

# Rapid Mobile Prototyping

# Why do it?



time



money



tangible

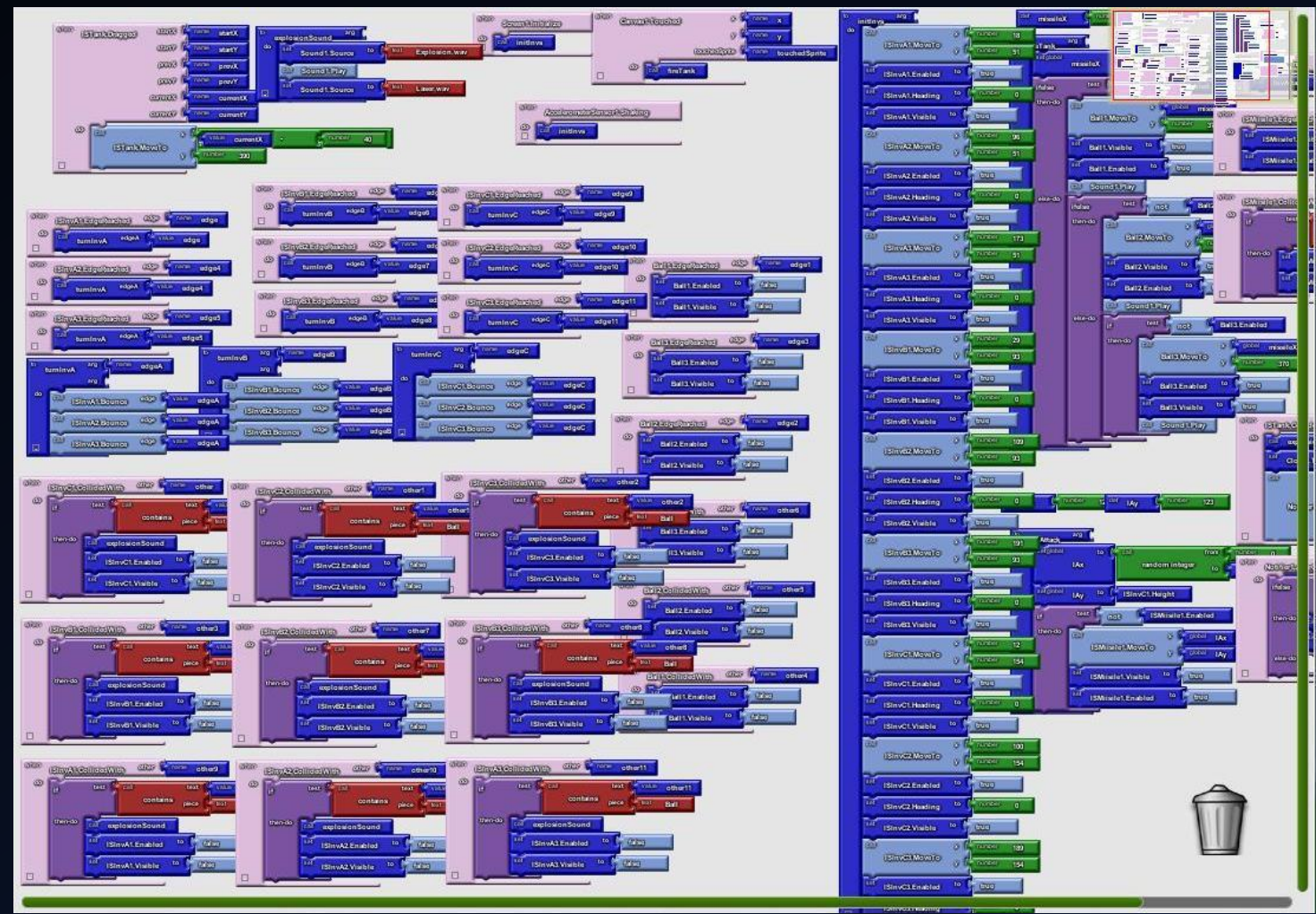
# Famous Apps



# Why use something like App Inventor?

- 1) Observe programming structures (which can be implemented in Java-based environment)
- 2) Results in a workable APK! (Can be reverse-engineered)

Two limitations:  
- Android  
- Limited size and features (but has plenty!)

