Curso
2016/2017

Aplicaciones móviles con APP Inventor

Curso de Iniciación al desarrollo de aplicaciones Android con App Inventor 2

Autor:
Antonio Ruiz Murcia
Profesor de matemáticas en el IES Blas Infante
Córdoba, Febrero 2017

CEP "Luisa Revuelta" de Córdoba

Descripción:

Hasta hace algunos años estaba extendida la creencia de que desarrollar programas de software requería grandes conocimientos informáticos, ser un experto en la materia.
Sin embargo, con la aparición de algunos lenguajes de programación educativos (LPE) como Scratch, que utilizan una técnica de cómputo creativo (entornos gráficos en los que los usuarios programan mediante bloques de acciones de una manera fácil e intuitiva), se ha comprobado que hasta un niño puede crear sus propios programas o juegos.
App Inventor es un entorno de desarrollo de aplicaciones para dispositivos Android creado por Google y el MIT aunque, en la actualidad, es este último quien se encarga de su mantenimiento. El servicio se ofrece a través del modelo "computación en la nube", lo que significa poder acceder y gestionar nuestro trabajo desde cualquier ordenador y lugar. Con una interfaz de programación que recuerda a Scratch cualquiera puede elaborar, en pocas horas, sus propias aplicaciones sin necesidad de disponer de conocimientos previos en programación.
Numerosos estudios indican que al aprender a programar, los estudiantes utilizan el pensamiento computacional adquiriendo habilidades del pensamiento procedimental que son utilizadas para el desarrollo de la lógica y la inteligencia en general (Barrera, 2013). Además, favorece la creatividad, mejora la motivación y facilita la adquisición y desarrollo de competencias clave.
En un principio, App Inventor estaba destinado al alumnado universitario para que pudiera desarrollar sus propias aplicaciones a la medida de sus necesidades. Pero pronto se descubrió que, por su facilidad de uso, estaba al alcance de estudiantes de etapas más tempranas. Hoy en día cuenta con una comunidad de cerca de 6 millones de usurarios registrados que representan a 195 países en todo el mundo.

Objetivos:

- Introducir al profesorado en el desarrollo de aplicaciones móviles con App Inventor
- Dar a conocer el potencial de App Inventor y las posibilidades que ofrece en el ámbito educativo
- Manejar y programar los diferentes sensores que se encuentran integrados en los dispositivos móviles (acelerómetro, ubicación, dirección, etc.)
- Utilizar los gestos que son detectados sobre la pantalla táctil del dispositivo y aplicarlos en el diseño de juegos
- Integrar la utilización de aplicaciones externas (Google Maps) en nuestra aplicación
- Crear archivos de geolocalización (kml) para generar vistas gráficas tridimensionales de rutas con Google Earth
- Compartir proyectos en la Galería de App Inventor
- Fomentar el uso de App Inventor como una herramienta para la adquisición y desarrollo de competencias clave del alumnado

Contenidos:

- El entorno de desarrollo: Gestión de Proyectos, Editor de Diseño y Editor de Bloques
- Conectando con el emulador: MIT AI2 Companion
- Componentes de interfaz de usuario
- Componentes Reconocimiento de Voz y traductor Yandex
- Lienzo y Sprites. Nuestro primer juego
- Variables y listas. Procedimientos. Diseñando un test
- El acelerómetro. Componente Reloj: Juego de la abeja
- Gestos detectados sobre la pantalla táctil. Dibujar sobre un lienzo. La cámara de fotos y uso de la galería. Pequeña base de datos. Guardar una imagen
- Animaciones
- Sensor de ubicación: localización GPS. Visores de listas y archivos de texto. Activity Starter (para la integración de aplicaciones externas). Archivos "kml"
- Sensor de Orientación. Construcción de una brújula
- Instalación de la APK en nuestro dispositivo móvil
- Compartir proyectos con otros usuarios en la Galería
- App Inventor en el aula. Experiencias

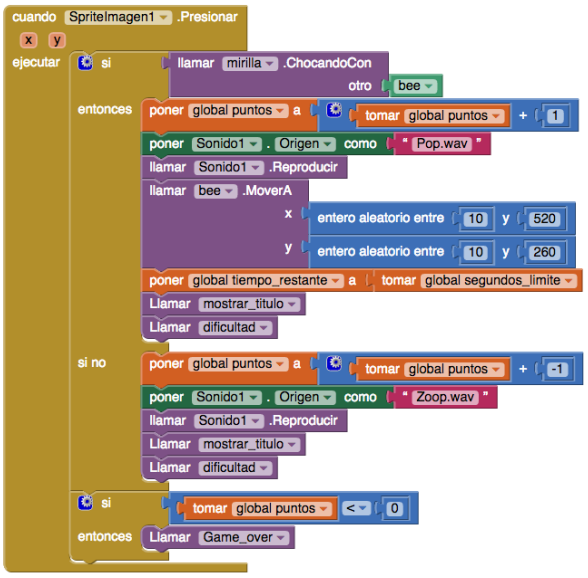
ÍNDICE

INTRODUCCIÓN

El uso de las TIC en el aula constituye una pieza fundamental para producir los cambios metodológicos que permitan el objetivo de mejora de la calidad educativa. Numerosos estudios apuntan las ventajas que ofrece su utilización para favorecer el aprendizaje del estudiante. Morrisey (2007) destaca que con estas herramientas se impregnan de dinamismo a las clases, enriqueciendo de forma motivadora el espacio de aprendizaje.

Por otro lado, diversos estudios demuestran que los niños que aprenden a programar obtienen mejores resultados en razonamiento matemático y resolución de problemas. Según indica Barrera (2013), al aprender a programar se adquieren habilidades del pensamiento procedimental que son utilizadas para el desarrollo de la lógica.

App Inventor es un entrono de desarrollo de aplicaciones para dispositivos Android creado por Google en 2010 junto con el MIT a partir de investigaciones previas en informática educativa. De hecho presenta una interfaz de programación que recuerda a Scratch, ya que se basa en bloques secuenciales que se ensamblan como los bloques físicos de LEGO. Además utiliza un editor de diseño “drag and drop” (arrastrar y soltar).



En 2011 Google deja de lado el proyecto pero antes lo publica como código abierto, y es el Instituto Tecnológico de Massachusetts (MIT) el encargado de continuar este proyecto.

App Inventor es un servicio que se ofrece a través del modelo “cloud computing” (computación en la nube), lo que significa poder acceder y gestionar nuestro trabajo desde cualquier ordenador y lugar.

La característica principal de App Inventor es que no se necesitan conocimientos previos de programación. Cualquiera que disponga de una cuenta de Google y un navegador Web puede empezar a desarrollar aplicaciones para dispositivos Android.

CÓMO ACCEDER AL SITIO DE APP INVENTOR

El servicio App Inventor se encuentra en la dirección:

http://ai2.appinventor.mit.edu/

y para acceder es necesario disponer de una cuenta Gmail.

BREVE DESCRIPCIÓN DEL ENTORNO DE DESARROLLO

Para desarrollar nuestras aplicaciones disponemos de tres herramientas:

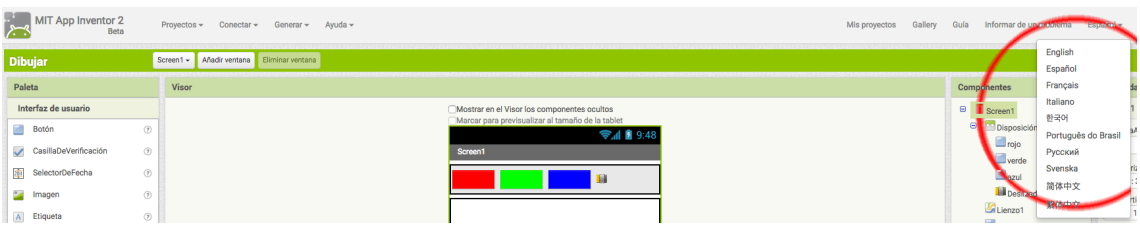
1. Gestor de Proyectos
2. Editor de Diseño
3. Editor de Bloques

GESTOR DE PROYECTOS

Nos permite configurar el idioma en el que deseamos trabajar, crear nuevos proyectos, guardarlos o importarlos así como compartirlos, informar de errores, acceder a ayuda y tutoriales, generar archivos .apk (para instalar la aplicación en el dispositivo Android), etc. Incluye además una herramienta que nos facilitará la elaboración y desarrollo de nuestras aplicaciones: podemos conectar un dispositivo móvil Android (Wifi o USB) y ver los cambios de nuestra aplicación conforme vamos modificando el diseño y los bloques de código.

