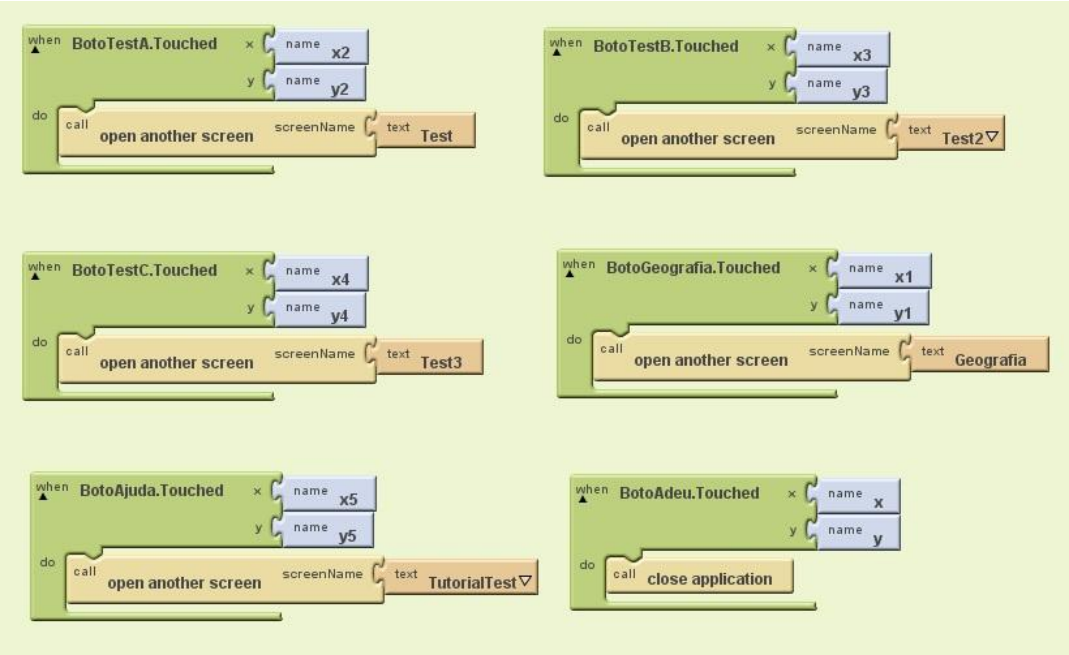


Imatge58

Pel que fa a la programació de la pantalla principal, seria la que veiem a la *Imatge59*, que són els blocs que fan que pasin d'una Screen a una altre per a cada botó corresponent. En el cas del Botó pel Test A (anomenat per mi a la interficie gràfica

com a "BotoTestA"), hem agafat el bloc que quan el toquem, se'n va a la Screen corresponent, en aquest cas, la Screen anomendada "Test", a on ens porta al TestA.

Pels altres botons és el mateix mètode excepte l'últim de tots, ja que aquest botó no ens porta a cap Screen concreta, volem que ens tanqui l'aplicació perquè volem sortir d'allà. En aquest cas, el bloc serà diferent i seria el que surt a la mateixa *Imatge59.*



Imatge59

4.1.1. Creació de Test A; "Conèixer la pronunciació"

L'objectiu principal del TestA és escollir la pronunciació correcte d'un caràcter japonès que surt de forma aleatoria sobre 4 possibles respostes.

Quan selecciones l'opció TestA hauria de sortir una cosa com *Imatge60*:

Imatge60

Com podem veure així de començament, hem hagut d'utilitzar diversos elements, alguns de repetits. Els elements utilitzats són:

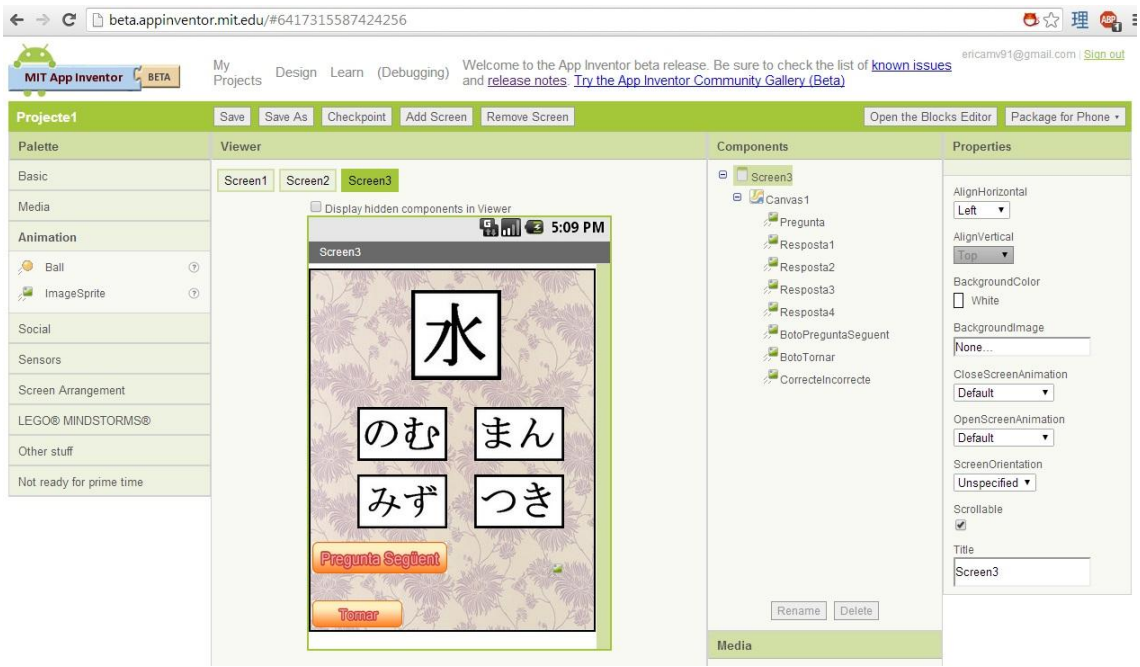
- 1 Canvas
- 8 ImageSprite en total.

El Canvas és principalment per poder posar posteriorment les ImageSprite (recordem que no es poden col·locar imatges sense el Canvas i, posteriorment, ImageSprite).

Les ImageSprites les hem utilitzat per posar la pregunta, les 4 possibles respostes, el botó de "Pregunta següent", el botó de "Tornar" i una imatge amagada que sortirà si has posat la resposta correcte o no, que com heu vist a la *Imatge60*, no es veu ja que està amagada fins que algú toqui alguna de les respostes.

Hem utilitzat les ImatgeSprite perquè volia donar un propi disseny creat per nosaltres ja que amb els Button no donava molta varietat pel que era en tema disseny.

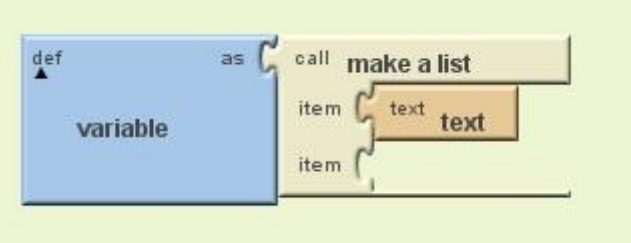
Sobre els elements i la interfície gràfica utilizada per aquest joc, és la que podem veure a continuació a la *Imatge61*.



Imatge61

Per la part de programació d'aquesta part, hem de fer diverses funcions per separat. En primer lloc, hem de crear una llista per totes les possibles preguntes i respostes que poden haver (la base de dades, com hem comentat abans, tracta sobre 80 preguntes diferents). Per una altre banda, les preguntes seran de forma aleatoria, les respostes també, s'ha de generar la pregunta, veure si és correcte o no i donar la informació d'aquesta.

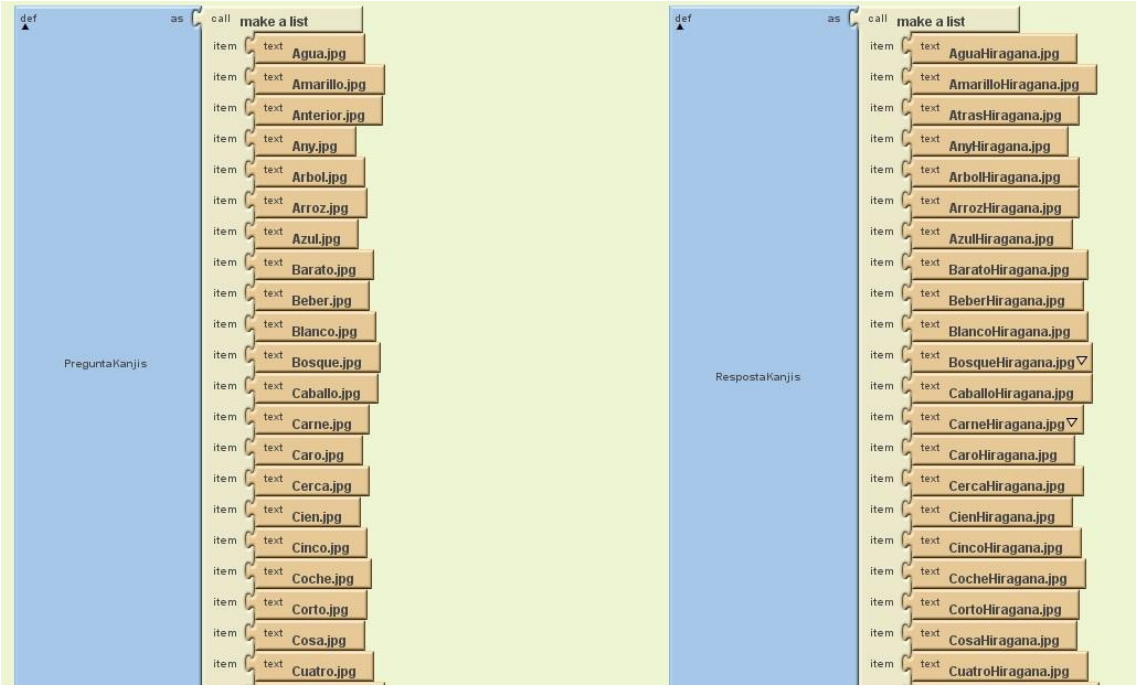
Comencem primer per la part d'afegir tota la base de dades de preguntes i respostes a través d'una llista, amb els seus corresponents blocs (*Imatge62*).



Imatge62

Per afegir totes les possibles preguntes i respostes, s'han de crear dos llistes a partir d'una variable, una variable que l'he anomenada PreguntaKanjis (per les preguntes que hi poden haver) i una altre llista creada a partir d'una variable anomenada RespostaKanjis (per les seves repectives respostes). Llavors, per una part tenim la base de dades de preguntes i per una altre banda la de respostes. Com podem saber si són correctes o no les respostes? Per poder crear bé aquesta part s'ha d'anar amb compte ja que el primer de la llista de la variable de PreguntaKanjis que posis, la seva resposta correcta també ha de ser la primera de la llista de la variable de RespostaKanjis. Han d'anar ordenades una per una. Per això, el programa detecta quina de les respostes és correcte i quina no. Per la resta de respostes, es generarà de forma alteatoria (Random).

Per veure com quedaria, a la *Imatge63* veiem una part de la llista de preguntes i respostes.



Imatge63

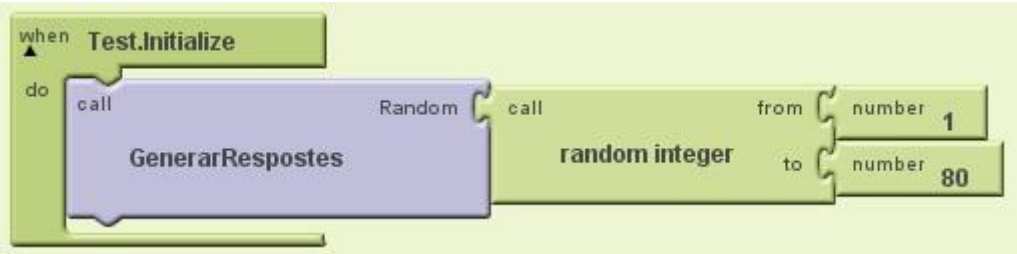
Un cop tenim tota la base de dades col·locades, tant preguntes com respostes (80 per cada llista, com he comentat abans), ja podem fer la programació per la creació de preguntes, generar el Random, etc.

Primer, hem de deixar clara la funció del Botó que torna al menú principal (BotoTornar), tal i com veiem a la *Imatge64.*

Imatge64

D'altra banda, també hem de crear la primera pregunta de totes, ja que quan prems el botó on et porta el Test A, ja ha de generar la primera pregunta amb les seves corresponents respostes per començar el test. Per a que es generin les preguntes i respostes res més entrar al Test A, s'hauria d'agafar el bloc que quan Test A s'inicialitzi (Test.Initialize), llavors genera les preguntes i respostes de forma aleatoria gràcies al bloc predefiniti que hi ha de "Random integer" i donar el valor de quin número fins quin altre vols que faci el Random. En aquest cas, voldrem que sigui de 1 a 80, ja que tenim 80 possibles preguntes en total, volem que faci el Random de tot.

Per tant, la programació de blocs quedaria com a la *Imatge65.*



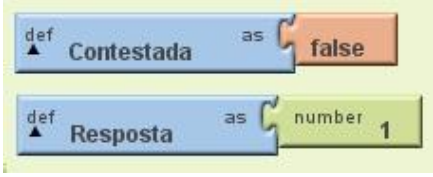
Imatge65

Una vegada s'hagi definit a l'atzar la pregunta a mostrar, s'haurà de definir la resposta correcta i 3 d'incorrectes. Tot això ho farem a la inicialització de les respostes.

Encara que la part de pregunta no està acabada, encara queda definir quina és la correcta i quina no, però ho mirarem més endavant. Això és la inicialització de les respostes.

Per la part de generar la pregunta, és una mica més complexe. Abans de ficar-nos en com es fa, hem de tenir en compte que hem de crear unes variables per després col·locarles a blocs necessaris (per exemple, al de generar preguntes) ja que sino no funcionarien. Hem de definir uan constant (Random) i 4 variables (RespostaCorrecte, Contestada, Resposta, Atzar) (*Imatge66* i *Imatge67*), i hem d'enrecordar-nos que tenim també les variables PreguntaKanjis i RespostaKanjis gràcies a la llista creada d'abans.

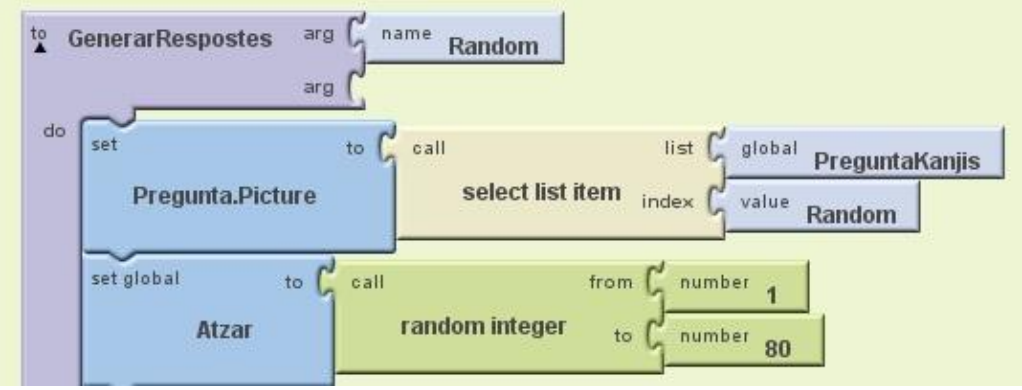
Imatge66



Imatge67

Per a què són aquestes variables? Per definir un número que, posteriorment, la funció la detectarà i haurà de comprar-la amb ella, com ja veurem més endavant.

Llavors, pel cas de generar preguntes, seria de la següent forma (*Imatge68*).

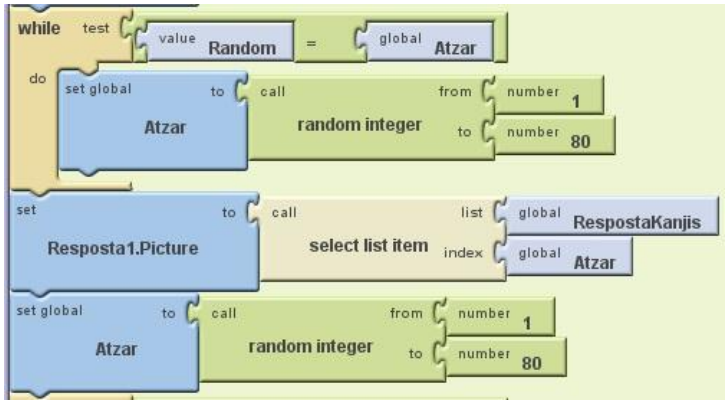


Imatge68

Com podem veure, en el bloc de Pregunta.Picture, posem les variables de PreguntaKanjis i Random, ja que hem d'escollir una pregunta de les que ja hi ha prèviament col·locades a través de la llista creada per la variable PreguntaKanjis. Però no és suficient, ja que necessitem que es generi de forma aleatòria. Per això, també s'ha creat la constant Random, per a que es faci aleatòriament la selecció de la pregunta a través també del "Random integer", que genera la pregunta de forma aleatòria per tota la llista sencera.

Un cop ja tenim les preguntes i respostes generades de forma aleatòria, hem de continuar amb les respostes, ja que encara no hem dit que surti la resposta correcta.

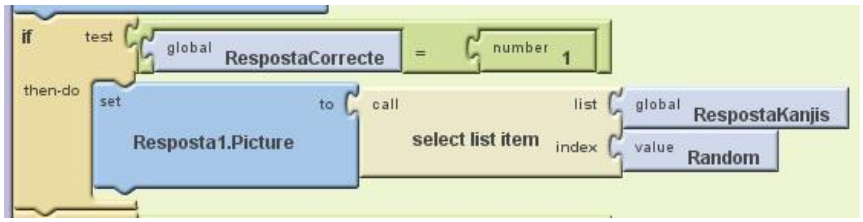
Per tant, per generar les respostes, hem de fer una per una. Posem com a exemple per a la Resposta1. S'ha de fer el mateix que per a la pregunta. Hem de col·locar el random a més d'especificar la variable RespostaKanjis i la constant Random per a que sigui una resposta aleatòria generada a través de la llista creada anteriorment (*Imatge69*).