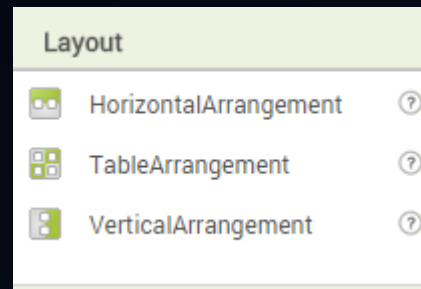


# Setting up your environment...

- STEP 1: If you do not have one, create a google e-mail address. This is required in order to use App Inventor.
- STEP 2: Go to [ai2.appinventor.mit.edu](https://ai2.appinventor.mit.edu) (important note: this is the second version of App Inventor. Make sure you are not using the first version as a lot of features will be missing and coding will be different)
- STEP 3: Connect your gmail address to the program. **'Please select an account that you would like to use'**.

**You're all set!**

# Layout



<http://ai2.appinventor.mit.edu/reference/components/layout.html>

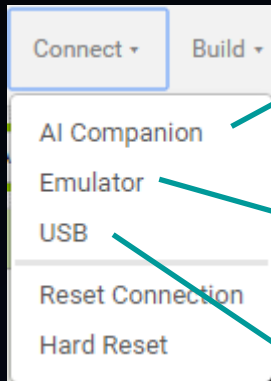
# Fill Parent

## NOTE:

If fill parent seems to not work, it is probably because you need to unclick the 'scrollable' value from the screen properties. You cannot fill parent if the screen height is  $\infty$ .

Avoid using table layout. It is buggy and difficult to control. You can achieve everything you need using a combination of the 'horizontal' and 'vertical' arrangements.

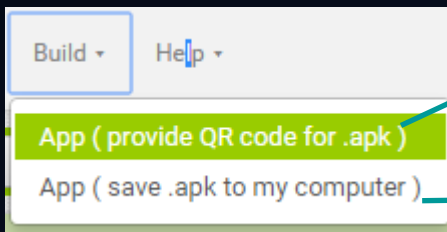
# Hello World (it has to be done...)



You can load this on your phone and live changes are made over the wireless network. This is easy if there is a relatively open network but can be hard over protected networks (like university networks can be)

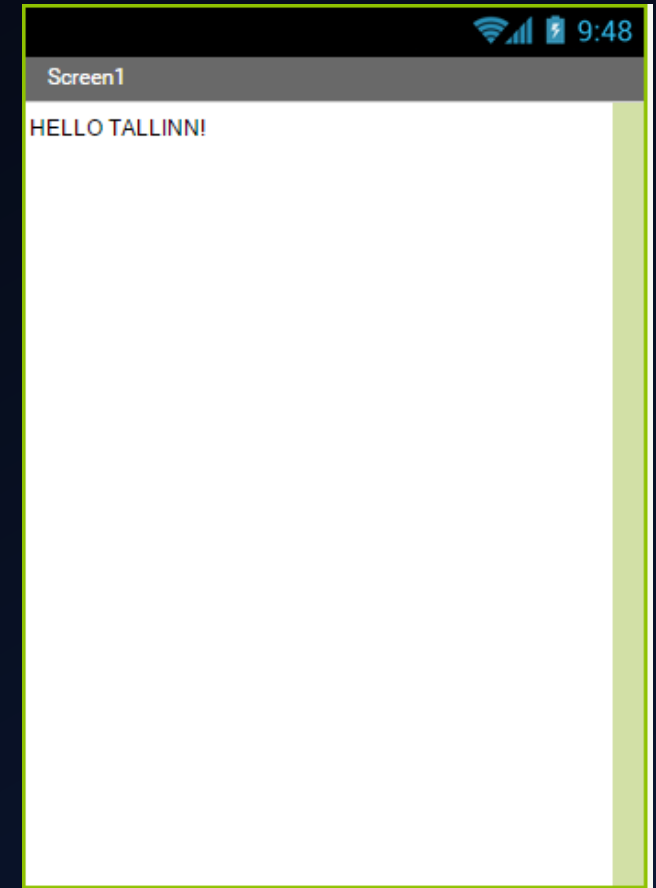
You can install an emulator on your computer. This simulates an android phone and shows you what the app would do on the phone. Does not support things like texting, accelerometer etc. for obvious reasons.