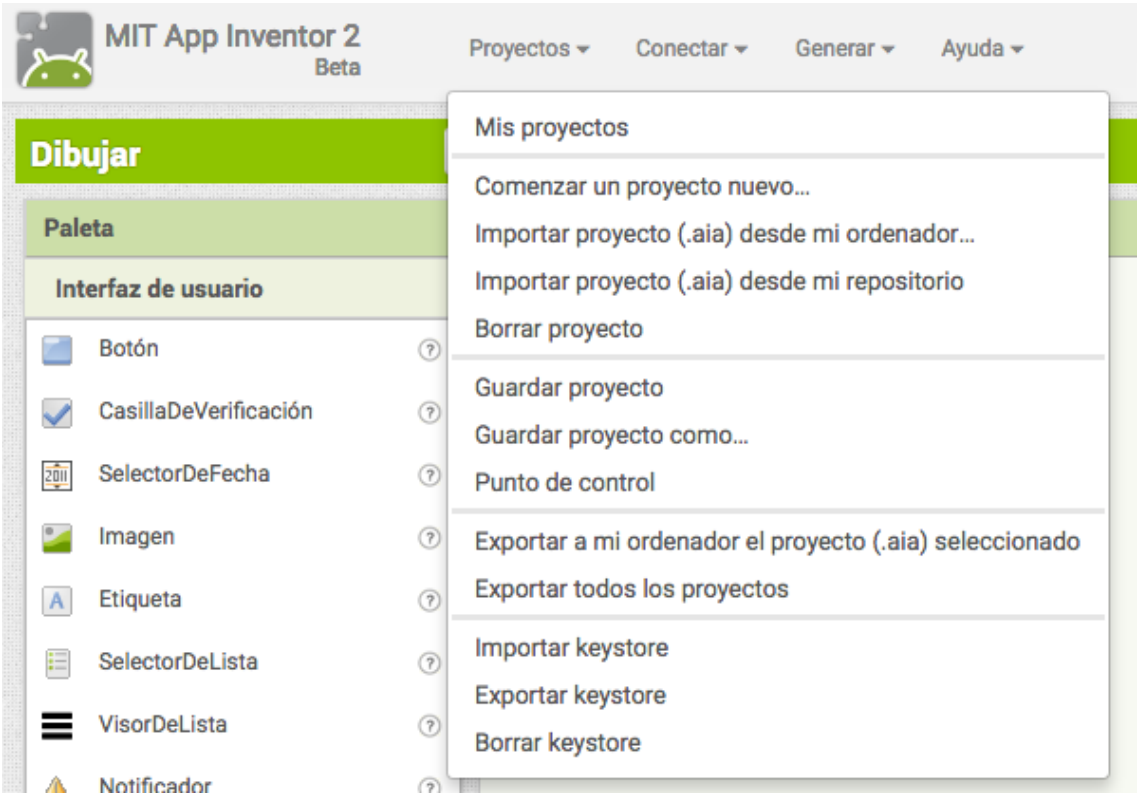
Configuración del idioma

Será la primera acción que realizaremos antes de empezar un nuevo proyecto.



Gestión de Proyectos

Podemos Guardar nuestro proyecto en los servidores de App Inventor o en nuestro ordenador en el formato .aia



En “Mis proyectos” se nos muestra la lista de proyectos que hemos desarrollado y nos ofrece la posibilidad de compartirlos (código y diseño) con otros usuarios de App Inventor a través de la opción “Publicar en la Galería”.

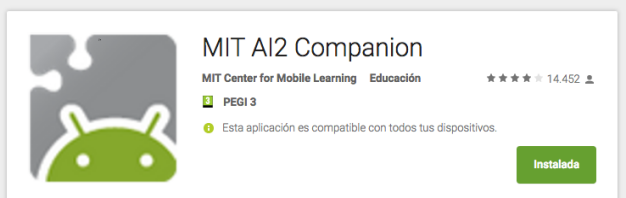
También es posible firmar el archivo de instalación .apk con una contraseña para su cifrado (Keystore)

Conectando con el Emulador o un dispositivo móvil Android (Wifi o USB)

Como se ha mencionado anteriormente, para poder ir comprobando tanto el diseño como el funcionamiento del código de nuestra aplicación, podemos instalar un Emulador en el ordenador o, mejor aún, conectar nuestro teléfono móvil o tableta mediante Wifi o un cable USB.

La opción más recomendable es utilizar nuestro dispositivo móvil como banco de pruebas a través de la conexión Wifi. Para ello se requiere que tanto el ordenador en el que estamos trabajando como nuestro dispositivo móvil se encuentre en la misma red (aunque el ordenador esté conectado a Internet a través de cable).

Para establecer la conexión necesitamos instalar en nuestro dispositivo Android la aplicación "***MIT AI2 Companion***" disponible en Google Play Store.



Ahora, para conectar App Inventor con el móvil haremos clic en **Conectar**, y elegiremos la opción **AI Companion**.

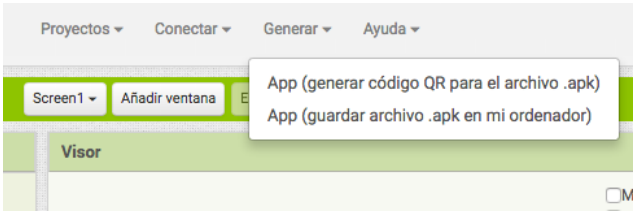
Se abrirá una ventana como esta:

A continuación abrimos en el móvil la aplicación que hemos descargado. Cuando la abramos elegiremos **Connect with code** (y escribiremos el código de letras y números que aparece en la ventana) o **Scan QR code** (sólo tenemos que apuntar al código QR y la conexión se establecerá automáticamente).

Generar el archivo APK

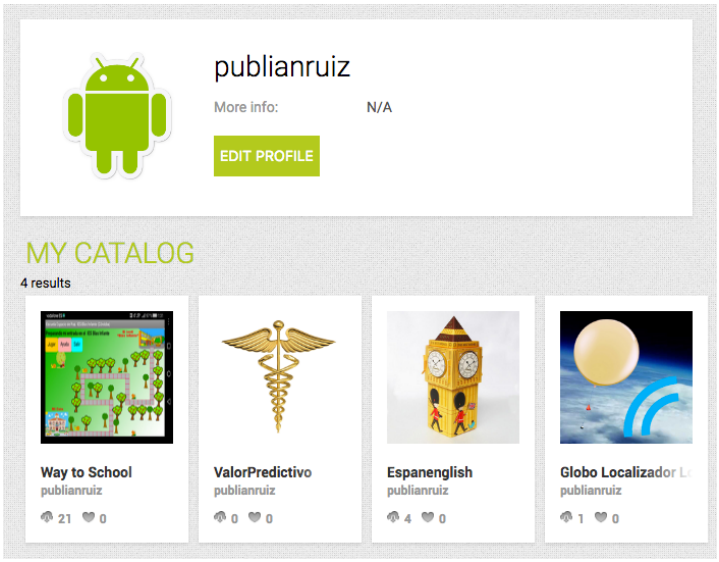
Una vez que hayamos finalizado nuestra aplicación tenemos que generar el archivo de instalación para el dispositivo móvil. Estos archivos tienen la extensión apk.
Se nos ofrece dos opciones:

- App (generar código QR para el archivo .apk). Usando cualquier lector de códigos QR en nuestro móvil, obtenemos el enlace de descarga de nuestra aplicación. Es conveniente recordar que debemos modificar los Ajustes en nuestro dispositivo para permitir la instalación de Apps que no preceden de Play Store (Configuración por defecto)
- App (guardar archivo .apk en mi ordenador). Ahora podemos publicar nuestra App en Play Store o enviarlo a nuestro dispositivo para su instalación.