Imatge69

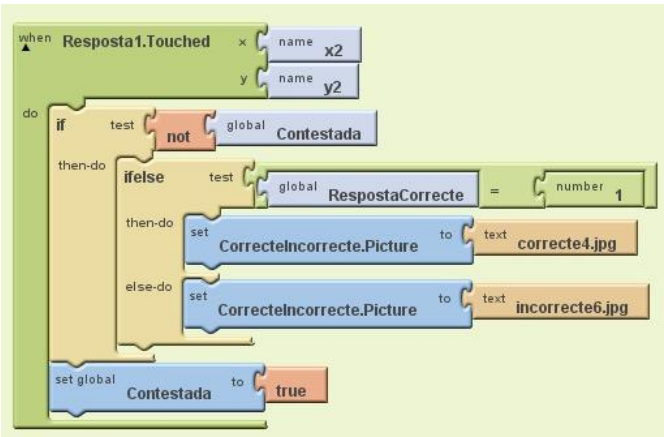
Però no acaba aquí. Ara s'ha generat la resposta totalment aleatòria però, com sabem que és la correcta? Ara ho hem de programar.

Per això, hem creat abans la variable RespostaCorrecte, ja que és la variable que tenia com a valor preferent un 0, però per a cada resposta li hem donat un valor qualsevol, en el cas de Resposta1, era 1. En el cas de Resposta2, era un 2 i així successivament.



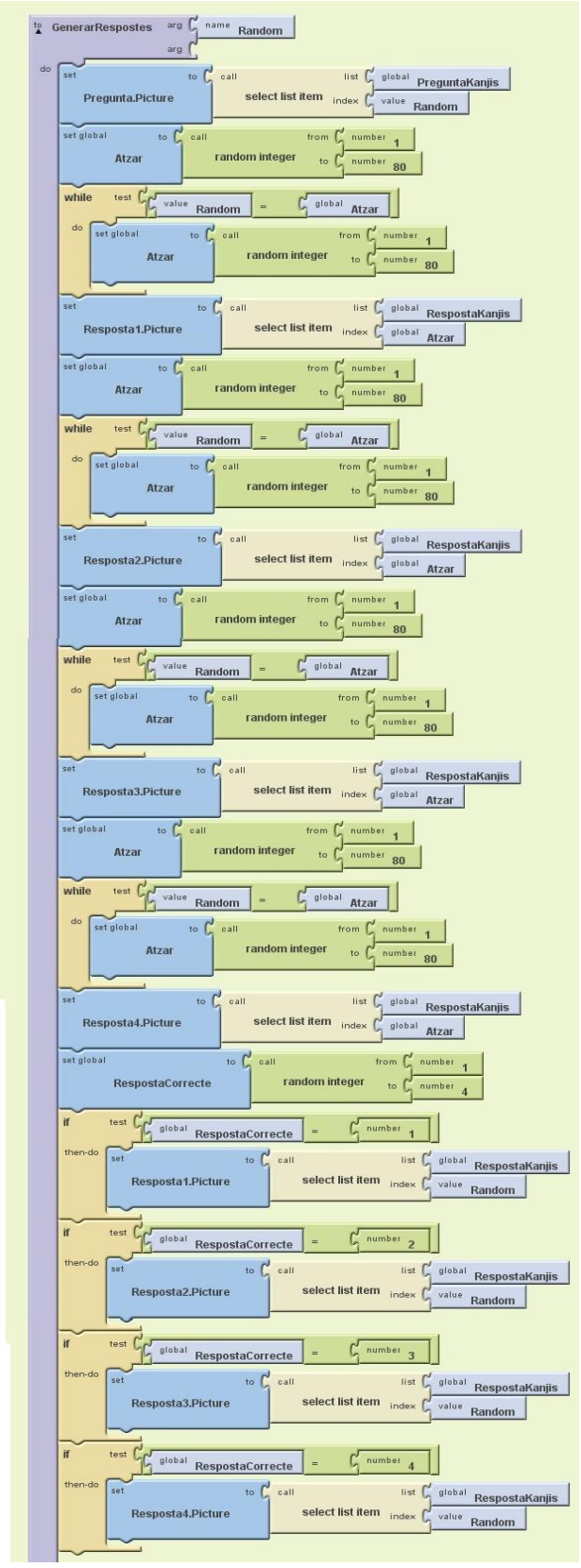
Imatge70

Per què donar aquests valors numèrics? Per a la següent funció que es farà. Per una altre banda, hem de veure si es correcte o no la resposta Ara comparem gràcies a la variable RespostaCorrecte si coincideix amb el mateix valor que li hem donat anteriorment. Si és el cas que coincideix, llavors és correcte i sortirà una imatge com que es la resposta correcta. Si no coincideix, sotirà una imatge com que és incorrecte (*Imatge71*).



Imatge71

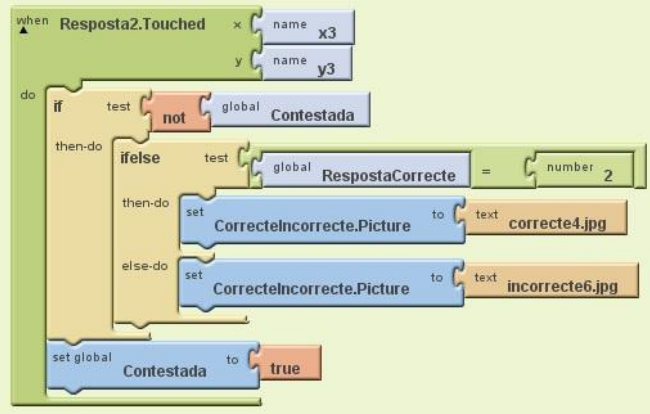
Ja tenim fet la programació per a una pregunta i la resposta donant si és correcte o no. Per aquesta programació s'ha de fer el mateix per les 4 respostes (*Imatge72*).



Imatge72

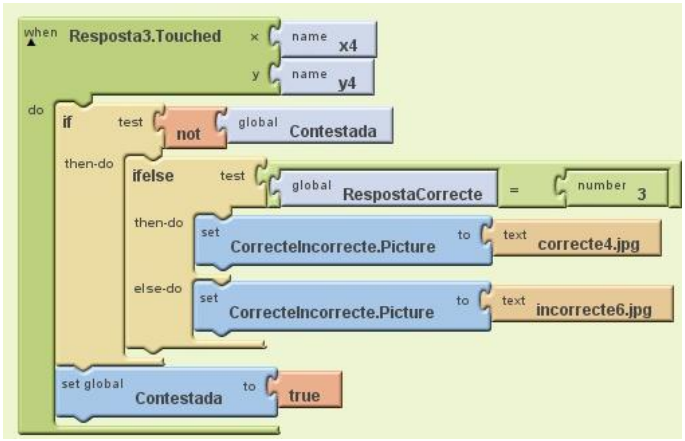
I pel que fa la comprovació de les 4 respostes que siguin correctes, és la mateixa metodologia feta que com hem explicat anteriorment per a la Resposta1.

Resposta2 (*Imatge73*):



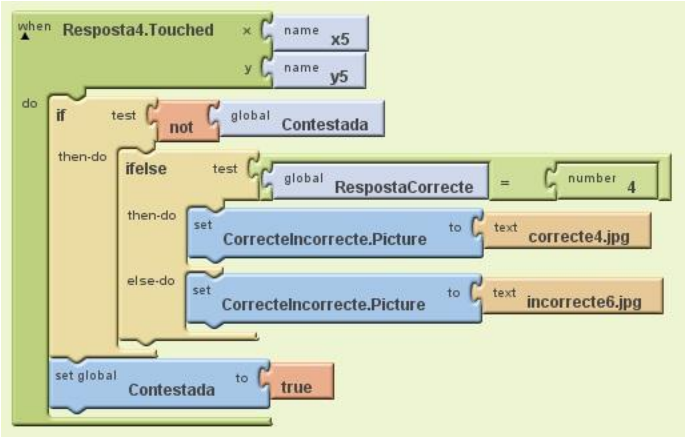
Imatge73

Resposta3 (*Imatge74*):



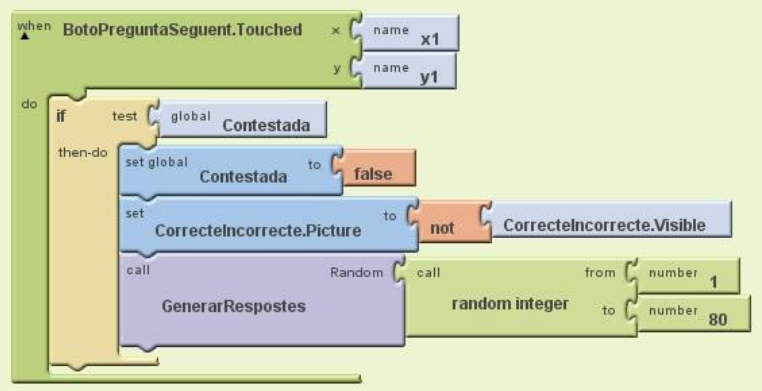
Imatge73

Resposta4 (*Imatge75*):



Imatge75

Per últim, només quedaria la programació per quan toquem el botó que fa la pregunta següent (BotoPreguntaSeguent), que quedaria com a la *Imatge76.*

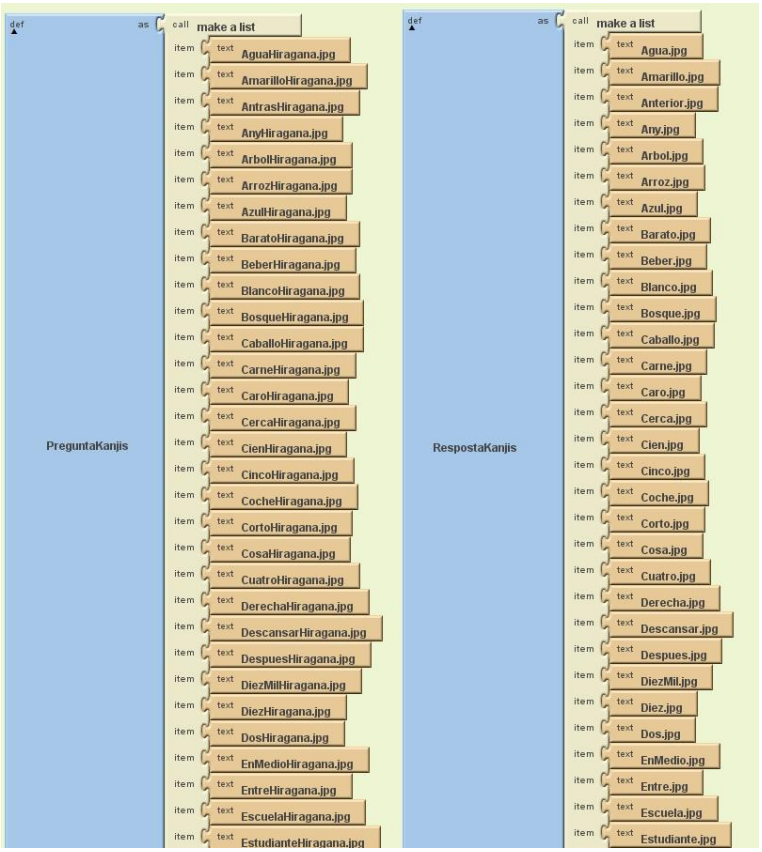


Imatge76

4.1.2. Creació de Test B; "Conèixer el caràcter"

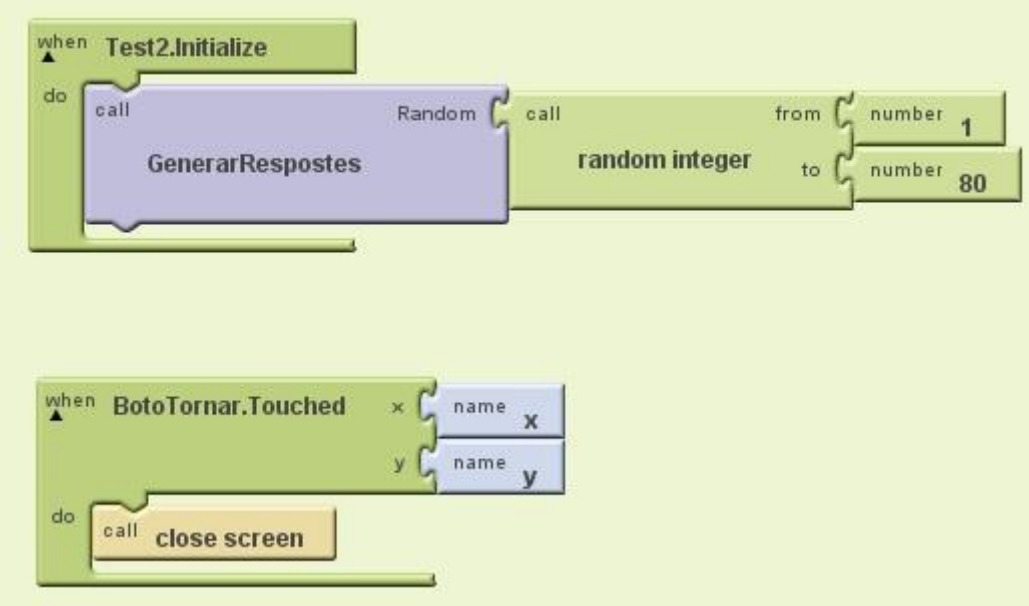
Per la creació d'aquest joc s'ha utilitzat la mateixa metodologia però canviant algunes variables i les llistes, ja que tenen diferents preguntes i respostes.

Per a la creació de les llistes, a la *Imatge77* tenim una petita part d'ella.



Imatge77

Pel que fa a quan s'inicialitza l'aplicació (*Imatge78*).



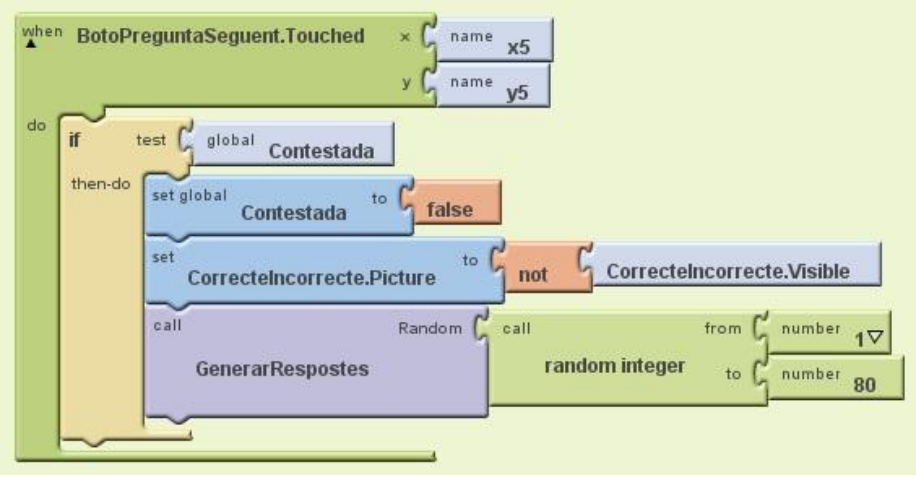
Imatge78

Pel que fa a les variables:

Imatge79

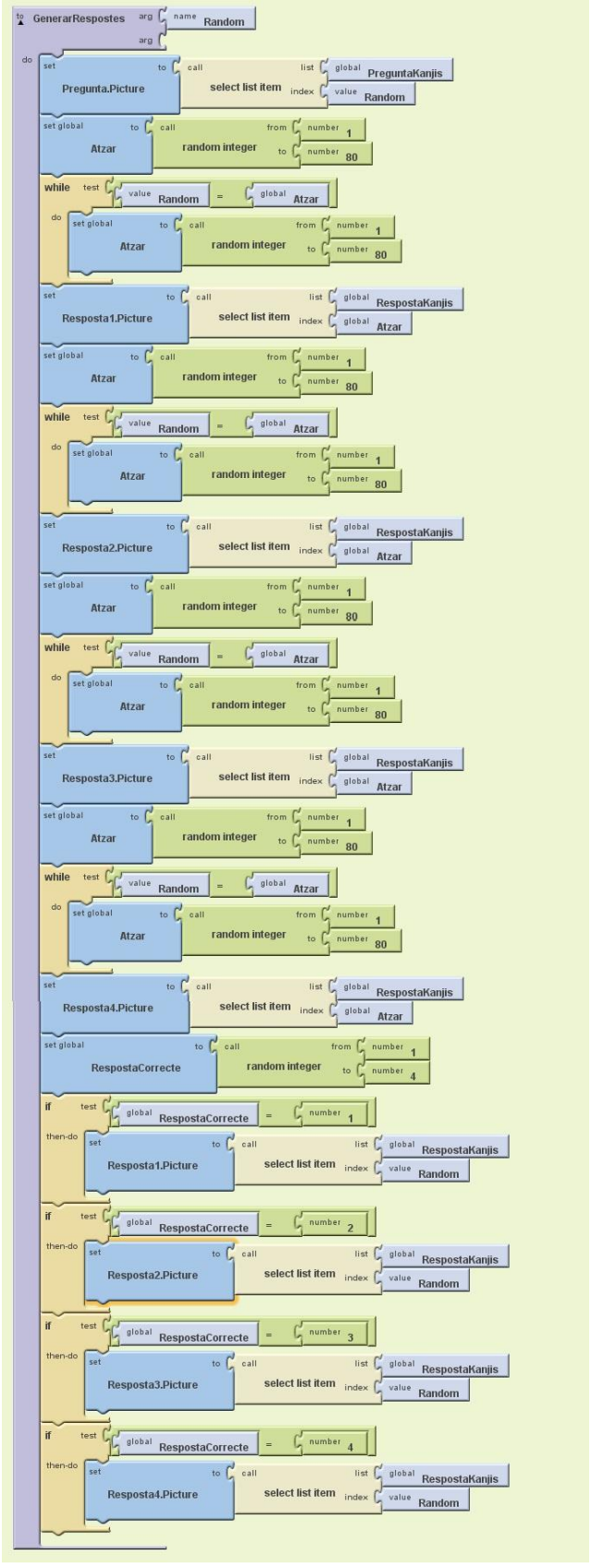
Imatge80

Pel que fa al botó de la pregüenta següent (BotoPreguntaSeguent):



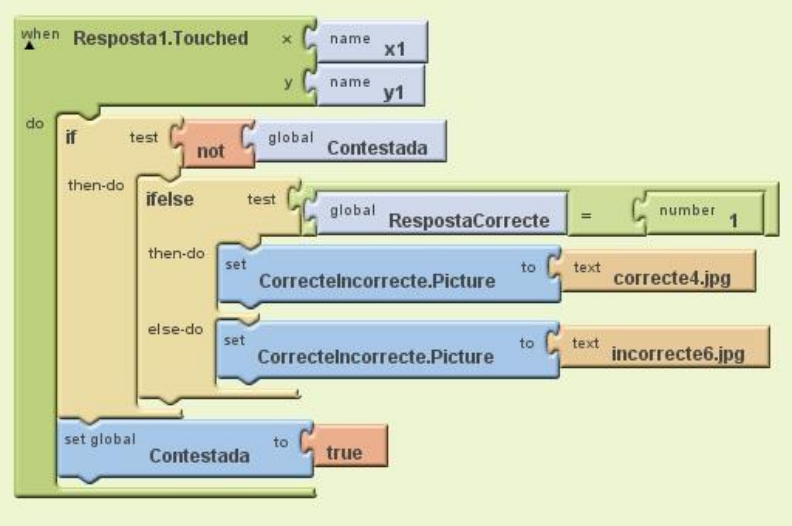
Imatge81

Pel que fa a la generació de preguntes i respostes (*Imatge82*):

