

Programación con MIT APP INVENTOR

1. Introducción

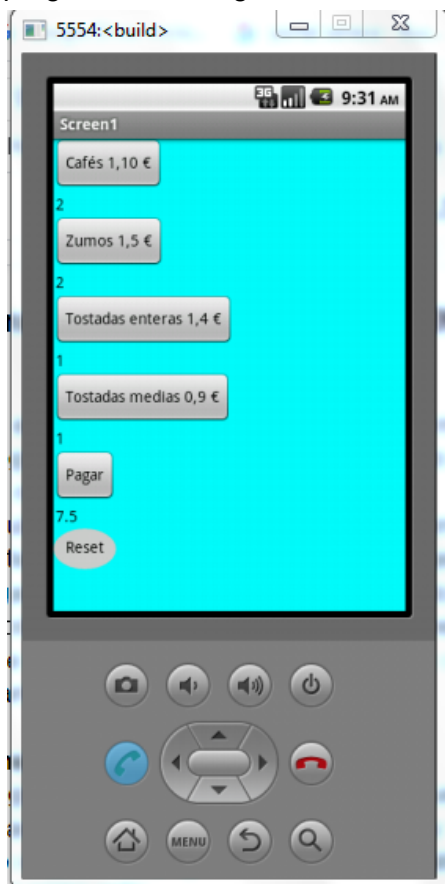
MIT APP Inventor es una aplicación online que nos permite crear aplicaciones con un lenguaje de programación orientado a objetos basado en bloques. Para acceder a la aplicación solo necesitamos una cuenta de Google.

Para utilizar el emulador y poder comprobar los programas que vayamos creando necesitaremos instalar en el ordenador un archivo ejecutable, el AIStarter, que podremos descargar desde la web APP Inventor. Esto permitirá que usemos el emulador, al que podremos acceder desde el menú Connect -> Emulador.

La primera vez que ejecutemos el emulador habrá que seguir las instrucciones de instalación correctamente para que funcione bien.

2. El primer programa

Nuestro primer programa consistirá en una aplicación que nos permita ayudar a los camareros en su tarea de recibir la comanda y poderla transmitir correctamente a cocina, obteniendo el precio a pagar de manera automática. La interfaz de nuestro programa sería algo como esto:



El diagrama de bloques que ejecuta el código del programa tendría la siguiente forma:

```
initialize global cafes to 0
initialize global zumos to 0
initialize global enteras to 0
initialize global medias to 0
initialize global total to 0
```

```
when cafes .Click
do set global cafes to get global cafes + 1
  set cantidadcafes .Text to get global cafes

when zumos .Click
do set global zumos to get global zumos + 1
  set cantidadzumos .Text to get global zumos

when enteras .Click
do set global enteras to get global enteras + 1
  set cantidadenteras .Text to get global enteras

when medias .Click
do set global medias to get global medias + 1
  set cantidadmedias .Text to get global medias
```

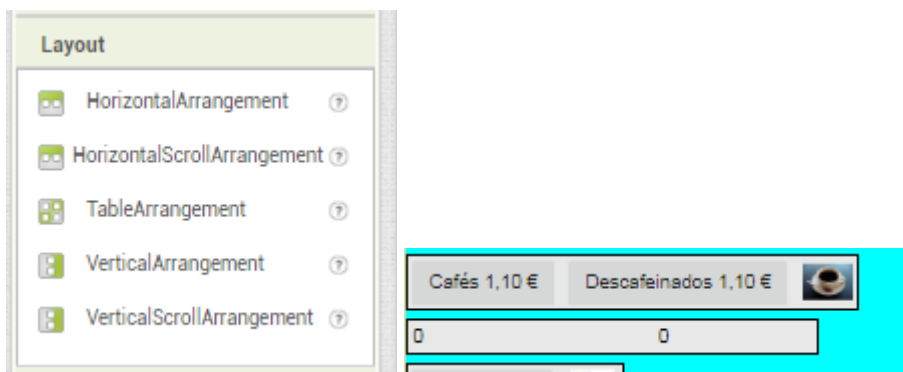
```
when pagar .Click
do set global total to 0
  get global cafes * 1.1 + get global zumos * 1.3 + get global enteras * 1.4 + get global medias * 0.2
  set cantidadpagar .Text to get global total
```

```
when reset .Click
do
  set global cafes to 0
  set cantidadcafes . Text to get global cafes
  set global zumos to 0
  set cantidadzumos . Text to get global zumos
  set global enteras to 0
  set cantidadenteras . Text to get global enteras
  set global medias to 0
  set cantidadmedias . Text to get global medias
  set global total to 0
  set cantidadapagar . Text to get global total
```

3. Layout (HorizontalArrangement)

Debajo de User Interface, donde están los objetos en App Inventor, encontramos la pestaña Layout, la cual nos va a permitir entre otras cosas colocar objetos de izquierda a derecha o de arriba a abajo.

Así si queremos colocar por ejemplo dos botones horizontales, antes elegiríamos un HorizontalArrangement, lo situaríamos en la pantalla de nuestro móvil emulado y ya podríamos arrojar allí los botones deseados. En caso de necesitar más espacio para colocar más botones y desbordar así nuestra pantalla, podríamos elegir la opción HorizontalScrollArrangement, que nos crea un scroll o barra de desplazamiento para evitar el desbordamiento.

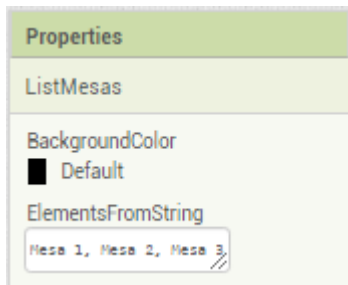


4. El objeto ListPicker

ListPicker



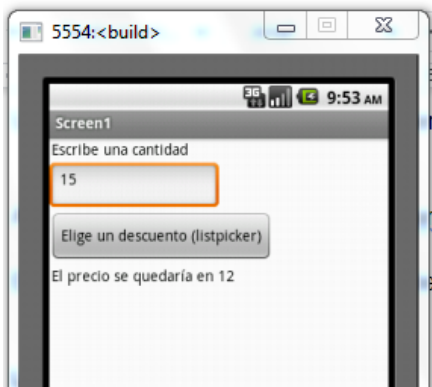
Este objeto nos permitirá crear una lista con una serie de elementos. Para conseguir añadir más de un elemento a la lista, debemos irnos a la propiedad ElementsFromString del objeto ListPicker y allí introducir los elementos de nuestra lista separados por comas.



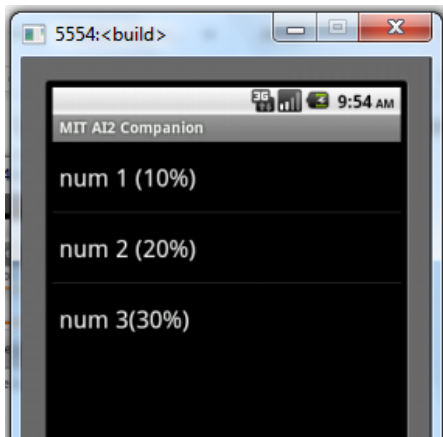
El código asociado a este objeto podría tener la siguiente forma:

```
when ListMesas .AfterPicking  
do set Labelnumerodemesa .Text to ListMesas .Selection
```

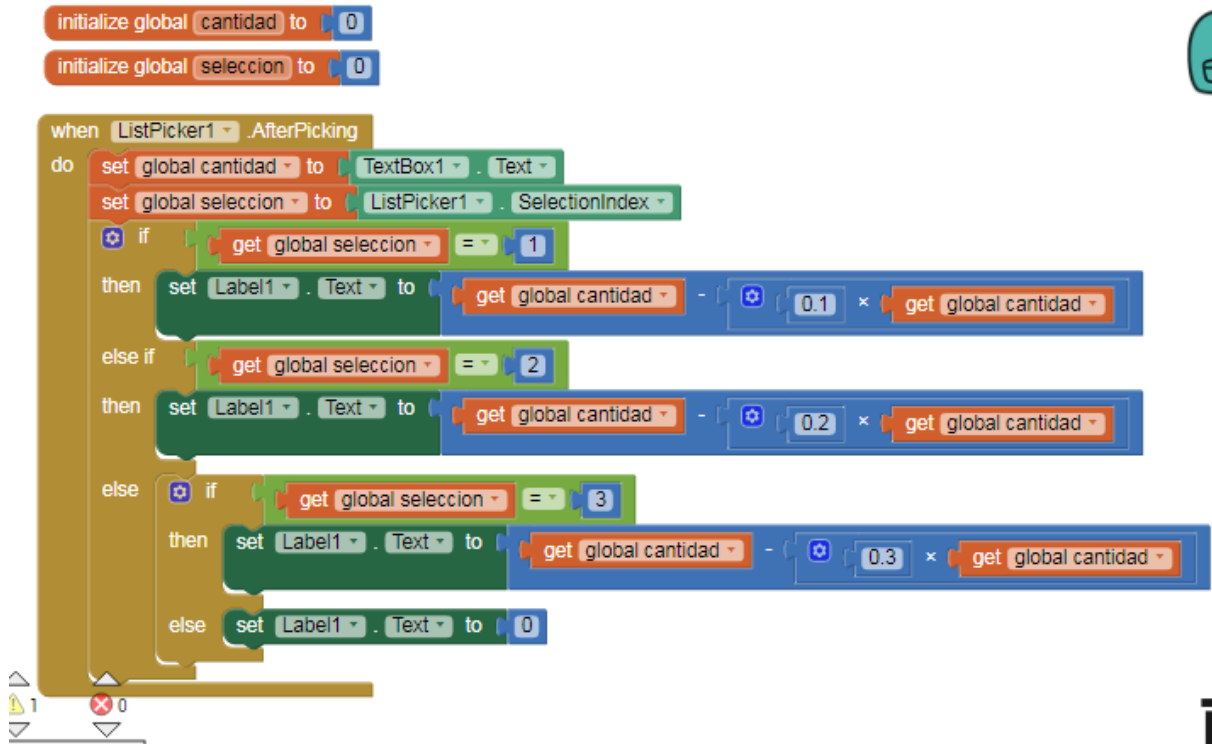
Un ejemplo de uso podría ser el siguiente:



Al pinchar sobre Elige un descuento, aparecerían 3 opciones:



En función de la opción elegida, se mostrará el precio con el descuento aplicado:



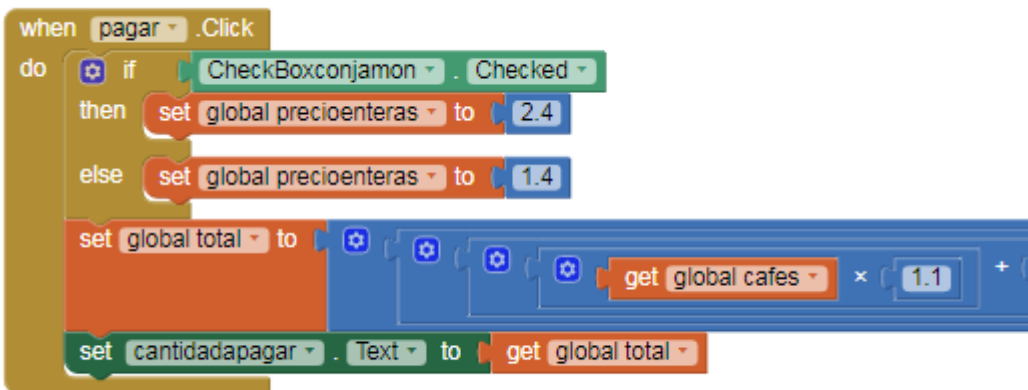
```
initialize global cantidad to 0
initialize global seleccion to 0

when ListPicker1 .AfterPicking
do
  set global cantidad to TextBox1 .Text
  set global seleccion to ListPicker1 .SelectionIndex
  if (get global seleccion = 1)
  then
    set Label1 .Text to (get global cantidad - (0.1 * get global cantidad))
  else if (get global seleccion = 2)
  then
    set Label1 .Text to (get global cantidad - (0.2 * get global cantidad))
  else
    if (get global seleccion = 3)
    then
      set Label1 .Text to (get global cantidad - (0.3 * get global cantidad))
    else
      set Label1 .Text to 0
```

5. La instrucción if.

Los bloques que hacen referencia a la instrucción if siempre deben incluirse dentro de los bloques de otros objetos.

Pongamos un ejemplo:



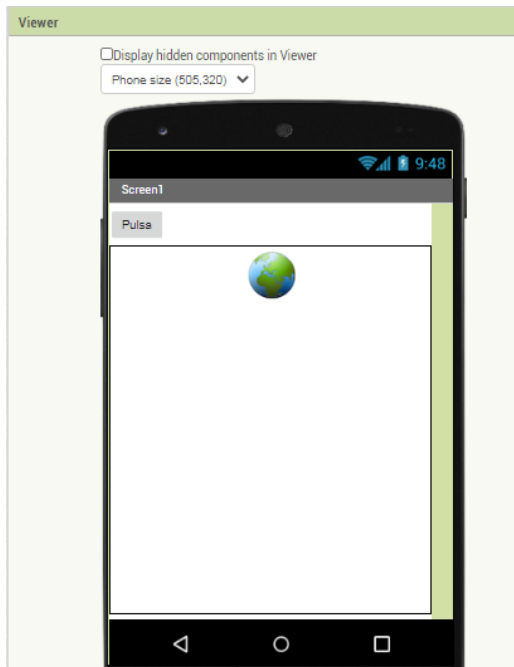
```
when pagar .Click
do
  if (CheckBoxconjamon .Checked)
  then
    set global precioenteras to 2.4
  else
    set global precioenteras to 1.4
  set global total to (get global cafes * 1.1 + (get global cantidad * global precioenteras))
  set cantidadapagar .Text to (get global total)
```

De esta manera cuando pulsemos el botón pagar, el programa evaluará si el objeto checkboxconjamon está checked o marcado y fijará el precio de la variable precioenteras a 2,4. Si no, lo fijará a 1,4.

6. El objeto Webviewer.



Este objeto nos permite navegar a cualquier página que queramos desde nuestra aplicación. Veamos este ejemplo de programa:



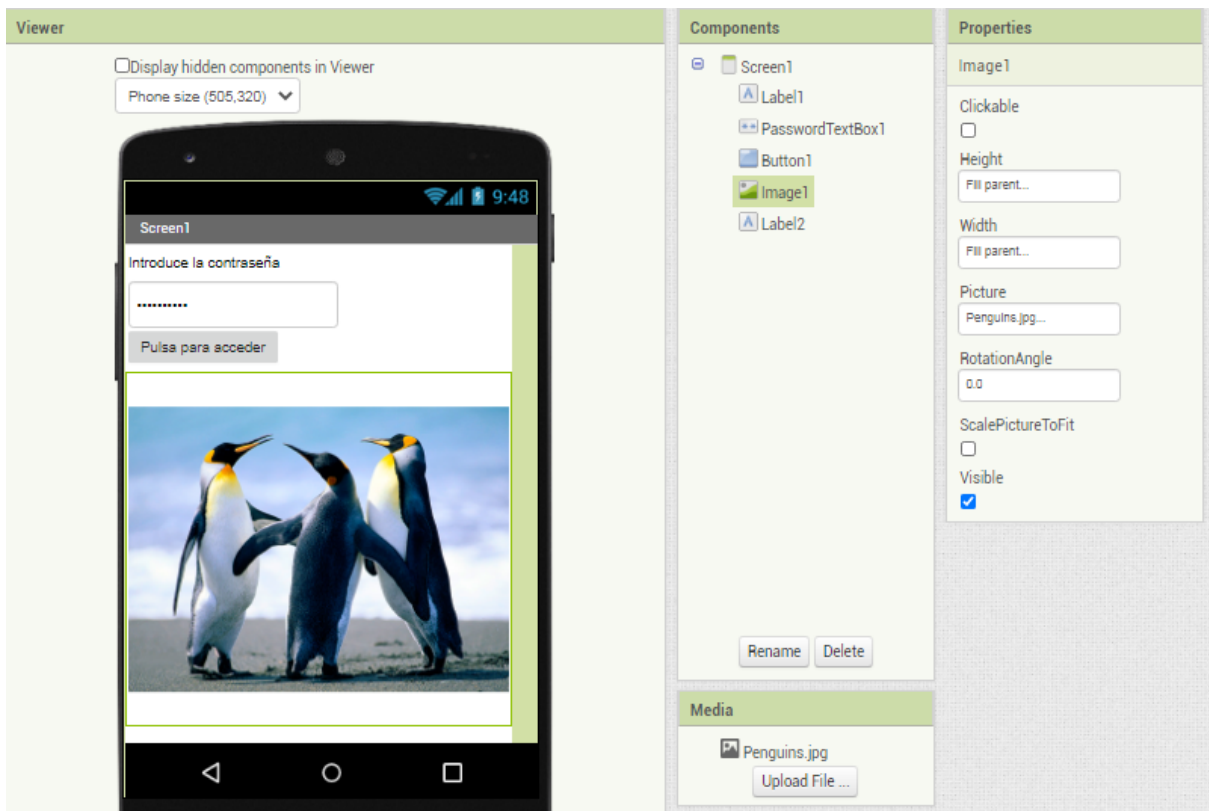
Su código asociado tendría la siguiente forma:

```
initialize global url to " https://manuelfloresiesbellavista.blogspot.com "
```

```
when Button1 .Click  
do set WebView1 . HomeUrl to get global url
```

7. El objeto PasswordTextBox

Quando necesitemos una contraseña de acceso para acceder a un enlace o hacer visible algún objeto, usaremos PasswordTextBox. Veamos el siguiente ejemplo:



El código de esta aplicación es el siguiente:

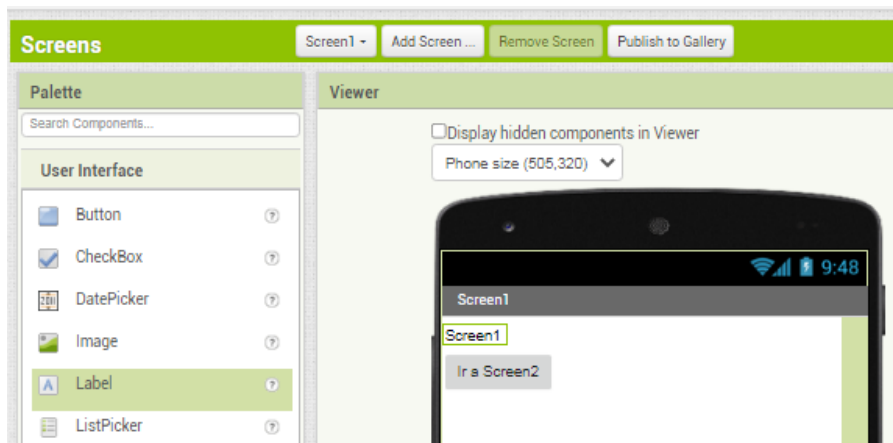
```

when Button1 .Click
do
  if
    compare texts PasswordTextBox1 . Text = " 1234 "
  then
    set Image1 . Visible to true
    set Label2 . Text to " Correcto "
  else
    set Image1 . Visible to false
    set Label2 . Text to " Error "

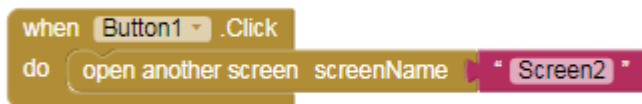
```

8. Cambiar entre varias pantallas o screens.

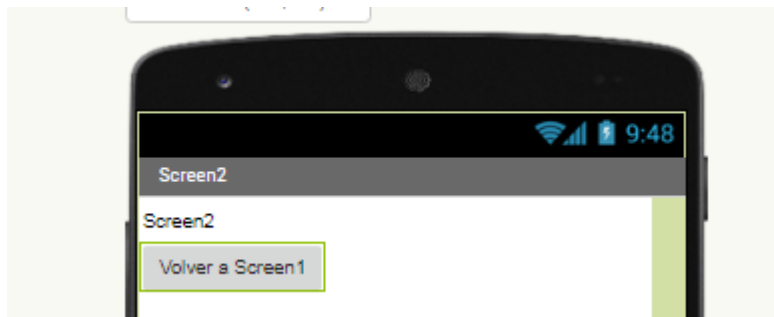
Si nuestra aplicación necesita más de una pantalla, podemos hacerlo añadiendo un nuevo Screen desde el botón Add screen ... del App Inventor:



Una vez hemos creado Screen2, el código dentro de Screen1 para ir a la segunda ventana tendría esta forma:



Así accederíamos a Screen2:



Y nuevamente para volver a Screen1, usaríamos los siguientes bloques de código:

