Trabajando con Listas

Presidents Quiz es un juego de preguntas y respuestas sobre los presidentes de los Estados Unidos. Aunque trate sobre los mandatarios de este país, podemos utilizar esta plantilla para crear un juego sobre cualquier tema.

En esta aplicación usaremos **dos** variables **list** donde guardarás los datos, en este caso, las preguntas y las respuestas. También usaremos una variable **index** para determinar dónde se encuentra el usuario dentro del juego.

Vamos a diseñar un juego de preguntas y respuestas donde el usuario pueda pasar a la siguiente cuestión haciendo clic en el **botón Siguiente** y recibir información sobre si la contestación es correcta o no.

¿ Qué aprenderé ?

En esta aplicación aprenderás:

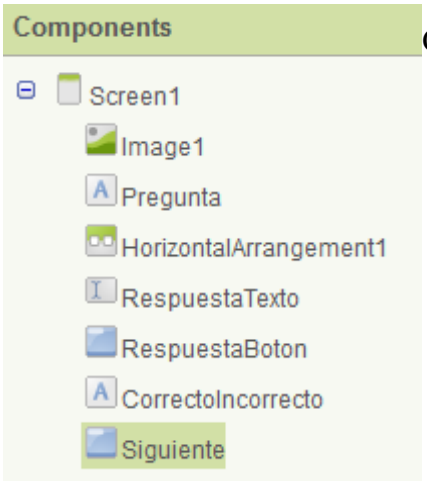
- Definir variables **list** para almacenar las preguntas y las respuestas.
- Utilizar un índice para moverse a través del contenido de una lista. Cuando el usuario toque el botón **Siguiente**, la aplicación muestra la siguiente pregunta.
- Emplear comportamientos condicionales (**if**), los cuales permiten desarrollar ciertas operaciones sólo cuando se cumplen unas determinadas condiciones. Recurriremos al bloque **if** cuando el usuario llegue al final del juego.
- Cambiar imágenes para mostrar una fotografía diferente con cada pregunta.

Diseñar los componentes

Crea un nuevo proyecto llamado **PresidentsQuiz** y cambia el título de la pantalla por **Presidents Quiz**. Para generar la interfaz deberás incorporar las imágenes del proyecto como ya has hecho en otras aplicaciones. Añade los componentes que aparecen en la siguiente tabla:

Tipo de componente	Grupo de Palette	Cómo lo llamaremos	Finalidad
Image	**Basic**	Image1	La imagen que se mostrará junto con las respuestas.
Label	**Basic**	Pregunta	Muestra la pregunta en la pantalla.
HorizontalArrangement	**Screen Arrangement1**	*Horizontal Arrangement1*	Coloca el contenido de la pregunta en la pantalla.
TextBox	**Basic**	*RespuestaTexto*	Cuadro de texto donde el usuario escribirá su respuesta.
Button	**Basic**	*RespuestaBoton*	Botón que utilizará el usuario

Tipo de componente	Grupo de Palette	Cómo lo llamaremos	Finalidad
			para enviar su respuesta.
Label	**Basic**	CorrectoIncorrecto	Muestra en la pantalla **correct!** o **incorrect!**.
Button	**Basic**	Siguiente	Botón para pasar a la siguiente pregunta.



Configura las **propiedades** de los **componentes**:

- Asigna a la propiedad **Image.Picture** el archivo *"roosChurch.gif"*. Es la primera imagen que debería parecer. Ajusta su ancho (**Width**) a Fill parent y su alto (**Height**) a 200.

- Da el valor **Pregunta...**a la propiedad **Pregunta.Text**. Colocaremos la primera pregunta desde **Blocks Editor**.

- Asigna el valor **Entra una respuesta** (escriba su respuesta) a la propiedad **RespuestaTexto.Hint**. Deja en blanco el valor de su **Text**. Colócalo dentro de **HorizontalArrangement1**.

- Cambia el valor de **RespuestaBoton.Text** por **Enviar** y colócalo dentro de **HorizontalArrangement1**.

- Deja en blanco el valor de la propiedad **CorrectoIncorrecto.Text**.

- Cambia la propiedad **Text** del botón Siguiente por **Siguiente**.

De momento habrás obtenido:

Asignar comportamientos a los componentes

- Cuando se abra la aplicación, mostrará la primera pregunta en la pantalla, incluyendo el texto y la imagen.

- Cuando el usuario pulse el botón **Siguiente**, aparecerá en la pantalla la segunda pregunta. Si vuelve a utilizar este botón, surgirá la tercera y así sucesivamente.