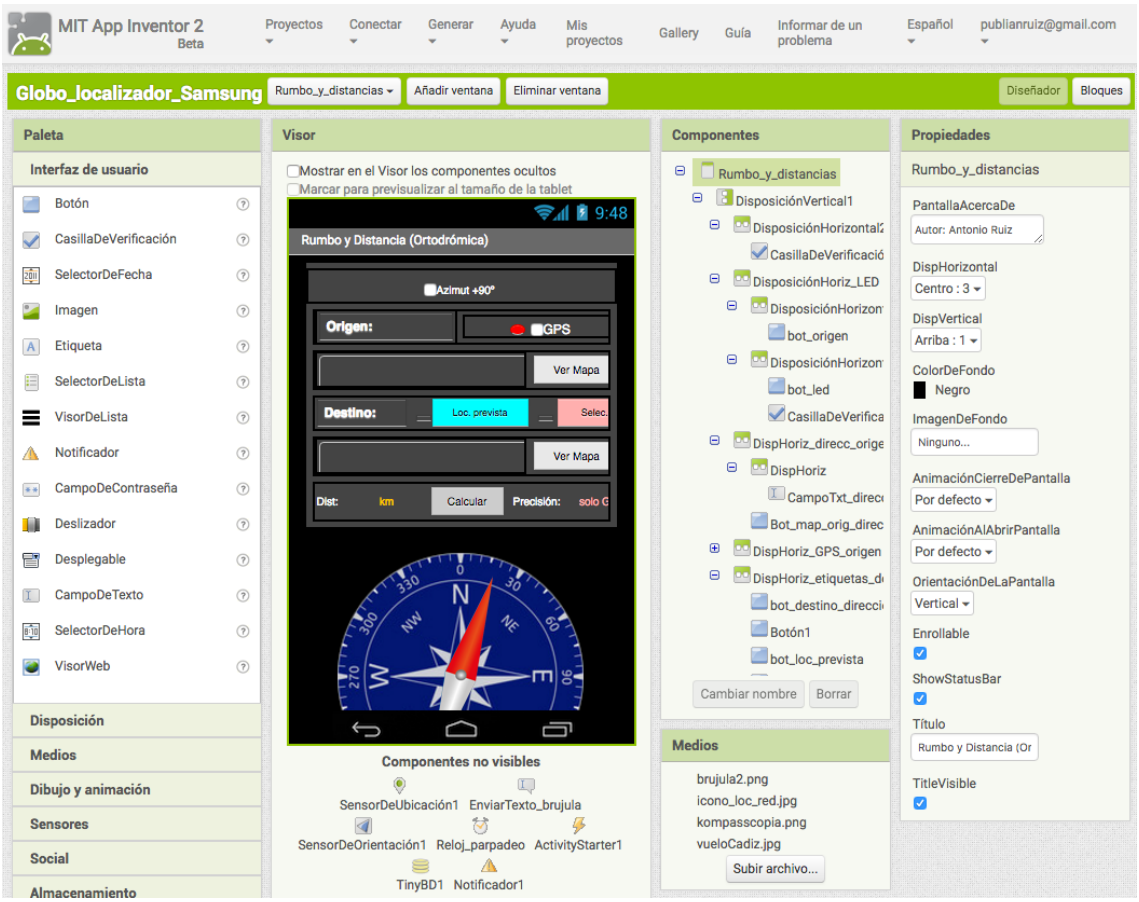
Proyectos de otros usuarios

Haciendo clic en **Galería** tenemos acceso a proyectos que han sido compartidos por otros usuarios. Este repositorio de recursos puede sernos de gran ayuda para resolver dudas y seguir profundizando en el uso y programación de algunos componentes que ofrece App Inventor.



EDITOR DE DISEÑO

Se encuentra dividido en cuatro columnas:



Para acceder al editor de diseño haremos clic en el botón "**Diseñador**" situado en la parte superior derecha (al lado del botón **Bloques**)

Paleta

Situada a la izquierda nos muestra los diferentes componentes disponibles, que pueden ser visibles (botones, etiquetas, deslizadores, etc.) o no-visibles (notificadores, traductor, sensores, etc.).

Visor

Representa la pantalla del dispositivo móvil. Desde la Paleta iremos arrastrando y soltando aquí los componentes que conformarán nuestra aplicación. Los componente no visibles se mostrarán en la parte inferior. Con el visor podemos diseñar el aspecto (la forma en que se distribuyen e integran los componentes visibles) de la app.

Componentes

Esta columna está integrada por los componentes que hemos agregado a la aplicación. El primero es siempre la pantalla (ventana) que estamos diseñando, que en el caso de ser la principal o la única de la app, se llamará **Screen1** (en este caso no se puede modificar el nombre).

Es conveniente asignarle un nombre a cada componente con el fin de identificarlo fácilmente cuando procedamos a su programación en el **Editor de Bloques**.

Debajo de **Componentes** tenemos el apartado de **Medios**. Desde aquí subiremos los archivos (imágenes, sonidos, etc.) que vayamos a utilizar en nuestra App.

Propiedades

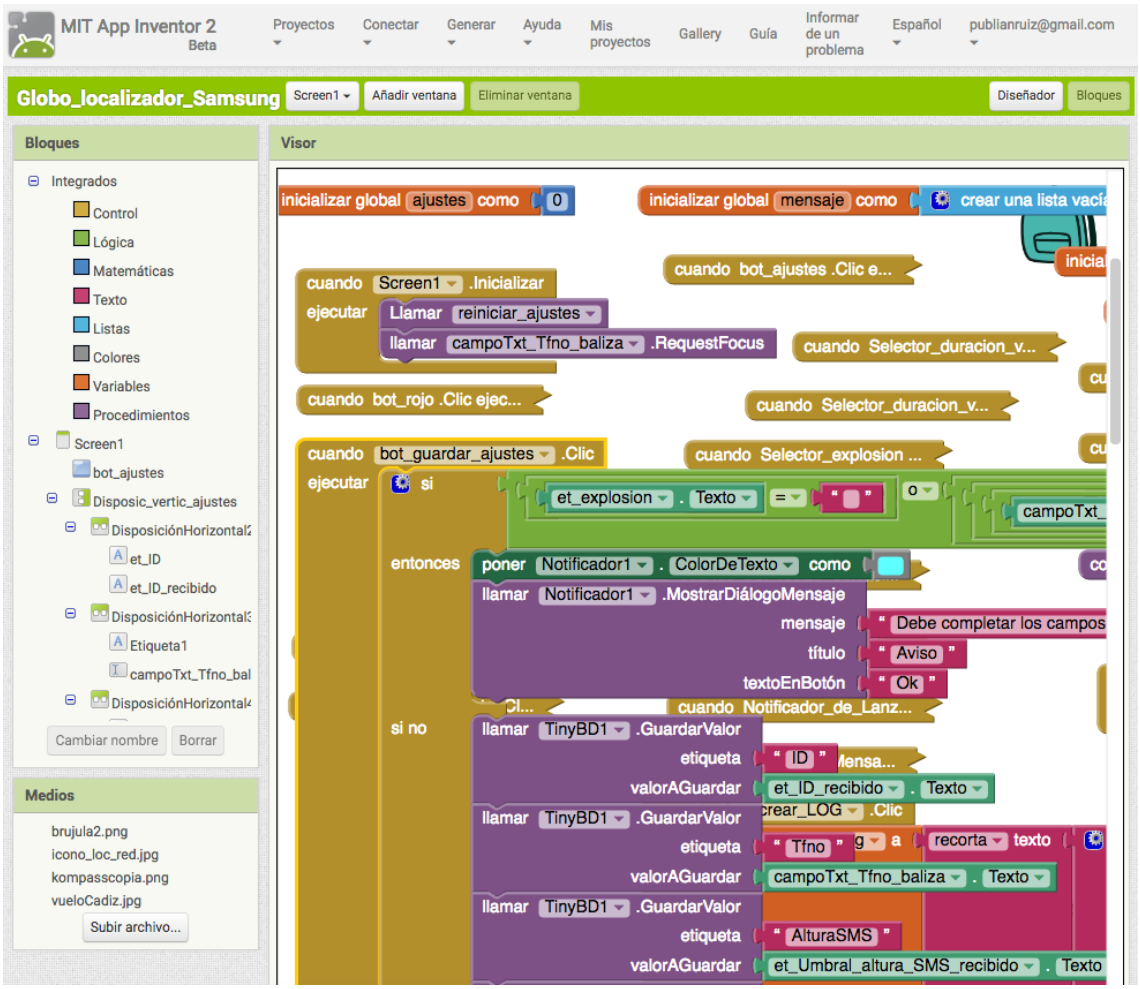
Nos permite establecer la configuración que tendrá por defecto cada uno de los componentes que conforman nuestra App (tanto visibles como no visibles). Estos valores iniciales los podemos modificar posteriormente desde el Editor de Bloques.

EDITOR DE BLOQUES

En esta pantalla programaremos cada uno de los componentes de la aplicación, es decir, iremos indicando las acciones que deberán realizarse cuando se produzca un ***evento*** (hacer clic en un botón, transcurra cierto tiempo de reloj, cambie nuestra ubicación, arrastrar un deslizador, colisionen dos sprites, etc.).

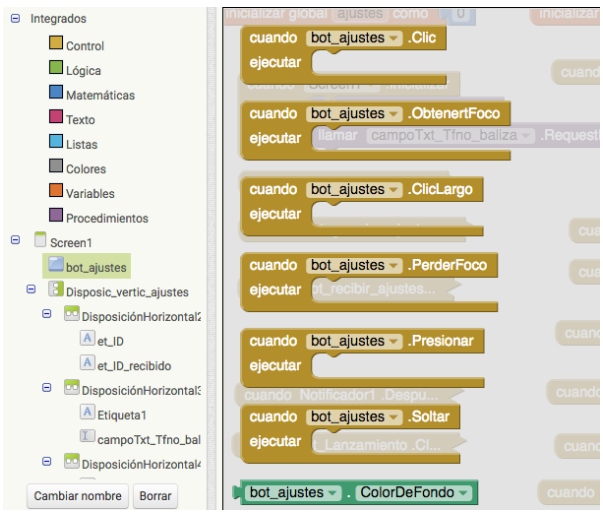
Se accede haciendo clic en el botón **Bloques** situado en la parte superior derecha (al lado del botón **Diseñador**)

Está formado por la columna **Bloques** (izquierda) y el **Visor**.



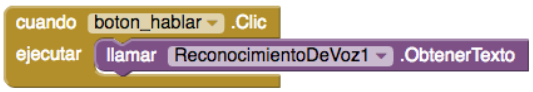
Dentro de **Bloques**, en el apartado de **Integrados**, tenemos de un conjunto de herramientas básicas para la programación. A continuación, se encuentran los componentes con los nombre que le asignamos desde el Editor de Diseño.