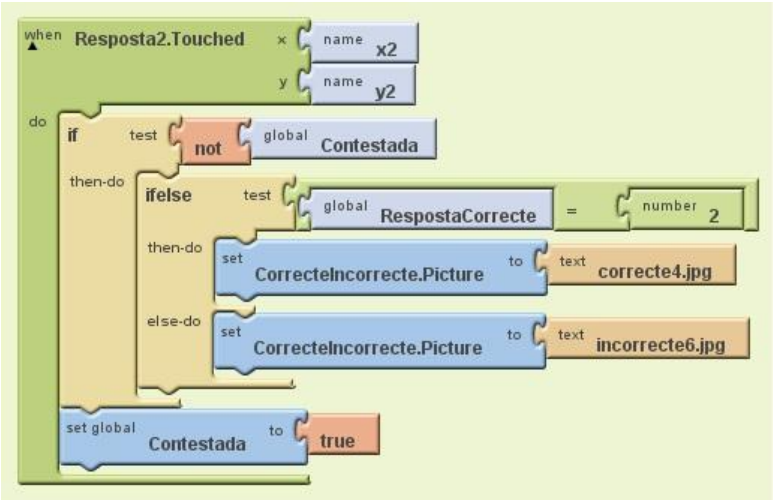


Imatge82

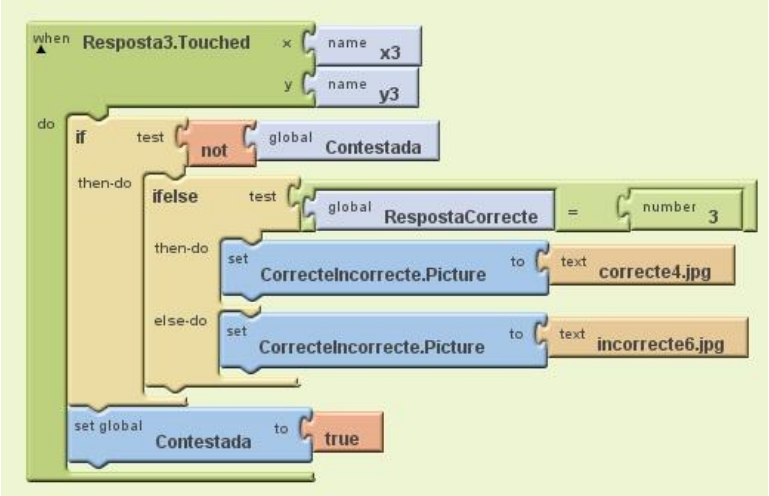
Pel que fa a la comprovació dela resposta correcta:



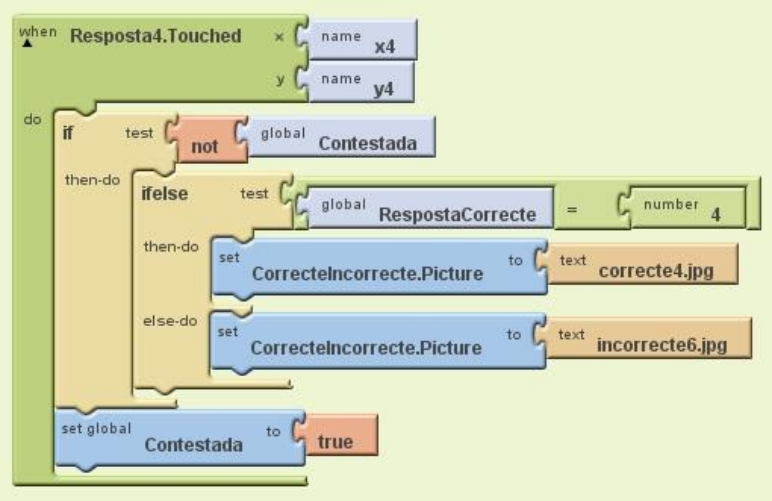
Imatge83



Imatge84



Imatge85

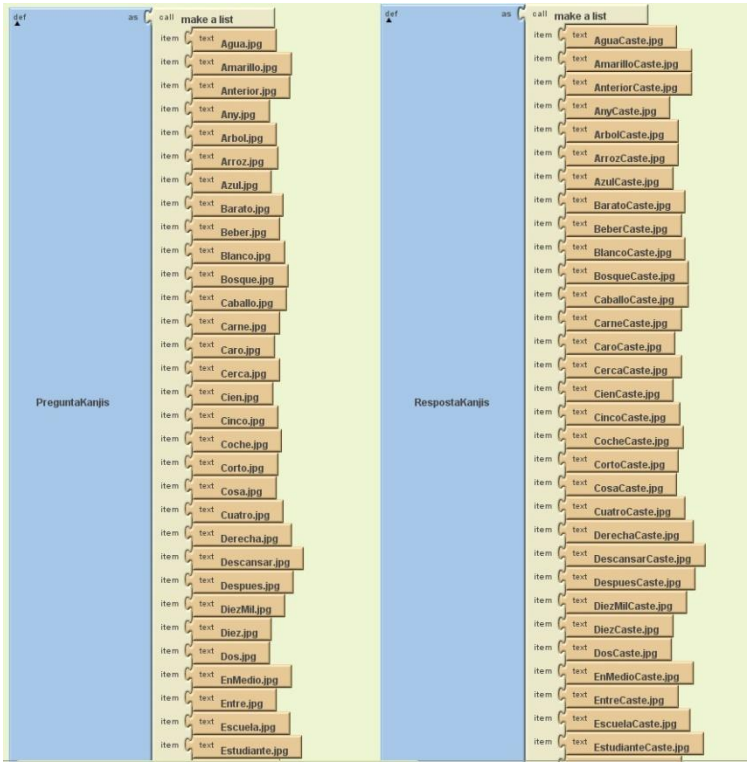


Imatge86

4.1.3. Creació de Test C; "Conèixer el significat"

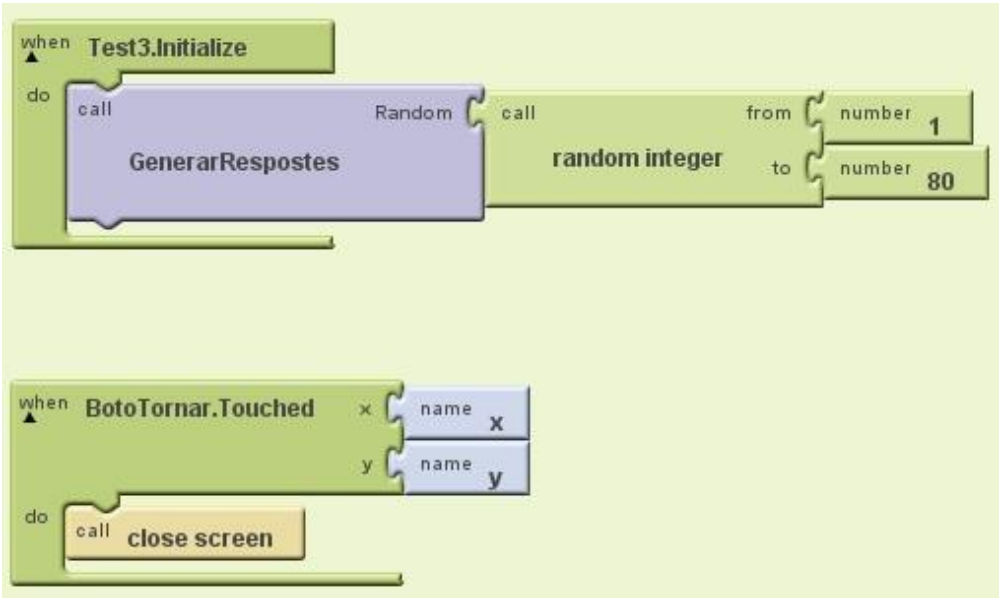
Per la creació d'aquest últim joc, també s'ha utilitzat la mateixa metodologia però canvien algunes variables i les llistes, ja que tenen diferents preguntes i respostes.

Per a la creació de les llistes, a la *Imatge* tenim una petita part d'ella.



Imatge87

Pel que fa a quan s'inicialitza l'aplicació (*Imatge88*).



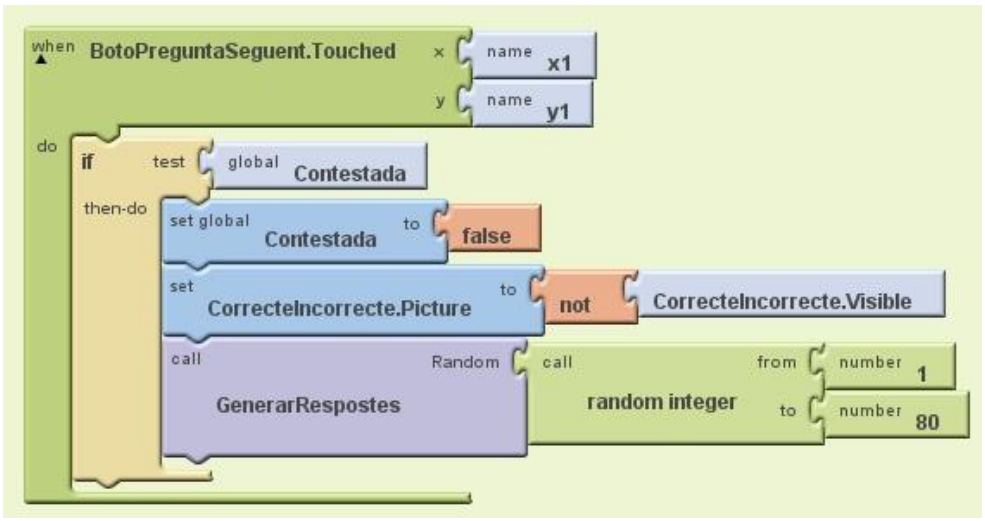
Imatge88

Pel que fa a les variables:

Imatge89

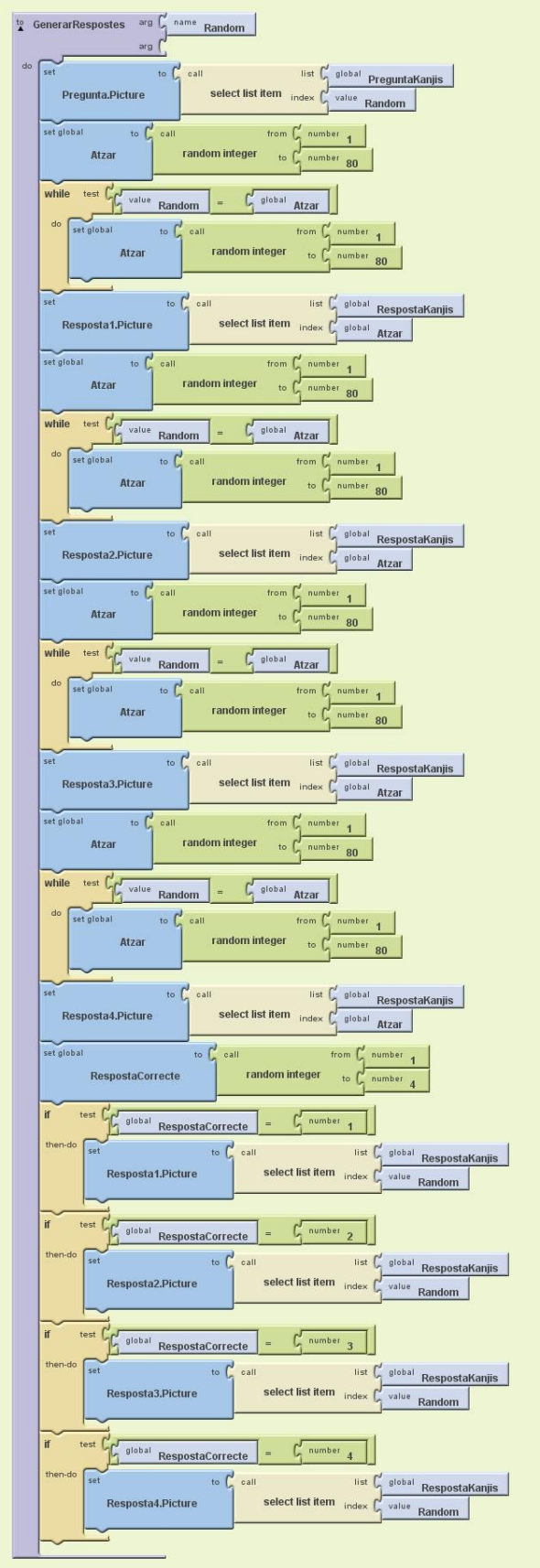
Imatge90

Pel que fa al botó de la pregüenta següent (BotoPreguntaSeguent):



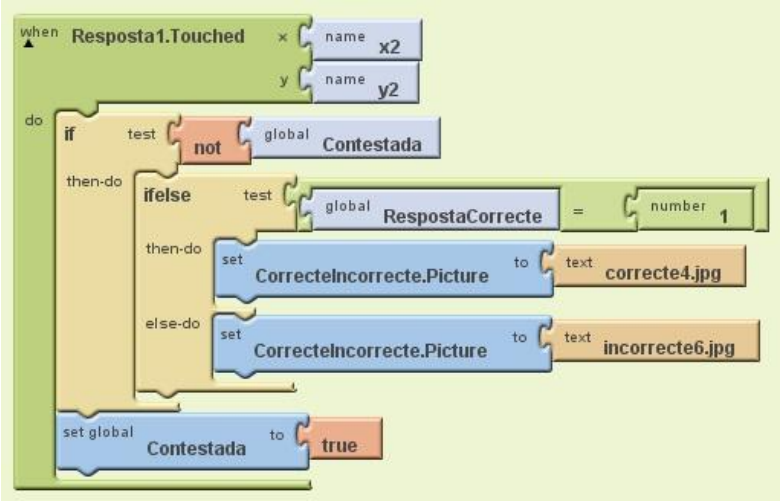
Imatge91

Pel que fa a la generació de preguntes i respostes (*Imatge92*):

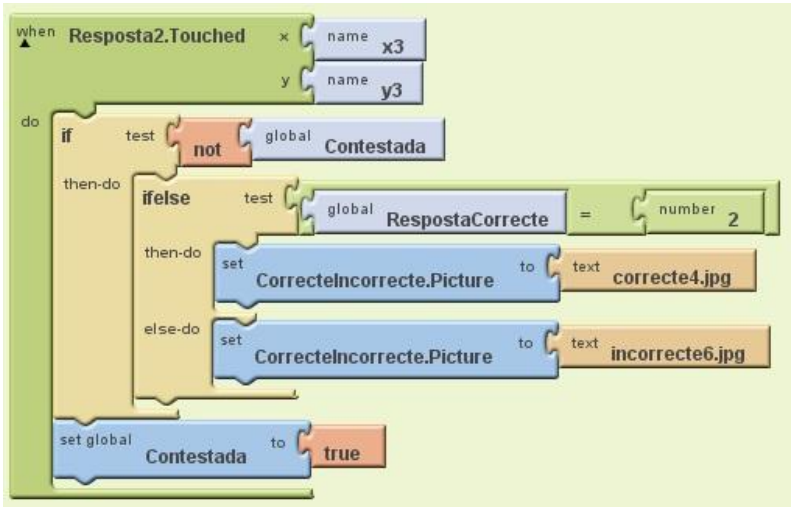


Imatge92

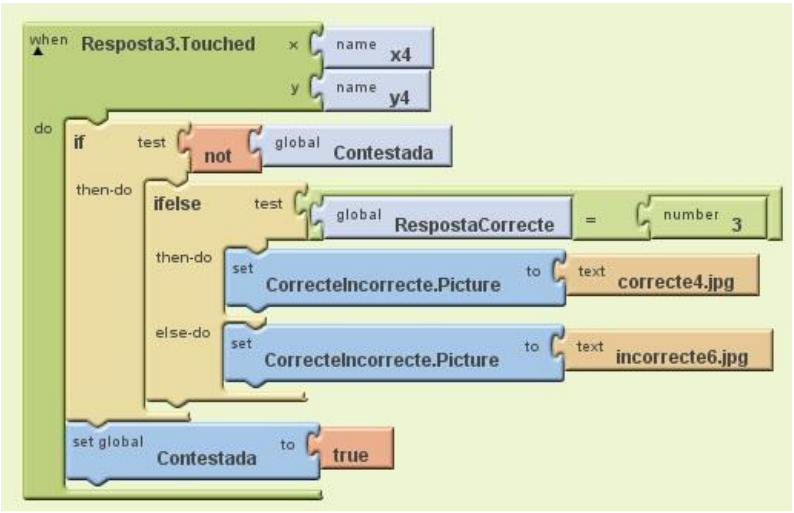
Pel que fa a la comprovació dela resposta correcta:



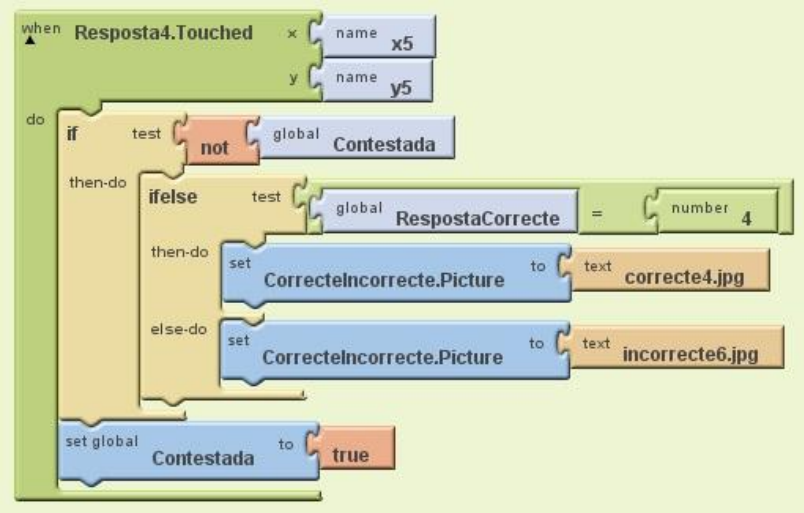
Imatge83



Imatge84



Imatge85



Imatge86

4.2. Geografia

En aquesta altre part de l'aplicació tracta sobre els coneixements geogràfics del Japó i dels caràcters de les prefectures i ciutats més importants del país.