You can connect your phone using a USB cable to load application this way.



A QR Code appears and using a QR scanner you can download and install the APK to your phone. You can also send the QR code to others. Keep in mind that the QR code is valid only for 2 hours after compiling.

Downloads the installer APK file to your computer. You can then email it to yourself or others or post it online.





# Button

## BackgroundColor

Returns the button's background color

## Enabled

If set, user can tap check box to cause action.

## FontBold

If set, button text is displayed in bold.

## FontItalic

If set, button text is displayed in italics.

## FontSize

Point size for button text.

## FontTypeface (designer only)

Font family for button text.

## Height

Button height (y-size).

## Image

Image to display on button.

## Shape (designer only)

Specifies the button's shape (default, rounded, rectangular, oval). The shape will not be visible if an Image is being displayed.



## ShowFeedback

Specifies if a visual feedback should be shown for a button that has an image as background.

## Text

Text to display on button.

## TextAlignment (designer only)

Left, center, or right.

## TextColor

Color for button text.

## Visible

Specifies whether the component should be visible on the screen. Value is true if the component is showing and false if hidden.

## Width

Button width (x-size).

# Button (2)

## Click()

User tapped and released the button.

## Gotfocus()

Indicates the cursor moved over the button so it is now possible to click it.

## Lostfocus()

Indicates the cursor moved away from the button so it is now no longer possible to click it.

## Touchdown()

Indicates that the button was pressed down.

## Touchup()

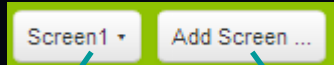
Indicates that a button has been released.

# Accelerometer



```
when AccelerometerSensor1 .Shaking  
do  
  call TextToSpeech1 .Speak  
  message " Stop Shaking Me! "
```