Al hacer clic sobre cualquier componente se nos despliega un conjunto de acciones disponibles para dicho componente. Las acciones se completarán con las herramientas del apartado **Integrados**.



Las acciones se realizarán siempre como consecuencia de la ocurrencia de un *evento*. Los *eventos* se pueden producir por la interacción del usuario con los componentes visibles (arrastrar un deslizador) o por los cambios de valor de un sensor (orientación del dispositivo móvil, acelerómetro, etc.).

Por ejemplo, si deseamos que al hacer clic sobre el botón *Hablar* (creado en la fase de diseño) se active el componente ***Reconocimiento de Voz*** construiríamos un bloque como el siguiente:



En la parte superior derecha del **Visor** encontramos una **Mochila** que nos será de utilidad cuando deseemos utilizar el mismo código en otra ventana u otra aplicación.

En la parte inferior izquierda del Visor se nos muestra un notificador de errores que nos ayudará a depurar nuestra app.

APRENDEMOS HACIENDO

Para tener una idea más precisa del funcionamiento de App Inventor, empezaremos con el desarrollo, paso a paso, de dos aplicaciones muy sencillas. Se recomienda llevarlas a cabo en el ordenador conforme se realiza la lectura.

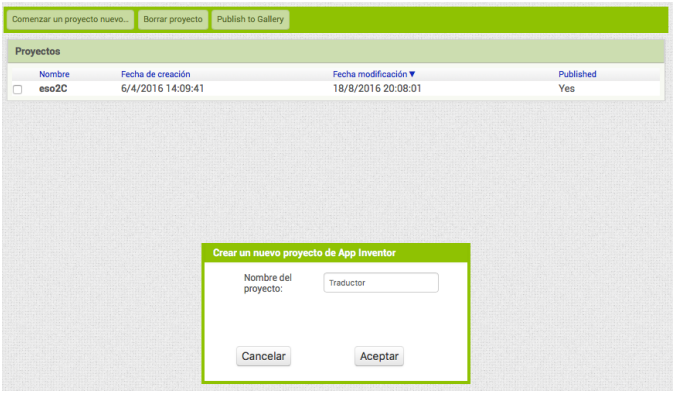
TRADUCTOR

Vamos a crear un sencillo traductor de Español-Inglés en el que el texto lo podremos introducir escribiendo o bien directamente hablando y haciendo uso del reconocimiento de voz.
Una vez realizada la traducción podremos indicarle que la lea (de texto a voz) usando el altavoz del dispositivo móvil.

Paso 1

Entramos en la dirección de App Inventor: http://ai2.appinventor.mit.edu
indicando nuestra cuenta de Gmail que utilizaremos para acceder al servicio.

Seleccionamos el **idioma** español y hacemos clic en el botón "**Comenzar un proyecto nuevo…**". Elegiremos un *nombre* para nuestro Proyecto, por ejemplo, "Traductor" (o cualquier otro).

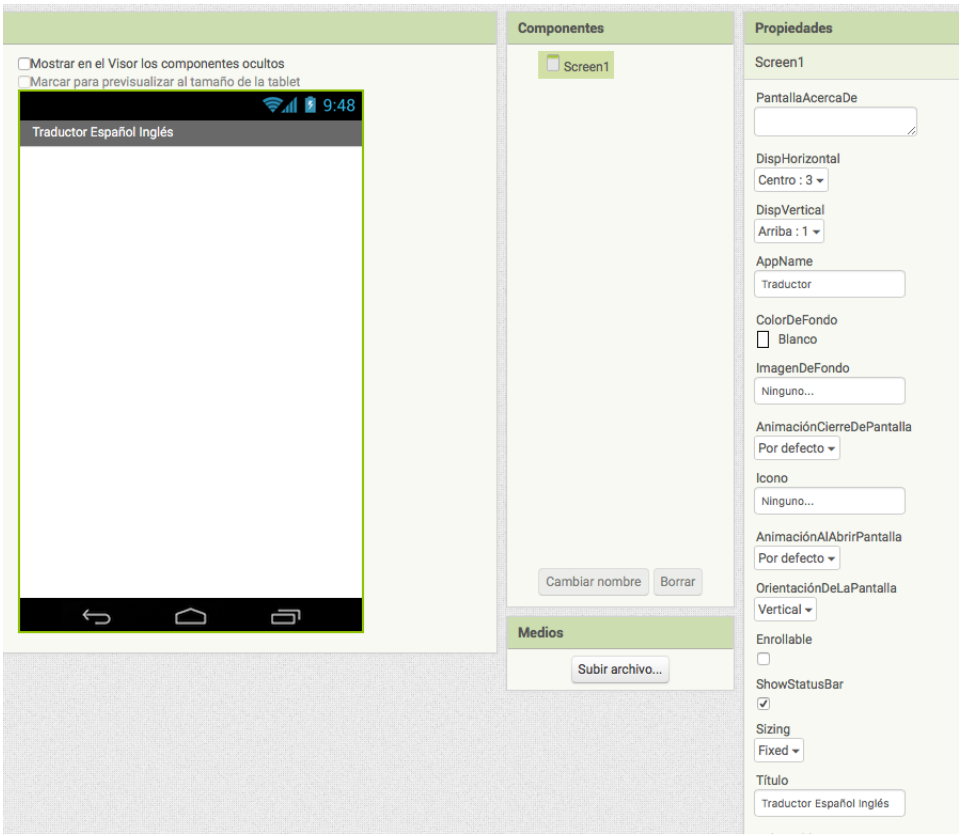


Una vez aceptada la operación se cargará en pantalla el **Editor de Diseño**.

Paso 2

En la columna **Componentes** estará seleccionada la ventana principal que por defecto es nombrada como "Screen1". Dicho nombre no se puede modificar. Veremos más adelante que cuando añadimos ventanas nuevas si será posible renombrarlas.

En la columna **Propiedades** (para este componente) indicaremos que deseamos una **disposición horizontal** "*centrada*", **nombre de la App**: *Traductor* (este será el nombre que aparecerá debajo del icono en nuestro dispositivo móvil) y seleccionamos **orientación** "*Vertical*" para la pantalla. En **Título** escribiremos "*Traductor Español Inglés*". Lo demás lo dejamos tal y como está.



Paso 3

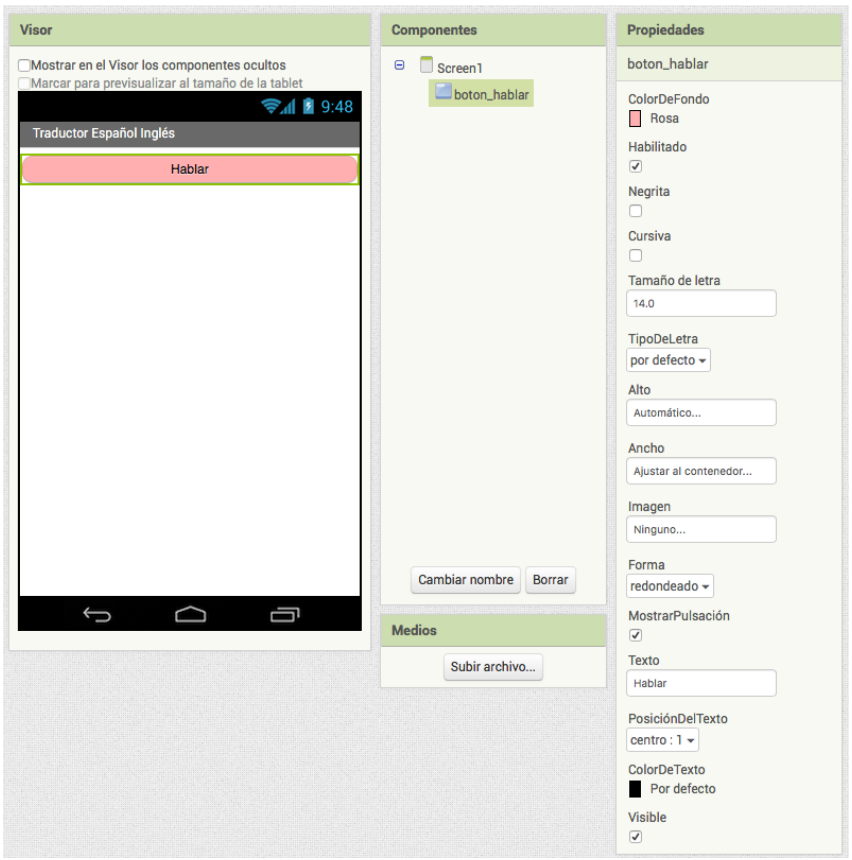
Ahora iremos añadiendo a la pantalla "Screen1" (Visor) los componentes necesarios, eligiendo el lugar en el que deseamos que aparezcan (aquellos que sean visibles). Es el momento de decidir el "aspecto" que tendrá nuestra App, por lo que elegiremos color, tamaño, forma, etc., para cada uno de los componentes.