En aquí, es comença amb un mapa del Japó amb les seves prefectures clarament separades. Pots clickar a qualsevol de les prefectures que, a la vegada, pots veure que sortirà els caràcters del nom de la ciutat en qüestió. En aquesta part és per veure fins quin punt pots llegir els caràcters de les prefectures més importants del país.

A més d'això, en cas de no saber-ho, pots clickar al nom en qüestió i l'aplicació et porta al lloc exacte a través de Google Maps. Gràcies a això, podem observar, a més del nom de la prefectura/ciutat en lletres occidentals, podem veure altres opcions de Google Maps com veure els carrers, com arribar fins allà, etc.

4.2.1. Creació

Primer s'ha de començar configurant la part de Google Maps. Per accedir-hi a Google Maps, s'ha d'incloure una petita variable anomenada ActivityStarter. És un d'aquests elements que no és visible, ja que a la pantalla del Viewer no sortirà res i l'element en si mateix estaria a sota de la pantalla.

Col·loquem l'element ActivityStarter de la Paletta (a l'apartat de Other Stuff) fins a la pantalla.

Un cop posat, ho tindrem col·locat i ja podem modificar aquests paràmetres.

En el cas d'utilitzar Google Maps només hem d'afegir 3 paràmetres que ens faciliten a continuació, a les parts corresponents com podem veure a la següent imatge:

Els paràmetres en qüestió són:

Action: android.intent.action.VIEW

ActivityClass: com.google.android.maps.MapsActivity

ActivityPackage: com.google.android.apps.maps

Per una altra part, un cop posat tots els paràmetres necessaris per Google Maps, hem de començar a crear el mapa.

En aquesta part, he optat per utilitzar només imatges ja pre-seleccionades anteriorment.

En primer lloc, els elements que hem utilitzat ha sigut un Canvas que ocupa tota la pantalla, a més de ImageSprite, tantes com necessitem per nombre de ciutats de prefectures del Japó (en aquest cas, 47).

Una d'aquestes ImageSprite farà de mapa sencer de Japó mentre que un altre farà del caràcter japonès del nom de la ciutat seleccionada a partir de l'usuari. La resta de ImageSprites són de les ciutats de les seves prefectures.

En aquesta apartat, modificarem la mida (Width i Height) i les coordenades (X, Y, Z) per a que quedi ben posicionat a la pantalla.

Comencem a ficar les imatges en cada ImageSprite. En aquest cas, primer fiquem el mapa del Japó i després les prefectures una per una ja seleccionades amb un color diferent per a que quan es clicki, canviï el color d'aquesta en concret. A més, hem d'afegir una petita imatge a sota del mapa que serà on es ficarà el caràcter de la ciutat clickada.

Un cop fet això, ja tenim una prefectura realitzada. Ara ho hem de fer per la resta de prefectures i sortirà una cosa com aquesta:

Imatge87

Això per la part de disseny. Ara pasem a la part de programació de blocs.

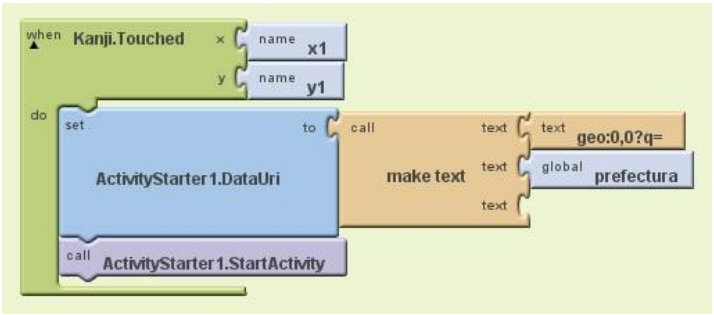
Per començar, hem de programar per a que quan es toqui una de les ciutats, quan surti el nom en caràcters japonesos, si clickem aquesta part, ens ha de portar a la ciutat a través de Google Maps.

Primer, Google Maps s'inicialitzarà un cop que l'usuari toca la ImageSprite anomenada Kanji (nom de la variable que he anomenat jo mateixa per definir la ImageSprite on sortirà els caràcters japonesos quan toques una ciutat a través del mapa).

Llavors, el bloc adequat per això és el de Kanji.Touched. Aquí s'activa el bloc específic de ActivityStarter1.DataUri, que afegim un text, que és el link que porta a Google Maps, a més d'una variable. Aquesta variable, és la de "Prefectura". Això significa que Google Maps buscarà la localització a través d'aquesta variable, ja que agafarà el valor del nom de la ciutat tocada al mapa.

A més, ha de trucar a la funció de ActivityStarter1.

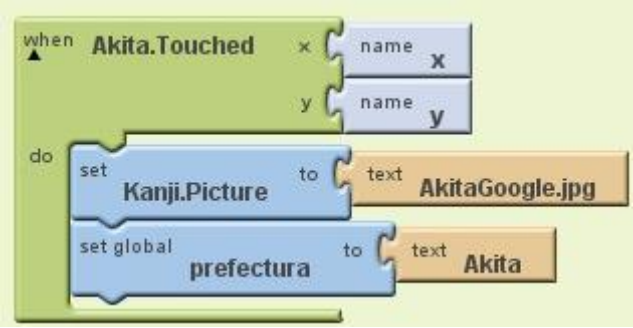
La programació quedaria tal com està a la *Imatge88*:



Imatge88

Un cop es veu que quan prems el kanji ja pots entrar a Google Maps i et dóna la localització correcte, només cal introduir aquesta part a cada una de les prefectures de tot el pais.

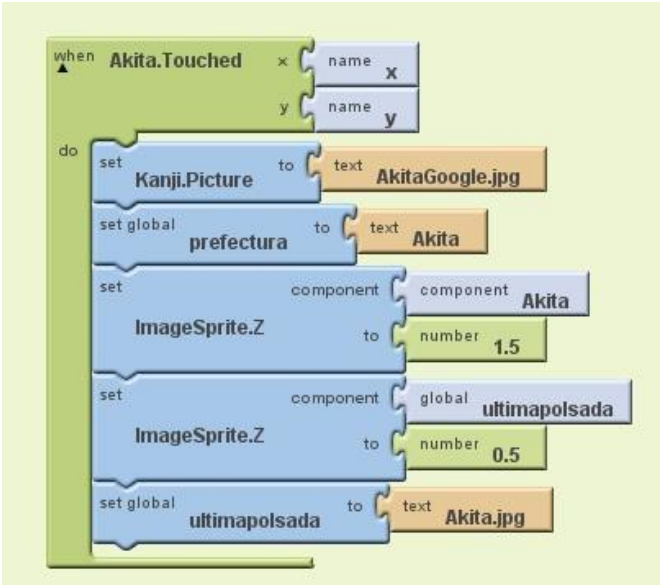
Veiem com es fa per a cada una. En aquest cas, el faré de la ciutat Akita (*Imatge89*):



Imatge89

Per aquesta programació, hem d'afegir el bloc de Kanji.Picture per afegir la imatge del carácter japonés de Akita. Després, hem d'afegir la variable "prefectura" per deixar en constancia el nom de la ciutat. En resum, quan l'usuari premi la ciutat d'Akita, sortirà la imatge del carácter de Akita i, a més, la variable prefectura agafa el valor del text "Akita". Això significa que quan premi a Kanji, em portarà a Google Maps (com hem explicat abans) i el mateix Google Maps, buscarà la localització d'Akita.

I per últim, afegir els blocs necessaris per a que la regió que clickem es quedi marcada, en aquest cas, de color verd. Per això, haurem de fer-ho a través de les posicions Z del mapa i la imatge de la regió de color verd (que ja haurem pujat al programa al principi, juntament amb la base de dades dels caràcters). Quan es clicka, la posición Z de la imatge canvia i es posa per damunt del mapa principal. D'aquesta forma es pot amagar o que estigui per a sobre del mapa.



Imatge90

5. PRESSUPOST

En aquest capítol s'explica les hores i diners invertits en el projecte.

5. Pressupost

Sobre el projecte, hi ha sempre un pressupost que cal definir per saber el valor que té.

En aquest cas, és difícil un pressupost adient i exacte per a un projecte final de carrera sobre un estudiant que, a més, no tenia coneixements previs d'Android i ni cap experiencia al sector. Per tant, el pressupost final podria ser per alguns força bé o per altres, fluix. Primerament, volia aclarar sobre les condicions de com s'ha fet aquest projecte.

El projecte fet és crear i dissenyar una aplicació pels dispositius amb sistema operatiu Android sobre una plataforma per l'estudi de japonès. Aquest projecte el podria separar per 2 parts:

1- Hores de formació bàsica
2- Hores de dedicació al projecte

Sobre les hores de formació, les comptaré com a extra ja que com la meva especialització no es programació ni res semblant al que es fa en altres titulacions o rames com seria l'enginyeria informàtica. Per tant, sobre les hores de formació, comptaré com unes 40 hores. 40 hores que seria per aprendre bé el programa AppInventor, anar provant programes senzills per veure com funciona, veure la seva interfície, etc.

Sobre les hores dedicades al projecte, un cop ja he rebut la formació bàsica per ja poder crear aplicacions per a Android ja vaig començar a fer la meva aplicació (que ja estava pensada d'abans de fer la formació bàsica i prèvia). L'aplicació es va crear i fer en un total de 250 hores aproximadament. Les 250 hores inclouen: programació de l'aplicació, recol·lecció d'imatges de tota la base de dades, disseny de l'aplicació i testejar-lo tantes vegades fins que sigui tal i com volia.

En resum, l'aplicació es va crear en: 40h formació + 250h creació i disseny. Seria un total de 290h. El preu per hora depèn moltíssim de per a quina empresa treballaríem o si som autònoms, etc però en aquest cas, faré una mitjana i posaré que siguin 12 € la hora treballada, com a becari en pràctiques en una empresa i tenint en compte que l'aplicació no és a mida per a un client. En aquest cas, ens surt un preu total de:

250h X 12 € = **3.000 €**

Ara anem a fer un estudi de mercat. He mirat a través de Google Play diferents aplicacions de la mateixa matèria i he trobat uns quants interessants. He agafat les que tenien preu per fer una comparació de preus i descàrregues ja que s'intenta veure a través d'altres aplicacions si surt a compte i veure si es possible que es pugui retornar tota la inversió feta.

Abans de fer l'estudi, s'ha de tenir unes coses clares a l'hora d'aconseguir diners a través de les aplicacions, ja que hi ha 2 formes:

1- Aplicacions de pagament. En aquest cas, el creador de l'aplicació s'emporta el 70% per a cada descarrega feta.
2- Aplicacions de franc però amb publicitat. En aquest cas, ja depèn de la publicitat però al principi no s'aconsegueix molt. Tot i així, quan més popular sigui i més descàrregues hi hagi, anirà en augment.

Especificat això, passem a fer l'estudi de mercat de diferents aplicacions.

Aquí troben les aplicacions més conegudes sobre la matèria:

- **Kanji Tutor.** Preu: 2'17€. +1000 descàrregues (desde 22/11/10).
  Partint a la base que hi ha 1000 descàrregues com a mínim, ho calculem amb aquest número. El 70% de 2'17€ = 1'519€. Per tant, 1'519€ X 1000 descàrregues = 1.519 € guanyats com a mínim.

- **Survive! Japanese.** Preu: 1'94€. +1000 descàrregues (Desde 31/3/13).
  Partint a la base que hi ha 1000 descàrregues com a mínim, ho calculem amb aquest número. El 70% de 1'94€ = 1'358€. Per tant, 1'358€ X 1000 descàrregues = 1.358 € guanyats com a mínim. Recordem que he posat el mínim de descàrregues ja que no ens donen les xifres exactes de descàrregues. Tenim en compte que en 1 any aproximadament s'ha guanyat quasi la meitat del preu de la meva aplicació.

- **Japanese Kanji.** Preu: 5'63 €. +100 descàrregues (desde 27/3/12)
  Partint a la base que hi ha 100 descàrregues com a mínim, ho calculem amb aquest número. El 70% de 5'63€ = 3'941€. Per tant, 3'941€ X 100 descàrregues = 394'1€ guanyats com a mínim.

- **Japanese 3 JLPT N3.** Preu: 1'93€. +100 descàrregues (desde 3/12/13).
  Partint a la base que hi ha 100 descàrregues com a mínim, ho calculem amb aquest número. El 70% de 1'93€ = 1'351€. Per tant, 1'351€ X 100 descàrregues = 135'1€ guanyats com a mínim.

- **1000 Kanji Word Quiz.** Preu: 0'99€. +100 descàrregues (desde 20/1/10).
  Partint a la base que hi ha 100 descàrregues com a mínim, ho calculem amb aquest número. El 70% de 0'99€ = 0'693€. Per tant, 0'693€ X 100 descàrregues = 69'3€ guanyats com a mínim.

- **Learn Japanese.** Preu: 0'73 €. +100 descàrregues (desde 15/3/13)
  Partint a la base que hi ha 100 descàrregues com a mínim, ho calculem amb aquest número. El 70% de 0'73€ = 0'551€. Per tant, 0'551€ X 100 descàrregues = 55'1€ guanyats com a mínim.

- **Japanese Vocabulary Quiz.** Preu: 0'70€. +50 descàrregues. Desde 6/11/10
  Partint a la base que hi ha 100 descàrregues com a mínim, ho calculem amb aquest número. El 70% de 0'70€ = 0'49€. Per tant, 0'49€ X 50 descàrregues = 29'5€ guanyats com a mínim.

La conclusió que podem arribar és que amb el temps podrem aconseguir la inversió. És una cosa que va més bé per sort i que agradi l'aplicació.

També cal tenir en compte que el sector de l'aprenentage dels idiomes, concretament del japonés, no és sobre el que hi ha més demanda al mercat. Però podem veure, que una aplicació que no sigui líder en el mercat, pot aconseguir recuperar la inversió i fins i tot tenir beneficis.

6. CONCLUSIONS

En aquest capítol s'explica totes les conclusions finals d'aquest projecte a més de les perspectives que podria tenir aquest projecte.

6. Conclusions

L'objectiu principal del projecte s'ha pogut realitzar ja que hem pogut analitzar bé el AppInventor i avaluar el seu funcionament. S'ha creat un petit manual de funcionament i, a més, a través d'aquest projecte s'ha pogut crear una aplicació Android sense haver tingut previs coneixements sobre el tema i guiant-nos a través del manual AppInventor creat i els coneixements bàsics de programació.

Amb aquest projecte, he volgut ensenyar una de les moltes i diferents eines que hi ha actualment a l'hora de crear i dissenyar aplicacions pels dispositius Android. Una eina que, sens dubte, facilita molt a l'hora de crear per la seva interfície intuïtiva, i que necessita pocs coneixements de programació, cosa que s'agraeix molt avui en dia.

L'aplicació creada és només un exemple per veure les capacitats de AppInventor, que això no vol dir que només es puguin fer coses d'aquest estil. Gràcies a la gran varietat de blocs que hi ha, es poden fer coses molt diferents. L'elecció d'aquest tipus d'aplicació va ser perquè avui en dia gràcies a les capacitats que donen els dispositius Smartphone o Tablets gran part de la població utilitza aplicacions per estudiar, poder repassar temaris o aprendre coses d'un tema concret, entre altres coses. Per això, vaig decidir fer una aplicació sobre una llengua estrangera ja que es necessita molt de temps fins a arribar a un nivell alt d'aquesta, i quina millor manera que fer-ho d'una manera més entretinguda per a l'usuari. A més, s'està apostant molt avui en dia per aplicacions sobre l'estudi de qualsevol camp concret.

Sobre les dificultats trobades a l'hora de realitzar el projecte, el més complicat que m'he trobat ha sigut el fet d'aprendre sobre un programa que només tenia una base molt bàsica, ja que a la rama principal dels meus estudis el nivell de programació no era suficient per després poder realitzar aplicacions com aquesta.

Però també, he d'agraïr l'esforç invertit en aprendre-ho ja que he aprés molt sobre aquest tipus de tecnologia, que avui en dia és un punt molt important i que està en ple auge, i tinc les idees més clares a l'hora de programar.

Per últim, també seria desitjar que aquest projecte sigui útil per a aquelles persones que tinguin dubtes o no sàpiguen molt sobre el tema Android i vulguin iniciar-se, ja perquè no hagin tingut l'oportunitat, no tenen prous coneixements o pel seu propi interès.

6.1. Perspectives

Si hagués de continuar l'aplicació, afegiria més paraules a la base de dades a més de poder fer lliçons senceres de tota la gramàtica bàsica i avançada. Pel que fa el tema programació, seria molt similar ficar tot aquest tipus d'informació. Si volgués afegir alguna cosa totalment nova, seria sobre tocar algun aspecte sobre la cultura del país, ja que una llengua no és només estudiar-la i traduir-la al castellà/català, també s'ha de

comprendre aquelles paraules que culturalment tenen un significat i que, per temes culturals, només es poden entendre si saps sobre la cultura del país.

Però no només de gramàtica com si fos un llibre qualsevol, seria també gramàtica i expressions que s'utilitzin avui en dia per les mateixes persones natives. Perquè ni el japonès ni cap llengua estrangera només és aprendre paraules (o caràcters en els casos de les llengües que no tinguin el nostre alfabet), també és aprendre la gramàtica i saber expressions utilitzades al dia a dia. Això últim és molt remarcable ja que a l'hora d'aprendre llengües a través de llibres o acadèmies està molt enfocat a nivell acadèmic i de vegades no és necessari saber-ho tant acadèmic, és suficient per la vida diària.

7. BIBLIOGRAFIA

En aquest capítol veurem tota la Bibliografia utilitzada per a la realització d'aquest projecte.

7. Bibliografia

[1] Pàgina d'accés a AppInventor. Consulta: 03/09/13

Disponible a: http://beta.appinventor.mit.edu/

[2] Pàgina Web oficial de AppInventor. Consulta: 03/09/13

Disponible a: http://appinventor.mit.edu/explore/

[3] Tutorial "Com col·locar un nombre random". Consulta: 25/10/13

Disponible a: http://www.appinventorblocks.com/appinventor-tutorials-tips/appinventor-randomImagePicker

[4] Tutorial "Passar d'una pantalla a una altre". Consulta: 17/11/13

Disponible a: http://blogromerojacfsmr.blogspot.com.es/2013/04/crear-una-aplicacion-con-dos-screen.html

[5] Tutorial "Fer un qüestionari". Consulta: 17/11/13

Disponible a: http://s445852199.mialojamiento.es/peponees/appinventorspanish/learn/tutorials/quizme/quizme.html

[6] AppInventor for Android. Autor: Jason Tyler. Editorial: WILEY. Primera edició 2011. Estats Units d'Amèrica. Codi ISBN: 978-1-119-99133-5.