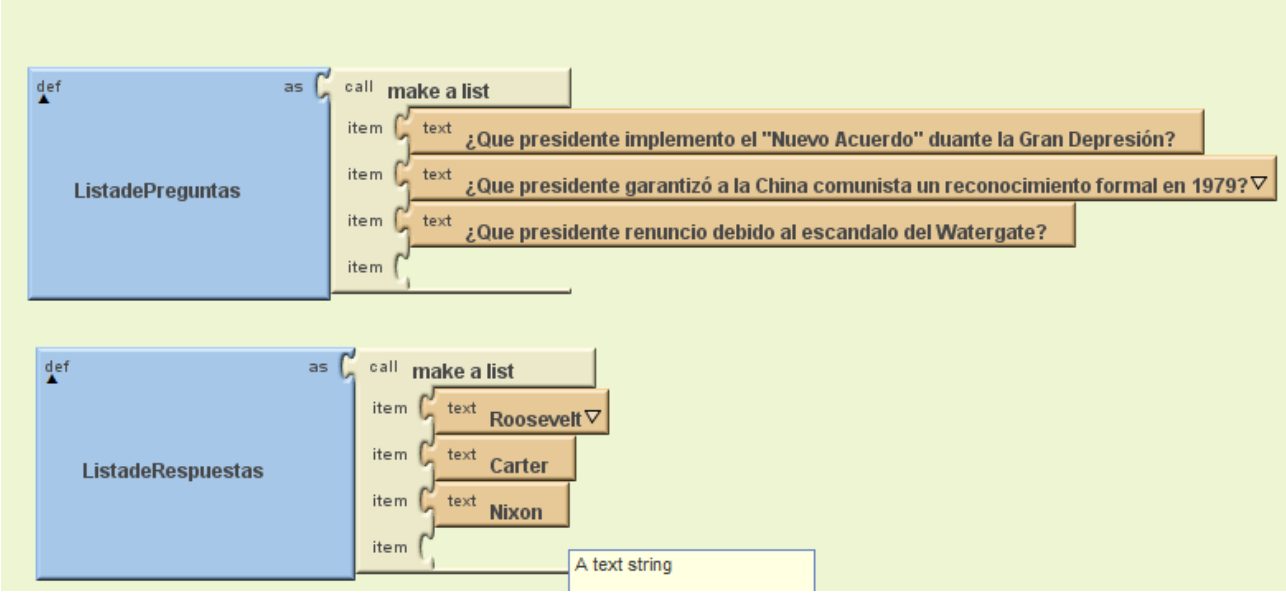
- Cuando el usuario llegue a la última pregunta y presione el botón **Siguiente**, la aplicación volverá a enseñar la primera.

- Cuando el usuario responda a la pregunta, la aplicación le informará si su respuesta ha sido correcta o no.

Para empezar, definiremos dos variables basándonos en los elementos de la tabla anterior. **ListadePreguntas** albergará la lista de las preguntas y **ListadeRespuestas** contendrá la de sus

correspondientes respuestas. La siguiente tabla muestra las dos listas que crearemos en **Blocks Editor.**

Tipo de bloque	Cajón	Finalidad
def variable ("ListadePreguntas")	**Definitions**	Guarda la lista de preguntas. Cambia su nombre por **ListadePreguntas**.
def variable ("ListadeRespuestas")	**Definitions**	Guarda la lista de las respuestas. Cambia su nombre por **ListadeRespuestas**.
make a list	**Lists**	Inserta los elementos de **ListadePreguntas**.
text (tres de ellos)	**Text**	Las preguntas.
make a list	**Lists**	Inserta los elementos de **ListadeRespuestas**.
text (tres de ellos)	**Text**	Las respuestas.



Definir la variable index

La aplicación deberá conocer la posición en la que se encuentra el usuario dentro de la lista de preguntas del concurso. Para ello, definiremos una variable llamada **Indice** que usaremos como

índice con **ListadePreguntas** y **ListadeRespuestas**. La siguiente tabla muestra los bloques necesarios para generar este comportamiento.

Tipo de bloque	Cajón	Finalidad
def variable ("Indice")	**Definitions**	Guarda la posición de la pregunta/respuesta actual.
number (1)	**Math**	Establece el valor inicial de **Indice** a 1 (es la primera pregunta)

Mostrar en pantalla la primera pregunta

Haremos referencia al contenido de la primera ranura del elemento **ListadePreguntas**, sin importar la cuestión que esté allí. De esta forma, los bloques seguirán funcionando aunque cambiemos el contenido almacenado en dicha ranura.

Para seleccionar un elemento en particular de la lista deberemos utilizar el bloque **select list item**, el cual nos pedirá que especifiquemos la lista y el índice del elemento deseado. Es decir, deberemos determinar en qué lista se encuentra dicho elemento y la posición que tiene dentro de ella. Si una tiene 3 elementos, podremos emplear como índice los números 1, 2 o 3. Para este primer comportamiento querremos que, cuando la aplicación se abra, seleccione el primer elemento de **ListadePreguntas** y lo coloque en **Pregunta**. Recurriremos a los bloques de la siguiente tabla:

Tipo de bloque	Cajón	Finalidad
Screen1.Initialize	**Screen1**	Controlador de eventos que se utiliza cuando se abre la aplicación.
set Pregunta.Text to	**Pregunta**	Coloca la primera pregunta en **Pregunta**.
select list item	**Lists**	Selecciona la primera pregunta de **ListadePreguntas**.
Global ListadePreguntas	**My Definitions**	La lista que contiene preguntas.
number (1)	**Math**	Selecciona la primera pregunta utilizando 1 como valor del índice.