Componente 1: Botón (Hablar)

Necesitaremos un **botón** (Hablar) para que al hacer clic sobre el se inicie el **Reconocimiento de voz.** Por consiguiente, hacemos clic sobre el componente **Botón** de la *Interfaz de Usuario* (columna **Paleta**) y lo arrastramos hasta el **Visor**, situándose en la parte superior de la pantalla principal (Screen1). Aparecerá centrado en la ventana pues, anteriormente, elegimos Disposición Horizontal "Centro" en las propiedades de Screen1. En la columna Componentes renombramos Botón1 por "boton_hablar". *Cuando en una aplicación tenemos un gran número de componentes, es muy conveniente renombrarlos adecuadamente para facilitar su programación (al recordarnos la función que tiene asignada)*

Para este componente **boton_hablar** estableceremos las siguientes **Propiedades** (columna derecha):

- **Color de Fondo**: *Rosa*
- **Ancho:** *Ajustar al contenedor*
- **Forma:** *Redondeado*
- **Texto:** *Hablar*



Componente 2: Campo de Texto (Texto para traducir)

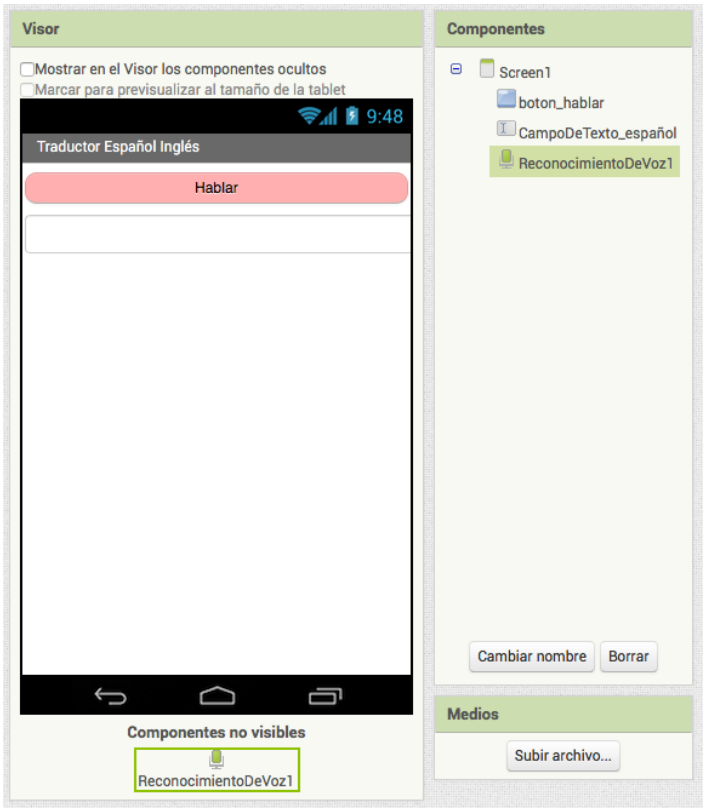
Arrastramos ahora el componente CampoDeTexto hasta situarlo debajo del botón Hablar. Utilizaremos esta caja de texto para recoger el texto obtenido por el reconocimiento de voz, o bien para introducir con el teclado el texto que deseemos. Al igual que antes, lo renombramos por CampoDetexto_español.

Propiedades:

- **Ancho**: *Ajustar al contenedor*
- **Pista**: *texto español*
- **Multilínea**: *activado* (si la longitud del texto supera el ancho de la pantalla, se ampliará el alto de la caja de texto para mostrar el texto íntegro en varias líneas)

Componente 3: Reconocimiento de Voz

En la ventana **Paleta**, dentro del bloque **Medios**, arrastramos hasta el interior del **Visor** el componente **Reconocimiento de Voz,** el cual aparecerá en la parte inferior fuera de la pantalla principal (Componentes no visibles).



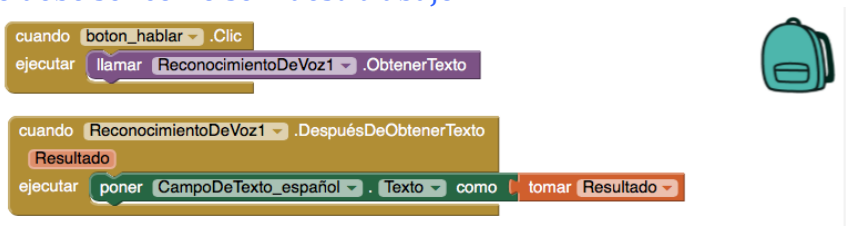
Para este componente no se requiere establecer propiedad alguna.

Paso 4

Vamos a proporcionarles funcionalidad a los componente que hemos agregado hasta este momento. Para ello entraremos en el **Editor de Bloques** haciendo clic en el botón **Bloques** (situado en la parte superior derecha).

- Haciendo clic sobre el componente llamado *boton_hablar* (en ventana Bloques) se desplegará las diferentes opciones disponibles. Elegimos la primera etiqueta: *cuando boton_hablar.Clic.*
- Hacemos clic sobre el componente *ReconocimientoDeVoz1* y posteriormente sobre la etiqueta *llamar ReconocimientoDeVoz1.ObtenerTexto*, que arrastraremos hasta el interior de la etiqueta anterior.
- Repetimos la operación anterior pero eligiendo esta vez la etiqueta *cuando ReconocimientoDeVoz1.DespuésDeObtenerTexto.*
- Hacemos clic sobre el componente *CampoDeTexto_español*, elegimos la etiqueta "poner *CampoDeTexto_español.Texto como"*, que la colocaremos en el interior de la etiqueta anterior.
- Situando el cursor del ratón (sin hacer clic) sobre *Resultado*, se nos despliega una ventana con dos opciones: *tomar Resultado* y *poner Resultado*. Elegimos la primara y la encajamos en la etiqueta "poner *CampoDeTexto_español.Texto como"* colocándola a su derecha.

El resultado debe ser como se muestra abajo



Paso 5

Conectamos nuestro dispositivo móvil con App Inventor, tal y como se describió anteriormente, para comprobar el buen funcionamiento de lo realizado hasta aquí.

Al hacer clic en el botón **Hablar** se iniciará el reconocimiento de voz. Pronunciamos alguna frase finalizando con un silencio. A los pocos segundos aparecerá en pantalla el texto correspondiente.

Si hay demasiado ruido ambiental, el reconocimiento de voz no funcionará correctamente y, en ese caso, es preferible introducir el texto a través del teclado.

Paso 6

Componente 4: Disposición horizontal

Continuamos con el diseño de nuestra App.

Volviendo al **Editor de Diseño** (botón **Diseñador**), añadiremos el componente visible *DisposiciónHorizontal*, que se encuentra dentro del apartado **Disposición**, arrastrándolo hasta situarlo debajo del CampoDeTexto.

Propiedades:

- **DispHorizontal**: *Centro*
- **DsipVertical**: *Centro*
- **Ancho**: *Ajustar al contenedor*

Paso 7

Componente 5: Botón Traducir
Componente 6: Botón Leer
Componente 7: Etiqueta1 (para separar los botones)

Al interior del componente *DisposiciónHorizontal* arrastraremos dos botones que renombraremos por *boton_Traducir* y *boton_Leer*, respectivamente, y una etiqueta para separar ambos botones entre sí.

Propiedades (botón **Traducir**):

- **ColorDeFondo**: *Turquesa*
- **Ancho**: *45 por ciento*
- **Forma**: *Redondeado*
- **Texto**: *Traducir*

Propiedades (botón **Leer**):

- **ColorDeFondo**: *Naranja*
- **Ancho**: *45 por ciento*
- **Forma**: *Redondeado*
- **Texto**: *Leer*

Propiedades (**Etiqueta1**):

- **Ancho**: *5 por ciento*
- **Texto**: (dejamos el campo en blanco)

Paso 8

Componente 8: Campo de Texto (texto traducido)

Añadimos un nuevo CampoDeTexto arrastrándolo hasta colocarlo debajo de la **Disposición Horizontal** anterior. Lo renombraremos por *CampoDeTexto_ingles*. Este caja de texto la utilizaremos para mostrar la traducción del texto en español y para introducir, desde el teclado, el texto en inglés que deseamos que lea.

Propiedades:

- **Ancho**: *Ajustar al contenedor*
- **Pista**: *texto inglés*
- **Multilínea**: *activado*

Componente 9: Texto a Voz (Leer)

También añadiremos el componente *TextoAVoz* que se encuentra en el apartado de **Medios**. Nos servirá para leer el texto en inglés.