[7] AppInventor. Create Your Own Android Apps. Autors: David Wolber, Hal Abelson, Ellen Spertus i Liz Looney,. Editorial: O'REILLY. Primera edició Abril 2011. Canadà. Codi ISBN: 978-1-449-39748-7.

[8] Informació sobre el programa SDK Android. Consulta: 15/12/13

Disponible a: http://developer.android.com/sdk/index.html

[9] Informació sobre el programa Titanium Studio. Consulta: 15/12/13

Disponible a: http://www.appcelerator.com/titanium/titanium-studio/

[10] Informació sobre el programa Mobincube. Consulta: 15/12/13

Disponible a: http://www.mobincube.com/es/

[11] Informació sobre el programa Nimbo. Consulta: 15/12/13

Disponible a: http://www.nimbosolutions.com/

[12] Botiga oficial Google Play. Consulta: 15/12/13

Disponible a: https://play.google.com/store/apps

8. ANNEXOS

Per últim, conclourem el projecte amb l'últim capítol, on podem mirar la base de dades i programació utilitzat i realitzat pel projecte.

8. Annexos

En aquest apartat, es publicarà tots els caràcters utilitzats per la creació de la base de dades de l'aplicació i programació completa de l'aplicació.

Base de dades:

水	みず	**Aigua**
黄色	きいろ	**Groc**
前	まえ	**Anterior**
年	ねん	**Any**
木	き	**Arbre**
米	こめ	**Arrós**
青い	あおい	**Blau**
安い	やすい	**Barato**
飲む	のむ	**Beure**
白い	しろい	**Blanc**
森	もり	**Bosc**
馬	うま	**Cavall**

肉	にく	**Carn**
高い	たかい	**Car**
近い	ちかい	**A prop**
百	ひゃく	**Cent**
五	ご	**Cinc**
車	くるま	**Cotxe**
短い	みじかい	**Curt**
物	もの	**Cosa**
四	よん	**Quatre**
右	みぎ	**Dreta**
休み	やすみ	**Descansar**
後	あと	**Després**
万	まん	**Deu mil**
十	じゅう	**Deu**
二	に	**Dos**

中	なか	**Meitat**
間	あいだ	**En mig**
学校	がっこう	**Escola**
学生	がくせい	**Estudiant**
火	ひ	**Foc**
猫	ねこ	**Gat**
好き	すき	**Agradar**
男	おとこ	**Home**
時	じ	**Lletra**
左	ひだり	**Esquerra**
漢字	かんじ	**Caràcter japonès**
長い	ながい	**Llarg**
読む	よむ	**Llegir**
遠い	とおい	**Lluny**
遅い	おそい	**Lent**

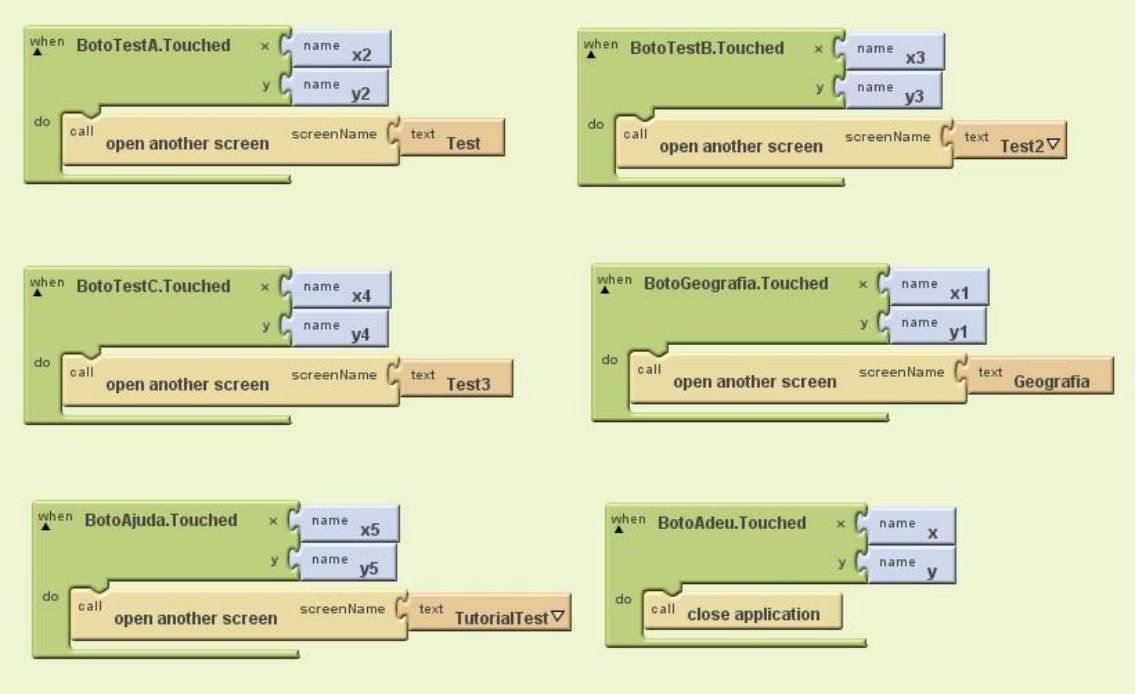
字	じ	**Lletra**
本	ほん	**Llibre**
月	つき	**Mes**
千	せん	**Mil**
分	ぶん	**Minut**
半	はん	**Mig**
猿	さる	**Mico**
山	やま	**Muntanya**
多い	おおい	**Molt**
女	おんな	**Dona**
黒い	くろい	**Negre**
子供	こども	**Nen**
九	きゅう	**Nou**
八	はち	**Vuit**
金	かね	**Or**

鳥	とり	**Ocell**
犬	いぬ	**Gos**
人	ひと	**Persona**
魚	さかな	**Peix**
先生	せんせい	**Professor**
速い	はやい	**Ràpid**
川	かわ	**Riu**
赤い	あかい	**Vermell**
六	ろく	**Sis**
七	なな	**Set**
日	ひ	**Dia**
茶	ちゃ	**Té**
時間	じかん	**Temps**
土	ち	**Terra**
働く	はたらく	**Treballar**

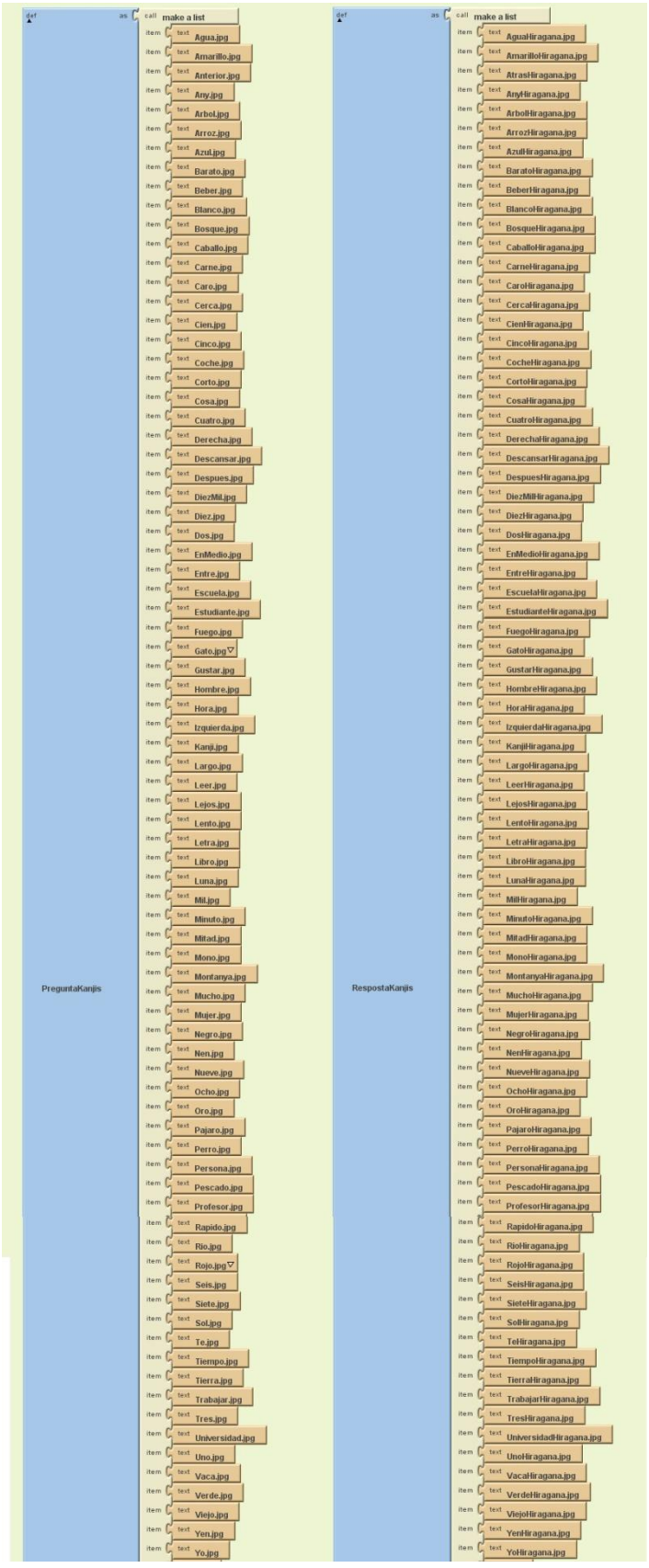
三	さん	**Tres**
大学	だいがく	**Universitat**
一	いち	**Un**
牛	うし	**Vaca**
緑	みどり	**Verd**
古い	ふるい	**Vell**
円	えん	**Yen**
私	わたし	**Jo**

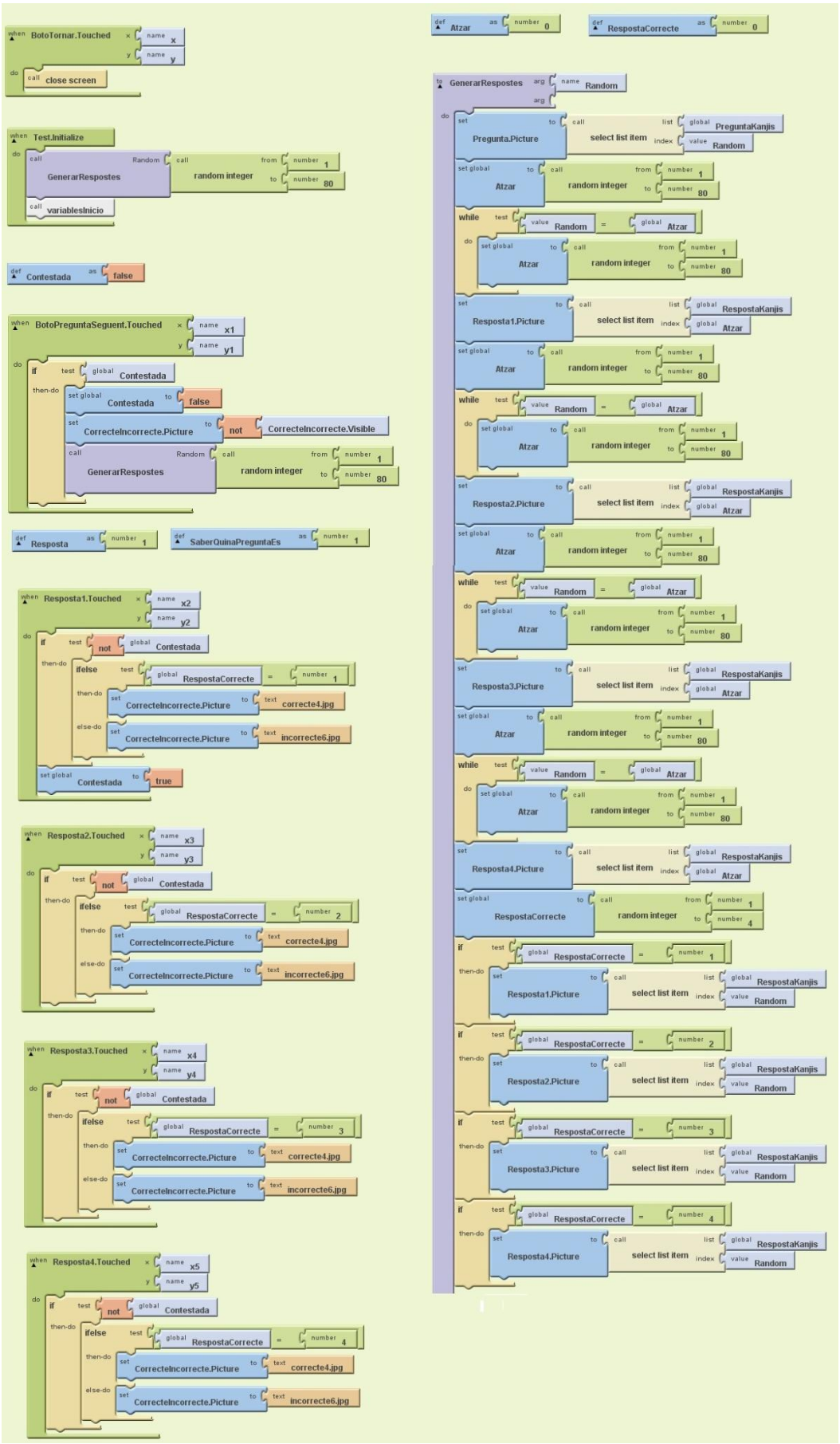
Programa:

Screen Screen1 (Pantalla principal):

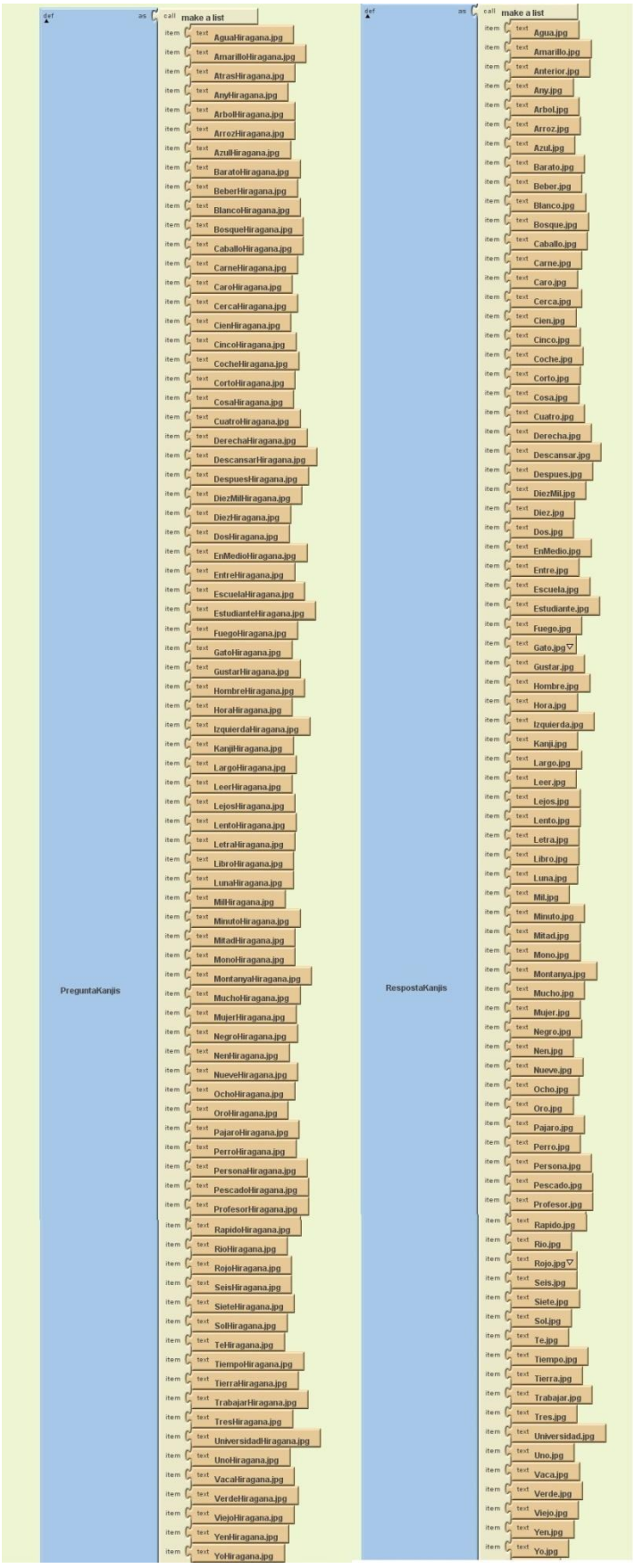


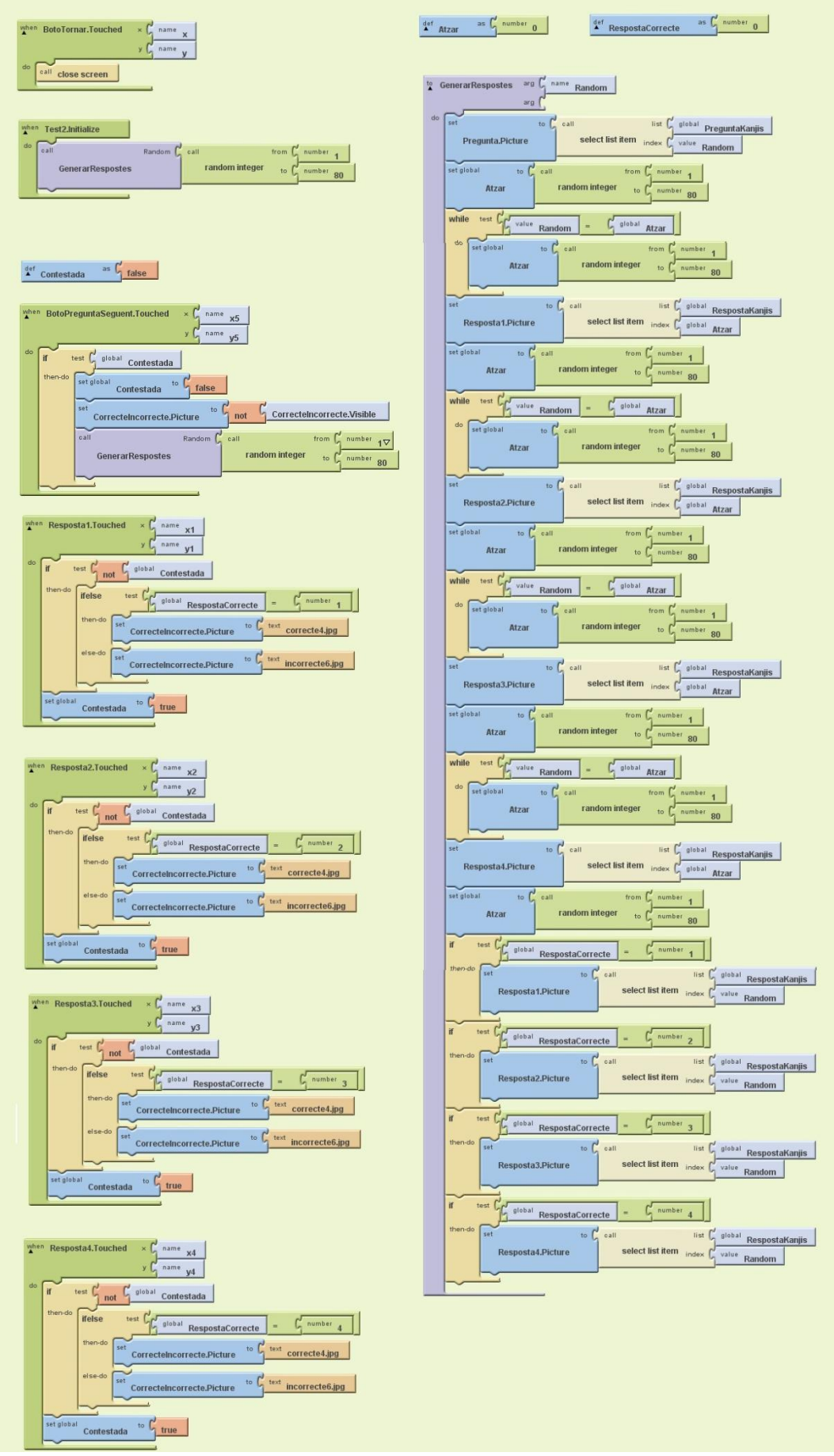
Screen Test (Test A):