# Changing Screens

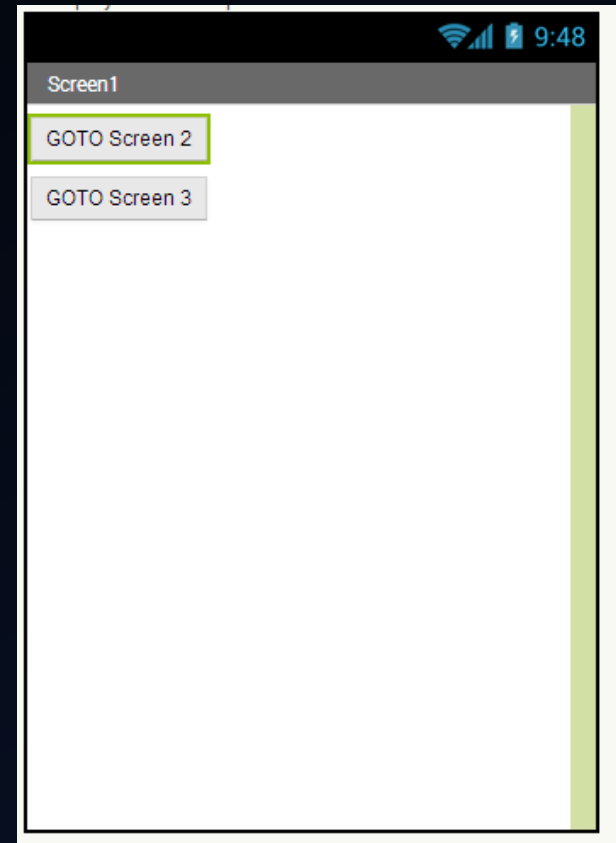


Selecting an Existing Screen



Add a new Screen

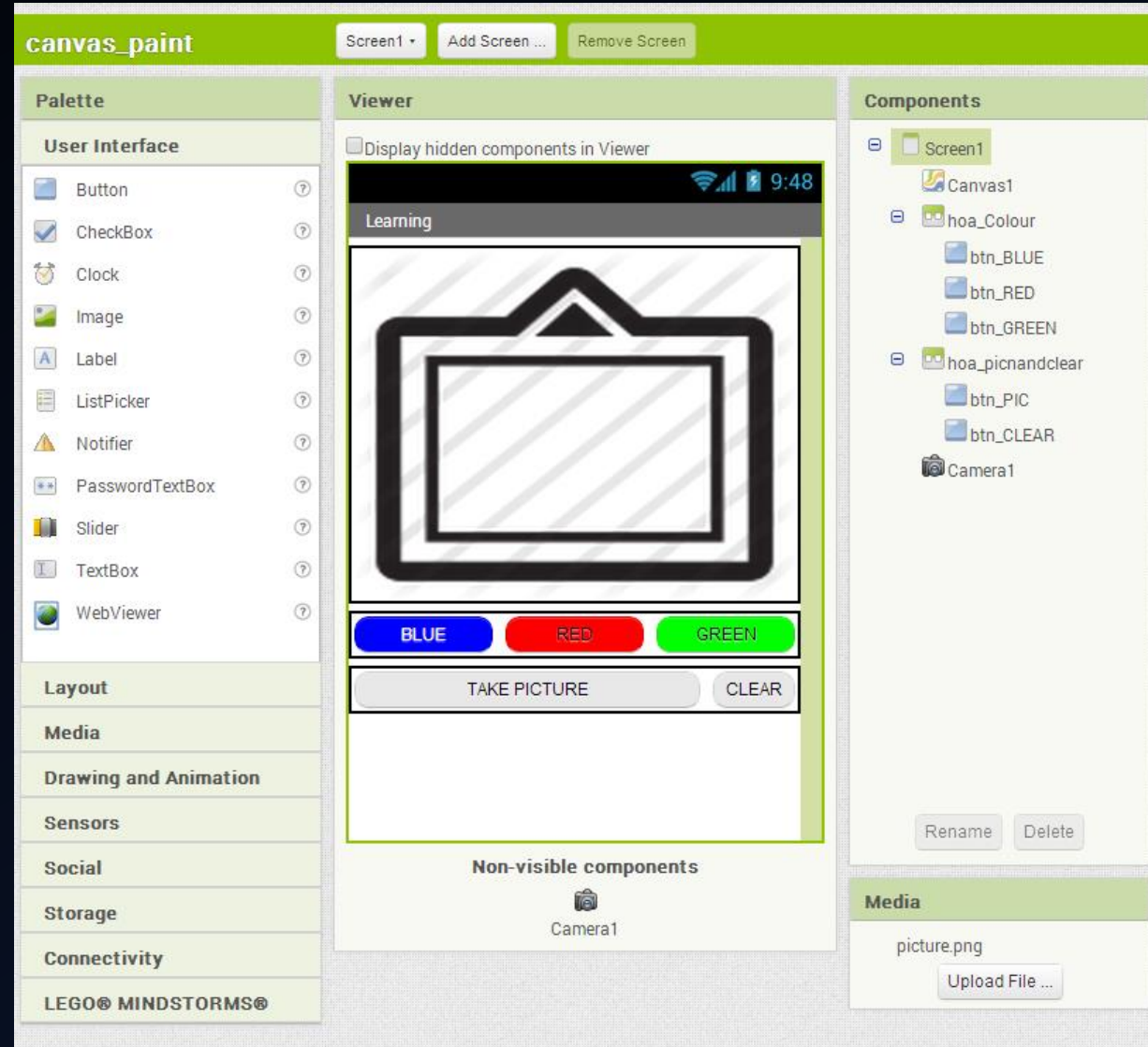
\*You cannot (presently) change the name of the initial screen. You can rename it's heading though.



```
when Button1 .Click
do open another screen screenName "Screen2"

when Button2 .Click
do open another screen screenName "Screen3"
```

# Using the Canvas - Drawing



# Using the Canvas - Drawing

canvas\_paint
Screen1 ▾ Add Screen ... Remove Screen

**Blocks**

- ☑ Built-in
  - Control
  - Logic
  - Math
  - Text
  - Lists
  - Colors
  - Variables
  - Procedures
- ☑ Screen1
  - Canvas1
  - ☑ hoa\_Colour
    - btn\_BLUE
    - btn\_RED
    - btn\_GREEN
  - ☑ hoa\_picandclear
    - btn\_PIC
    - btn\_CLEAR
  - Camera1

Rename Delete

**Viewer**

```

when Canvas1 .Touched
  x y touchedSprite
do call Canvas1 .DrawCircle
  x get x
  y get y
  r 5

when btn_BLUE .Click
do set Canvas1 .PaintColor to blue

when btn_RED .Click
do set Canvas1 .PaintColor to red

when btn_GREEN .Click
do set Canvas1 .PaintColor to green

when Canvas1 .Dragged
  startX startY prevX prevY currentX currentY draggedSprite
do call Canvas1 .DrawLine
  x1 get prevX
  y1 get prevY
  x2 get currentX
  y2 get currentY

when btn_CLEAR .Click
do call Canvas1 .Clear

when btn_PIC .Click
do call Camera1 .TakePicture

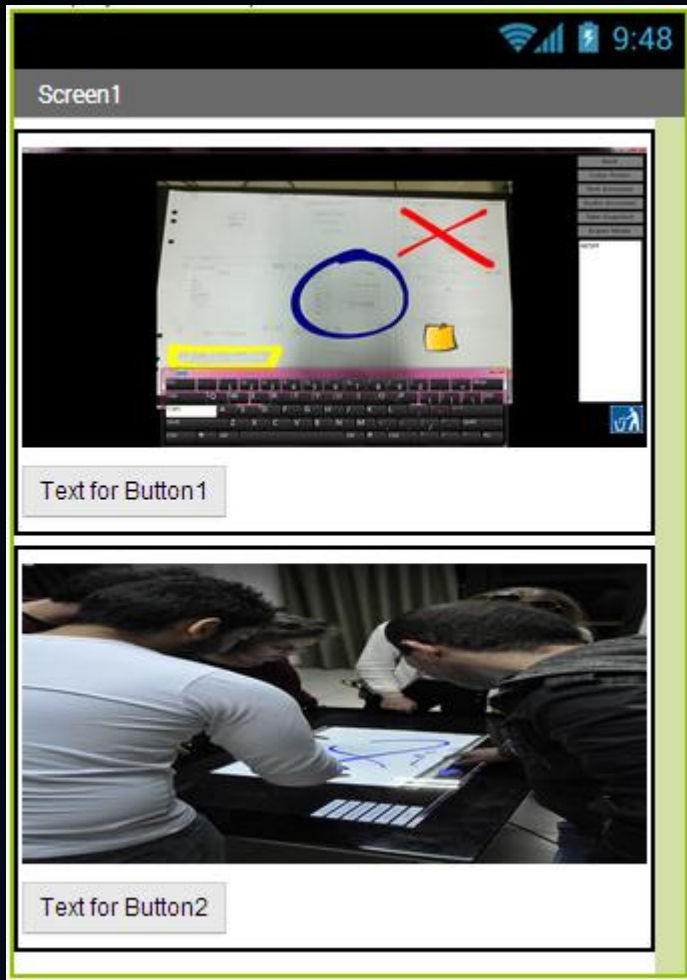
when Camera1 .AfterPicture
  image
do set Canvas1 .BackgroundImage to get image
          
```

**Media**

picture.png

Upload File ...

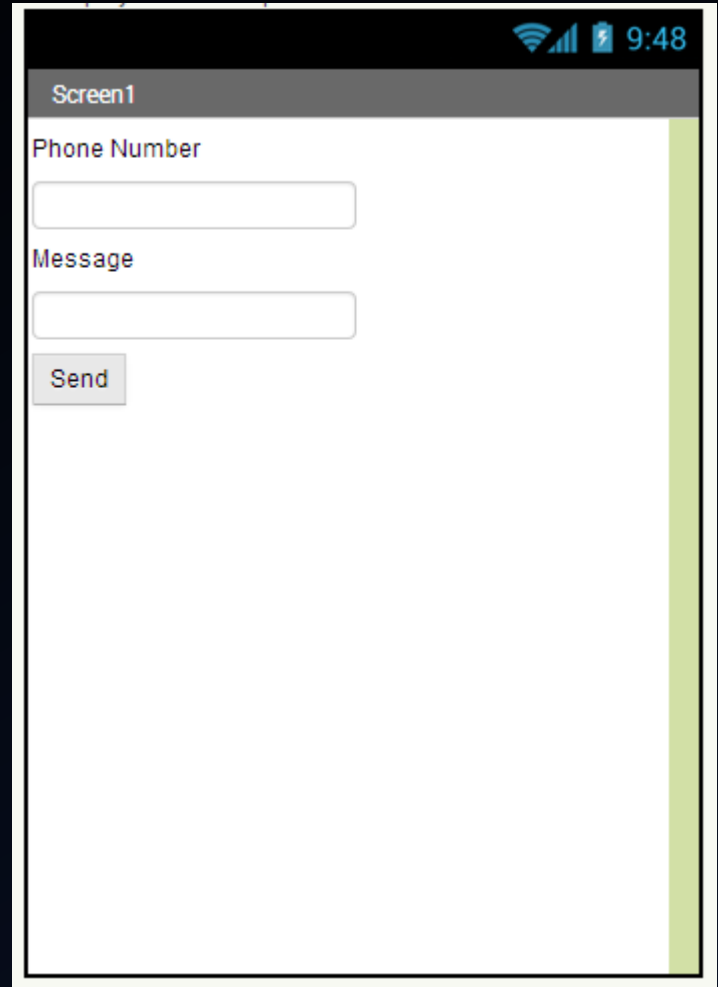
# Simulating Multiple Screens In One



```
when Button1 .Click  
do set VerticalArrangement1 . Visible to false  
   set VerticalArrangement2 . Visible to true
```


```
when Button2 .Click  
do set VerticalArrangement2 . Visible to false  
   set VerticalArrangement1 . Visible to true
```

# SMS



```
when btn_Send .Click
do
  set Texting1 . PhoneNumber to txt_Number . Text
  set Texting1 . Message to txt_Message . Text
  call Texting1 .SendMessage
```

**Non-visible components**



Texting1



# SOME PHYSICS

```

when GolfBall .Flung
  x y speed heading xvel yvel
do
  set GolfBall . Heading to get heading
  set GolfBall . Speed to get speed x 7

when TimerClock .Timer
do
  if GolfBall . Speed > 0.5
  then set GolfBall . Speed to GolfBall . Speed - 0.5
  else set GolfBall . Speed to 0

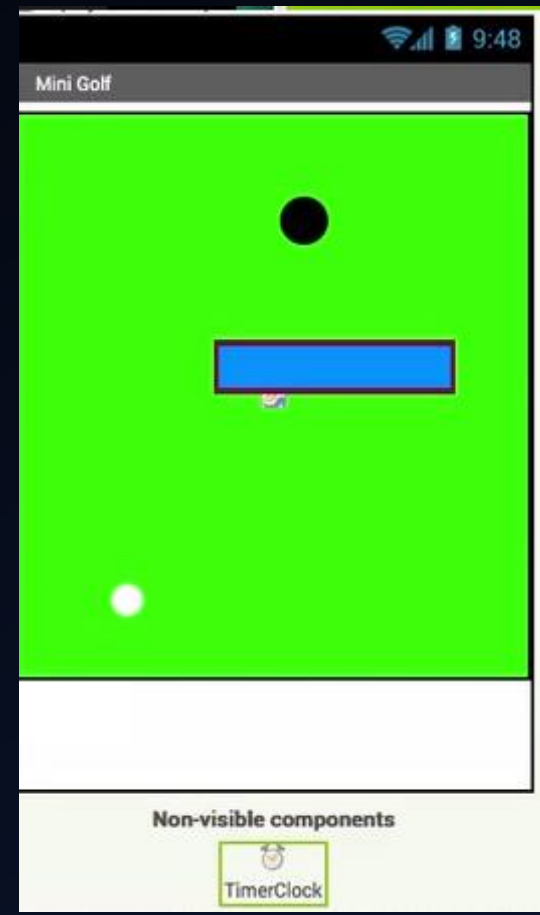
when GolfBall .EdgeReached
  edge
do
  call GolfBall .Bounce
  edge get edge
    
```

```

when GolfBall .CollidedWith
  other
do
  if get other == Hole
  then
    set GolfBall . Speed to 0
    set GolfBall . X to Hole . X
    set GolfBall . Y to Hole . Y
    
```

```

when Obstacle .CollidedWith
  other
do
  set GolfBall . Heading to 0 - GolfBall . Heading
    
```



For a more comprehensive list of the use of App Inventor components have a look at:

<http://ai2.appinventor.mit.edu/reference/components/>