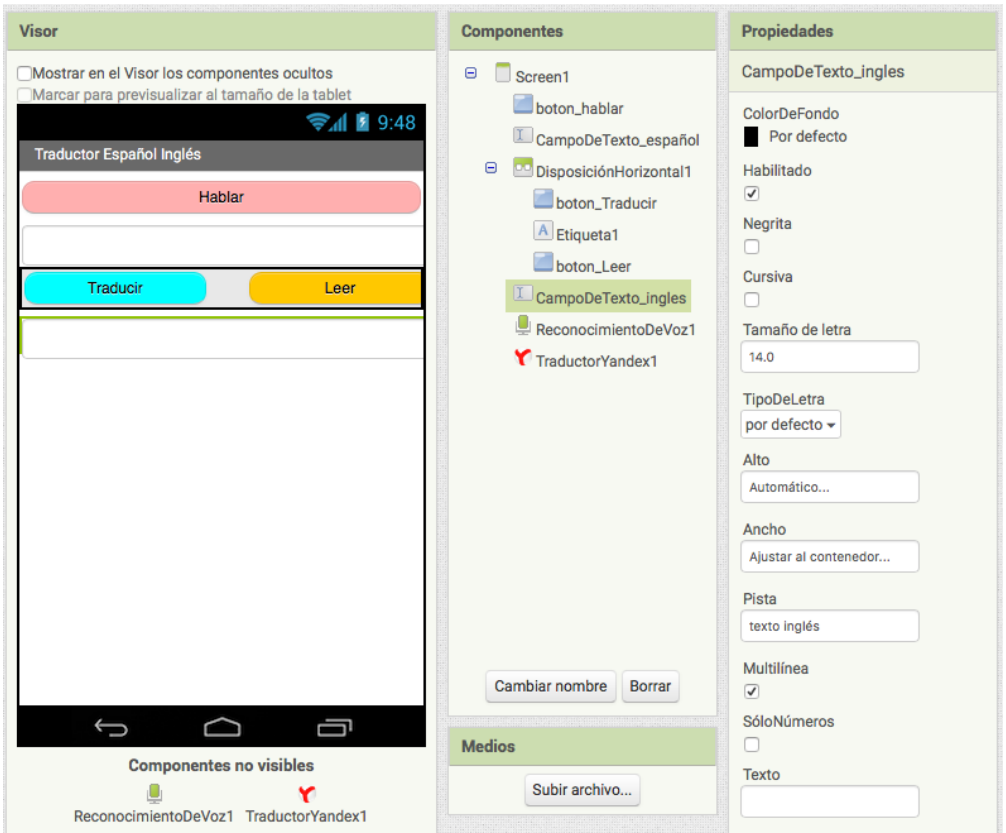
Propiedades:

- **País**: *GBR*
- **Idioma**: *en (English)*

Paso 9

Componente 10: Traductor Yandex

Como deseamos que la aplicación traduzca el texto al inglés, necesitamos añadir el componente (no visible) **TraductorYandex** que se encuentra en el apartado **Medios**. El idioma se establecerá en los bloques de programación (Editor de Bloques).

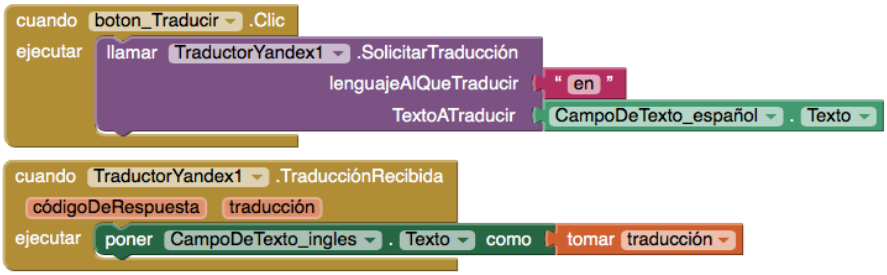


Paso 10 (Traducimos el texto)

Entramos nuevamente en el **Editor de Bloques** (botón **Bloques**) para programar el botón **Traducir**.

Cuando hagamos clic sobre él, llamaremos al traductor Yandex indicándole el *idioma* al que deseamos traducir y el *lugar* donde se encuentra el texto a traducir. Asimismo, una vez realizada la traducción, debemos indicar donde queremos que la muestre. En nuestro caso será en CampoDeTexto_ingles.

Arrastrando la etiquetas pertinentes debemos obtener unos bloques así:



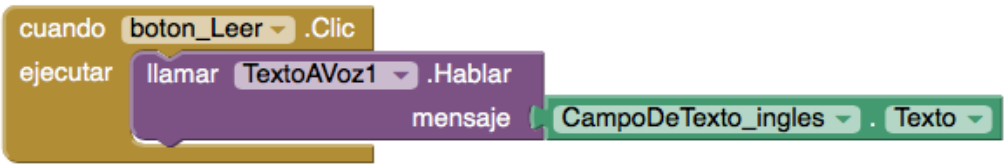
La etiqueta "*tomar traducción*" se obtiene al hacer clic sobre ella cuando aparezca al situar el cursor sobre "traducción" (etiqueta Traductor Yandex1.Traducción Recibida).

La etiqueta:

se utiliza para escribir texto. Se obtiene en la herramienta básica Texto dentro del apartado Integrados.

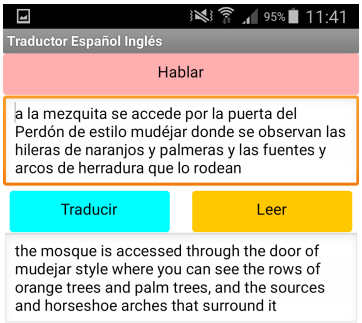
Paso 11 (Leer en inglés)

Finalmente, programamos el botón Leer. Cuando hagamos clic en el botón leer llamaremos al componente *TextAVoz1* indicándole, con la etiqueta correspondiente, que el texto a leer se encuentra en el *CampoDeTexto_ingles*.
El bloque quedaría así:



Paso 12 (Comprobamos funcionamiento y generamos archivo .apk)

Testeamos la aplicación con nuestro dispositivo móvil (debemos mantenerlo conectado a App Inventor) y, si todo va bien, podemos **generar** el archivo de instalación .apk.
Como se indicó anteriormente, podemos generar un código QR (necesitamos un el lector QR) o bien guardarlo en el ordenador.



Actividad 1

Modificar el programa anterior para que la traducción y lectura la realice en el idioma francés.

Actividad 2

Modificar la App para que al *agitar* el dispositivo móvil lea (altavoz) el texto "*Esta es mi primera práctica*"
Para ello necesitamos añadir dos componentes más.

Componente 11. Acelerómetro

Este componente (no visible) se encuentra en el apartado de **Sensores**.

Propiedades:

- **IntervaloMínimo**: *400* milisegundos (entre dos sacudidas)
- **Sensibilidad**: *moderada*

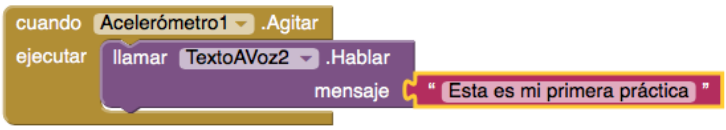
Componente 12. TextoAVoz2 (para leer en español)

Se encuentra en el apartado **Medios**.

Propiedades:

- **País**: *ESP*
- **Idioma**: *es (español)*

Programación de los componentes:



Actividad 3

Añadimos un icono a nuestra aplicación y la instalamos en nuestro dispositivo móvil. App Inventor utiliza el siguiente icono por defecto:

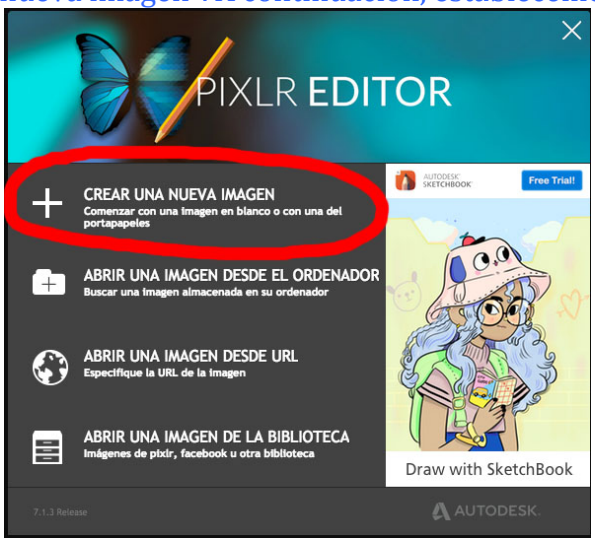
Para cambiar el icono debemos subir una imagen (Columna **Componentes** apartado **Medios**). En la propiedad *Icono* del componente *Screen1*, seleccionamos el nombre del archivo de nuestra imagen.