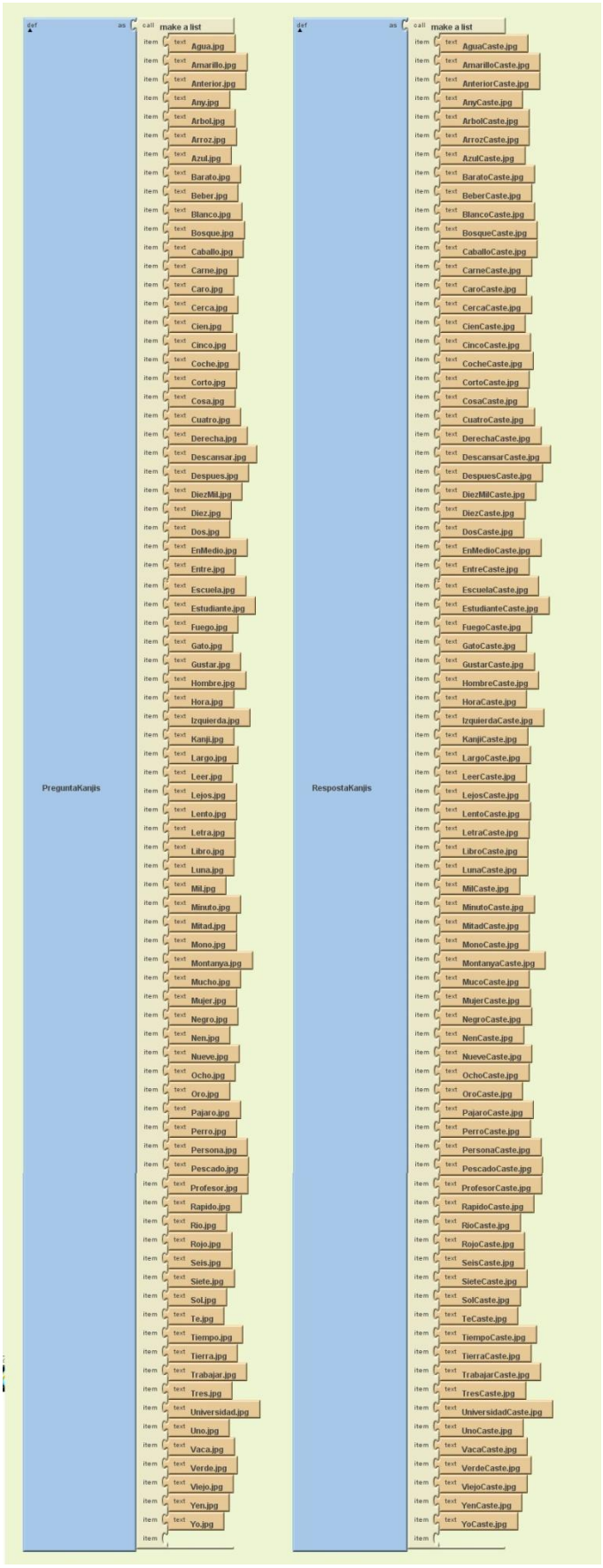

when BotoTornar.Touched x y do call close screen
when Test.Initialize do call GenerarRespostes Random call random integer from 1 to 80 call variablesInicio
def Contestada as false
when BotoPreguntaSeguent.Touched x y do if test global Contestada then-do set global Contestada to false set CorrecteIncorrecte.Picture to not CorrecteIncorrecte.Visible call GenerarRespostes Random call random integer from 1 to 80
def Resposta as 1
def SaberQuinaPreguntaEs as 1
when Resposta1.Touched x y do if test not global Contestada then-do ifelse test global RespostaCorrecte = 1 then-do set CorrecteIncorrecte.Picture to correcte4.jpg else-do set CorrecteIncorrecte.Picture to incorrecte6.jpg set global Contestada to true
when Resposta2.Touched x y do if test not global Contestada then-do ifelse test global RespostaCorrecte = 2 then-do set CorrecteIncorrecte.Picture to correcte4.jpg else-do set CorrecteIncorrecte.Picture to incorrecte6.jpg
when Resposta3.Touched x y do if test not global Contestada then-do ifelse test global RespostaCorrecte = 3 then-do set CorrecteIncorrecte.Picture to correcte4.jpg else-do set CorrecteIncorrecte.Picture to incorrecte6.jpg
when Resposta4.Touched x y do if test not global Contestada then-do ifelse test global RespostaCorrecte = 4 then-do set CorrecteIncorrecte.Picture to correcte4.jpg else-do set CorrecteIncorrecte.Picture to incorrecte6.jpg
def Atzar as 0
def RespostaCorrecte as 0
to GenerarRespostes arg Random do set Pregunta.Picture to select list item list global PreguntaKanjis index value Random set global Atzar to random integer from 1 to 80 while test value Random = global Atzar do set global Atzar to random integer from 1 to 80 set Resposta1.Picture to select list item list global RespostaKanjis index global Atzar set global Atzar to random integer from 1 to 80 while test value Random = global Atzar do set global Atzar to random integer from 1 to 80 set Resposta2.Picture to select list item list global RespostaKanjis index global Atzar set global Atzar to random integer from 1 to 80 while test value Random = global Atzar do set global Atzar to random integer from 1 to 80 set Resposta3.Picture to select list item list global RespostaKanjis index global Atzar set global Atzar to random integer from 1 to 80 while test value Random = global Atzar do set global Atzar to random integer from 1 to 80 set Resposta4.Picture to select list item list global RespostaKanjis index global Atzar set global RespostaCorrecte to random integer from 1 to 4 if test global RespostaCorrecte = 1 then-do set Resposta1.Picture to select list item list global RespostaKanjis index value Random if test global RespostaCorrecte = 2 then-do set Resposta2.Picture to select list item list global RespostaKanjis index value Random if test global RespostaCorrecte = 3 then-do set Resposta3.Picture to select list item list global RespostaKanjis index value Random if test global RespostaCorrecte = 4 then-do set Resposta4.Picture to select list item list global RespostaKanjis index value Random

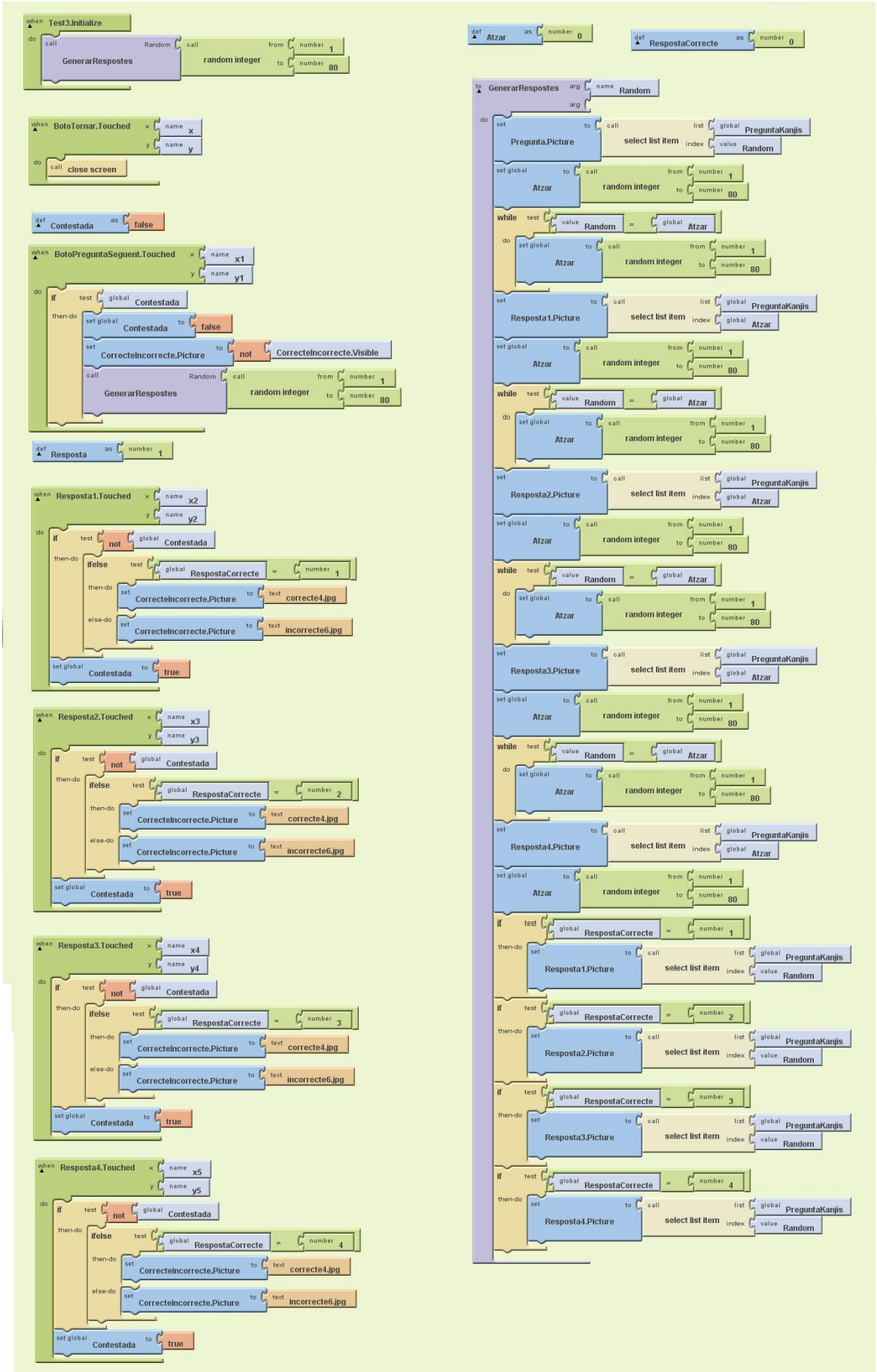
Scrren Test2 (Test B):



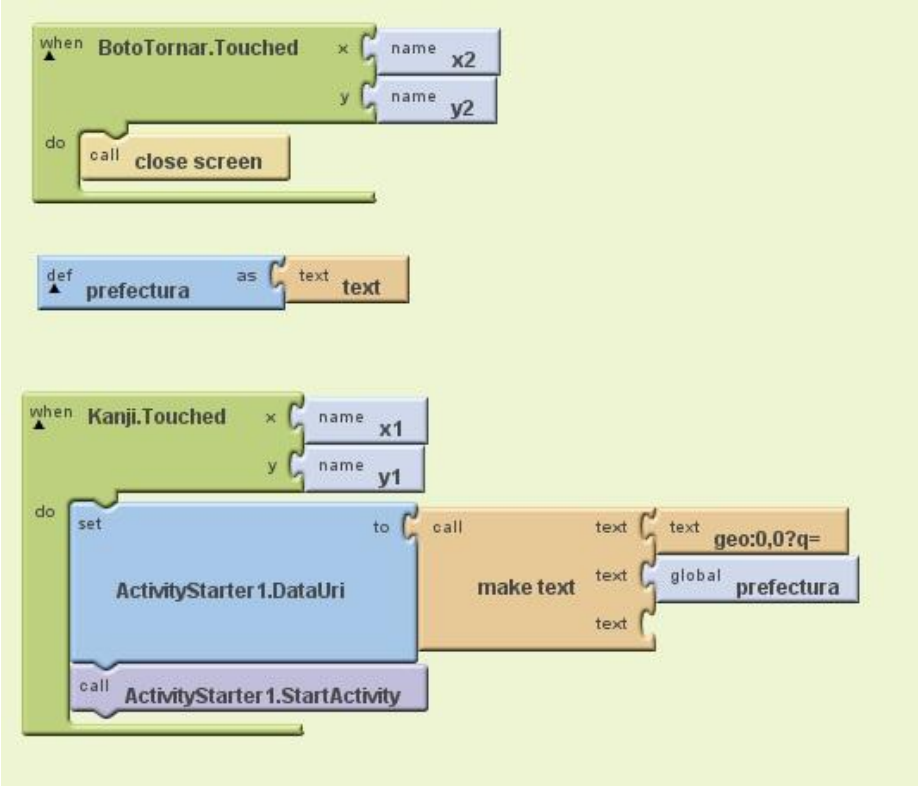

when BotoTornar.Touched x y do call close screen
when Test2.Initialize do call GenerarRespostes Random call random integer from 1 to 80
def Contestada as false
when BotoPreguntaSeguent.Touched x y do if test global Contestada then-do set global Contestada to false set CorrecteIncorrecte.Picture to not CorrecteIncorrecte.Visible call GenerarRespostes Random call random integer from 1 to 80
when Resposta1.Touched x y do if test not global Contestada then-do ifelse test global RespostaCorrecte = number 1 then-do set CorrecteIncorrecte.Picture to correcte4.jpg else-do set CorrecteIncorrecte.Picture to incorrecte6.jpg set global Contestada to true
when Resposta2.Touched x y do if test not global Contestada then-do ifelse test global RespostaCorrecte = number 2 then-do set CorrecteIncorrecte.Picture to correcte4.jpg else-do set CorrecteIncorrecte.Picture to incorrecte6.jpg set global Contestada to true
when Resposta3.Touched x y do if test not global Contestada then-do ifelse test global RespostaCorrecte = number 3 then-do set CorrecteIncorrecte.Picture to correcte4.jpg else-do set CorrecteIncorrecte.Picture to incorrecte6.jpg set global Contestada to true
when Resposta4.Touched x y do if test not global Contestada then-do ifelse test global RespostaCorrecte = number 4 then-do set CorrecteIncorrecte.Picture to correcte4.jpg else-do set CorrecteIncorrecte.Picture to incorrecte6.jpg set global Contestada to true
def Atzar as number 0
def RespostaCorrecte as number 0
to GenerarRespostes arg Random do set Pregunta.Picture to select list item list global PreguntaKanjis index value Random
set global Atzar to random integer from 1 to 80
while test value Random = global Atzar do set global Atzar to random integer from 1 to 80
set Resposta1.Picture to select list item list global RespostaKanjis index global Atzar
set global Atzar to random integer from 1 to 80
while test value Random = global Atzar do set global Atzar to random integer from 1 to 80
set Resposta2.Picture to select list item list global RespostaKanjis index global Atzar
set global Atzar to random integer from 1 to 80
while test value Random = global Atzar do set global Atzar to random integer from 1 to 80
set Resposta3.Picture to select list item list global RespostaKanjis index global Atzar
set global Atzar to random integer from 1 to 80
while test value Random = global Atzar do set global Atzar to random integer from 1 to 80
set Resposta4.Picture to select list item list global RespostaKanjis index global Atzar
set global Atzar to random integer from 1 to 80
if test global RespostaCorrecte = number 1 then-do set Resposta1.Picture to select list item list global RespostaKanjis index value Random
if test global RespostaCorrecte = number 2 then-do set Resposta2.Picture to select list item list global RespostaKanjis index value Random
if test global RespostaCorrecte = number 3 then-do set Resposta3.Picture to select list item list global RespostaKanjis index value Random
if test global RespostaCorrecte = number 4 then-do set Resposta4.Picture to select list item list global RespostaKanjis index value Random

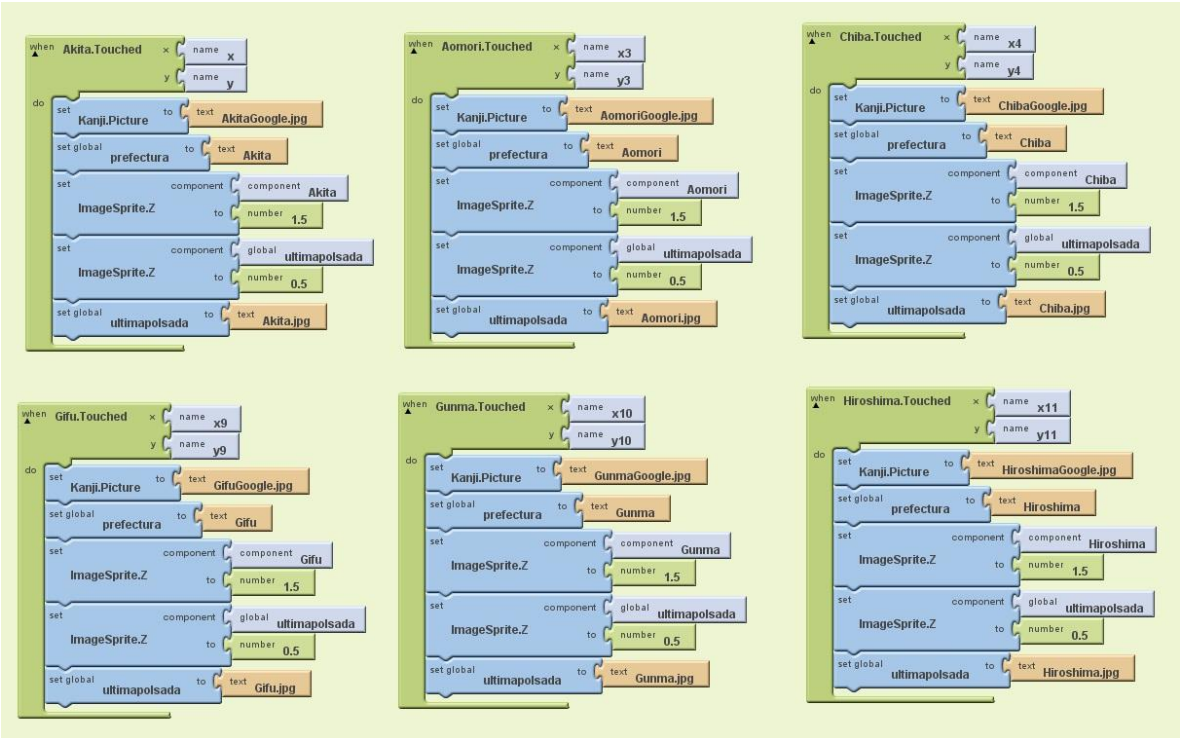
Screen Test3 (Test C):

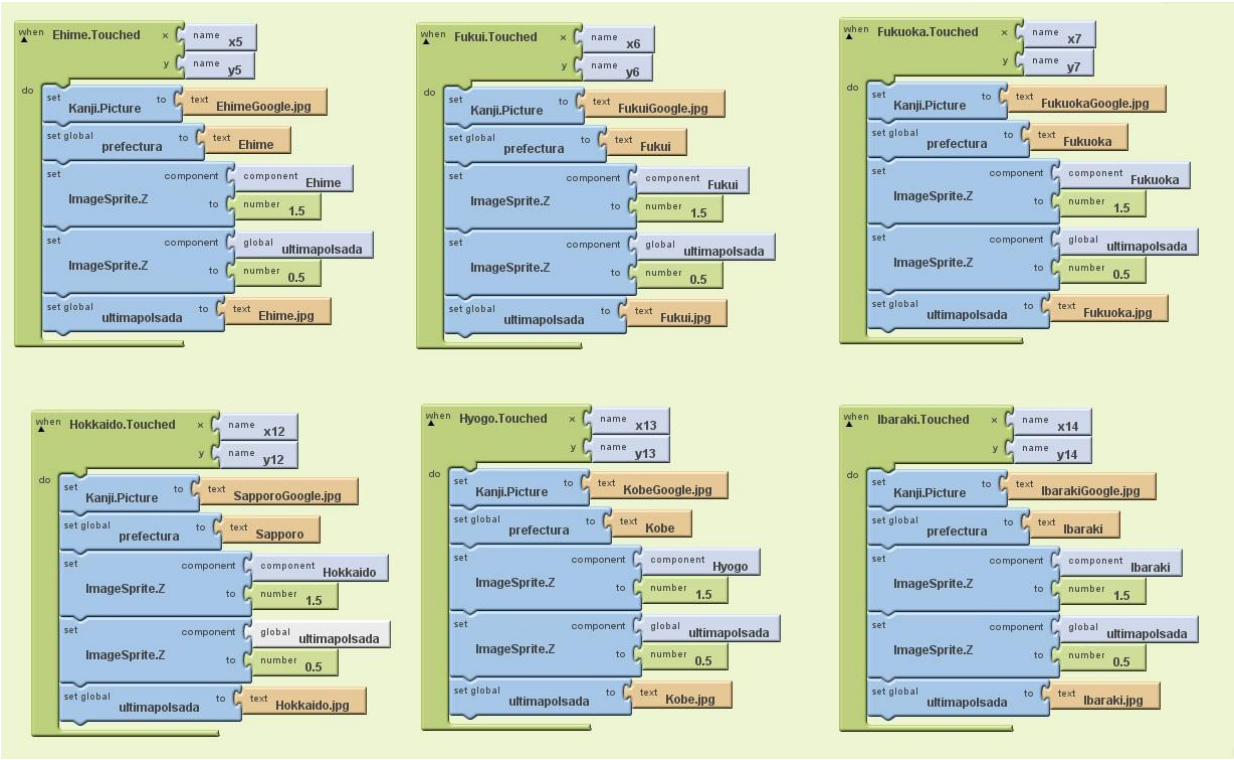


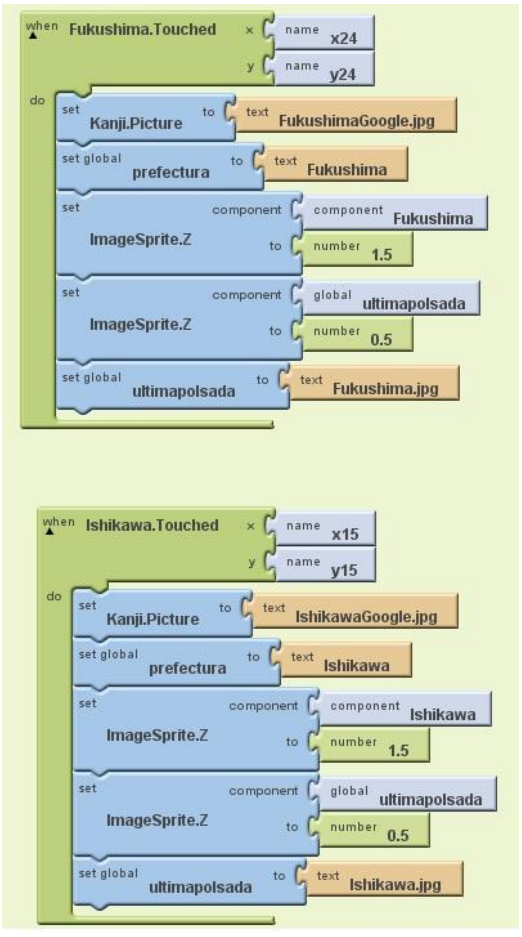

when Test3.Initialize
do call GenerarRespostes Random call random integer from number 1 to number 80
when BotoTornar.Touched x name x y name y
do call close screen
def Contestada as false
when BotoPreguntaSeguent.Touched x name x1 y name y1
do if test global Contestada then-do set global Contestada to false
set CorrecteIncorrecte.Picture to not CorrecteIncorrecte.Visible
call GenerarRespostes Random call random integer from number 1 to number 80
def Resposta as number 1
when Resposta1.Touched x name x2 y name y2
do if test not global Contestada then-do ifelse test global RespostaCorrecte = number 1
then-do set CorrecteIncorrecte.Picture to text correcte4.jpg
else-do set CorrecteIncorrecte.Picture to text incorrecte6.jpg
set global Contestada to true
when Resposta2.Touched x name x3 y name y3
do if test not global Contestada then-do ifelse test global RespostaCorrecte = number 2
then-do set CorrecteIncorrecte.Picture to text correcte4.jpg
else-do set CorrecteIncorrecte.Picture to text incorrecte6.jpg
set global Contestada to true
when Resposta3.Touched x name x4 y name y4
do if test not global Contestada then-do ifelse test global RespostaCorrecte = number 3
then-do set CorrecteIncorrecte.Picture to text correcte4.jpg
else-do set CorrecteIncorrecte.Picture to text incorrecte6.jpg
set global Contestada to true
when Resposta4.Touched x name x5 y name y5
do if test not global Contestada then-do ifelse test global RespostaCorrecte = number 4
then-do set CorrecteIncorrecte.Picture to text correcte4.jpg
else-do set CorrecteIncorrecte.Picture to text incorrecte6.jpg
set global Contestada to true
def Atzar as number 0
def RespostaCorrecte as number 0
to GenerarRespostes arg name Random
do set Pregunta.Picture to call select list item list global PreguntaKanjis index value Random
set global Atzar to call random integer from number 1 to number 80
while test value Random = global Atzar
do set global Atzar to call random integer from number 1 to number 80
set Resposta1.Picture to call select list item list global PreguntaKanjis index global Atzar
set global Atzar to call random integer from number 1 to number 80
while test value Random = global Atzar
do set global Atzar to call random integer from number 1 to number 80
set Resposta2.Picture to call select list item list global PreguntaKanjis index global Atzar
set global Atzar to call random integer from number 1 to number 80
while test value Random = global Atzar
do set global Atzar to call random integer from number 1 to number 80
set Resposta3.Picture to call select list item list global PreguntaKanjis index global Atzar
set global Atzar to call random integer from number 1 to number 80
while test value Random = global Atzar
do set global Atzar to call random integer from number 1 to number 80
set Resposta4.Picture to call select list item list global PreguntaKanjis index global Atzar
set global Atzar to call random integer from number 1 to number 80
if test global RespostaCorrecte = number 1 then-do set Resposta1.Picture to call select list item list global PreguntaKanjis index value Random
if test global RespostaCorrecte = number 2 then-do set Resposta2.Picture to call select list item list global PreguntaKanjis index value Random
if test global RespostaCorrecte = number 3 then-do set Resposta3.Picture to call select list item list global PreguntaKanjis index value Random
if test global RespostaCorrecte = number 4 then-do set Resposta4.Picture to call select list item list global PreguntaKanjis index value Random

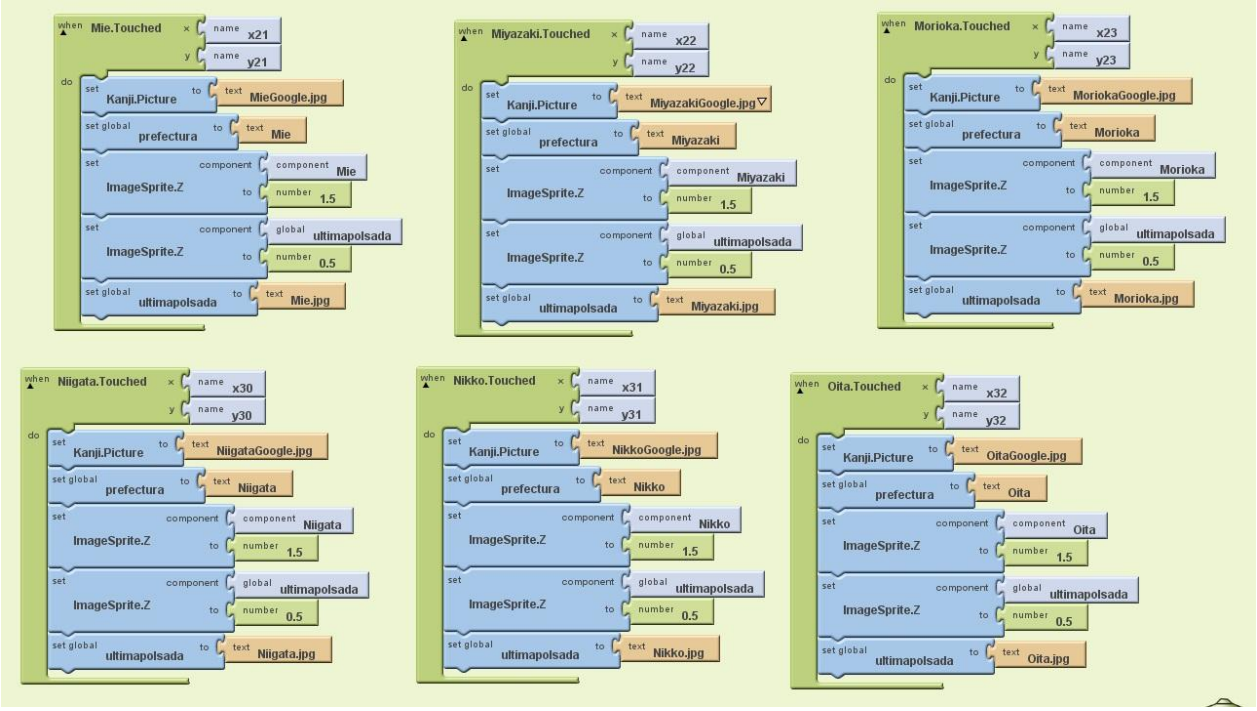
Screen Geografia:

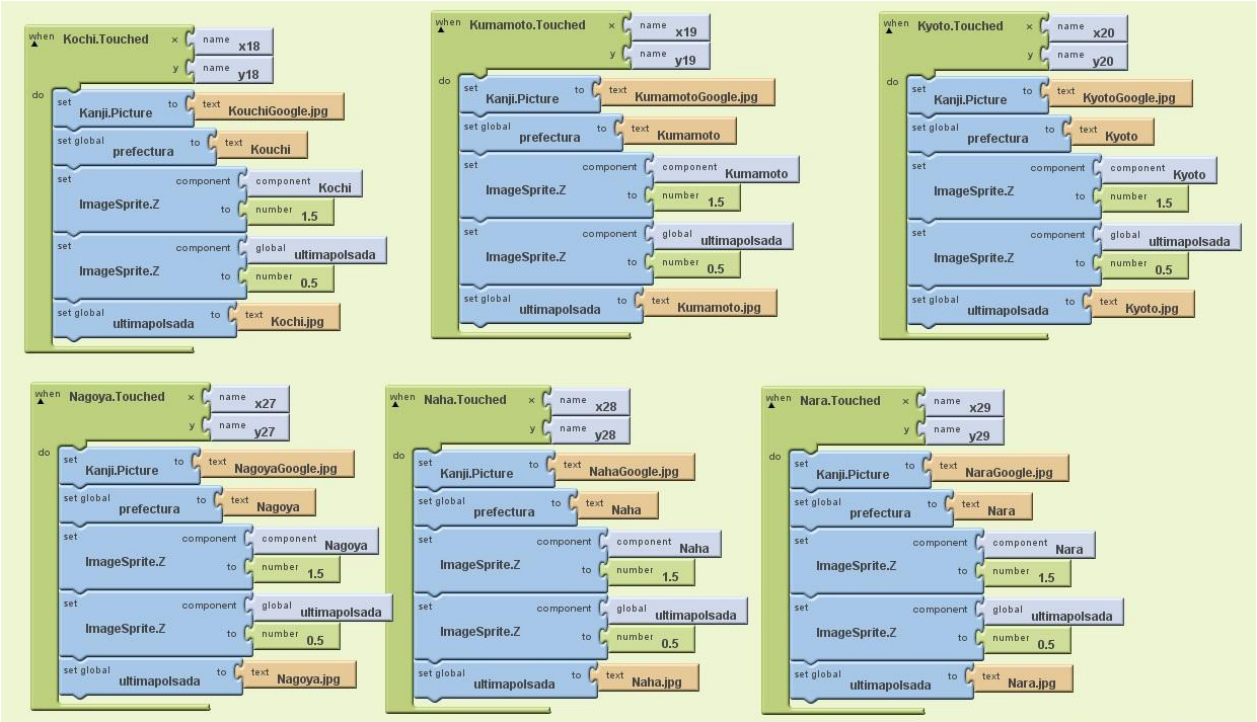


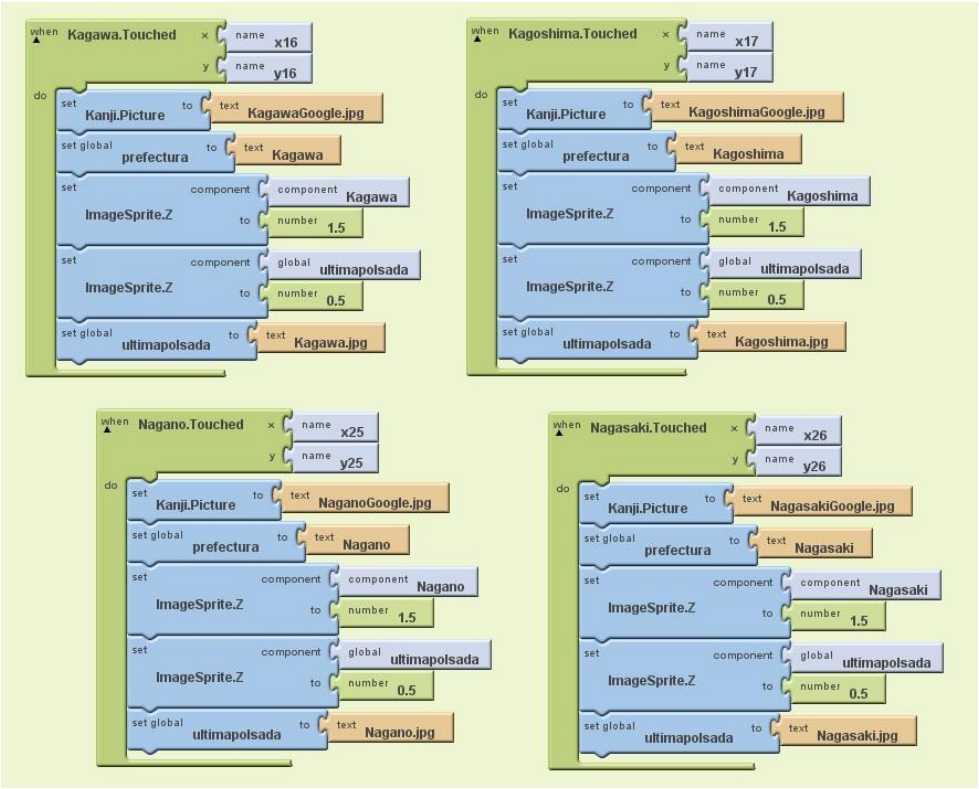


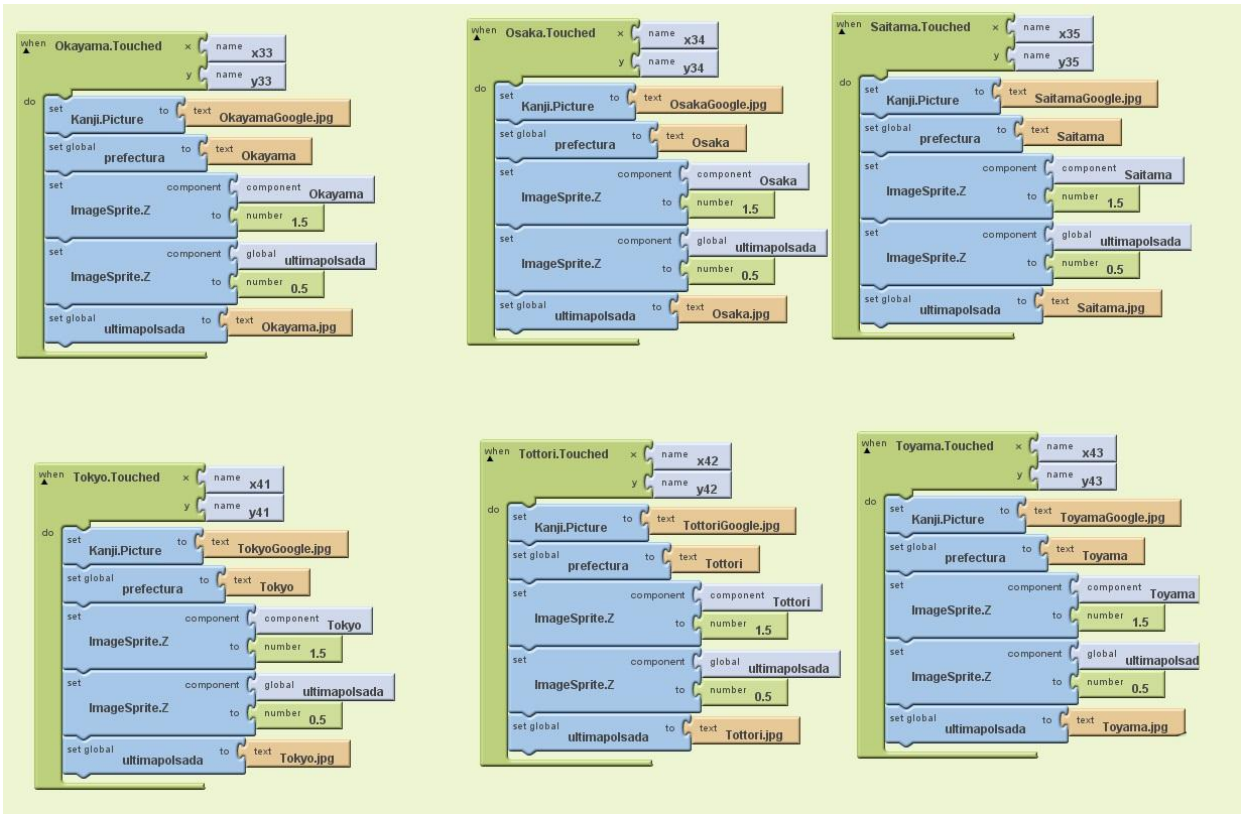

when Ehime.Touched x y
name x5
name y5
do
set Kanji.Picture to text EhimeGoogle.jpg
set global prefectura to text Ehime
set ImageSprite.Z component component Ehime to number 1.5
set ImageSprite.Z component global ultimapolsada to number 0.5
set global ultimapolsada to text Ehime.jpg
when Fukui.Touched x y
name x6
name y6
do
set Kanji.Picture to text FukuiGoogle.jpg
set global prefectura to text Fukui
set ImageSprite.Z component component Fukui to number 1.5
set ImageSprite.Z component global ultimapolsada to number 0.5
set global ultimapolsada to text Fukui.jpg
when Fukuoka.Touched x y
name x7
name y7
do
set Kanji.Picture to text FukuokaGoogle.jpg
set global prefectura to text Fukuoka
set ImageSprite.Z component component Fukuoka to number 1.5
set ImageSprite.Z component global ultimapolsada to number 0.5
set global ultimapolsada to text Fukuoka.jpg
when Hokkaido.Touched x y
name x12
name y12
do
set Kanji.Picture to text SapporoGoogle.jpg
set global prefectura to text Sapporo
set ImageSprite.Z component component Hokkaido to number 1.5
set ImageSprite.Z component global ultimapolsada to number 0.5
set global ultimapolsada to text Hokkaido.jpg
when Hyogo.Touched x y
name x13
name y13
do
set Kanji.Picture to text KobeGoogle.jpg
set global prefectura to text Kobe
set ImageSprite.Z component component Hyogo to number 1.5
set ImageSprite.Z component global ultimapolsada to number 0.5
set global ultimapolsada to text Kobe.jpg
when Ibaraki.Touched x y
name x14
name y14
do
set Kanji.Picture to text IbarakiGoogle.jpg
set global prefectura to text Ibaraki
set ImageSprite.Z component component Ibaraki to number 1.5
set ImageSprite.Z component global ultimapolsada to number 0.5
set global ultimapolsada to text Ibaraki.jpg

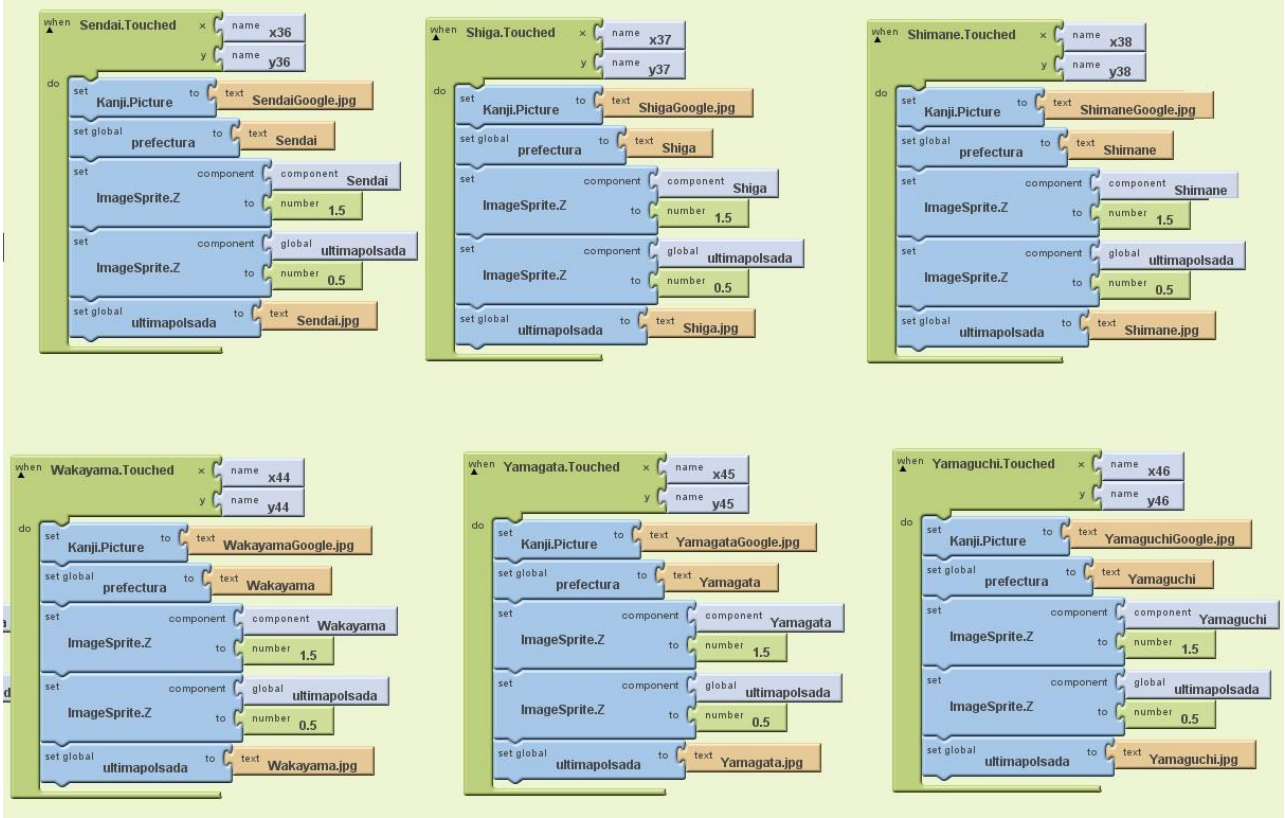

when Fukushima.Touched x y
name x24
name y24
do
set Kanji.Picture to text FukushimaGoogle.jpg
set global prefectura to text Fukushima
set ImageSprite.Z component component Fukushima to number 1.5
set ImageSprite.Z component global ultimapolsada to number 0.5
set global ultimapolsada to text Fukushima.jpg
when Ishikawa.Touched x y
name x15
name y15
do
set Kanji.Picture to text IshikawaGoogle.jpg
set global prefectura to text Ishikawa
set ImageSprite.Z component component Ishikawa to number 1.5
set ImageSprite.Z component global ultimapolsada to number 0.5
set global ultimapolsada to text Ishikawa.jpg

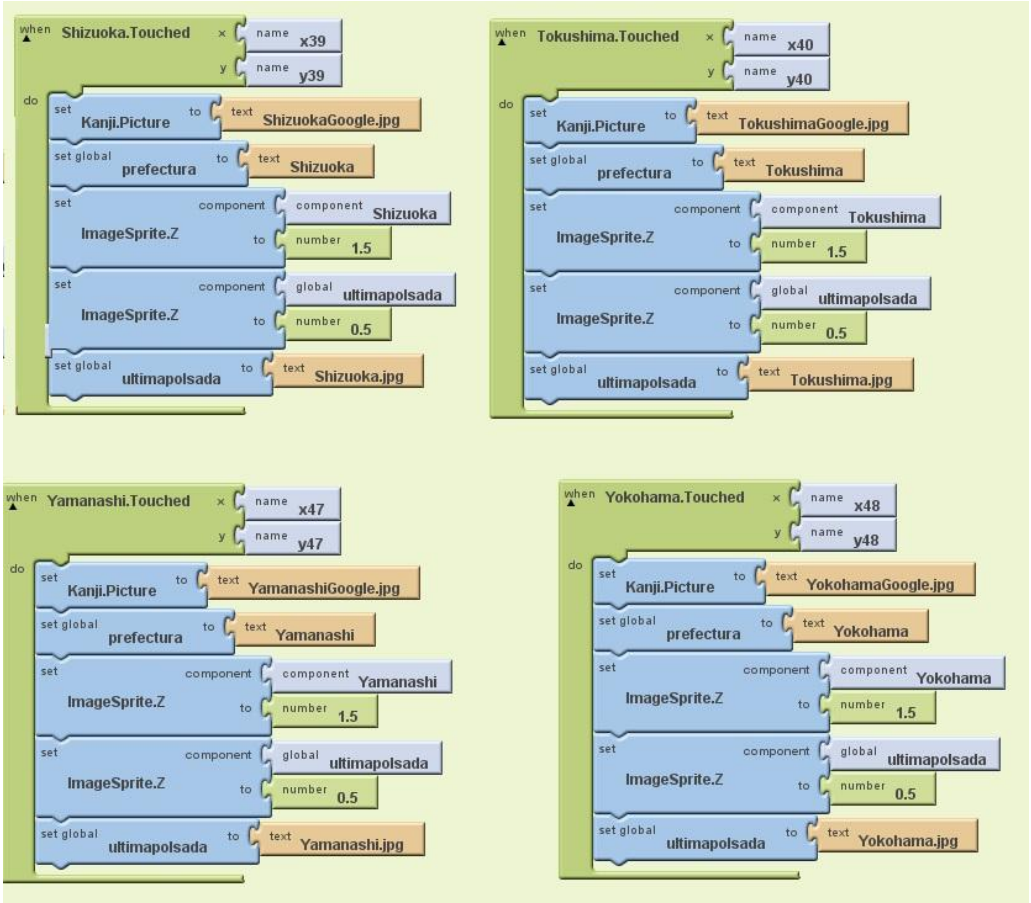

when Mie.Touched x y
name x21
name y21
do
set Kanji.Picture to text MieGoogle.jpg
set global prefectura to text Mie
set component ImageSprite.Z to component Mie number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Mie.jpg
when Miyazaki.Touched x y
name x22
name y22
do
set Kanji.Picture to text MiyazakiGoogle.jpg
set global prefectura to text Miyazaki
set component ImageSprite.Z to component Miyazaki number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Miyazaki.jpg
when Morioka.Touched x y
name x23
name y23
do
set Kanji.Picture to text MoriokaGoogle.jpg
set global prefectura to text Morioka
set component ImageSprite.Z to component Morioka number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Morioka.jpg
when Niigata.Touched x y
name x30
name y30
do
set Kanji.Picture to text NiigataGoogle.jpg
set global prefectura to text Niigata
set component ImageSprite.Z to component Niigata number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Niigata.jpg
when Nikko.Touched x y
name x31
name y31
do
set Kanji.Picture to text NikkoGoogle.jpg
set global prefectura to text Nikko
set component ImageSprite.Z to component Nikko number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Nikko.jpg
when Oita.Touched x y
name x32
name y32
do
set Kanji.Picture to text OitaGoogle.jpg
set global prefectura to text Oita
set component ImageSprite.Z to component Oita number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Oita.jpg


when Kochi.Touched x y
name x18
name y18
do
set Kanji.Picture to text KouchiGoogle.jpg
set global prefectura to text Kouchi
set component ImageSprite.Z to component Kochi number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Kochi.jpg
when Kumamoto.Touched x y
name x19
name y19
do
set Kanji.Picture to text KumamotoGoogle.jpg
set global prefectura to text Kumamoto
set component ImageSprite.Z to component Kumamoto number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Kumamoto.jpg
when Kyoto.Touched x y
name x20
name y20
do
set Kanji.Picture to text KyotoGoogle.jpg
set global prefectura to text Kyoto
set component ImageSprite.Z to component Kyoto number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Kyoto.jpg
when Nagoya.Touched x y
name x27
name y27
do
set Kanji.Picture to text NagoyaGoogle.jpg
set global prefectura to text Nagoya
set component ImageSprite.Z to component Nagoya number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Nagoya.jpg
when Naha.Touched x y
name x28
name y28
do
set Kanji.Picture to text NahaGoogle.jpg
set global prefectura to text Naha
set component ImageSprite.Z to component Naha number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Naha.jpg
when Nara.Touched x y
name x29
name y29
do
set Kanji.Picture to text NaraGoogle.jpg
set global prefectura to text Nara
set component ImageSprite.Z to component Nara number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Nara.jpg


when Kagawa.Touched
name x16
y name y16
do
set Kanji.Picture to text KagawaGoogle.jpg
set global prefectura to text Kagawa
set component ImageSprite.Z to component Kagawa number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Kagawa.jpg
when Kagoshima.Touched
name x17
y name y17
do
set Kanji.Picture to text KagoshimaGoogle.jpg
set global prefectura to text Kagoshima
set component ImageSprite.Z to component Kagoshima number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Kagoshima.jpg
when Nagano.Touched
name x25
y name y25
do
set Kanji.Picture to text NaganoGoogle.jpg
set global prefectura to text Nagano
set component ImageSprite.Z to component Nagano number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Nagano.jpg
when Nagasaki.Touched
name x26
y name y26
do
set Kanji.Picture to text NagasakiGoogle.jpg
set global prefectura to text Nagasaki
set component ImageSprite.Z to component Nagasaki number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Nagasaki.jpg


when Okayama.Touched
name x33
y name y33
do
set Kanji.Picture to text OkayamaGoogle.jpg
set global prefectura to text Okayama
set component ImageSprite.Z to component Okayama number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Okayama.jpg
when Osaka.Touched
name x34
y name y34
do
set Kanji.Picture to text OsakaGoogle.jpg
set global prefectura to text Osaka
set component ImageSprite.Z to component Osaka number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Osaka.jpg
when Saitama.Touched
name x35
y name y35
do
set Kanji.Picture to text SaitamaGoogle.jpg
set global prefectura to text Saitama
set component ImageSprite.Z to component Saitama number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Saitama.jpg
when Tokyo.Touched
name x41
y name y41
do
set Kanji.Picture to text TokyoGoogle.jpg
set global prefectura to text Tokyo
set component ImageSprite.Z to component Tokyo number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Tokyo.jpg
when Tottori.Touched
name x42
y name y42
do
set Kanji.Picture to text TottoriGoogle.jpg
set global prefectura to text Tottori
set component ImageSprite.Z to component Tottori number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Tottori.jpg
when Toyama.Touched
name x43
y name y43
do
set Kanji.Picture to text ToyamaGoogle.jpg
set global prefectura to text Toyama
set component ImageSprite.Z to component Toyama number 1.5
set component ImageSprite.Z to global ultimapolsad number 0.5
set global ultimapolsada to text Toyama.jpg


when Sendai.Touched x y name x36 name y36
do set Kanji.Picture to text SendaiGoogle.jpg
set global prefectura to text Sendai
set component ImageSprite.Z to component Sendai number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Sendai.jpg
when Shiga.Touched x y name x37 name y37
do set Kanji.Picture to text ShigaGoogle.jpg
set global prefectura to text Shiga
set component ImageSprite.Z to component Shiga number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Shiga.jpg
when Shimane.Touched x y name x38 name y38
do set Kanji.Picture to text ShimaneGoogle.jpg
set global prefectura to text Shimane
set component ImageSprite.Z to component Shimane number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Shimane.jpg
when Wakayama.Touched x y name x44 name y44
do set Kanji.Picture to text WakayamaGoogle.jpg
set global prefectura to text Wakayama
set component ImageSprite.Z to component Wakayama number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Wakayama.jpg
when Yamagata.Touched x y name x45 name y45
do set Kanji.Picture to text YamagataGoogle.jpg
set global prefectura to text Yamagata
set component ImageSprite.Z to component Yamagata number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Yamagata.jpg
when Yamaguchi.Touched x y name x46 name y46
do set Kanji.Picture to text YamaguchiGoogle.jpg
set global prefectura to text Yamaguchi
set component ImageSprite.Z to component Yamaguchi number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Yamaguchi.jpg


when Shizuoka.Touched x y name x39 name y39
do set Kanji.Picture to text ShizuokaGoogle.jpg
set global prefectura to text Shizuoka
set component ImageSprite.Z to component Shizuoka number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Shizuoka.jpg
when Tokushima.Touched x y name x40 name y40
do set Kanji.Picture to text TokushimaGoogle.jpg
set global prefectura to text Tokushima
set component ImageSprite.Z to component Tokushima number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Tokushima.jpg
when Yamanashi.Touched x y name x47 name y47
do set Kanji.Picture to text YamanashiGoogle.jpg
set global prefectura to text Yamanashi
set component ImageSprite.Z to component Yamanashi number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Yamanashi.jpg
when Yokohama.Touched x y name x48 name y48
do set Kanji.Picture to text YokohamaGoogle.jpg
set global prefectura to text Yokohama
set component ImageSprite.Z to component Yokohama number 1.5
set component ImageSprite.Z to global ultimapolsada number 0.5
set global ultimapolsada to text Yokohama.jpg

Screen TutorialTest (Ajuda):

Configurar Local de AppInventor:

Per a configurar el Local de AppInventor, s'ha de fer abans una prèvia configuració pel seu bon funcionament.

1- Hem d'haver instal·lat correctament el Java amb la versió que ens exigirà el mateix AppInventor pel seu bon funcionament.

2- Un cop veiem que funciona tot correctament, hem de tenir uns arxius com a extra, que ho podrem descarregar en un arxiu .rar que ens facilita ja a través de la pàgina web oficial de AppInventor (https://sites.google.com/site/aprendeappinventor/app-inventor-en-servidor-local).

3- Un cop tenim l'arxiu .rar amb els arxius que necesitem, el descomprimim i obrim la carpeta. En aquesta carpeta, hi haurà diverses carpetes. Haurem d'anar a la carpeta AppInventor i allà trobarem un arxiu anomenat "startAI.cmd".

4- Executem aquest arxiu i ens sortirà una finestra de processador de comandes.

5- Esperem que carregui i al final ens sortirà un avís de que el servidor local ja està llest en http://localhost:8888.

6- Un cop ja hem rebut aquest missatge, ja podem obrir el navegador que utilitzem i hem de posar com a URL el mateix http://localhost:8888.

7- Entrareu a una pàgina web molt semblant a la del núvol de AppInventor, us demanaran també el correu electrònic Gmail i un cop que ho introduïu, el servidor local estarà llest.