Cuando se abra la aplicación, se inicia el evento **Screen1.Initialize**. Como puedes observar en la figura de abajo, se selecciona el primer elemento de la variable **ListadePreguntas** y se coloca en **Pregunta.Text**. Así, cuando se activa el programa, el usuario ve la primera pregunta en la pantalla.

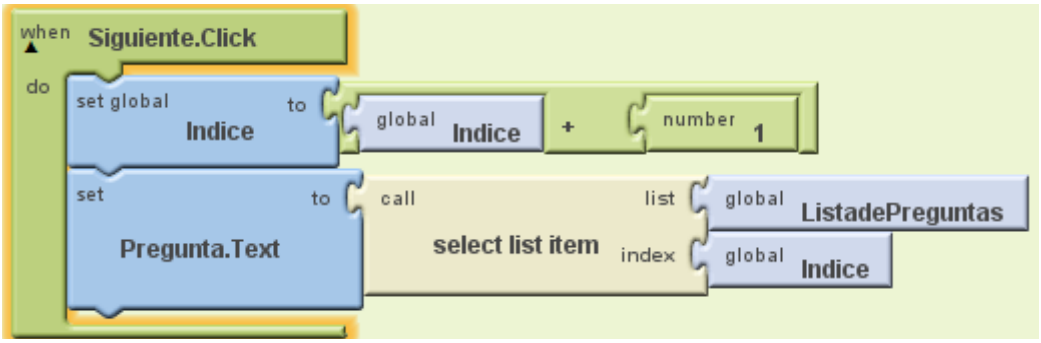
Guarda primero el proyecto con el nombre **PresidentsQuiz**, se generará el fichero **"PresidentsQuiz.apk"** que es el que debes copiar a tu teléfono. Cuando se abra la aplicación deberás ver el primer elemento de la lista **ListadePreguntas**.

PRUEBA!!!!

Moverse por las preguntas

Vamos a programar el comportamiento de **Siguiente**. Ya hemos definido **Indice** para que recuerde la pregunta en la que se encuentra el usuario.

Cuando hagas clic sobre el botón Siguiente, la aplicación incrementará en una unidad el valor de Indice. Es decir, pasará de 1 a 2 o de 2 a 3, etc. Utilizaremos el valor resultante para seleccionar la nueva pregunta que se mostrará en la pantalla.



La primera fila de bloques incrementa el valor de la variable **Indice**. Si su valor es 1, lo cambiará por 2. Si su valor es 2, lo cambiará por 3 y así, sucesivamente. Una vez que se ha modificado el valor de la variable Indice, la aplicación la utilizará para seleccionar la siguiente pregunta que debe mostrar en la pantalla. La primera vez que el usuario emplee el botón **Siguiente**, el bloque cambiará el valor de *Indice* de 1 a 2. Entonces, la aplicación escogerá el segundo elemento de la lista *ListadePreguntas*: **¿Que presidente garantizó a la China comunista un reconocimiento formal en 1979?**. La segunda vez que el usuario utilice el botón Siguiente, el valor de Indice pasará de 2 a 3 y se seleccionará la tercera pregunta de la lista **¿Que presidente renuncio debido al escándalo del WaterGate?.**

Cuando el usuario emplea el botón **Siguiente** y el valor de **Indice** pasa de 3 a 4, el programa llama a **select list item** para recoger el siguiente elemento de la lista. **Select list item** utilizará el valor de Indice para localizarlo (en este momento es 4). Como solamente hay 3 elementos en la variable ListadePreguntas, el dispositivo Android no sabrá qué hacer y cerrará la aplicación (****compruebalo en tu teléfono**. Habla con tu profesor). Así pues, ¿cómo podemos hacer que el programa sepa si ha alcanzado el último elemento de la lista?.

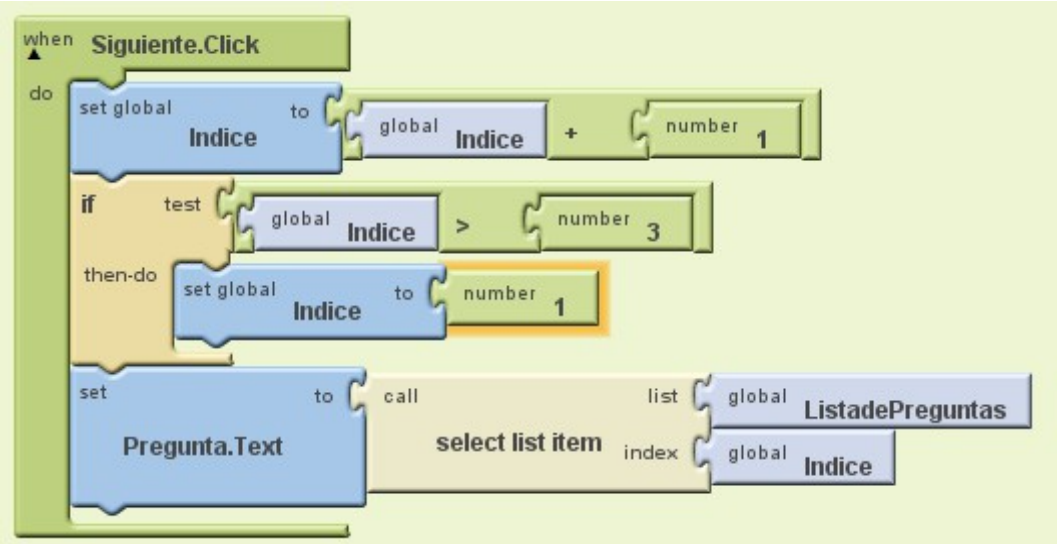
Guarda primero el proyecto con el nombre **PresidentsQuiz2**, se generará el fichero **"PresidentsQuiz2.apk"** que es el que debes copiar a tu teléfono.

PRUEBA!!!!

Hemos de preguntar pues **si la variable Indice es mayor que 3**. Si la respuesta es afirmativa, deberemos hacer que el valor de **Indice vuelva a ser 1**, con lo que la aplicación mostrará de nuevo la primera pregunta en la pantalla. La siguiente tabla muestra los bloques necesarios:

Tipo de bloque	Cajón	Finalidad
if	**Control**	Trata de establecer si el usuario se encuentra en la última pregunta.
=	**Math**	Comprueba si Indice es 3.
global Indice	**My definitions**	Coloca el valor de Indice a la izquierda del signo =.

Tipo de bloque	Cajón	Finalidad
number (3)	**Math**	Coloca este número a la derecha del signo =, ya que la lista contiene 3 elementos.
set global Indice to	**My definitions**	Asigna el valor 1 para que la aplicación vuelva a la primera pregunta.
number (1)	**Math**	Asigna el valor 1 al índice.



Guarda primero el proyecto con el nombre **PresidentsQuiz3**, se generará el fichero **"PresidentsQuiz3.apk"**que es el que debes copiar a tu teléfono.

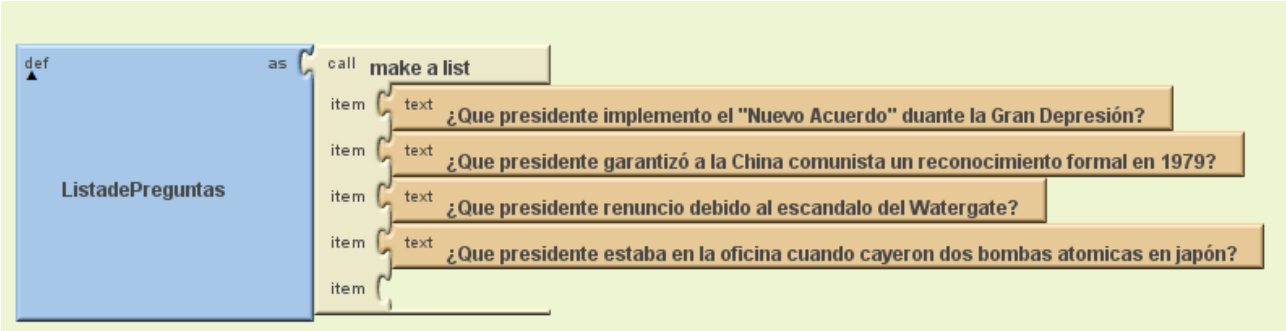
PRUEBA!!!

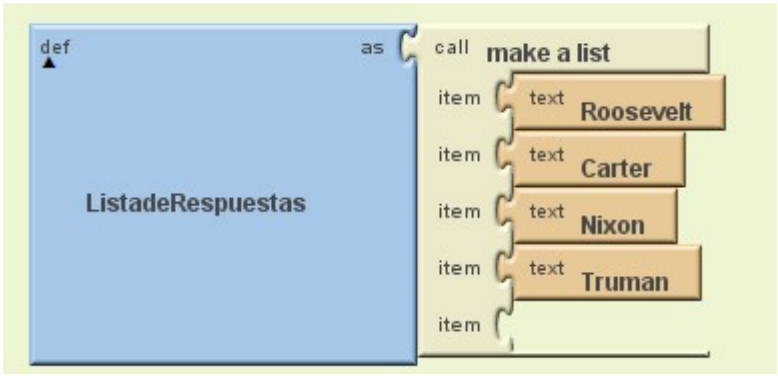
Facilitar la modificación del cuestionario

¿Qué ocurre si añadimos una cuarta pregunta?. Añadimos a **ListadePreguntas** siguiente pregunta:

¿Que presidente estaba en la oficina cuando cayeron dos bombas atomicas en Japon?

Y añadimos la respuesta a **ListadeRespuestas**: **Truman**.

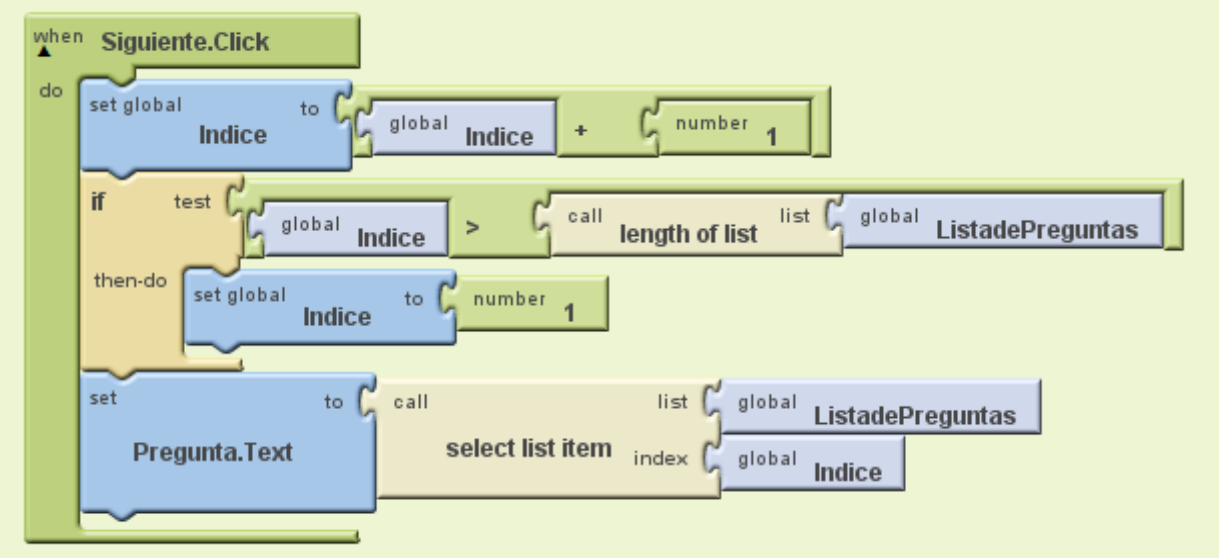




Si pruebas ahora la aplicación comprobarás que falla. El problema reside en que la comprobación que efectúa la aplicación para determinar si el usuario se encuentra en la última pregunta es muy específica. Pregunta si el valor de la variable **Indice** es **3**. Si cambiamos este valor por 4, el programa funcionará correctamente, pero si queremos volver a añadir otra pregunta nos encontraremos en la misma tesitura.

Deberemos preguntar si el valor de la variable **Indice** es mayor que el **número de elementos de ListadePreguntas**. Así pues, modifica **Siguiente.Click** y sustituye la comprobación que habíamos hecho con el número 3 por la que hay en la figura de abajo, previamente presentamos la tabla con los bloques necesarios:

Tipo de bloque	Cajón	Finalidad
length of list	**Lists**	Pregunta el número de elementos que hay en **ListadePreguntas**.
global ListadePreguntas	**My definitions**	Colocará este elemento en la ranura **list** de **length of list**.



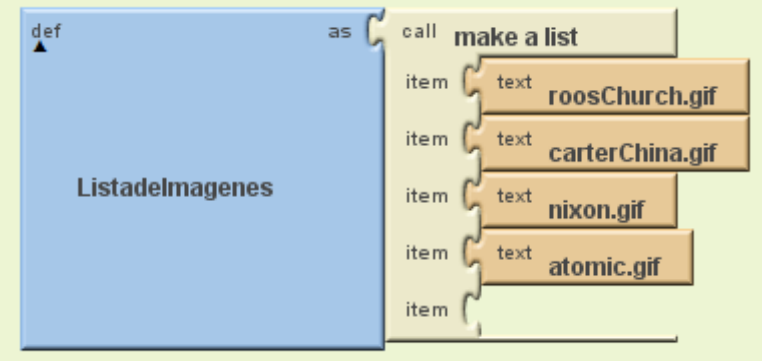
Guarda primero el proyecto con el nombre **PresidentsQuiz4**, se generará el fichero **"PresidentsQuiz4.apk"** que es el que debes copiar a tu teléfono.

PRUEBA!!!!

Ahora, el bloque **if** compara el valor de **currentListadePreguntas** con la longitud de la lista **ListadePreguntas**. Este comportamiento funcionará siempre, sin importar la cantidad de elementos que tenga la lista de preguntas.

Mostrar la imagen de cada pregunta

Podemos enseñar una imagen relacionada con cada pregunta. Al principio del proyecto incorporamos en la sección Media de Component Designer cuatro imágenes. Para utilizarlas crearemos una nueva lista llamada **PictureList**, que albergará el nombre de las imágenes que se

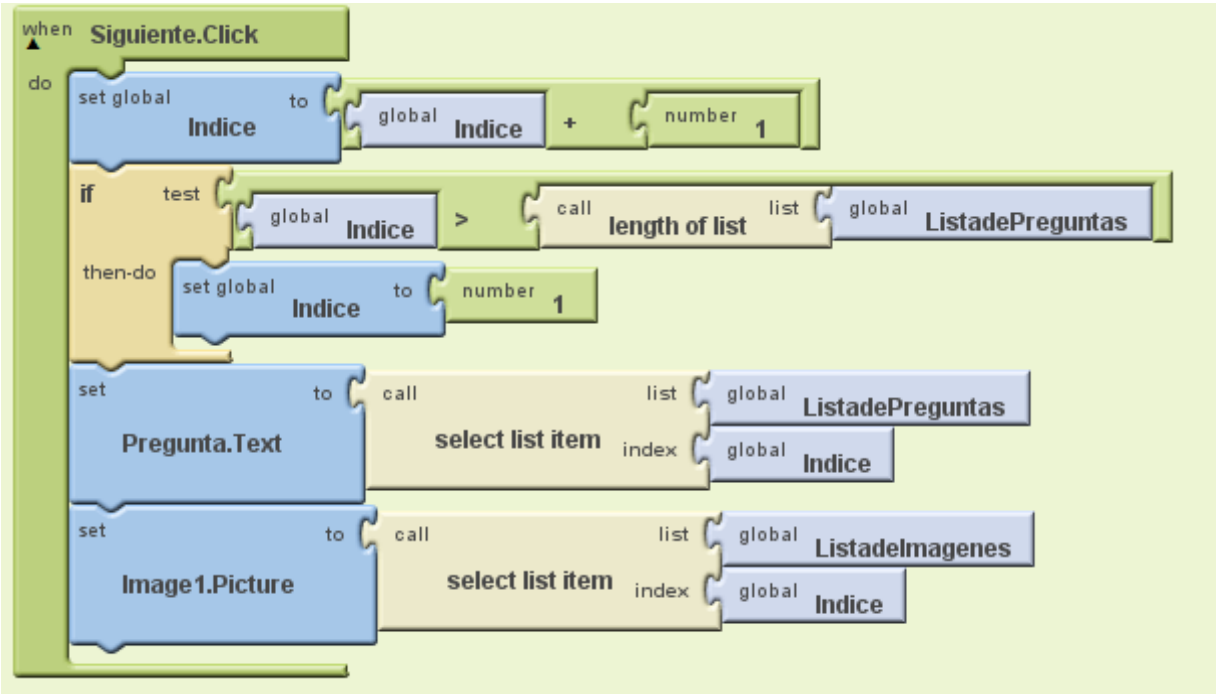


utilizan:

También debemos modificar **Siguiente.Click** para que las muestre de una en una con las preguntas. Para ello, utilizaremos la propiedad **Image.Picture** que se encargará de cambiar la fotografía. Los bloques necesarios son los siguientes:

Tipo de bloque	Cajón	Finalidad
set Image1.Picture to	**Image1**	Cambiará la imagen.
select list item	**Lists**	Selecciona la imagen correspondiente a la pregunta que aparece en la pantalla.
global PictureList	**My Definitions**	Elige un nombre de archivo de la lista.
global Indice	**My Definitions**	Escoge el elemento de la lista cuyo índice coincida con el valor de Indice.

La variable **Siguiente.Click** hace las funciones de índice de **Siguiente.Click** y **PictureList**. Como hemos configurado las listas correctamente, de modo que la primera pregunta se corresponde con la primera imagen, la segunda con la segunda, etc. podremos utilizar el mismo índice con ambas listas como vemos en la figura siguiente:



Guarda primero el proyecto con el nombre **PresidentsQuiz5**, se generará el fichero **"PresidentsQuiz5.apk"** que es el que debes copiar a tu teléfono.

PRUEBA!!!!

Comprobar las respuestas del usuario

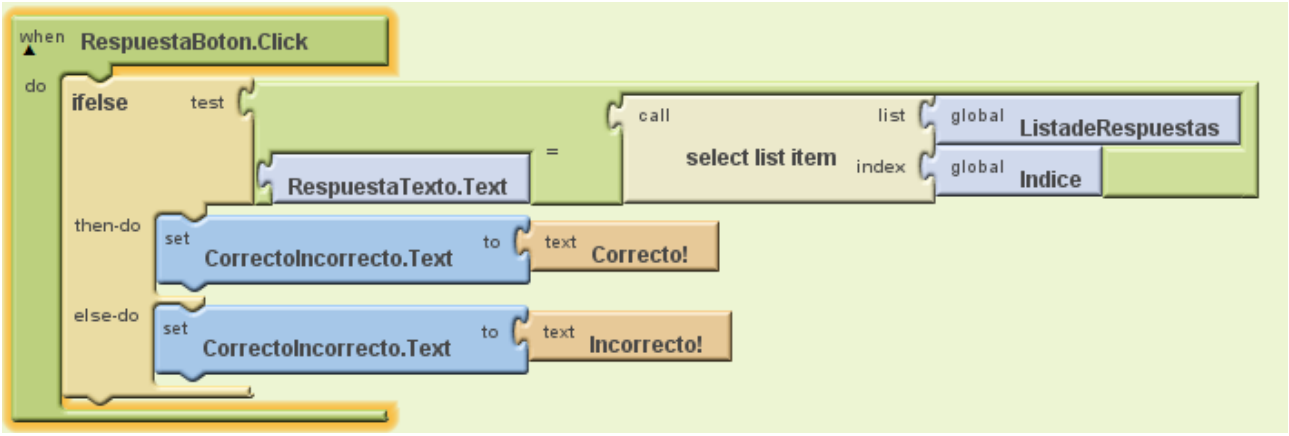
Hasta el momento, hemos creado una aplicación que tan sólo se mueve por unas listas de preguntas y respuestas y tiene asociada una imagen a cada cuestión. Es un buen ejemplo de cómo se pueden desarrollar programas que utilicen listas pero, para que se convierta en un verdadero juego de preguntas y respuestas, tendremos que proporcionar al usuario información sobre si ha contestado correctamente o no. Así pues, vamos a centrarnos ahora en los bloques que se encargarán de suministrar esta información. Hemos configurado la interfaz para que el usuario pueda introducir su respuesta en un campo de texto (**RespuestaTexto**) y luego la envíe a través del botón **RespuestaBoton**. La aplicación debe comparar la entrada del usuario con la respuesta de la pregunta que aparece en pantalla, para lo que recurrirá a un bloque **ifelse**. Luego, se deberá modificar el bloque **CorrectoIncorrecto** para informar si la respuesta es la correcta o no. Los bloques necesarios son los siguientes:

Tipo de bloque	Cajón	Finalidad
RespuestaBoton.Click	**RespuestaBoton**	Se lanzará cuando el usuario emplee el botón **RespuestaBoton**.
ifelse	**Control**	Si la respuesta es correcta hará una cosa. Si no, hará otra diferente.
=	**Math**	Pregunta si la respuesta es correcta.
RespuestaTexto.Text	**RespuestaTexto**	Contiene la respuesta del usuario.
select list item	**Lists**	Selecciona la respuesta de la lista **ListadeRespuestas**.
global ListadeRespuestas	**My Definitions**	La lista desde donde se seleccionará la respuesta.
global Indice	**My Definitions**	El número de la pregunta y la respuesta del usuario.

Tipo de bloque	Cajón	Finalidad
set CorrectoIncorrecto.Text to	**CorrectoIncorrecto**	Aquí muestra la respuesta.
text ("correct!")	**Text**	Indica que la respuesta fue correcta.
set CorrectoIncorrecto.Text to	**CorrectoIncorrecto**	Aquí muestra la respuesta.
text ("incorrect!")	**Text**	Indica que la respuesta fue incorrecta.

Cómo muestra la figura de abajo, el bloque **ifelse** comprobará si la respuesta escrita por el usuario (**RespuestaTexto.Text**) es igual al elemento de la lista **ListadeRespuestas** que se encuentra en la posición determinada por el valor de **Indice**. Por ejemplo, si el valor de la variable **Indice** es **1**, la aplicación comparará la respuesta del usuario con el primer elemento de la lista **ListadeRespuestas**: **Roosevelt**.

Si el valor de la variable **Indice** fuese **2**, la aplicación comparará la respuesta con la segunda de la lista: **Carter**. Y así sucesivamente. Si la comparación es positiva (**true**), entonces se ejecutarán los bloques **then-do** y se asignará el valor **correct!** a **CorrectoIncorrecto**. Si el valor de la comprobación fuese **false**, es decir, si el usuario hubiese respondido incorrectamente, entonces se ejecutarán los bloques **else-do** y el valor de la variable **CorrectoIncorrecto** será **incorrect!**.

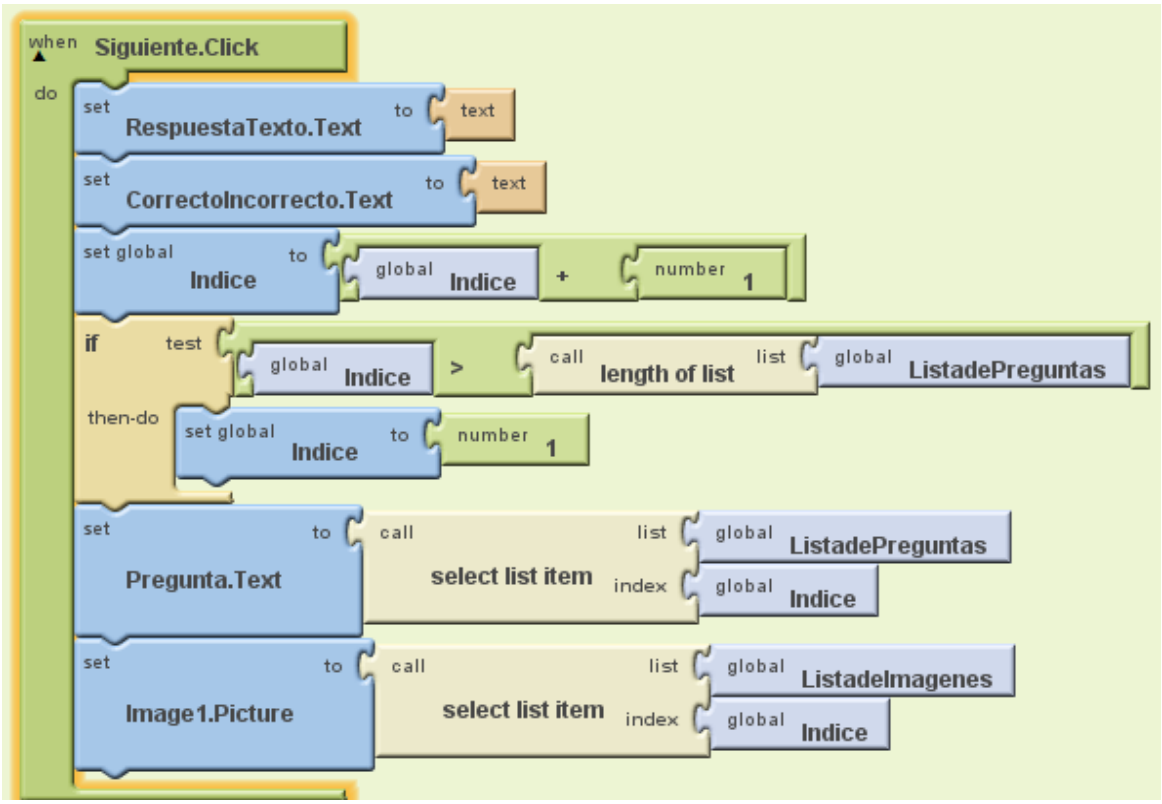


Guarda primero el proyecto con el nombre **PresidentsQuiz6**, se generará el fichero **"PresidentsQuiz6.apk"** que es el que debes copiar a tu teléfono. Prueba a contestar correcta e incorrectamente. Ten en cuenta que el programa sólo considera respuesta correcta aquella que coincide exactamente a la escrita en la lista **ListadeRespuestas**.

PRUEBA!!!!

La aplicación debe funcionar, pero cuando utilizas el botón **Siguiente** aún se muestra el texto **correcto!** o **incorrecto!**. Para limpiar el contenido de las variables **CorrectoIncorrecto** y **RespuestaTexto** debes colocar los siguientes bloques dentro de **Siguiente.Click**:

Tipo de bloque	Cajón	Finalidad
set CorrectoIncorrecto.Text to	**CorrectoIncorrecto**	Ésta es la etiqueta que se borrará.
text (“”)	**Text**	Cuando el usuario emplee el botón **Siguiente**, la aplicación borrará el resultado de la respuesta anterior.
set RespuestaTexto.Text to	**RespuestaTexto**	La respuesta del usuario a la última pregunta.
text (“”)	**Text**	Cuando el usuario emplee el botón **Siguiente**, la aplicación borrará la respuesta anterior.



Guarda primero el proyecto con el nombre **PresidentsQuiz7**, se generará el fichero **"PresidentsQuiz7.apk"** que es el que debes copiar a tu teléfono.

PRUEBA!!!

Variaciones

- En vez de limitarse a mostrar una imagen por cada pregunta, trata de reproducir un sonido o un vídeo corto. Al trabajar con sonido, podrías crear un programa que invitase al usuario a adivinar qué canción está sonando.

- Este juego es muy estricto desde el punto de vista de las cadenas de texto que admite como contestaciones válidas, utiliza el bloque **text.contains** para ver si la respuesta del usuario contiene la almacenada en la aplicación. Otra opción es proporcionar varias contestaciones posibles a cada pregunta y dejar que el usuario seleccione la correcta. También es posible que el usuario escriba algún espacio de más o mayúsculas. Considera todos estos comportamientos antes de dar por incorrecta una respuesta.

- Modifica la aplicación para que admita varias respuestas. Deberás agregar listas adicionales que contengan todas las soluciones posibles a cada pregunta. Entonces, tendremos que éstas serán una lista cuyos elementos serán otras listas. Cada una de estas sublistas albergará las contestaciones. Utiliza el componente **ListPicker** para permitir que el usuario seleccione una respuesta.