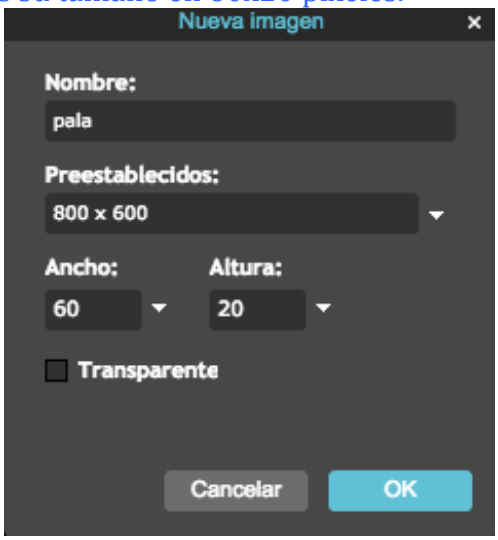
JUEGO DEL FRONTÓN

Es un clásico de los juegos para ordenador. Se trata de una bola que rebota en los bordes de la pantalla y una pala (rectángulo) que es manejado por el jugador. Cuando la bola rebasa a la pala y colisiona con el borde inferior de la pantalla, el jugador pierde y el juego se detiene.
Existen diferentes maneras para controlar el movimiento de la pala (Sprite). Nosotros utilizaremos aquí el "arrastre" con el dedo (por su simplicidad en la programación).
Para diseñar la "pala" utilizaremos el editor de imágenes "pixlr web" disponible en la dirección: https://pixlr.com/editor/

Paso 1. Diseñando la "pala"

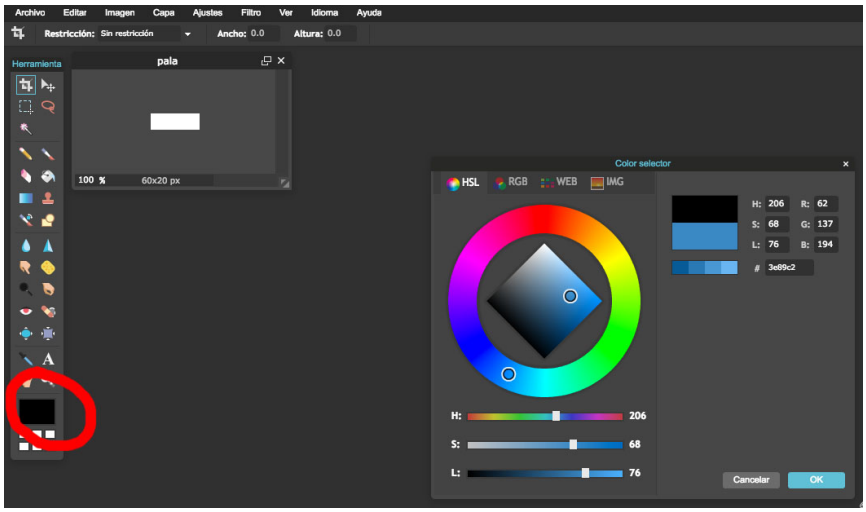
Una vez hayamos entrado en el sitio Web "**pixlr.com**" seleccionamos "Crear una nueva imagen". A continuación, establecemos su tamaño en 60x20 píxeles.



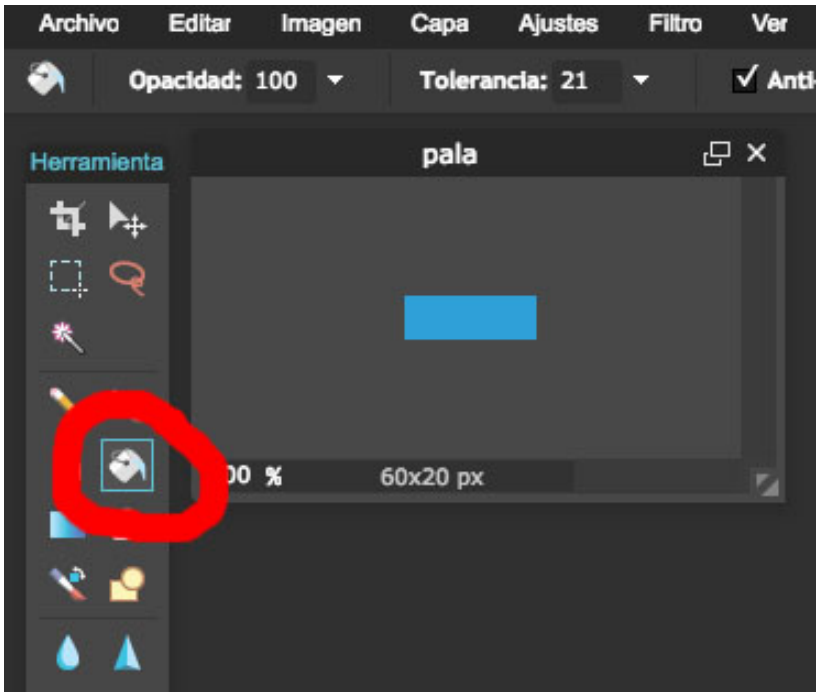


La "pala" es la propia imagen que hemos creado, un rectángulo que solo tenemos que rellenar de color.

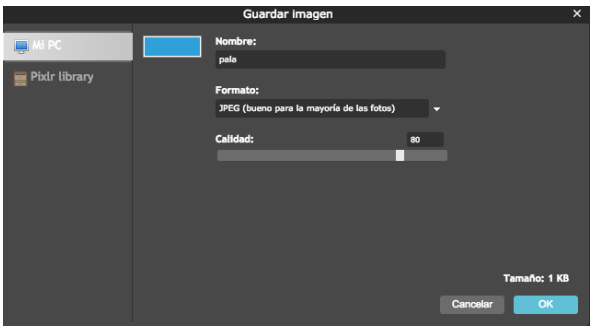
Seleccionamos el color en la paleta



Para rellenar de color el rectángulo utilizamos el "bote de pintura" haciendo clic en el interior de la imagen



Por último, ***guardamos*** el archivo con el nombre "*pala*" y formato "*jpg*" para subirlo a App Inventor.



Paso 2. Creamos el Proyecto en App Inventor

Al igual que en el ejemplo anterior, haciendo clic en "**Proyectos**" seleccionamos "**Comenzar un proyecto nuevo…**" y le asignamos el nombre que deseemos, por ejemplo "*Fronton*" (no está permitido el uso de tildes para nombrar un proyecto).

Propiedades del componente Screen1:

- **OrientaciónDeLaPantalla**: *Vertical*
- **Título**: *Frontón*

Paso 3. Añadimos los componentes

En la parte superior de la pantalla colocaremos un componente de **Disposición Horizontal** donde alojaremos, de momento, un **botón** de *Inicio*. Esta estructura nos facilitará posibles ampliaciones futuras (activar sonido, contador de puntos, otros botones, etc.)
Debajo de la Disposición Horizontal añadiremos un componente **Lienzo**, que albergará un componente **Pelota** y un componente **SpriteImage**. Todos ellos se encuentran dentro del apartado **Dibujo y animación**.
Subimos a App Inventor el archivo "***pala.jpg***" creado anteriormente (*Botón "**Subir archivo…**"*)

El **Lienzo** es un panel rectangular sensible al tacto dentro del cual se puede dibujar y también mover Sprites (imágenes). Hay eventos que permiten saber cuándo y dónde se ha tocado un Lienzo, o hacia dónde se ha arrastrado un Sprite (Imagen o Pelota). Existen métodos para dibujar puntos, líneas y círculos como veremos en otras prácticas.

Los ***Sprites*** son objetos gráficos que podemos mover, arrastrar, lanzar y hacerlos interactuar con otros ***Sprites***.

El componente **Pelota** es un Sprite que viene predeterminado (por su uso frecuente en juegos) en App Inventor. En la columna Propiedades (Editor de Diseño) podremos indicarle el radio, color, velocidad, dirección, etc.

El componente **SpriteImage** es un Sprite que viene representado por una imagen que hemos subido previamente a App Invnetor. Constituyen la base de los juegos gráficos basados en movimiento.

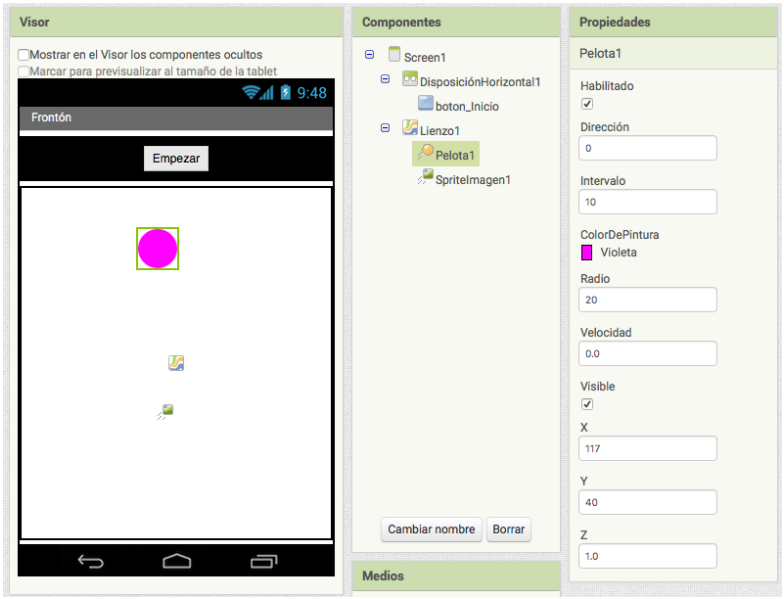
Propiedades de DisposiciónHorizontal1:
- **DispHorizontal**: *Centro*
- **DispVertical**: *Centro*
- **Alto**: *10 por ciento*
- **Ancho**: *Ajustar al contenedor*

