Capítulo 4

Una primera aplicación con App Inventor

Contenido

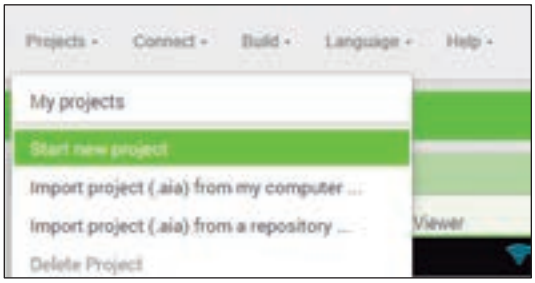
1. Introducción
2. Crear el proyecto
3. Elegir los componentes de la aplicación, añadirlos y establecer sus propiedades
4. Implementar la lógica de la aplicación desde el editor de bloques
5. Probar la aplicación
6. Empaquetar la aplicación
7. Resumen

1. Introducción

Ya se han adquirido los conocimientos mínimos y necesarios para desenvolverse por *MIT App Inventor 2.* A lo largo de esta unidad, se va a realizar una primera aplicación. Se trata de una aplicación básica y sencilla con la que se comprobará el flujo de desarrollo que se sigue en la mayoría de las aplicaciones que uno se disponga a realizar. En este caso, la tarea va a consistir en crear una pequeña aplicación que señale el movimiento del dedo por la pantalla con el objetivo de unir una serie de puntos para descubrir una imagen. El trazado se podrá realizar en diferentes colores y se podrá además limpiar la pantalla cuando se considere oportuno para comenzar de nuevo. Por último, se aprenderá a comprobar el resultado de los ensayos a través de un dispositivo *Android* real o mediante el emulador.

2. Crear el proyecto

El primer paso para el desarrollo de la aplicación será la creación del proyecto. Para este primer paso se debe estar identificado en la aplicación *MIT App Inventor 2,* tal y como se aprendió en unidades anteriores, y acceder a la pantalla principal de la misma. Acto seguido a la identificación solo se tendrá que hacer clic sobre el menú **Projects** de la barra de la aplicación y seguidamente elegir la opción **Start new projects…**



Crear un nuevo proyecto

Recuerde

Desde la opción My projects, ubicada en la parte derecha de la barra superior de la aplicación, se podrá acceder a todos los proyectos. Desde aquí se podrá crear un proyecto nuevo a través de la opción New Project que aparece sobre la lista de proyectos.

El siguiente paso será elegir el nombre para el proyecto. Este nombre debe ser una única palabra compuesta de letras, números o guiones bajos, sin espacios ni caracteres especiales, como signos de puntuación o letras con tilde, entre otros. Desde aquí se ha elegido como nombre *unidad4primeraApp*. Introdúzcase y hágase clic en el botón **OK.**

Nombre del proyecto

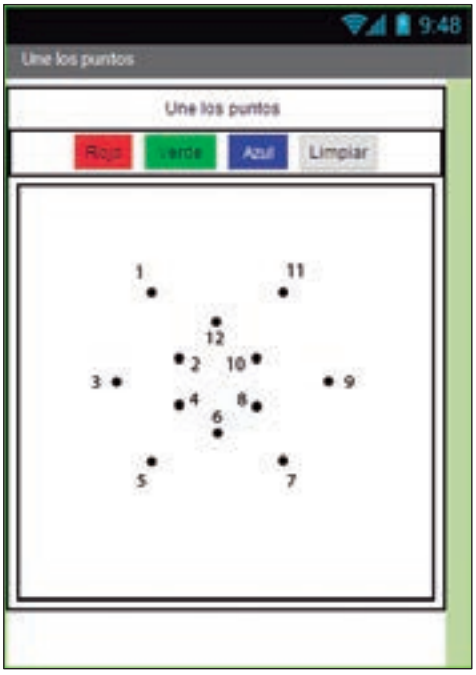
Actividades

1. Cree dos proyectos de prueba de cada una de las formas que puede hacerse. Nómbrelos prueba1 y prueba2.

3. Elegir los componentes de la aplicación, añadirlos y establecer sus propiedades

Después de crear el proyecto y otorgarle un nombre, se va a ser direccionado al *diseñador* de *MIT App Inventor 2* para comenzar con el diseño de la interfaz gráfica. Desde aquí se podrán elegir los componentes que determinarán el aspecto que tendrá la aplicación.

En la siguiente imagen se podrá observar una previsualización, en el panel visor del diseñador, de la interfaz de la primera aplicación se que va a desarrollar con *MIT App Inventor.*



Previsualización de la interfaz gráfica

Entre los componentes que se pueden observar, el primero que se añadirá al visor será **VerticalArrangement.** Este componente, que se encontrará en la categoría *Layout* de la paleta de componentes, será el que organice de forma vertical a los demás componentes. Las propiedades que debe modificar del

componente **VerticalArrangement** son las relativas al tamaño, **Width** y **Height** que establecerá a **Fill parent,** con lo que el componente rellenará el espacio hasta completar el tamaño del componente padre, que en este caso es **Screen.** Al establecer las propiedades **Width** y **Height** a **Fill Parent** se habilitarán las propiedades **AlignHorizontal** y **AlignVertical,** que inicialmente se encontraban deshabilitadas. Establecer el valor **Center** para la correspondiente a la alineación horizontal.

Propiedades modificadas del componente ***VerticalArrangement***

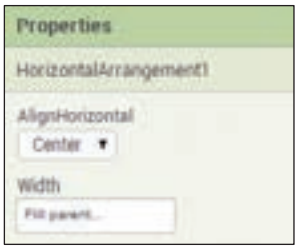
Obsérvese que la interfaz se estructura claramente en tres apartados. El primero de ellos es el título. Añádase un componente **Label** para establecer un título para la aplicación. Las propiedades que se deberán establecer para este componente son **Text,** para establecer el texto del título, **TextAlignment** a **Center,** para alinear el texto al centro, y el ancho, que se establece mediante la propiedad **Width** a **Fill parent** para que ocupe todo el ancho. Sin embargo, la propiedad **Height,** correspondiente a la altura, se dejará **Automatic** para que se adapte al tamaño del texto, que por defecto es 14.0.

Propiedades modificadas del componente ***Label***

Recuerde

Para añadir los diferentes componentes que integran la interfaz gráfica de la aplicación se deberán arrastrar con el ratón desde la paleta de componentes al visor. Posteriormente se deberán definir sus propiedades en función de los requisitos de la aplicación.

El segundo apartado se compone de los botones correspondientes a los diferentes colores que pueden tomar los trazos, además de otro que limpie de la pantalla los trazos actuales y permita comenzar de nuevo el dibujo. Estos componentes **Button** se situarán dentro de otro componente dedicado a la disposición de pantalla, como es **HorizontalArrangement,** que los ordenará de manera horizontal en el ancho de la pantalla. Entre las propiedades que se deben personalizar se encuentra **AlignHorizontal,** que se establecerá a **Center** para que los botones se coloquen alineados en el centro, y **Width** se fijará a **Fill parent** para que los botones, a la vez que alineados al centro, ocupen el ancho de la pantalla.



Propiedades modificadas del componente ***HorizontalArrangement***

Actividades

2. ¿Qué ocurrirá si en lugar de elegir el componente HorizontalArrangement para la disposición de los botones elige VerticalArrangement?

En líneas generales, cada botón deberá definirse con unas propiedades similares a las de los demás, pero con la salvedad de tener algunas propias, como son el color de fondo del botón, el texto y el color del texto, es decir, las propiedades **BackgroundColor, Text** y **TextColor** respectivamente. Adáptense en función de los colores establecidos para cada botón. En la siguiente imagen se observarán las propiedades correspondientes al botón de color azul.

Propiedades modificadas del componente ***Button***

En última instancia, se observa la imagen desde la que el usuario de la aplicación comenzará a realizar los trazos para unir los puntos y formar la figura. Esta imagen se establece dentro de un componente **Canvas** que permite dejar marcados los trazos que realice el usuario con el dedo sobre la pantalla del dispositivo y será el idóneo para indicar el camino que determinará la unión de los puntos de la figura de la aplicación. Las propiedades del componente **Canvas** que se tienen que establecer son, por un lado, **BackgroundImage,** desde la que cargará la imagen sobre la que se dibujarán los trazos, y por otro, las correspondientes al tamaño, **Width** y **Height.** Para esta aplicación se ha utilizado una imagen de 300x300 píxeles, con lo que el valor de estas propiedades se podrá establecer a **Automatic.** Sin embargo, si se utiliza una imagen con un tamaño mayor se deberá ajustar para que quede adaptada y no se produzca ningún desbordamiento en la pantalla.

Propiedades modificadas del componente ***Canvas***

Con las indicaciones realizadas en este apartado se debe ser capaz de reproducir fielmente en el proyecto la interfaz gráfica de la aplicación que se está diseñando a modo de ejemplo para que se establezcan las bases del diseño de aplicaciones *Android* mediante *MIT App Inventor 2*. Se recomienda no continuar hasta tener claros los conceptos expuestos aquí a lo largo del ejemplo.

Actividades

3. Pruebe a insertar los valores 1200 y 1200 para las propiedades Width y Height del componente Canvas. ¿Qué ocurre?

4. Implementar la lógica de la aplicación desde el editor de bloques

Después de diseñar la interfaz gráfica de la aplicación, ya se dispone de una pantalla compuesta de una serie de componentes que completan su aspecto, pero que no responden a ninguna acción del usuario. Por el momento, la aplicación y sus componentes no están preparados para contestar a los diferentes eventos que el usuario ocasione a través de sus acciones, como por ejemplo tocar alguno de estos componentes reflejados en la pantalla del dispositivo.

El siguiente paso al diseño de la interfaz será analizar los requisitos de nuestra aplicación y diseñar a través del editor de bloques la lógica necesaria para que esta realice las acciones que de ella se esperan. Mediante la correcta elección y combinación de los diferentes bloques se podrá conseguir el funcionamiento que se desea para la aplicación.

Se comenzará analizando los requisitos de la aplicación. Esta se compone de una serie de botones. Tres de ellos serán los encargados de establecer el color del trazado del dibujo, aunque inicialmente el color será amarillo. Un cuarto botón tendrá como labor limpiar de la pantalla los trazos dibujados. Por otro lado, se tiene un componente **Canvas** que muestra la imagen y que tiene por tarea señalar el recorrido por el que el usuario pasa su dedo sobre la imagen. Acto seguido, se van a implementar estos requisitos sobre el editor de bloques añadiendo los bloques necesarios.

En primera instancia, se define el color inicial del trazado a amarillo. Este requisito se establece al iniciar la aplicación, con lo que hay que detectar cuándo esta es iniciada. Para ello, habrá que ayudarse de un bloque muy importante, dado que es el encargado de establecer ciertas condiciones para su aplicación en el momento que esta sea iniciada. Este bloque en cuestión es **WHEN Screen1.Initialize DO acciones** y se podrá encontrar seleccionando el componente **Screen1** que se corresponde con la pantalla de la aplicación desde el panel **Blocks,** en el apartado de los componentes que se han añadido. El significado de este bloque se podría traducir de la siguiente forma: "cuando la aplicación sea iniciada, hacer las acciones indicadas". Este bloque controla lo que se conoce como **evento** y que se dominará a lo largo de este apartado.

Sabía que...

Cada vez que se seleccione un componente desde el panel Blocks aparecerá a la derecha el listado de bloques asociados a este y solo con hacer clic sobre uno de ellos se podrá colocar sobre el visor.

Para establecer el color inicial amarillo se deberá conectar al bloque **WHEN Screen1.Initialize DO** la acción a realizar, así que hay que seleccionar ahora el componente **DrawingCanvas** ubicado bajo la estructura de bloques contenidos en **Screen1.** Seguidamente eligir el bloque correspondiente a la propiedad **PaintColor.** Este bloque establece el color del trazado mediante el comando **SET propiedad TO valor,** es decir, establecer la propiedad **PaintColor** del componente **DrawingCanvas** a amarillo, que se determina por el valor conectado. Para elegir un valor se podrá definir un bloque de color que asigne el código representativo de dicho color y que se podrá hacer de la siguiente forma: primero, seleccionar el apartado **Built-In** desde el panel **Blocks;** a continuación, la categoría **Colors;** y, por último, el bloque **Yellow,** correspondiente al color amarillo. Para asignar el color inicial del trazado a amarillo solo se deberá conectar este bloque correspondiente al color amarillo que se acaba de definir al bloque que determina la propiedad **PaintColor** del componente **DrawingCanvas.** En la siguiente imagen se podrá comprobar el resultado de las indicaciones dadas.

*Asignar el color amarillo a la propiedad **PaintColor** del componente **DrawingCanvas** al iniciar la aplicación*

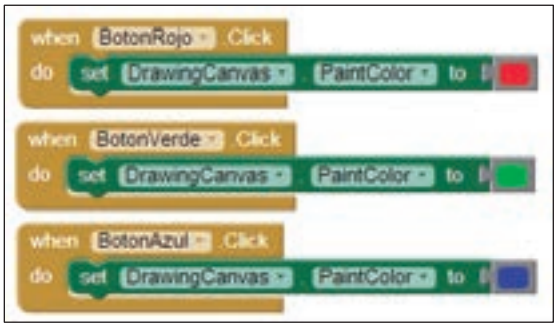
Como se ha dicho, inicialmente los trazos realizados sobre el componente **Canvas** serán de color amarillo pero en función de la elección de alguno de los botones de color se podrá modificar el color del trazo. Para ello la aplicación deberá detectar la acción de pulsar un botón y definir el color en función del botón elegido. Este hecho es lo que se conoce como **evento.**

Recuerde

Un *evento* es una acción o suceso provocada por el usuario, como por ejemplo pulsar un botón.

La aplicación debe estar preparada a niveles de desarrollo para ser capaz de responder a dichos eventos o, lo que es lo mismo, reaccionar a ellos. ¿Y de qué manera puede una aplicación reaccionar a un evento? Pues en este caso reaccionará asignando el color del botón pulsado al del trazado. La aplicación podrá reaccionar en función de las diferentes situaciones en las que se pueda encontrar y sus requisitos.

Para captar la pulsación de un botón determinado, es decir, detectar que se ha producido el evento, "pulsar el botón rojo", por ejemplo, se deberá añadir un bloque específico que se encargará de estar alerta y detectar dicha elección del usuario. Este evento se denomina **Click** y trabaja de la siguiente forma: **WHEN acción DO respuesta,** o lo que es lo mismo, cuando se pulse un botón hacer que se modifique el color del trazado. Para añadirlo desde el editor de bloques se ha de seleccionar el componente **BotonRojo** que se encuentra en el panel **Blocks.** Acto seguido se debe elegir el bloque **WHEN BotonRojo.Click DO.** Ya se tiene definido el evento para el botón rojo. Esto se ha de hacer para cada botón de color. Ahora queda especificar la respuesta al mismo. Al igual que se hiciera para establecer a amarillo el color inicial del trazo, se realizarán ahora los mismos pasos para cada color conectándolos a su correspondiente evento **Click.** El resultado se podrá verificar en la siguiente imagen.



*Capturar evento **Click** y definir colores sobre el trazado*

El botón **Limpiar** es un caso especial dado que no realizará la misma acción que los otros. En esta ocasión, se trata de detectar la pulsación de este botón y realizar un borrado de los trazos existentes en el componente **Canvas.** De igual forma, se hará uso del bloque **WHEN acción DO respuesta** que, personalizado

a esta situación, será **WHEN BotonLimpiar.Click DO respuesta.** La respuesta en este caso se obtendrá a través del **método Clear** aplicado al componente **DrawingCanvas.**

Recuerde

Un método es una sentencia o grupo de ellas que realizan una acción.

Este método borra lo que haya dibujado sobre el **Canvas** pero no borra, en cuanto al fondo se refiere, ni el color ni la imagen que hayan sido prestablecidos para tal efecto. Este método tiene la nomenclatura **CALL DrawingCanvas. Clear** y se puede encontrar seleccionando el componente **DrawingCanvas.** Los bloques de métodos se traducen de la siguiente forma: llamar al método del componente para realizar la acción. En algunos casos se podrán encontrar métodos que reciban parámetros y que operen con ellos para realizar su cometido o devolver algún valor que en ellos se calcule. Obsérvese la imagen siguiente para comprobar el resultado.

*Método **Clear** aplicado al componente **DrawingCanvas** en respuesta al evento **Click** del botón **BotonLimpiar***

Por último, se va a implementar la lógica necesaria para que se dibuje el trazado sobre el componente **Canvas** al paso del dedo del usuario por la pantalla. El evento encargado de estar alerta cuando se produce la acción de deslizar el dedo por la pantalla es **Dragged** y se aplica sobre el componente **DrawingCanvas.** Para añadirlo desde el editor de bloques, seleccionar el componente **DrawingCanvas.** Finalmente, elegir el bloque **WHEN DrawingCanvas.Dragged DO** que se traduce

de la siguiente forma: cuando se perciba un movimiento de arrastre dibujarlo sobre el componente Canvas. Obsérvese que el bloque que se acaba de añadir contiene una serie de valores predefinidos denominados **argumentos** o **atributos** a la entrada del mismo. Estos valores están establecidos por defecto y no deben modificarse, dado que llevan el control sobre la posición inicial del movimiento de arrastre del dedo, así como de la trayectoria que toma.

Recuerde

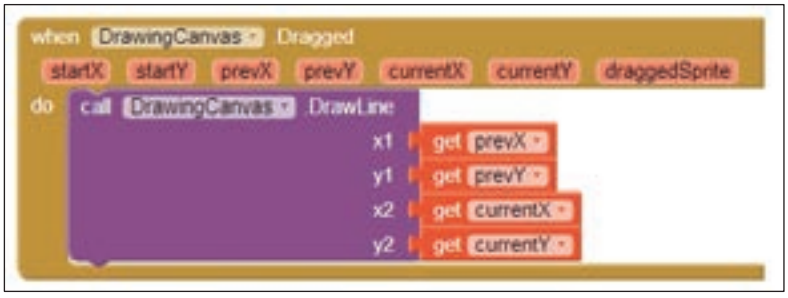
Un atributo es una propiedad que diferencia a un componente y que determina su comportamiento, estado o apariencia.

Además, se deberá completar la lógica de la acción que se realizará cuando se detecte el evento **Dragged.** Para ello, se añadirá una llamada al método **DrawLine** sobre el componente **DrawingCanvas** que encontrará al seleccionar dicho componente desde el panel **Blocks.** Una vez situado dentro de la zona de acción del evento **Dragged,** completará los argumentos **x1, x2, y1** e **y2** respectivamente. Estos determinarán los puntos de inicio y fin del trayecto a dibujar. Para definirlos se deberán conectar a los puntos **x1** e **y1** los valores previos a estos que se obtendrán de los propios argumentos del bloque correspondiente al evento **Dragged, prevX** y **prevY** respectivamente. Conectar dichos valores es sencillo, solo se ha de dejar el puntero del ratón sobre cada uno de ellos y aparecerán dos bloques; uno para obtener el valor del argumento elegido y otro para establecerlo al valor que se indique a continuación. En este caso, se hará clic sobre el bloque **get prevX** y **get prevY** respectivamente, seguidamente conéctense a **x1** e **y1.** En cuanto a los puntos **x2** e **y2, se** hará lo propio pero con los valores actuales del movimiento de arrastre, que se obtendrá desde los parámetros **currentX** y **currentY** respectivamente.

Actividades

4. ¿Qué método utilizaremos para establecer el valor de una propiedad nada más lanzar la aplicación?

En la siguiente imagen se puede comprobar el resultado de realizar el proceso descrito para dibujar sobre el componente **Canvas.**



*Método **DrawLine** aplicado al componente **DrawingCanvas** en respuesta al evento **Dragged** sobre el propio **Canvas***

Nota

Se ha de tener en cuenta que cuando se habla de deslizar el dedo por la pantalla o tocar algún botón se hace referencia al uso de la aplicación en un dispositivo Android real y no al emulador. En el caso del emulador, estas acciones o gestos se realizarán mediante clics y movimientos de ratón.

En este momento, ya se puede afirmar que está desarrollada la interfaz de la aplicación junto con la lógica que determinará el comportamiento de la

misma. En los apartados siguientes se aprenderá a probar la aplicación que se acaba de desarrollar.

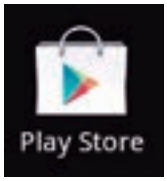
5. Probar la aplicación

Al término de los desarrollos con *MIT App Inventor* y antes de dar la aplicación como válida, se deberá someter a revisión, y qué mejor modo de hacerlo que ejecutarla en un entorno *Android.* Para ello, se puede desplegar la aplicación en un emulador, así como en un dispositivo real. La aplicación no se transferirá al emulador durante esta fase de revisión, sino que realizará una emulación de la misma.

5.1. Wifi

En este apartado se van a explicar los pasos a seguir para probar las aplicaciones a través de *AI Companion,* aplicación que se puede descargar de manera gratuita desde *Google Play,* siendo esta una manera rápida e innovadora de probar las aplicaciones. A continuación, se explicará cómo descargar la aplicación y ponerla en marcha.

Desde el dispositivo *Android,* comenzar abriendo la aplicación *Google Play.* Obsérvese en la siguiente imagen el icono de su acceso directo. Solo deberá tocarse y esta se abrirá.

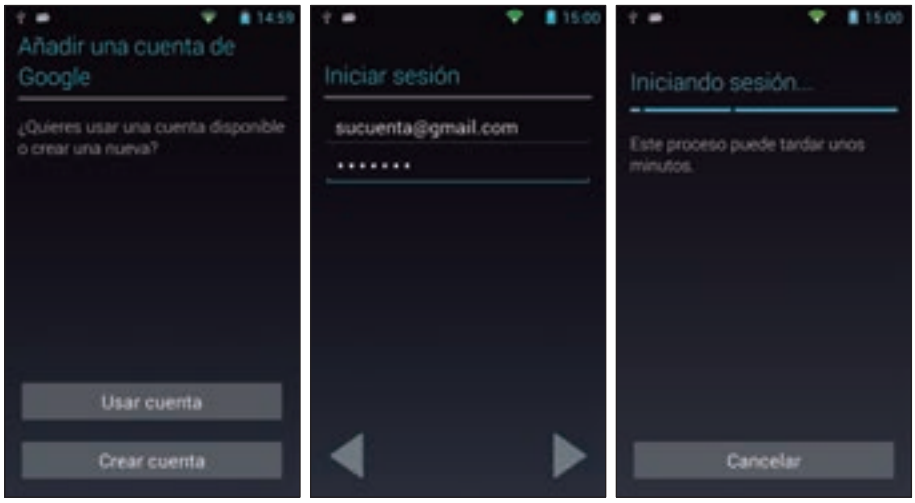


Icono de Google Play

Una vez abierto se observará que, para acceder a *Google Play*, es necesario registrar una cuenta de usuario de Google. Si no se posee una, se podrá crear y

añadirla en este momento. Al finalizar el asistente, se podrá acceder a Google Play.

Si ya se dispone de una cuenta, solo deberá añadirse introduciendo el usuario y contraseña de la misma. El proceso se muestra en las siguientes imágenes.



Iniciar sesión con una cuenta de Google a Google Play

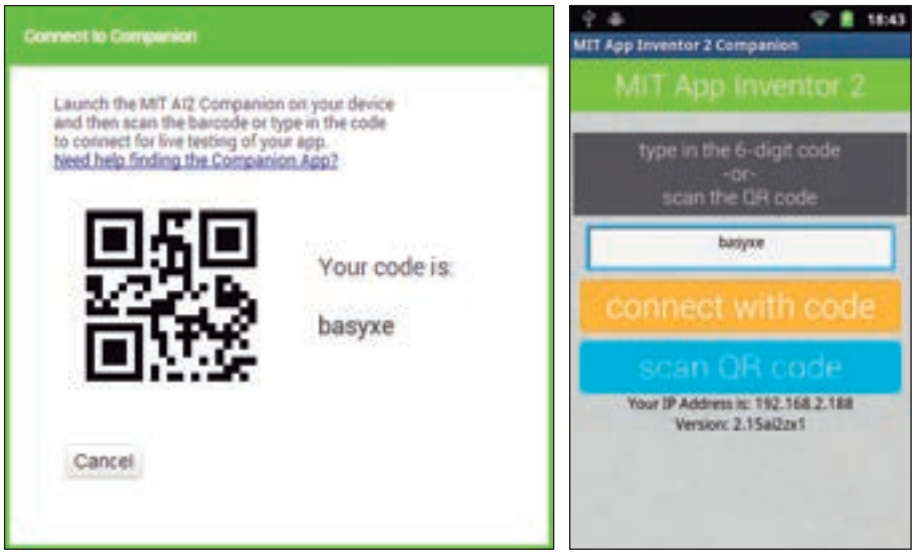
Una vez abierto *Google Play,* obsérvese que dispone de un buscador para facilitar la localización de contenidos a los usuarios y un filtro de contenido, ya que además de aplicaciones proporciona juegos, películas, música y libros.

Google Play

Pulsar sobre la lupa que se localiza en la esquina superior derecha e introducir el nombre de la aplicación, *MIT AI2 Companion.* Aparecerá como resultado la aplicación que se desea instalar en el dispositivo *Android.* Selecciónese y, a continuación, tocar sobre el botón **Instalar** para que comience la descarga e instalación.

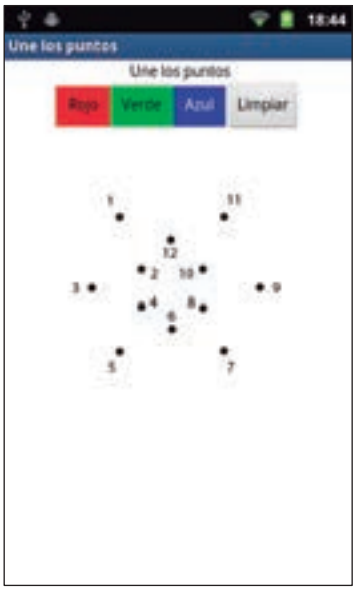
Instalación MIT AI2 Companion

Al finalizar la descarga e instalación se ofrecerá la oportunidad de abrir por primera vez dicha aplicación. Esto se debe hacer seleccionando el botón **Abrir.** Una vez abierta, se podrá comprobar que se dispone de dos posibilidades para conectar la aplicación que se ha desarrollado con *MIT App Inventor 2* y su dispositivo móvil *Android.* La primera de ellas, se realizará introduciendo el código que aparece en *MIT App Inventor 2* al seleccionar el menú *Connect* y la opción *AI Companion*. La segunda de ellas, leerá el código QR que de igual forma aparece en la misma opción antes mencionada. En la siguiente imagen se podrá ver un ejemplo de un código suministrado por *MIT App Inventor 2* y la pantalla de la aplicación *MIT AI Companion.*



Conectar con MIT AI Companion

En el dispositivo *Android* se desplegará automáticamente la aplicación desarrollada. Si esta requiere ser modificada se podrá hacer mientras que esté desplegada, aplicándose los cambios sobre ella sobre la marcha. En la siguiente imagen se ve cómo quedará la aplicación desarrollada a lo largo del capítulo en un dispositivo *Android.*



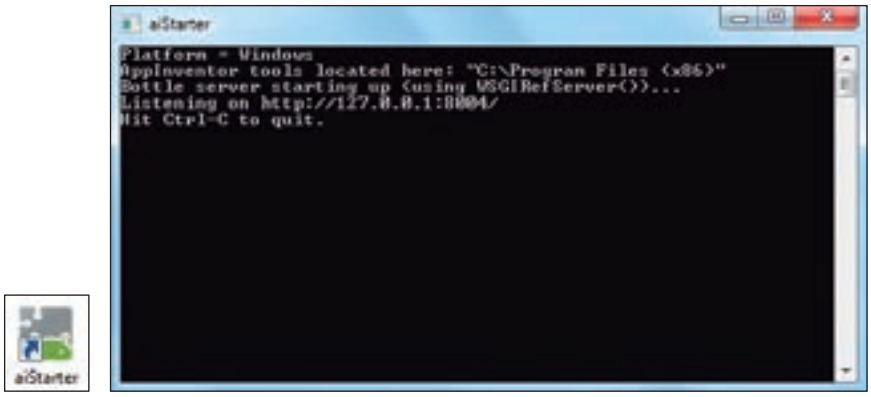
Conectar con AI Companion

Actividades

5. Discuta la veracidad de la siguiente frase. Es obligatorio descargar la aplicación *AI Companion* para probar cualquier aplicación.

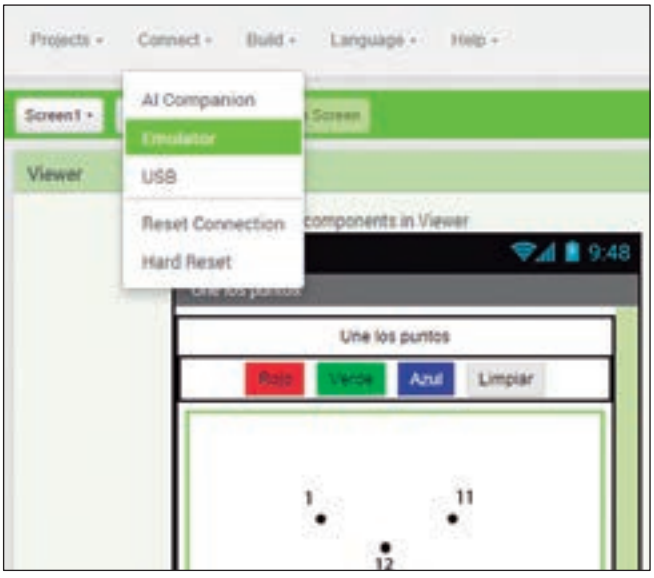
5.2. El emulador

Antes de lanzar el emulador habrá que descargarse e instalar el *software* de instalación de *App Inventor*, *aiStarter*. Para ello, acceder a la dirección http://appinventor.mit.edu/explore/ai2/setup-emulator.html, elegir su sistema operativo en el paso 1 y hacer clic sobre **Download the installer.** En unos instantes comenzará la descarga. A su término hacer doble clic sobre el fichero que se acaba de descargar y completar la instalación. Al finalizar la instalación se coloca un acceso directo a *aiStarter* en el escritorio, que se ha de ejecutar antes de lanzar el emulador desde *MIT App Inventor 2*.



aiStarter

Tras ejecutar *aiStarter*, se deberá hacer clic sobre el menú **Connect,** que se encuentra en la barra de opciones superior de la ventana de la aplicación de *MIT App Inventor 2* y, seguidamente, elegir la opción **Emulator.** Aparecerá un aviso que notificará que se está iniciando el emulador y pedirá paciencia. Al cabo de unos minutos, el emulador aparecerá a modo de ventana con el aspecto de la siguiente imagen.



Desplegar en emulador

Una vez iniciado el emulador, se deberá esperar hasta que se cargue la aplicación correspondiente al proyecto que se tenga abierto en *App Inventor.* Es posible que durante el despliegue en el emulador aparezca un mensaje comunicando que esa versión de *MIT AI2 Companion* se encuentra desactualizada y ofrezca actualizarla. Se puede realizar el proceso. Tras la instalación, se deberá abrir dicha aplicación para continuar con el despliegue. Este proceso podrá tardar unos minutos. Después de desplegarse podrán revisarse y retocarse los puntos que no parezcan correctos en *MIT App Inventor* y los cambios se irán viendo reflejados sobre el emulador.

Emulador

5.3. USB

En caso de no poder hacer uso de una red wifi para probar la aplicación se dispone de un tercer método para ello. Quizás es el más complejo de los tres y por ello se desaconseja desde la página oficial de *MIT App Inventor.* Aun así, se exponen a continuación los pasos a seguir para su configuración y uso.

El primer paso necesario para conectar el dispositivo a través del cable USB es instalar el *software* de *App Inventor aiStarter*, al igual que se hiciera para el emulador. Además, desde el dispositivo *Android* se deberá tener instalada la app *MIT AI2 Companion,* al igual que se requirió para el primero de los métodos de prueba mediante la red wifi.

Con ambas aplicaciones instaladas, en el equipo y dispositivo *Android* respectivamente, ejecutar *aiStarter*. Esto deberá hacerse cada vez que se quieran realizar las pruebas de la aplicación en un dispositivo *Android* mediante el cable USB.

Seguidamente, desde el dispositivo *Android* se deberá activar la opción *Depuración USB,* que se localiza en los ajustes del sistema y desde las **Opciones de desarrollo.**

Ahora se deberá conectar el dispositivo *Android* mediante el cable USB al ordenador. Si el equipo no dispone de los *drivers* necesarios para que la conexión se haga de forma correcta, se deberán instalar los *drivers* que proporcione el fabricante de su dispositivo *Android.*

Nota

Para obtener ayuda sobre la instalación de los drivers del dispositivo en el ordenador, se puede consultar la siguiente dirección:

<http://appinventor.mit.edu/explore/ai2/connect-USB-win.html>

Por último, desde la aplicación *MIT App Inventor 2,* hacer clic en la opción **USB,** que se encontrará en el menú **Connect** de la barra de opciones superior de la ventana de la aplicación.

Nota

Si tiene problemas durante la conexión a través del cable USB, consultar la siguiente dirección web para obtener ayuda y orientación:

<http://appinventor.mit.edu/explore/ai2/setup-device-usb.html>

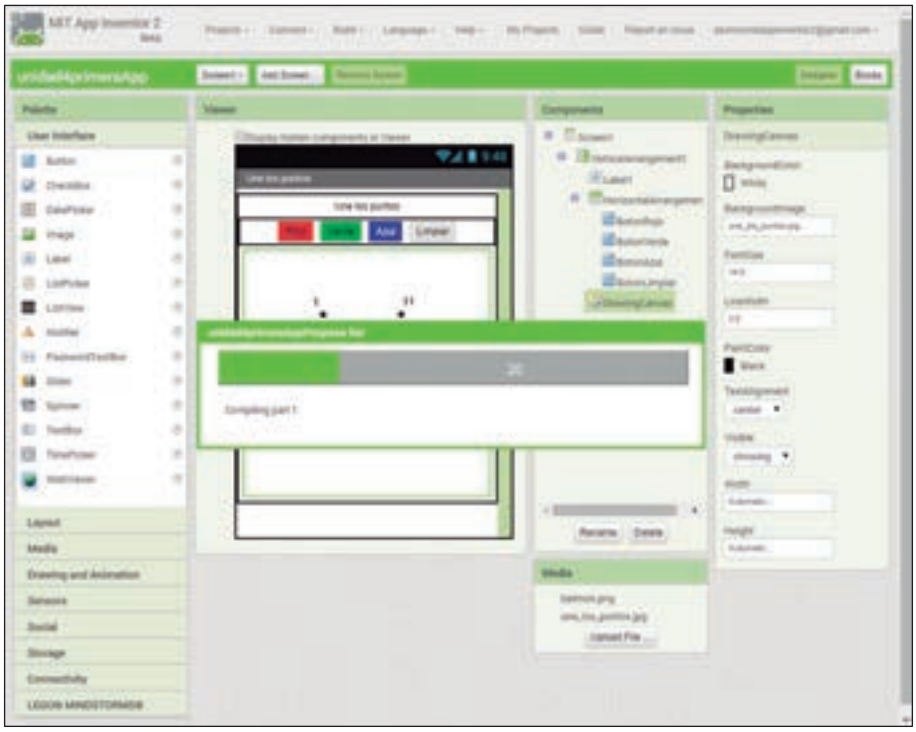
6. Empaquetar la aplicación

El empaquetado de la aplicación consiste en agrupar todo el contenido desarrollado en la aplicación. Entre este se encuentran los componentes de su interfaz gráfica y los generados por los bloques que definen la lógica y comportamiento de la aplicación a través del editor de bloques, así como los elementos multimedia que se hayan podido añadir. Toda esta agrupación de contenido se convierte a código entendible por un dispositivo *Android* en forma de archivo instalador de dicho sistema que se trata de un archivo "apk".

La acción del empaquetado se realiza cuando la aplicación se encuentra a un nivel óptimo y aceptable en cuanto a diseño y operatividad, pudiéndose llevar a cabo a través de las dos formas que ya se conocen. Desde aquí se llevará a cabo el proceso desde ambas opciones con el objetivo de que se conozcan para poder elegir cómo proceder en el futuro.

6.1. App (provide QR code for .apk)

La primera de ellas es mediante la opción *App (provide QR code for .apk)* que se encontrará desde el menú **Build.** Al seleccionar esta opción comenzará la compilación. En la siguiente imagen se podrá comprobar el comienzo del proceso.



Empaquetar aplicación

Al finalizar el proceso de compilación se mostrará el código QR que se deberá escanear desde *MIT AI Companion.*

Empaquetar aplicación

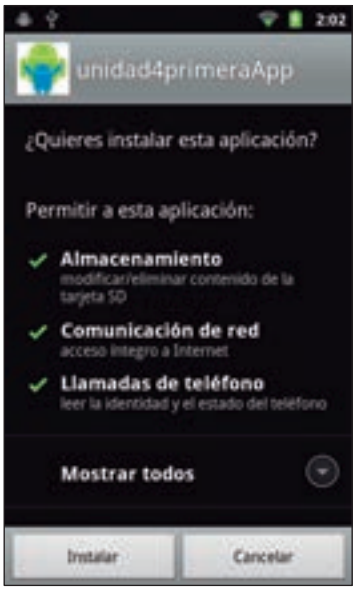
Definición

Compilar
Acción de traducir código escrito en un lenguaje de programación de alto nivel a lenguaje máquina con el fin de que este sea entendido por un equipo informático.

Una vez leído este, aparecerá la dirección correspondiente a la localización del fichero "apk" de la aplicación desarrollada. Obsérvese cómo aparece relleno con la citada dirección el cuadro de entrada de texto de *MIT AI Companion.*

Empaquetar aplicación

Al cabo de unos segundos se ocultará la aplicación *MIT AI Companion* para dar paso al proceso de instalación de la aplicación en el dispositivo *Android,* proceso que se podrá comprobar en la siguiente imagen. La diferencia existente entre conectar y/o desplegar la aplicación con un dispositivo *Android* físico para probarla y empaquetarla e instalarla estriba en que, en el primero de los casos, la aplicación no permanecerá en el dispositivo, mientras que en el segundo de ellos, sí.



Empaquetar aplicación

Además, a través de este método de empaquetado se podrá obtener el fichero instalador "apk" de la aplicación introduciendo la dirección obtenida, después de leer el código QR desde *MIT AI Companion,* en el navegador de internet. Acto seguido comenzará la descarga del fichero en el equipo.

6.2. App (save .apk to my computer)

De otro modo, si se necesita simplemente el fichero de instalación del curso, se podrá obtener eligiendo la segunda de las opciones ofrecidas por el menú **Build,** que es *App (save .apk to my computer)*. Este método consiste en la compilación, empaquetado y descarga de la aplicación a un fichero instalador del curso en formato "apk". Dicho fichero será ubicado en la carpeta de descargas del usuario con el que se esté trabajando en el sistema operativo.

El proceso durará varios segundos y acto seguido comenzará la descarga. Obsérvese en la siguiente imagen el mensaje mostrado al finalizar el proceso.

Empaquetar aplicación

Aplicación práctica

Haga lo necesario para empaquetar los proyectos prueba 1 y prueba 2 que creó anteriormente mediante cada una de las opciones disponibles, además instálelos y ejecute ambas aplicaciones.

SOLUCIÓN

Opción 1:

- Seleccione el menu Build y la opción App (provide QR code for .apk).
- Abra la aplicación AI Companion desde su móvil y escanee el código QR mostrado en la pantalla del ordenador.
- Una vez escaneado el código, aparecerá una dirección en la aplicación AI Companion. Espere unos segundos y comenzará el proceso de instalación.
- Seguidamente, instale la aplicación pulsando sobre el botón Instalar.
- Al terminar la instalación se le propondrá abrirla. Pulse sobre el botón Abrir.

Opción 2:

- Seleccione el menu Build y la opción App (save .apk to my computer).
- El fichero se creará en la carpeta de descargas de su usuario.
- Conecte su dispositivo Android al ordenador y transfiera el fichero instalador apk de su aplicación hasta la memoria del mismo.
- Una vez copiado, búsquelo en su ubicación e instálelo.
- De igual modo que en la opción 1 se le propondrá abrirlo al terminar la instalación.
- Toque sobre el botón Abrir.

7. Resumen

Para crear un proyecto se deberá hacer clic sobre el menú **Projects** de la barra de la aplicación y seguidamente elegir la opción **Start new projects...**

Seguidamente se debe elegir un nombre para el proyecto. Este nombre debe estar compuesto por una única palabra y sin espacios. Después, hacer clic sobre el botón **OK.**

Será direccionado al *diseñador* de *MIT App Inventor 2* para comenzar con el diseño de la interfaz gráfica. Desde aquí, se podrán elegir los componentes que determinarán el aspecto que tendrá la aplicación.

El siguiente paso al diseño de la interfaz será analizar los requisitos de la aplicación y diseñar a través del editor de bloques la lógica necesaria para que esta realice las acciones que de ella se esperan. Mediante la correcta elección y combinación de los diferentes bloques se podrá conseguir el funcionamiento que se desea para la aplicación.

Finalmente, se deberá someter la aplicación a revisión antes de darla por concluida. Para ello se podrá hacer ejecutándola en un entorno *Android* como el que brinda el emulador de *MIT App Inventor 2*. Para lanzarlo se deberá hacer clic sobre el menú **Connect** y seleccionar la opción **Emulator** que se encuentra en la barra de opciones de la aplicación. De otro modo, también se podrá examinar la aplicación desde un dispositivo *Android* real a través de las opciones **AI Companion** y **USB** del mismo menú.

Como último paso del proceso de creación de aplicaciones podrá empaquetarla con el objetivo de agrupar y compilar todo su contenido en un fichero entendible por un sistema Android. Este fichero es el propio instalador de la aplicación y posee extensión *apk*. Podrá completar este paso a través de la lectura de un código QR mediante *MIT AI Companion* o de otro modo descargándolo directamente a su ordenador.