Propiedades de Botón1:
Renombramos **Botón1** por *Boton_Inicio*
- **Texto**: *Empezar*

Propiedades de Lienzo1:
- **Alto**: *85 por ciento*
- **Ancho**: *Ajustar al contenedor…*
- **ColorDeFondo:** *Negro*

Propiedades de Pelota1:
- **Radio**: *20*
- **ColorDePintura**: *Violeta*
- **Intervalo**: 10 (milisegundos)

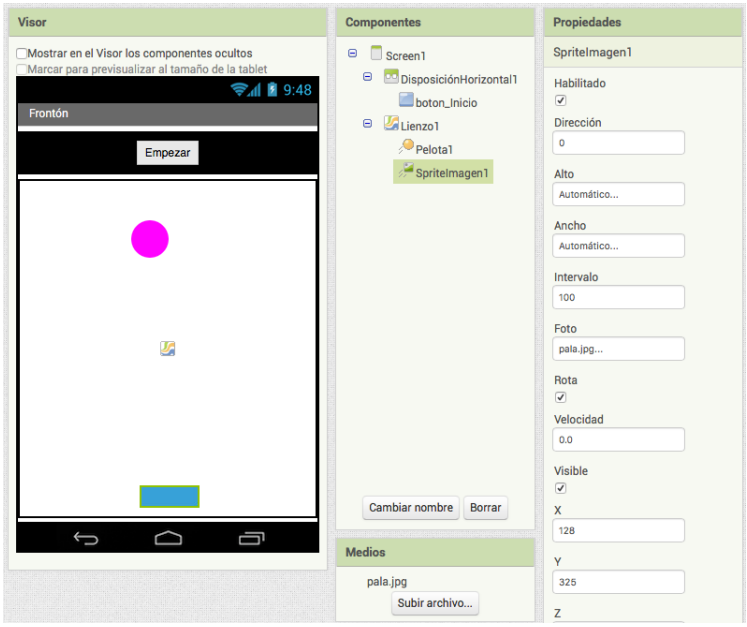
El **radio** de Pelota indica número de píxeles. Un **intervalo** de 10 (milisegundos) y una **velocidad** de 5 (píxeles) indica que cada 10 milisegundos la pelota se moverá 5 píxeles (en la dirección indicada)
Dirección es un ángulo que viene expresado en grados (sexagesimales).
0 grados indica la derecha. 90 grados, arriba. 180, a la izquierda. 270, hacia abajo.



Propiedades de SpriteImage:

- **Foto:** Seleccionamos "*pala.jpg*"
- **X:** *137*
- **Y:** *325*

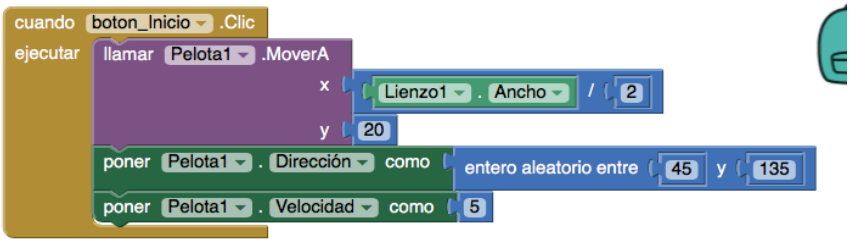
La posición X:0 , Y:0 representa la esquina superior izquierda del lienzo. X:25 Y:50 indica 25 píxeles hacia la derecha y 50 hacia abajo (coordenadas cartesianas).



Paso 4. Programamos el botón Empezar (Inicio)

Conectamos nuestro dispositivo móvil con App Inventor para ir comprobando el funcionamiento de la aplicación.

Dentro del **Editor de Bloques** (botón **Bloques**) seleccionamos cuando *boton_inico.clic* del componente *boton_inicio*.
Añadimos en su interior la etiqueta llamar "*Pelota.MoverA*" para situar el Sprite "Pelota1" siempre en la misma posición al iniciar la partida. La posición horizontal (X) la establecemos a la mitad del ancho del lienzo. Para ello necesitamos las etiquetas dividir y numérica ("2") que se encuentran en el apartado **Matemáticas** (Integrados). Para la posición vertical (Y) seleccionamos una etiqueta numérica escribiendo el valor "20" es su interior.
Nota. Haciendo clic con el botón derecho del ratón sobre cualquier etiqueta nos permite "Duplicarla", lo que nos facilitará la escritura de código.
Establecemos para el Sprite "Pelota1" una velocidad igual a"5" y una dirección aleatoria entre 45º y 135º con las etiquetas "poner Pelota1. Dirección" y "poner Pelota1.Velocidad". El bloque quedaría como se muestra en la imagen.

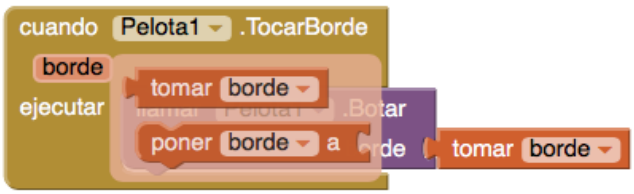


Si pulsamos el botón **Empezar** de nuestro dispositivo móvil observamos que el Sprite "Pelota" se mueve hacia la parte superior del Lienzo y se desliza hasta alcanzar una de las esquinas.

Nuestro siguiente paso será conseguir que la bola rebote cuando toque los bordes.

Paso 5. Rebote de la Pelota

Al hacer clic sobre el componente "Pelota1" nos aparecerá la etiqueta "*cuando Pelota1.TocarBorde*" que deberemos seleccionar. En su interior colocamos la etiqueta "*llamar Pelota1.Botar*" con la etiqueta "tomar borde" (se obtiene al pasar el cursor sobre la palabra "Borde" de la etiqueta ".TocarBorde") como se indica en la figura.



De esta forma conseguimos un movimiento continuo de la pelota.

Debemos establecer ahora el manejo y control de la "pala".

Paso 6. Arrastre de la Pala

Como se indicó anteriormente se pueden definir diferentes mecanismos para el manejo de la "pala". En nuestro caso utilizaremos la etiqueta "*cuando SpriteImage1.Arrastrado*". En su interior agregaremos la etiqueta "llamar SpriteUmage:MoverA", indicando como valor para X: XActual (etiqueta "tomar XActual") y para Y: la etiqueta numérica "325" (valor establecido durante el diseño). De esta manera, cuando arrastremos con el dedo la "pala" sobre el lienzo, está se moverá sólo en la dirección horizontal hasta la posición actual de nuestro dedo.

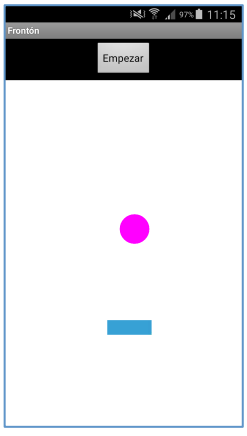
Paso 7. Colisión Pelota-Pala

Deseamos que la Pelota rebote en la Pala cuando colisione con ella. Para que tenga este comportamiento debemos seleccionar la etiqueta "cuando Pelota1:EnColisionCon" del componente "Pelota1", y en su interior colocar la etiqueta "poner Pelota1.Dirección como" y añadimos "entero aleatorio entre 45º y 135º. Es decir, cambiando la dirección del Sprite Pelota conseguimos el efecto rebote.

Comprobamos el funcionamiento en nuestro dispositivo móvil.

Paso 8. Generamos .apk y guardamos

Finalmente, **generamos** el archivo .apk y los **guardamos** en el ordenador o bien **generamos** el *código QR* para instalarlo en nuestro dispositivo móvil. Antes de realizar esta operación podemos añadir una imagen como icono de nuestra aplicación.



Actividad 4

Es evidente que la aplicación que hemos desarrollado admite muchas mejoras. Podemos ponerle sonido (para los rebotes), contador de puntos, botón de Pausa y final de partida

En esta actividad nos planteamos programar un final de partida. Cuando la bola toque la parte inferior de la pantalla el juego se detendrá.

Para proporcionar esta nueva función solo debemos modificar la etiqueta "*cuando Pelota1. TocarBorde*" agregando la etiqueta "*Si entones*" del bloque de **Control**.

App Inventor le asigna a cada borde un número (se puede consultar en la ayuda integrada) siendo +1 para el borde superior, -1 para el inferior, +3 para el derecho y -3 para el izquierdo.

Nosotros deseamos que cuando la bola rebase a la Pala y toque el borde inferior el juego se detenga.

Es decir, **Si** Pelota1 toca borde = -1 **entonces** "poner Pelota1.Velocidad = 0". El bloque quedaría así: