



開放源碼技術與應用

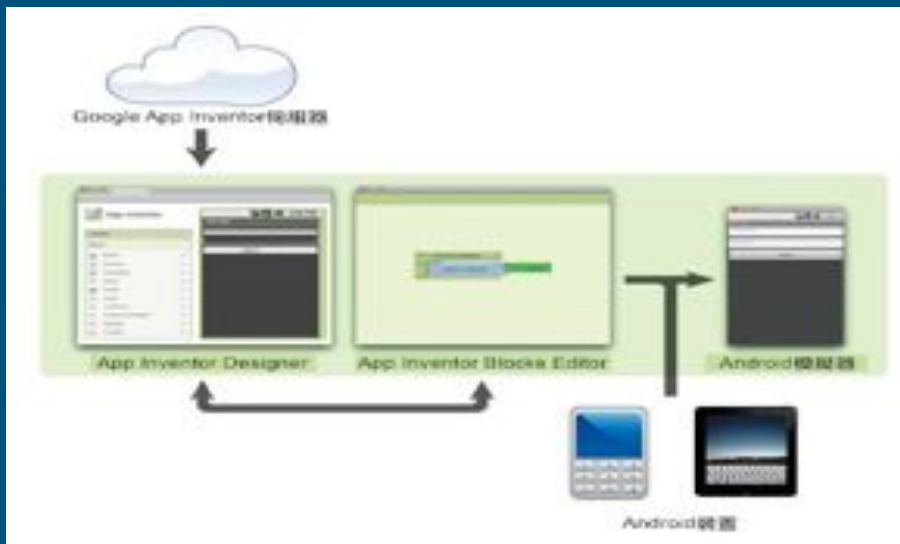
藍牙通訊及**App Inventor**



Android手機程式編寫

App Inventor 原是Google實驗室(Google Lab)的一個子計畫，由一群Google工程師與勇於挑戰的Google使用者共同參與。Google App Inventor是一個完全線上開發的Android程式環境，拋棄複雜的程式碼而使用樂高積木式的堆疊法來完成您的Android程式。除此之外它也正式支援樂高NXT機器人，對於Android初學者或是機器人開發者來說是一大福音。因為對於想要用手機控制機器人的使用者而言，他們不大需要太華麗的介面，只要使用基本元件例如按鈕、文字輸入輸出即可。

App Inventor於 2012年1月1日移交給麻省理工學院行動學習中心，並已於3月4日公佈使用。

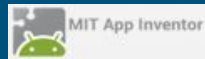


App Inventor 2 來了

App Inventor 一是由Google Lab所建置，一套免費的雲端Android程式開發工具，可以讓使用者快速建置個人的Android程式，就算沒有工程背景也能輕易上手!!!

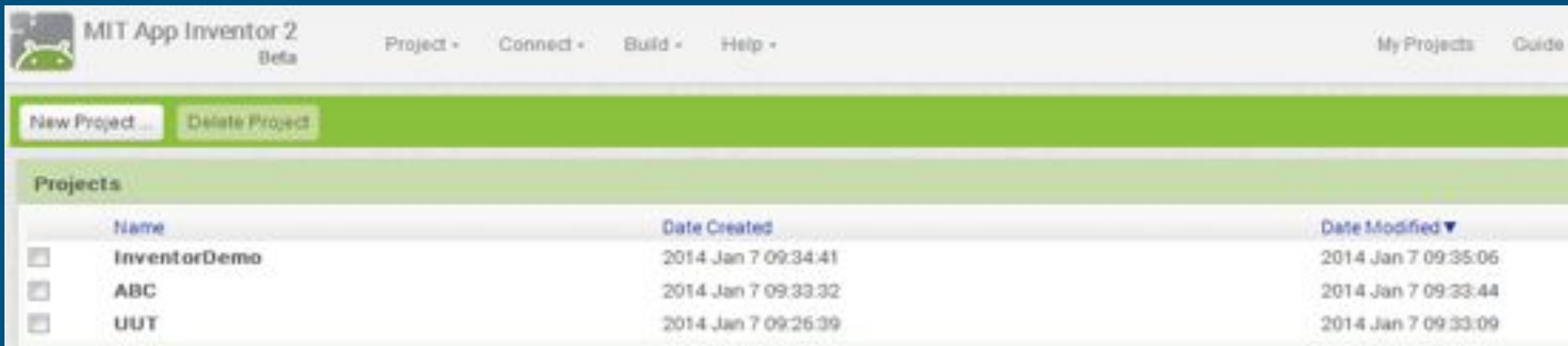
在2013年底，MIT 行動學習中心已發表 App Inventor 2 (AI2) 一操作程序與原先的 App Inventor十分類似，但兩者為不同系統，故程式不能通用。

App Inventor 2 & App Inventor Classic 的變更



- 程式設計概念不變，先在 Designer 前置面板決定好畫面元件的配置，再到 Blocks Editor 新增指令來決定程式的行為。
- 原有的 App Inventor 更名為 App Inventor Classic，未來可能會停用。
- App Inventor Classic 的原始檔格式為 .zip，而 AI2 是 .aia，兩者不通用。
- 省略了需要使用 Java 才能開啟的 Blocks Editor，整合成在網頁中即可編輯。
- Blocks 後置面板的頁面也可輸入正體中文了！是開發者的一大福音。
- 新版本請先安裝 MIT AI2 Companion 才能進行無線同步。

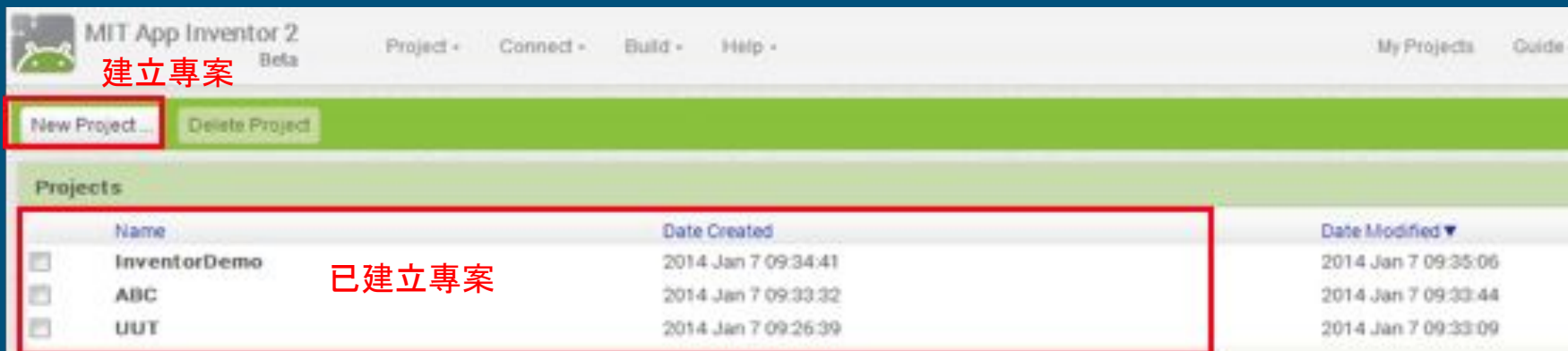
Google搜尋app inventor (ai2.appinventor.mit.edu), 登入個人帳號後即可進入以下畫面, 開始撰寫程式。



The screenshot shows the MIT App Inventor 2 Beta web interface. At the top, there is a navigation bar with the MIT App Inventor logo, the text "MIT App Inventor 2 Beta", and menu items: "Project", "Connect", "Build", and "Help". On the right side of the navigation bar are links for "My Projects" and "Guide". Below the navigation bar is a green bar containing two buttons: "New Project ..." and "Delete Project". Underneath this is a section titled "Projects" which contains a table of existing projects.

| | Name | Date Created | Date Modified ▼ |
|--------------------------|--------------|---------------------|---------------------|
| <input type="checkbox"/> | InventorDemo | 2014 Jan 7 09:34:41 | 2014 Jan 7 09:35:06 |
| <input type="checkbox"/> | ABC | 2014 Jan 7 09:33:32 | 2014 Jan 7 09:33:44 |
| <input type="checkbox"/> | UUT | 2014 Jan 7 09:26:39 | 2014 Jan 7 09:33:09 |

點選New Project後，建立一個新的測試程式



The screenshot shows the MIT App Inventor 2 Beta web interface. The top navigation bar includes the MIT App Inventor logo, the text 'MIT App Inventor 2 Beta', and menu items: 'Project', 'Connect', 'Build', and 'Help'. On the right side of the top bar are links for 'My Projects' and 'Guide'. Below the top bar is a green horizontal bar containing two buttons: 'New Project...' (highlighted with a red box) and 'Delete Project'. Below this is a section titled 'Projects' which contains a table of existing projects. The table has three columns: 'Name', 'Date Created', and 'Date Modified'. Three projects are listed: 'InventorDemo', 'ABC', and 'UIT'. The first two columns of the table are enclosed in a red box. The text '已建立專案' (Project already created) is written in red next to the 'InventorDemo' row.

MIT App Inventor 2 Beta

Project • Connect • Build • Help •

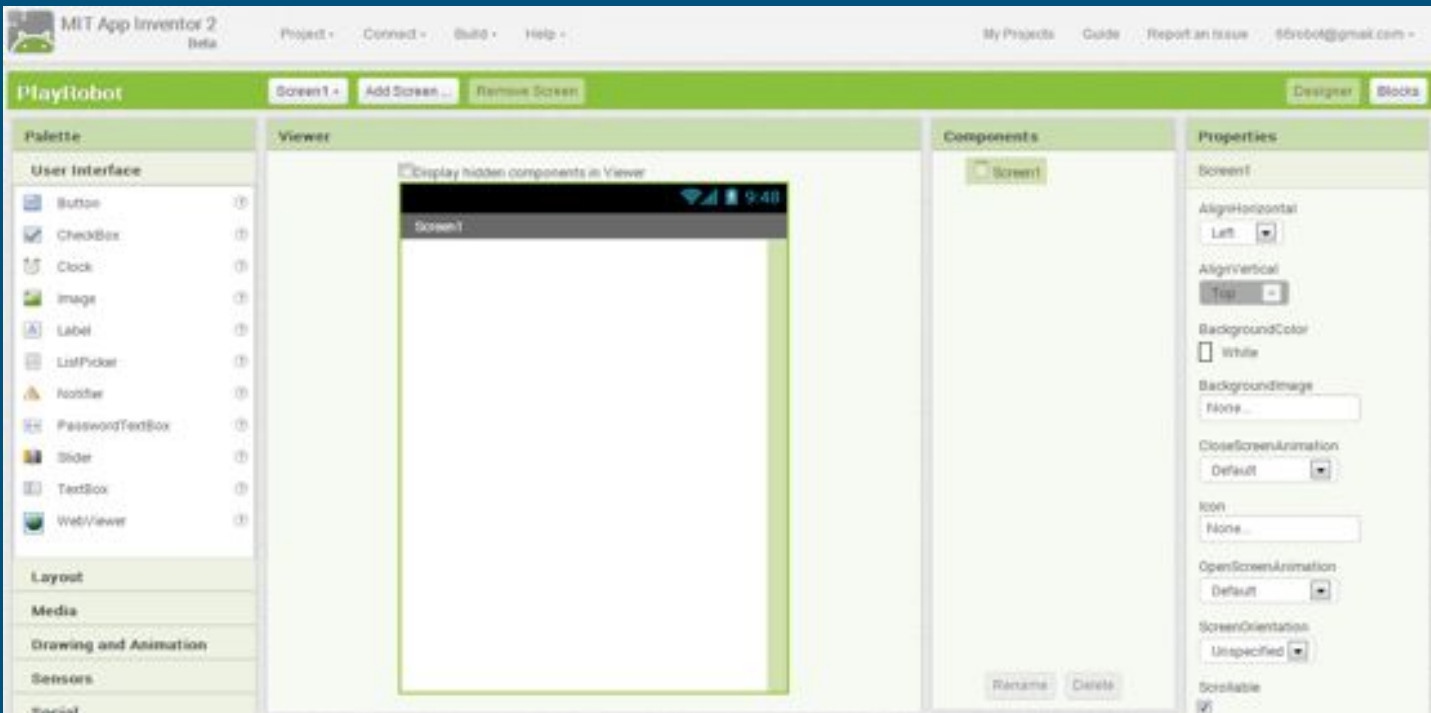
My Projects Guide

New Project... Delete Project

Projects

| Name | Date Created | Date Modified ▼ |
|---------------------------------------|---------------------|---------------------|
| <input type="checkbox"/> InventorDemo | 2014 Jan 7 09:34:41 | 2014 Jan 7 09:35:06 |
| <input type="checkbox"/> ABC | 2014 Jan 7 09:33:32 | 2014 Jan 7 09:33:44 |
| <input type="checkbox"/> UIT | 2014 Jan 7 09:26:39 | 2014 Jan 7 09:33:09 |

已建立專案



Projects 專案：

開啟、新增、儲存專案。

Connect 連結：

將程式經由 USB 同步到您的手機操作。

Build 建置：

提供 QR code或是.apk 安裝檔，讓我們可選擇掃描或下載的途徑來安裝程式。

Help 輔助：

提供教學資源一如：Library 指令說明，Get Started 新手上路、Tutorials 範例教學、Troubleshooting 問題排除以及 Forums 論壇等。

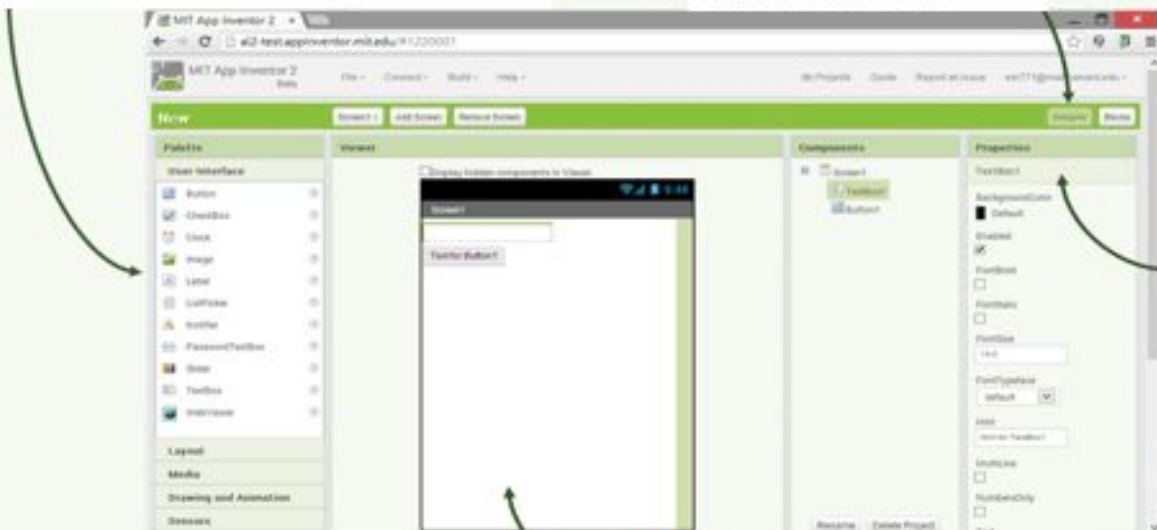


Palette :

將您需要的材料(Component)拉至前置面板上，作為個人app的顯示介面

Designer Button :

按此前往前置面板



Properties屬性：
由此改變元件的
顏色、字體大小、
字型...等多種顯
示設定

Viewer :

即是您的app呈現給使用者的樣貌

Built-In Drawers :

將需要的圖塊(Blocks)拉至後置面板，
要求app執行對應的功能

Blocks Button :

按此返回後置面板

Component-Specific

Drawers :

指定特定元件
(Component)的執行
動作

Block: Snap Blocks
together to set app
behavior.

Viewer :

後置面板的介面，由此顯示各個圖
塊的對應關係與執行動作



MIT App Inventor 2 Beta

Project = Connect = Build = Help =

My Projects Guide Report an Issue appinventor@mit.edu

TalkToMe Screen1 Add Screen Remove Screen Designer Blocks

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - Button1
 - TextToSpeech1
- Any component

Viewer

Built-in Blocks :
指令管理包含數學、文字、邏輯、控制...等分類

Component Blocks選擇對應到特定元件 (Component) 的執行指令

Workspace工作區

Trash垃圾桶 :
欲丟棄的Blocks

Show Warnings

Rename Delete

ai2.appinventor.mit.edu





The application MIT AppInventor Version 2 is requesting permission to access your Google Account.

Please select an account that you would like to use.

● playrobot2013@gmail.com

Google is not affiliated with the contents of MIT AppInventor Version 2 or its owners. If you sign in, Google will store other personal information.

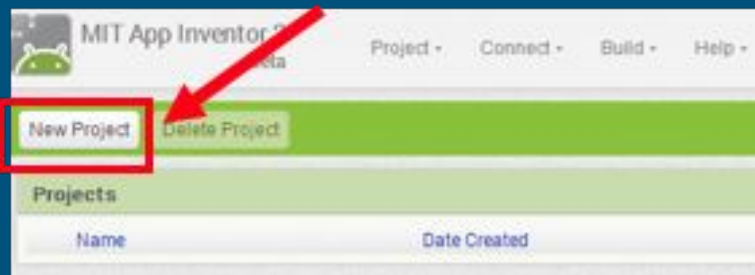
Allow

No thanks

[Sign in to another account](#)

☒ Remember this approval for the next 30 days

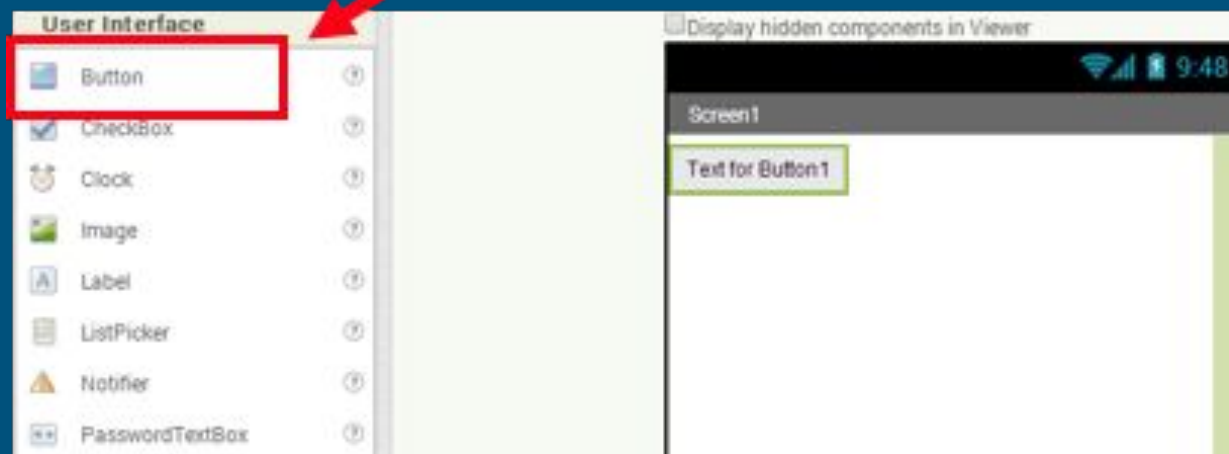
選定帳號後按下此紐



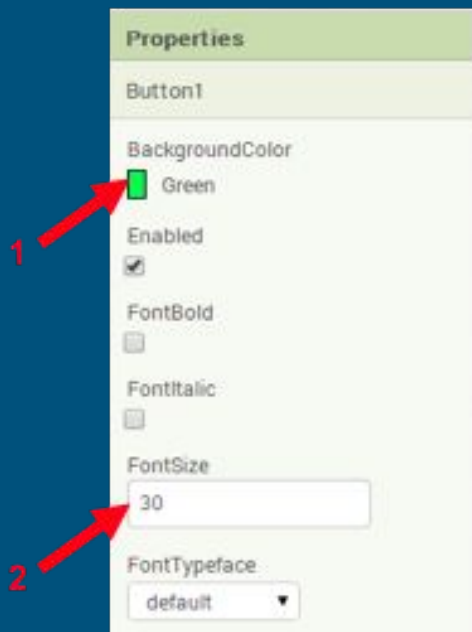
The image shows a dialog box titled 'Create new App Inventor project'. It has a green header bar with the title. Below the header, there is a text input field labeled 'Project name:'. A red arrow points from the right towards the text input field. At the bottom of the dialog box, there are two buttons: 'Cancel' and 'OK'.

輸入"HelloAppInventor"

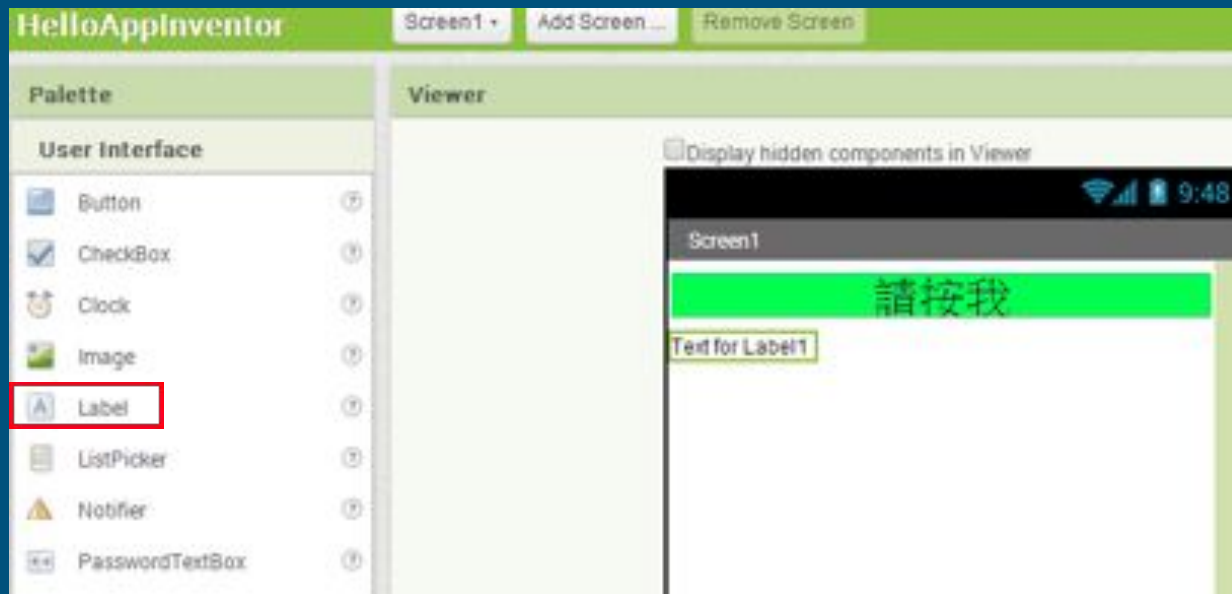
拖曳至右邊



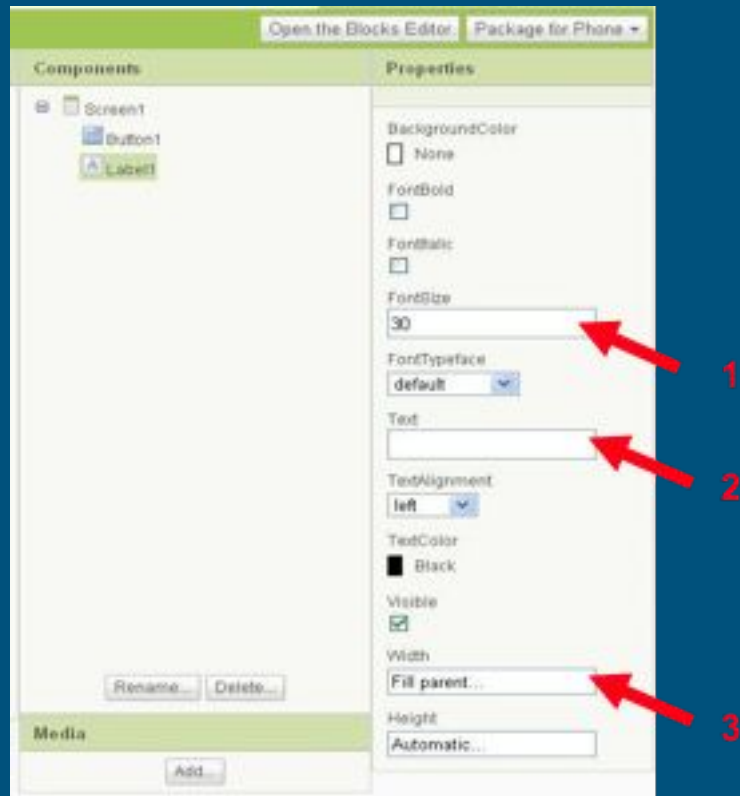
1. 背景顏色設為綠色,
2. 字體大小設為30
3. 文字設為"請按我",
4. 寬度設為"Fill Parent".



由左側的Basic 元件區新增一個標籤Label



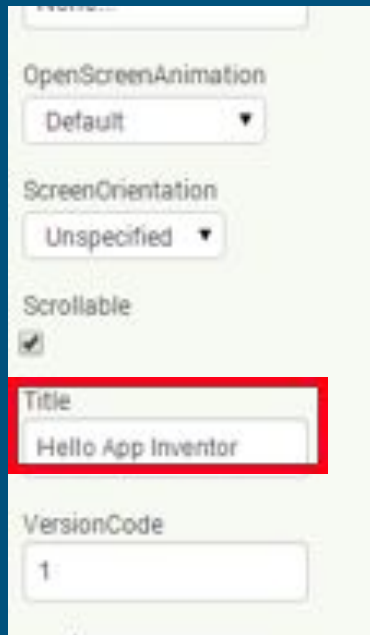
1. 字體大小設為30,
2. 文字清空 (無內容, 因為要另外指定內容)
3. 寬度設為"Fill Parent". 其餘不變.

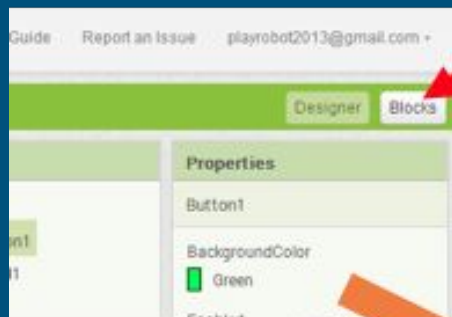


點選Screen1

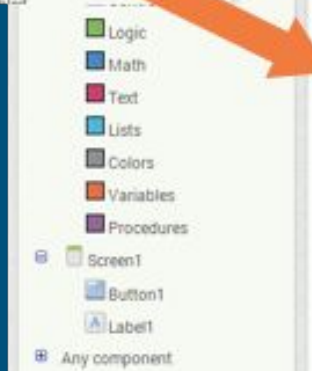


更改右側Title為
"Hello App Inventor"





點選右邊的Blocks鍵



切換至後置面板



點選Button1

The image shows a visual programming environment with two main panels: 'Blocks' and 'Viewer'.

Blocks Panel:

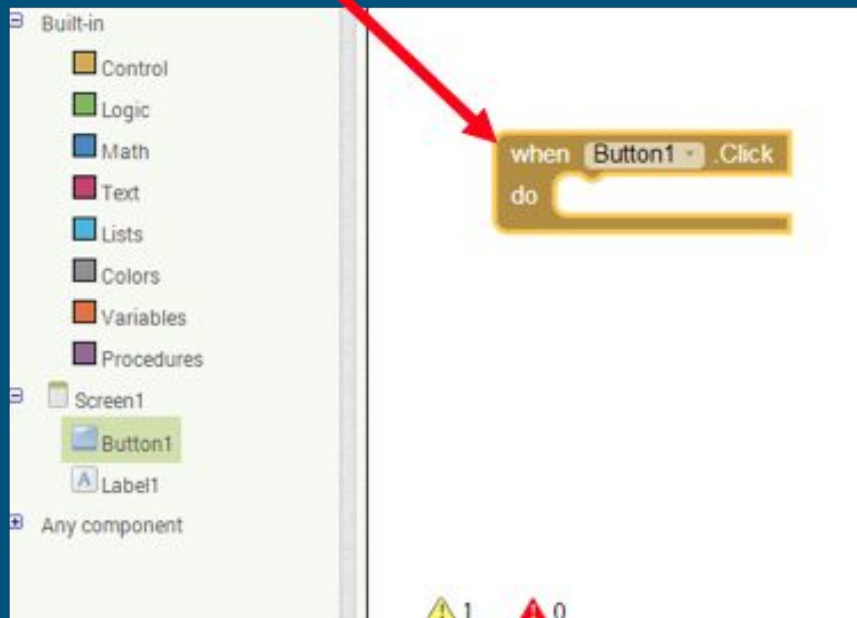
- Expanded category: Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
 - Screen1
 - Button1** (highlighted with a red arrow)
 - Label1
- Any component

Viewer Panel:

The Viewer panel displays a sequence of code blocks for 'Button1':

- when Button1 . Click
 - do
- when Button1 . GotFocus
 - do
- when Button1 . LongClick
 - do
- when Button1 . LostFocus
 - do
- Button1 . BackgroundColor
- set Button1 . BackgroundColor to
- Button1 . Enabled

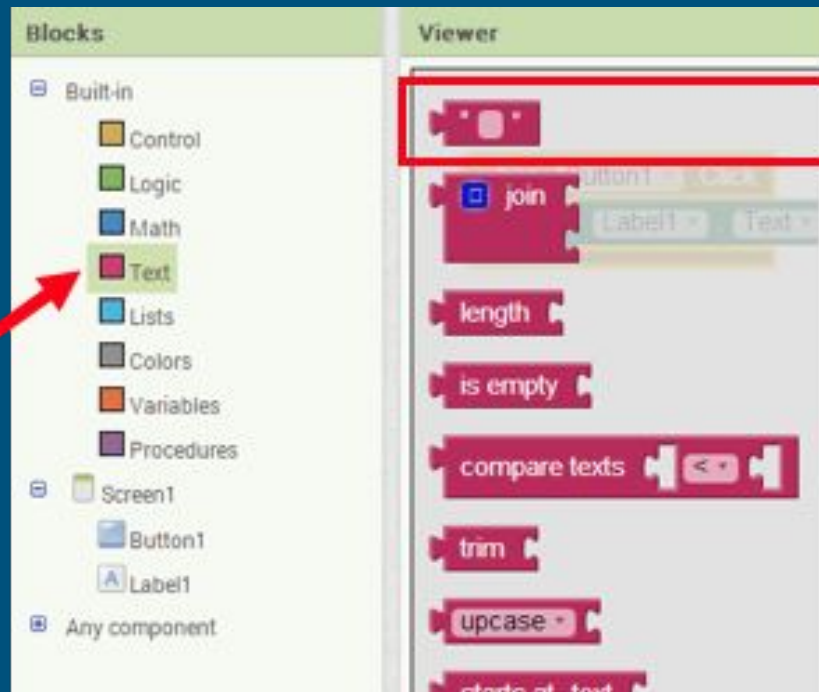
選擇第一個選項拖曳置後置面板







點選Text



選擇此項



The screenshot displays the App Inventor interface with two main panes: 'Blocks' on the left and 'Viewer' on the right. The 'Blocks' pane shows a category list with 'Text' selected. The 'Viewer' pane shows a code block for the 'when Button1.Click' event, containing a 'do' block with the instruction 'set Label1.Text to' followed by a text block containing the string 'Hello App Inventor'. A red arrow points from the Chinese text below to the text block in the code.

Blocks

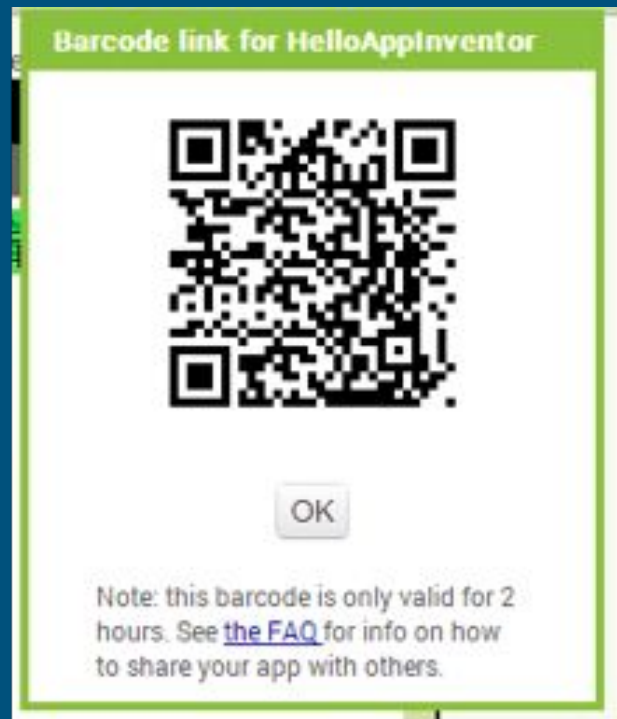
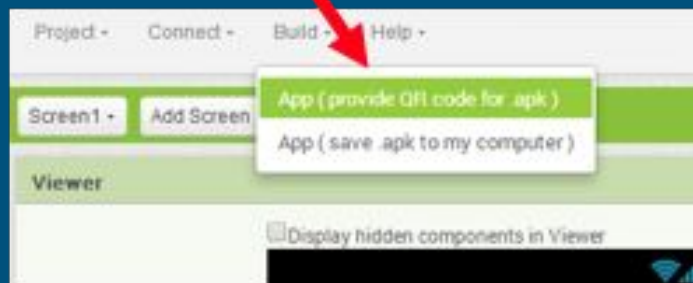
- Built-in
 - Control
 - Logic
 - Math
 - Text**
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - Button1
 - Label1
- Any component

Viewer

```
when Button1.Click  
do  
  set Label1.Text to "Hello App Inventor"
```

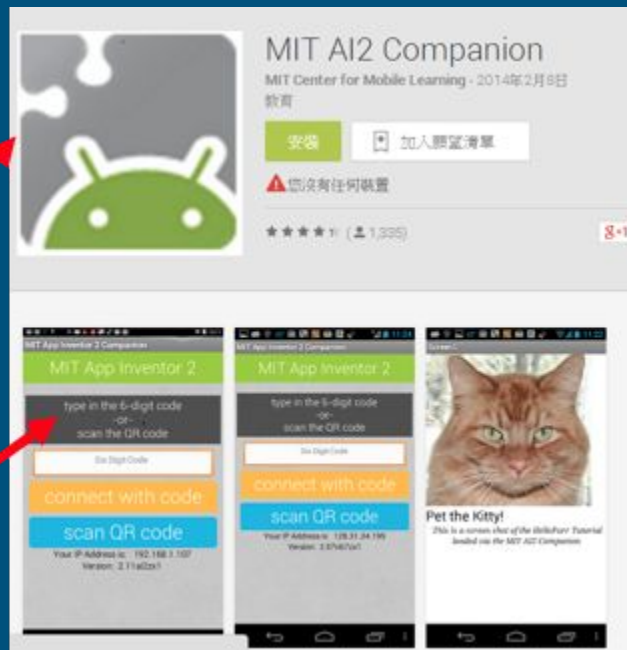
將“Text 的空白方塊”拖曳到“Label1.Text to”的方塊後方，並於內容中打上“Hello App Inventor”

點選上方選擇建立QR Code



從Google “Play商店”中下載並安裝 “MIT AI2 Companion” 這個App到手機裡。

MIT AI2 Companion圖示



執行圖示



此應用APP程式只要兩者皆在同一個WiFi網域下就可以直接使用QR碼來安裝設計完成的APP程式，如無法順利產生安裝檔，則僅需重新生成APP Inventor的QR碼再重新掃描即可



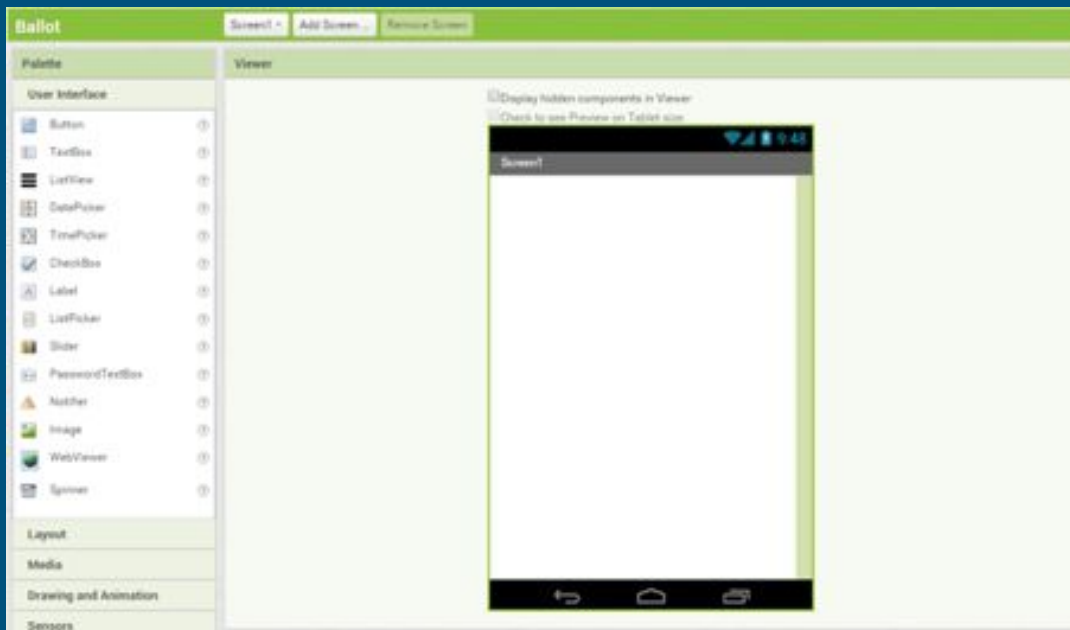


練習

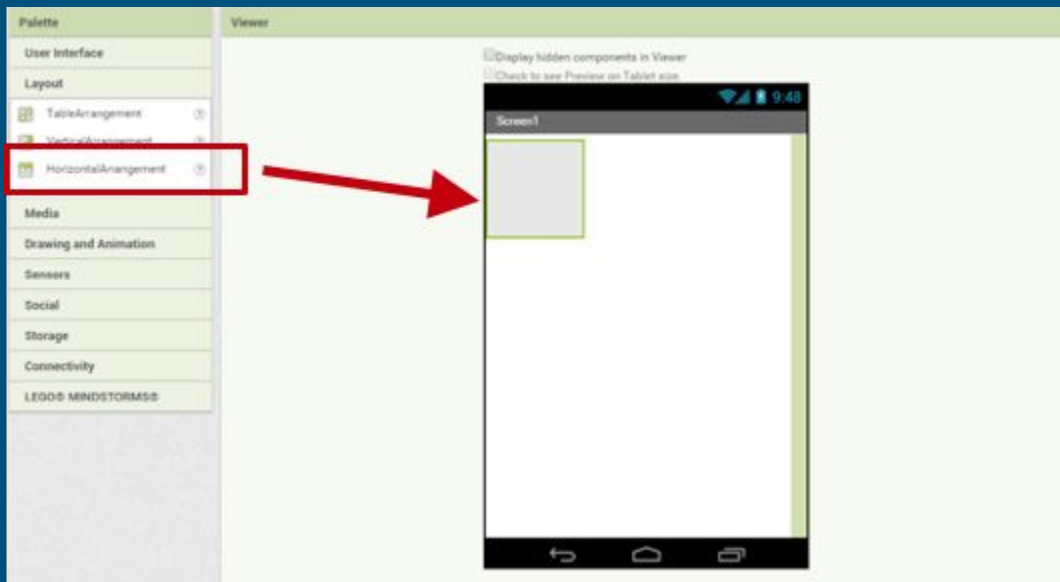
新增一個按鈕控制文字變成 Hello Arduino

抽籤

新建立一個專案，命名為“Ballot”



首先新增一個水平Layout。



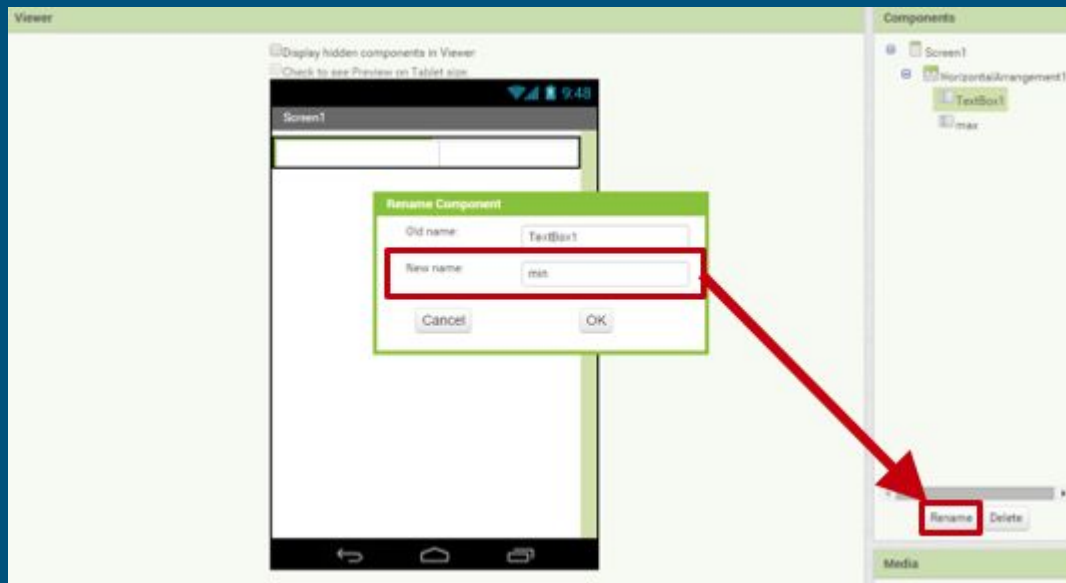
在Layout的Width屬性中，改成“Fill parent”。



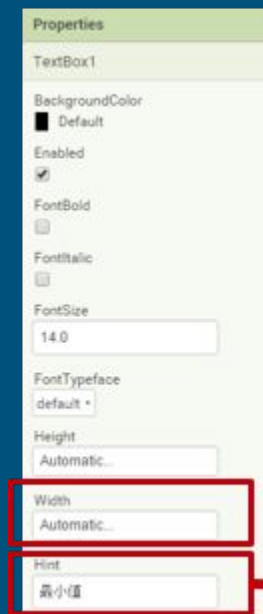
在Layout裡，新增兩個TextBox。



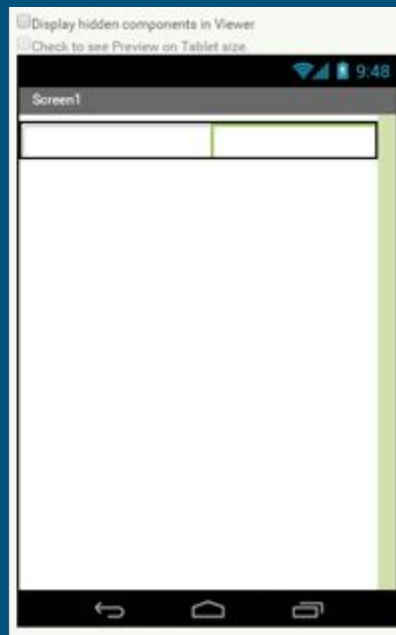
TextBox1的名稱, 把它改成“min”, TextBox2的名稱改成“max”。



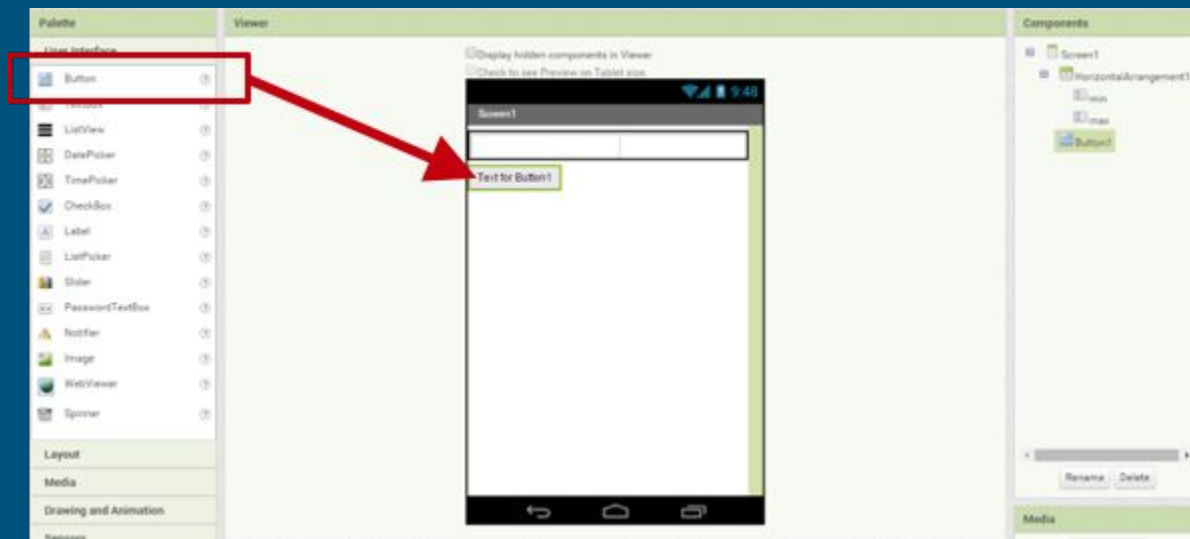
在TextBox的Width屬性中，改成“Fill parent”，把TextBox1的Hint屬性改成“最小值”。



輸入預設顯示文字



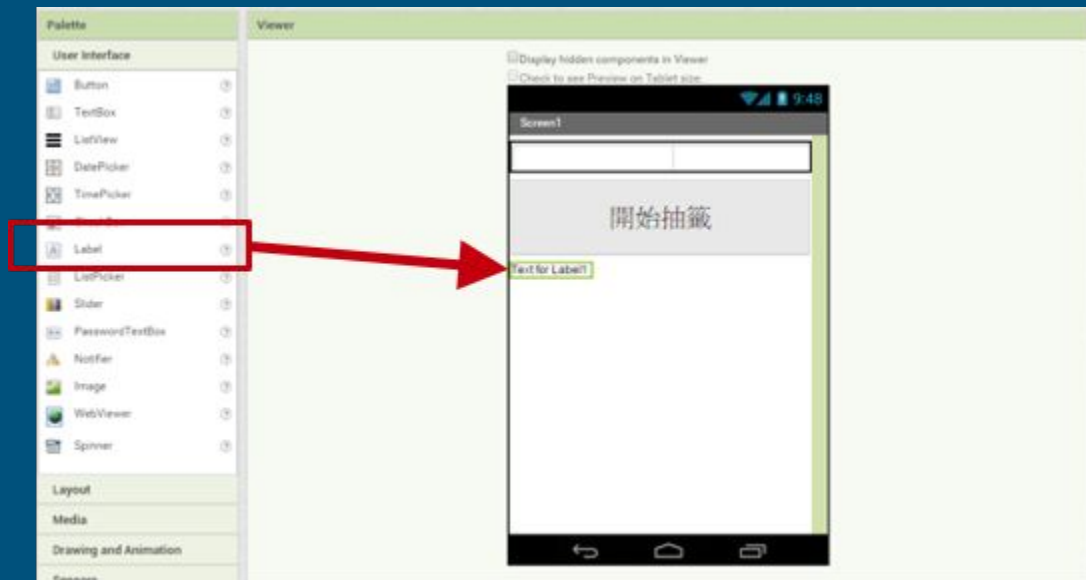
新增一個Button。



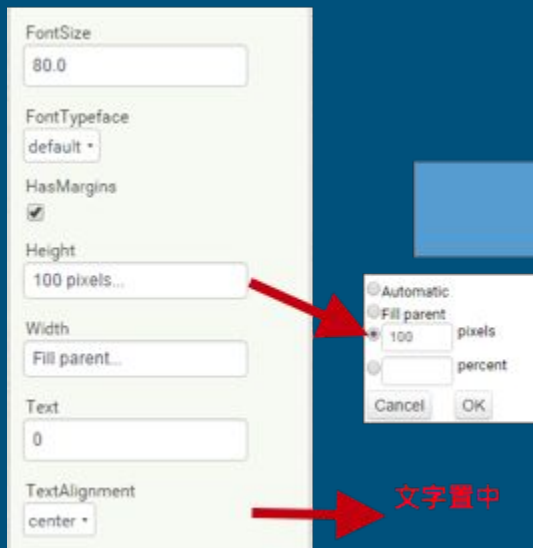
在Button的Width屬性中，改成"Fill parent"，Height改成80，把按鈕的Text改成"開始抽籤"，字型大小改成30.0。



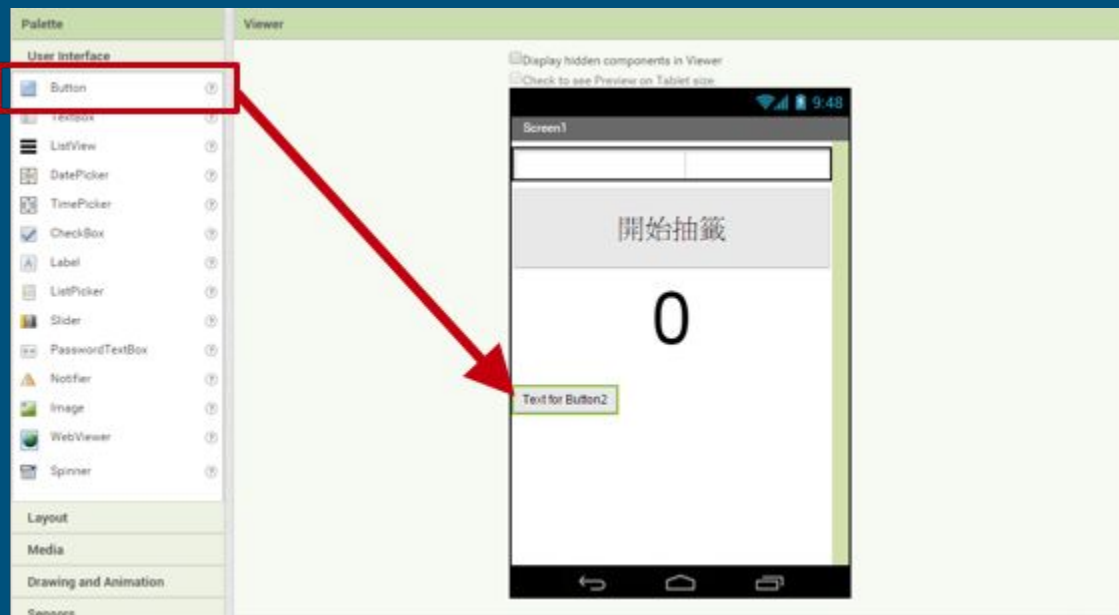
新增一個Label。



Step 10、在Label的Width屬性中，改成"Fill parent"，把Height屬性改成100，文字輸入"0"，文字置中，字型大小改成80.0。



新增一個Button。



在Label的Width屬性中，改成“Fill parent”，文字輸入“重新設定”。



輸入顯示
文字

新增一個Notifier警示訊息提示。



編輯程式



Blocks

Viewer

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Button2

0

=

+

-

×

/

^

random integer from

1

to

100

initialize global setup to

0

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables**
- Procedures

Screen1

- HorizontalArrangement1
 - min
 - max
- Button1
- LabelResult
- Button2
- Notifier1

Viewer

initialize global name to

get

set to

initialize local name to

in

initialize local name to

in

initialize global setup to

0

initialize global list to

Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Button2

Notifier1

Any component

Viewer

create empty list

make a list

add items to list
list
itemis in list? thing
list

length of list list

is list empty? list

pick a random item list

index in list thing
listinitialize global **list** to 0initialize global **list** to create empty list

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Viewer

to procedure
do

to procedure
result

call reset

initialize global setup to 0

initialize global list to create empty list

to reset
do

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - Horizontal arrangement1
 - min**
 - Button1
 - LabelResult
 - Button2
 - Notifier1
- Any component

Viewer

call min .RequestFocus

min . BackgroundColor

set min . BackgroundColor to

min . Enabled

set min . Enabled to

min . FontSize

set min . FontSize to

min . Height

set min . Height to

set min . HeightPercent to

initialize global setup to 0

initialize global list to create empty list

to reset

set min . Enabled to

The image displays the Scratch Blocks and Viewer panels. The **Blocks** panel on the left shows the **Logic** category selected, with a red box highlighting it. A red arrow points from this box to a **true** block in the **Viewer** panel. Another red arrow points from the **true** block in the **Viewer** panel to a **true** block within a **do** loop in the main script area.

Blocks Panel:

- Built-in
 - Logic (highlighted)
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
- HorizontalArrangement1
 - min
 - max
- Button1
- LabelResult
- Button2
- Notifier1

Viewer Panel:

- true (highlighted)
- false
- not
- =
- and
- or

Main Script Area:

- initialize global **setup** to 0
- initialize global **list** to create empty list
- to reset
 - do
 - set min - Enabled to true (highlighted)

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

HorizontalArrangement1

min

max

LabelResult

Button2

Notifier1

Viewer

call max . RequestFocus

max . BackgroundColor .

set max . BackgroundColor . to

max . Enabled .

set max . Enabled . to

max . FontSize .

set max . FontSize . to

max . Height .

set max . Height . to

set max . HeightPercent . to

initialize global setup to 0

initialize global list to create empty list

to reset

set min . Enabled . to true

set max . Enabled . to true

The image shows the Scratch IDE interface with the 'Blocks' panel on the left and the 'Viewer' panel on the right. In the 'Blocks' panel, the 'Variables' category is highlighted with a red box. A red arrow points from this box to a 'set * to' block in the 'Viewer' panel, which is also highlighted with a red box. Another red arrow points from the 'set * to' block to a 'do' block in the 'Viewer' panel.

Blocks Panel:

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables**
 - Media
- Screen1
 - HorizontalArrangement1
 - min
 - max
 - Button1
 - LabelResult
 - Button2
 - Notifier1

Viewer Panel:

Code blocks visible in the Viewer:

- initialize global name to
- get
- set * to
- initialize local name in
- initialize local name to in
- initialize global setup to 0
- initialize global list to create empty list
- to reset do
 - set min Enabled to true
 - set max Enabled to true
 - set global setup to 0

Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Button2

Notifier1

Any component

Viewer

initialize global name to

get

set to

initialize local name

in

initialize local name to

in

initialize global setup to 0

initialize global list to create empty list

to reset

do

set min Enabled to true

set max Enabled to true

set global setup to 0

set global list to create empty list

The image shows the MIT App Inventor web interface. On the left is the **Blocks** palette, which is organized into categories: Built-in (Control, Logic, Math, Text, Lists, Colors, Variables, Procedures), Screen1, HorizontalArrangement1 (min, max, Button1, **Button2**), and Any component. The **Button2** component is highlighted with a red box. Two red arrows originate from this box: one points to the **when Button2.Click** block in the **Viewer** pane, and the other points to the **when Button2.Click** block in the **Code** view.

The **Viewer** pane displays a series of event-driven blocks for **Button2**:

- when Button2.Click do
- when Button2.GotFocus do
- when Button2.LongClick do
- when Button2.LostFocus do
- when Button2.TouchDown do
- when Button2.TouchUp do

The **Code** view shows the following code blocks:

```
initialize global setup to 0
initialize global list to create empty list

to reset
do
  set min.Enabled to true
  set max.Enabled to true
  set global setup to 0
  set global list to create empty list

when Button2.Click
do
```


Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Media

Procedures

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Button2

Notifier1

Any component

Viewer

to procedure

do

to procedure

result

call reset

initialize global setup to 0

initialize global list to create empty list

to reset

do

set min - Enabled - to true

set max - Enabled - to true

set global setup to 0

set global list to create empty list

when Button2 - Click

do call reset

Blocks

Viewer

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

HorizontalArrangement1

min

Button1

LabelResult

Button2

Notifier1

Any component

when Button1 .Click

do

when Button1 .GotFocus

do

when Button1 .LongClick

do

when Button1 .LostFocus

do

when Button1 .TouchDown

do

when Button1 .TouchUp

do

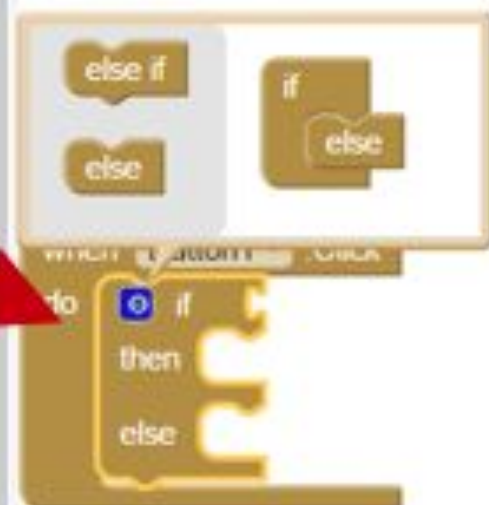
when Button1 .Click

do

Blocks



Viewer



Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Button2

Notifier1

Viewer

true

false

not

=

and

or

when Button1 Click

do

then

else

and

Blocks

Built-in

Logic

Text

Lists

Colors

Variables

Procedures

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Button2

Notifier1

Any component

Viewer

true

false

not

=

and

or

when Button1 Click

do if

then

else

and

and

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

- HorizontalArrangement1
 - min
 - max
- Button1
- LabelResult
- Button2
- Notifier1

Any component

Media

Viewer

round

ceiling

floor

modulo of

sin

cos

tan

atan2

y

x

convert radians to degrees

format as decimal number

places

is number?

when button Click

do


if

then

else

is number? and is number? and

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
 - Screen1
 - min -  min
 - Button1
 - LabelResult
 - Button2
 - NumVar1
 - Any component
- Rename Delete

Viewer



```
set min Hint to
min MultLine
set min MultLine to
min NumbersOnly
set min NumbersOnly to
min Text
set min Text to
min TextColor
set min TextColor to
min Visible
set min Visible to
min Width
```

Click

if number? min Text and if number? max Text and

Blocks

Built-in

Control

Math

Lists

Colors

Variables

Procedures

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Button2

Notifier1

Any component

Viewer

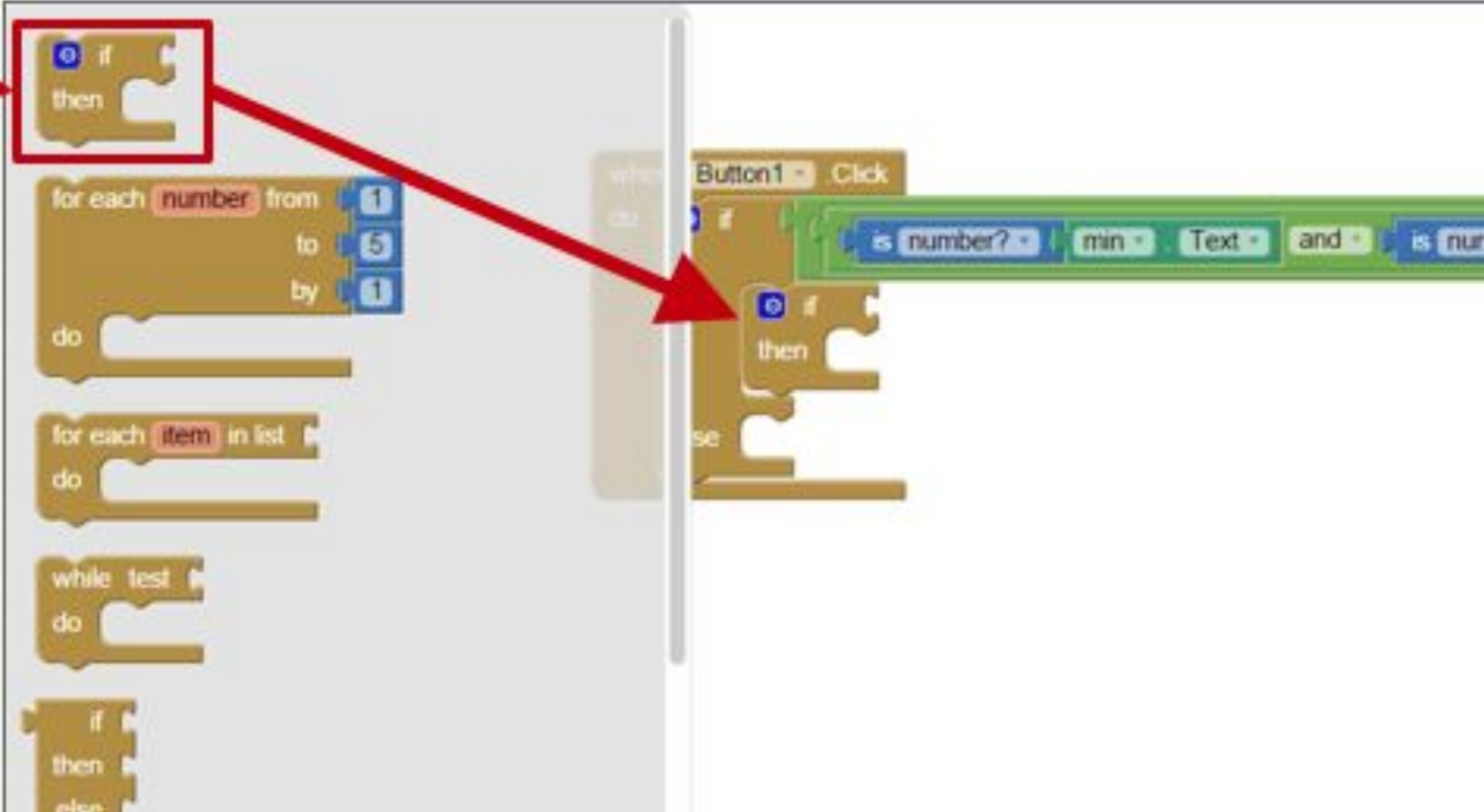
The Viewer displays a sequence of blocks in a script area. The first block is a blue '0' block. The second block is a blue 'Math' block with an equals sign icon, highlighted by a red box. The third block is a blue 'Math' block with a plus sign icon. The fourth block is a blue 'Math' block with a minus sign icon. The fifth block is a blue 'Math' block with a multiply sign icon. The sixth block is a blue 'Math' block with a divide sign icon. The seventh block is a blue 'Math' block with a power sign icon. The eighth block is a blue 'random integer from' block with '1' and '100' in the input fields. The ninth block is a blue 'random fraction' block. The tenth block is a blue 'random set seed to' block.

The HorizontalArrangement1 block contains the following expressions: 'max - Text -', 'and -', 'min - Text -', '< -', 'max - Text -'.

Blocks

- Control
- Math
- Text
- Lists
- Colors
- Variables
- Procedures
- Screen1
- HorizontalArrangement1
 - min
 - max
- Button1
- LabelResult
- Button2
- Notifier1
- Any component

Viewer



Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Variables
- Procedures

Screen1

- HorizontalArrangement1
 - min
 - max
- Button1
- LabelResult
- Button2
- Notifier1

Viewer

initialize global name to

get

set to

initialize local name to

in

initialize local name to

in

when Button1 Click

do

then

is number? min Text and is number? max

get global setup

global list

✓ global setup

Blocks

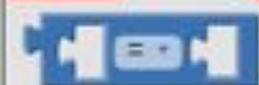
Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

- HorizontalArrangement1
 - min
 - max
- Button1
- LabelResult
- Button2
- Notifier1

Viewer



The image displays the Scratch interface, divided into a **Blocks** panel on the left and a **Viewer** panel on the right.

Blocks Panel:

- Built-in** category is expanded, showing sub-categories: Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures.
- The **Control** sub-category is highlighted with a red box.
- A red arrow points from the **Control** sub-category to the **for each** block in the Viewer.

Viewer Panel:

- The **for each** block is highlighted with a red box.
- A red arrow points from the **for each** block to its instance in the script area.
- The script area shows a sequence of blocks:
 - when Button1 Click** block.
 - if** block with a condition `is number? - min - Text - and - is num`.
 - then** block containing:
 - if** block with condition `get global setup - == 0`.
 - for each** block (highlighted with a red box) with parameters:
 - number** (variable)
 - from** `1`
 - to** `5`
 - by** `1`
 - do** block.
 - else** block.

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists**
- Colors
- Variables
- Procedures

Screen1

HorizontalArrangement1

- min
- max
- Button1
- LabelResult
- Button2
- Notifier1

Any component

Viewer

create empty list

make a list

add items to list list
item

is in list? thing
list

length of list list

is list empty? list

pick a random item list

index in list thing
list

Click

is number? min Text and is number?

get global setup = 0

for each number from min Text
to max Text
by 1

add items to list list
item

Blocks

Built-in

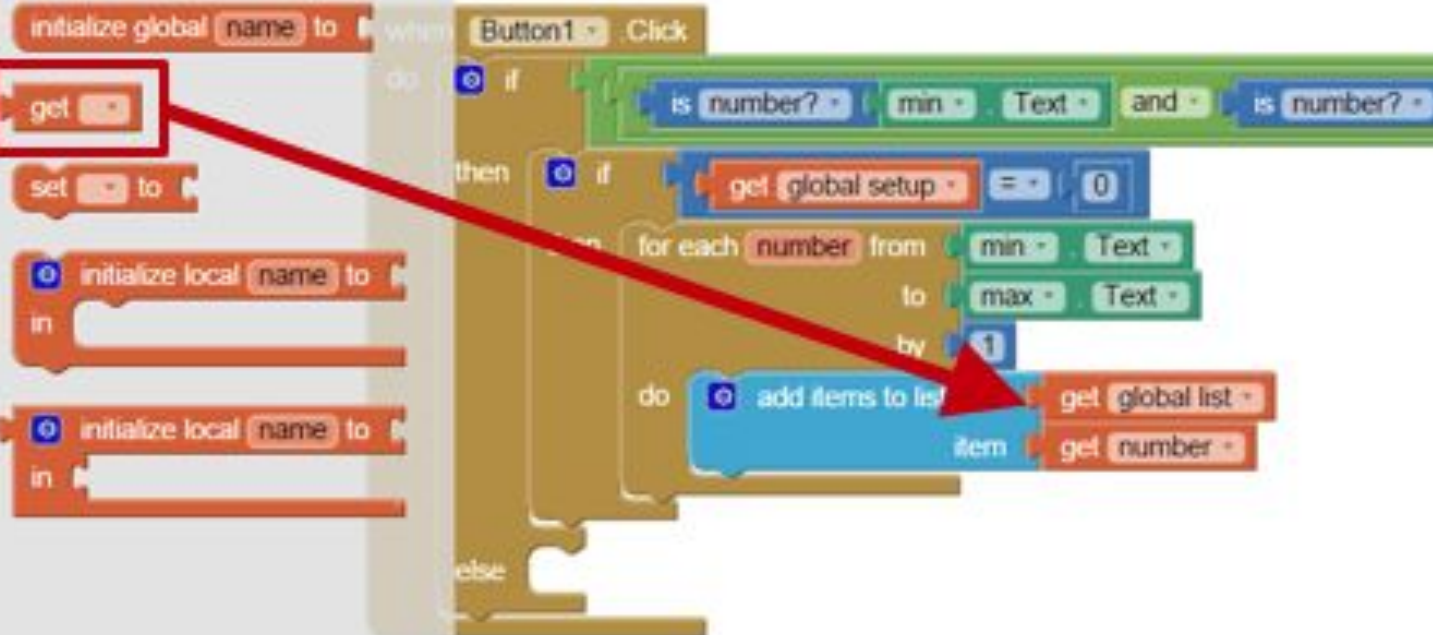
- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables**
- Procedures

Screen1

HorizontalArrangement1

- min
- max
- Button1
- LabelResult
- Button2
- Notifier1

Viewer



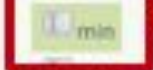
Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

Horizontal Arrangement1



- max
- Button1
- LabelResult
- Button2
- Notifier1

Any component

Viewer

```
call min . RequestFocus  
min . BackgroundColor .  
set min . BackgroundColor . to  
min . Enabled .  
set min . Enabled . to  
min . FontSize .  
set min . FontSize . to  
min . Height .  
set min . Height . to  
set min . HeightPercent . to
```

```
Click  
is number? . min . Text . and . is number? .  
get global setup . = . 0  
for each number from min . Text .  
to max . Text .  
by 1  
do  
add items to list list get global list .  
item get number .  
set min . Enabled . to false .  
set max . Enabled . to false .
```

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables**
- Procedures

Screen1

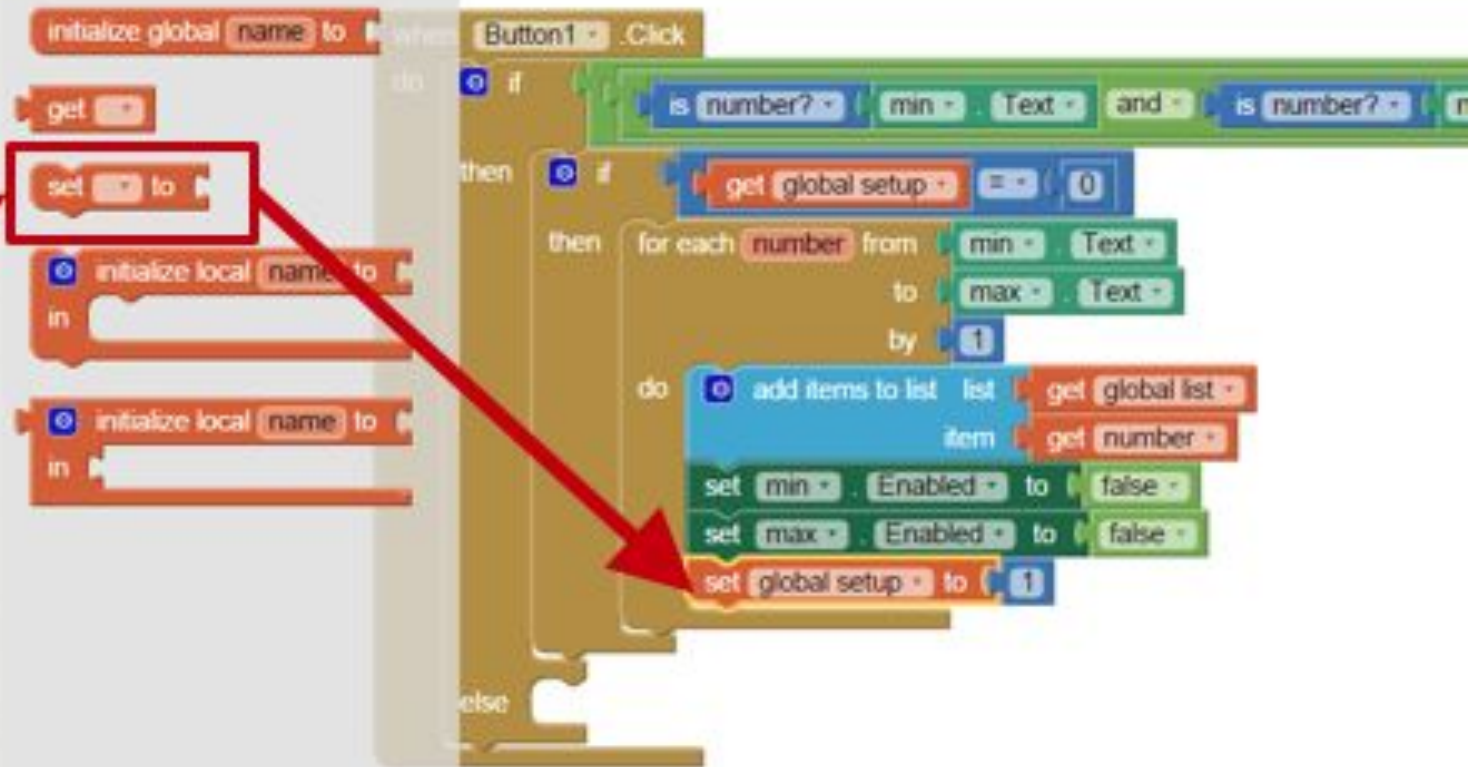
HorizontalArrangement1

- min
- max

- Button1
- LabelResult
- Button2
- Notifier1

Any component

Viewer



Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Button2

Notifier1

Any component

Viewer

set LabelResult . FontSize to

LabelResult . HasMargins

set LabelResult . HasMargins to

LabelResult . Height

set LabelResult . Height to

set LabelResult . HeightPercent to

LabelResult . Text

set LabelResult . Text to

LabelResult . TextColor

set LabelResult . TextColor to

when Button1 . Click

do

then

for each number from min . Text
to max . Text
by 1

do

add items to list list
item get global list
get number

set min . Enabled to false
set max . Enabled to false
set global setup to 1

set LabelResult . Text to

else

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists**
- Variables
- Procedures

Screen1

- HorizontalArrangement1
 - min
 - max
- Button1
- LabelResult
- Button2
- Notifier1

Viewer

is in list? thing list

length of list list

is list empty? list

pick a random item list

index in list thing list

select list item list index

insert list item list index item

when Button1 Click

do

if

is number? min Text and is number? max Text

then

if

get global setup == 0

then

for each number from min Text to max Text by 1

do

add items to list list item

get global list

set min Enabled to false

set max Enabled to false

set global setup to 1

set LabelResult Text to pick a random item list get global list

新增一個get

The image shows the Scratch interface with the 'Lists' category selected in the 'Blocks' palette. A red box highlights the 'Lists' category, and a red arrow points from it to the 'remove list item' block in the 'Viewer' area. Another red box highlights the 'remove list item' block, and a red arrow points from it to the 'get global list' block in the script area. The script area shows a complex script with multiple 'get global list' blocks highlighted by red boxes. The text '此三個get方塊是一樣的' (These three get blocks are the same) is written in Chinese at the bottom right.

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists**
 - Colors
 - Variables
 - Procedures
- Screen1
 - HorizontalArrangement1
 - min
 - max
 - Button1
 - LabelResult
 - Button2
 - Notifier1
 - Any component

Viewer

insert list item list index item

replace list item list index replacement

remove list item list index

append to list list1 list2

copy list list

is a list? thing

list to new name. Set

if

is number? min Text and is number? max Text

then

if

get global setup = 0

then

for each number from min Text to max Text by 1

do

add items to list list item

get global list

get number

set min Enabled to false

set max Enabled to false

set global setup to 1

set LabelResult to random item list

get global list

remove list item list index

get global list

else

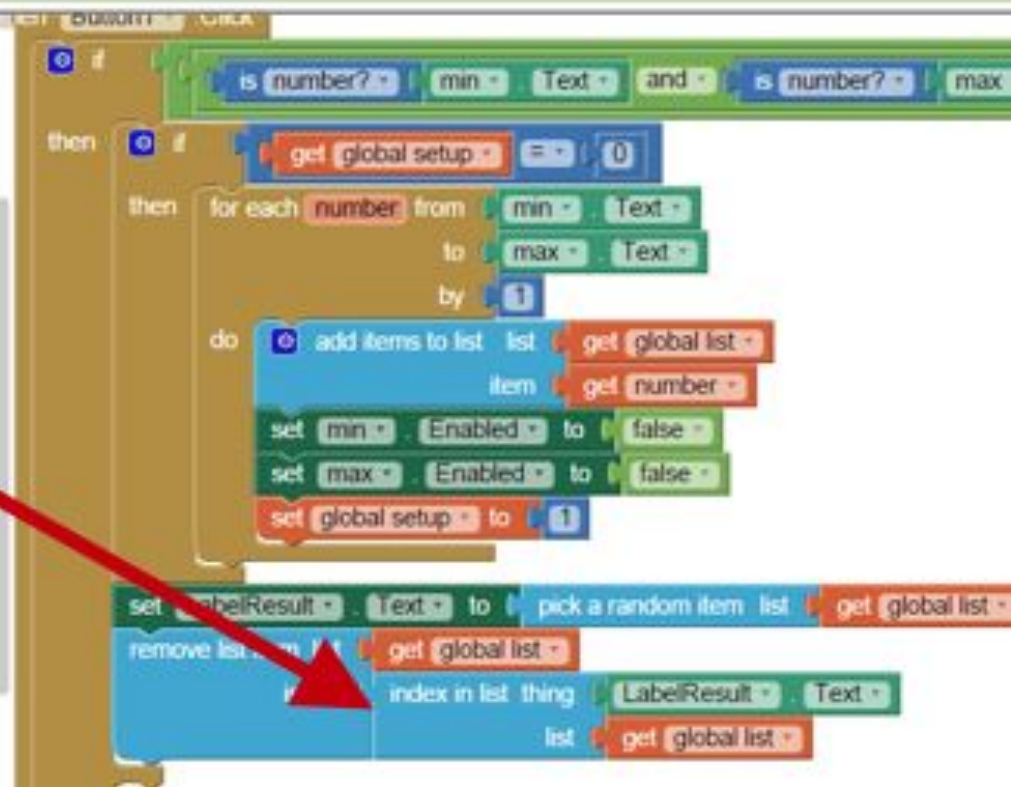
此三個get方塊是一樣的

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - HorizontalArrangement1
 - min
 - max
 - Button1
 - LabelResult
 - Button2
 - Notifier1
- Any component

Viewer

- is in list? thing list
- length of list list
- is list empty? list
- pick a random item list
- index in list thing list
- select list item list index
- insert list item list index item
- replace list item list index



Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Button2

Notifier1

Any component

Viewer

if
then

for each number from 1 to 5 by 1

do

for each item in list

do

while test

do

if
then

set min . Enabled

set max . Enabled

set global setup . to

set LabelResult . Text . to

remove list item list
indexget global list
index in listif
then

else

Blocks

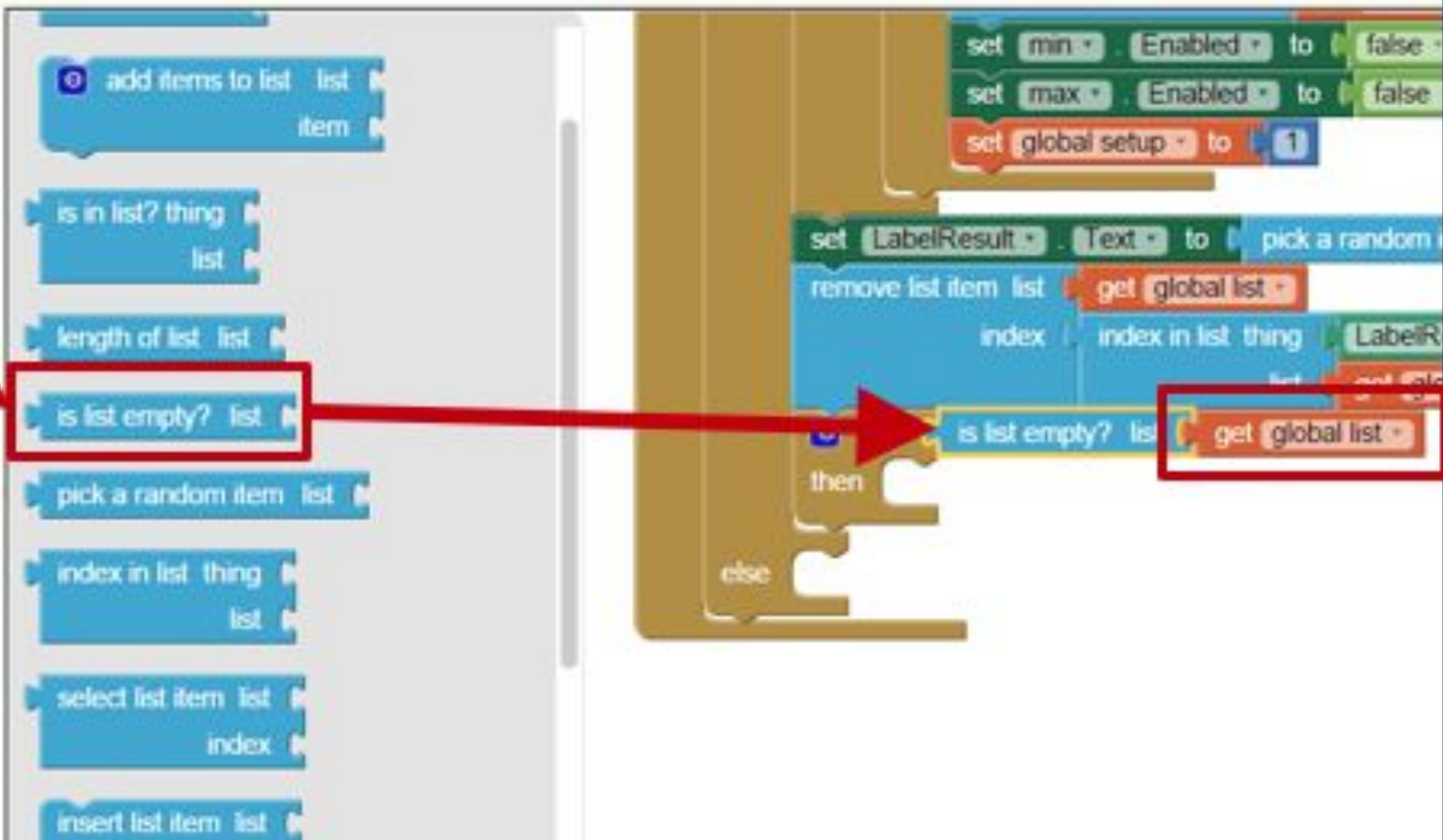
Built-in

- Control
- Logic
- Math
- Lists**
- Colors
- Variables
- Procedures

Screen1

- HorizontalArrangement1
 - min
 - max
 - Button1
 - LabelResult
 - Button2
 - Notifier1
- Any component

Viewer



Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors

Procedures

Screen1

HorizontalArrangement1

- min
- max
- Button1
- LabelResult
- Button2
- Notifier1

Any component

Viewer

to procedure

do

to procedure

result

call reset

set min - Enabled - to fa

set max - Enabled - to fa

set global setup - to 1

set LabelResult - Text - to pick a rand

remove list item list get global list

index index in list thing

list get

if is list empty? list get global list

call reset

else

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

HorizontalArrangement1

min

max

Button1

LabelResult

Notifier1

Viewer

message

call Notifier1 .LogInfo

message

call Notifier1 .LogWarning

message

call Notifier1 .ShowAlert

notice

call Notifier1 .ShowChooseDialog

message

title

button1Text

button2Text

cancelable

true

call Notifier1 .ShowMessageDialog

set min .Enabled to false

set max .Enabled to false

set global setup to 1

set LabelResult .Text to pick a random item list

remove list item list

index

index in list thing

LabelResult .Text

list

get global list

if is list empty? list

then

call reset

else

call Notifier1 .ShowAlert

notice

Error

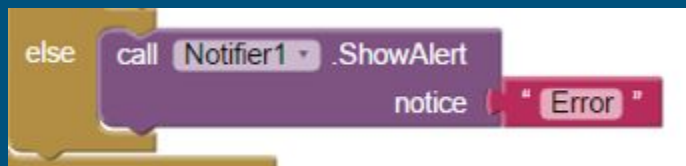
我們根據 setup 這個 flag 變數來判斷，在執行完上下限數值所決定的次數之前，兩個輸入欄位都不能再輸入數值 (當然按下 [重設] 按鈕就重來了)。例如下限為1，上限為10，則您需要按10次 [抽籤] 按鈕才算是一次完整的抽籤完成程序。第一次抽籤時建立一個包含所選範圍內所有數值的 list 陣列。



使用 pick random item 指令從 list 陣列中隨機抽取內容，並把抽取出來的 item 從 list 中移除，這樣就不會抽到重複的數字囉。
如果 list 內容為空，則呼叫 reset 副程式來重設畫面。



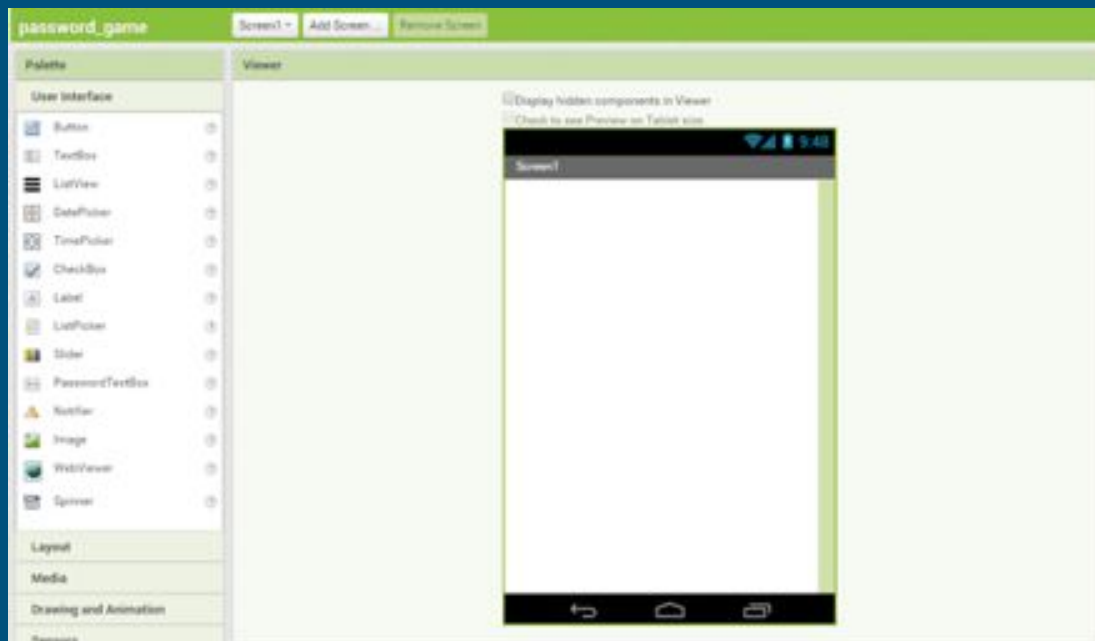
如果有錯誤則使用 Notifier元件顯示錯誤訊息，例如下限值比上限值來得大或是欄位為空等等。





終極密碼

新建立一個專案，命名為“password_game”



Palette

Viewer

User Interface

- Button
- TextBox
- ListView
- DatePicker
- TimePicker
- CheckBox
- Label**
- DatePicker
- Slider
- PasswordTextBox
- Notifier
- Image
- WebView
- Spinner

Layout

Media

Drawing and Animation

Sensors

Display hidden components in Viewer

Check to see Preview on Tablet size

Screen1

Text for Label1

Properties

Label1

BackgroundColor

☐ None

FontBold

☐

FontItalic

☐

FontSize

14.0

FontTypeface

default *

HasMargins

☒

Height

Automatic...

Width

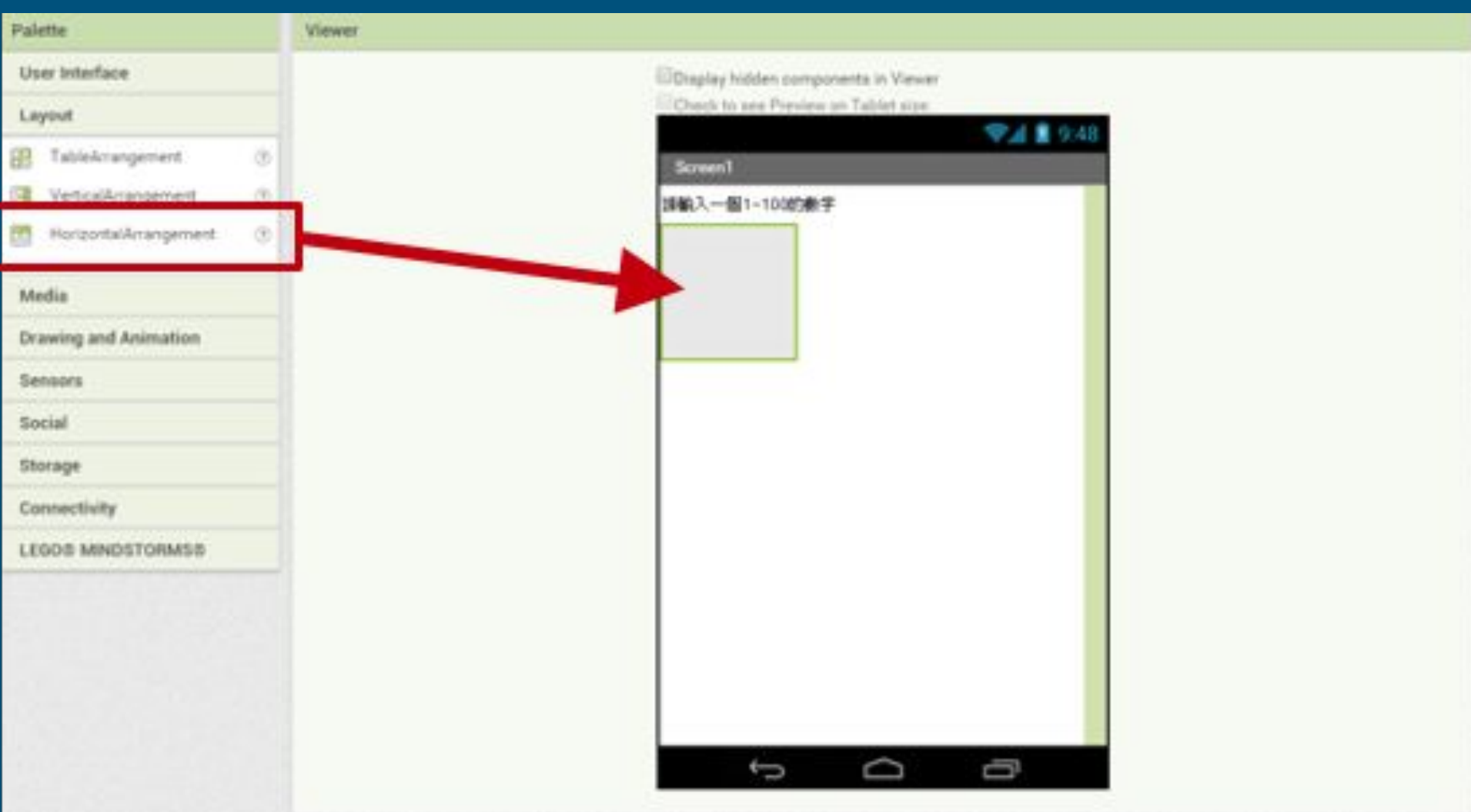
Automatic...

Text

請輸入一個1~100的數字



輸入顯示
文字





Properties

TextBox1

BackColor

Default

Enabled



FontBold



FontItalic



FontSize

14.0

FontTypeface

default ▾

Height

Automatic...

Width

Automatic...

Hint

請輸入數字



☐ Display hidden components in Viewer

☐ Check to see Preview on Tablet size.



輸入提示
文字

FontSize

14.0

FontTypeface

default •

Height

Automatic...

Width

Automatic...

Image

None...

Shape

default •

ShowFeedback



Text

確定

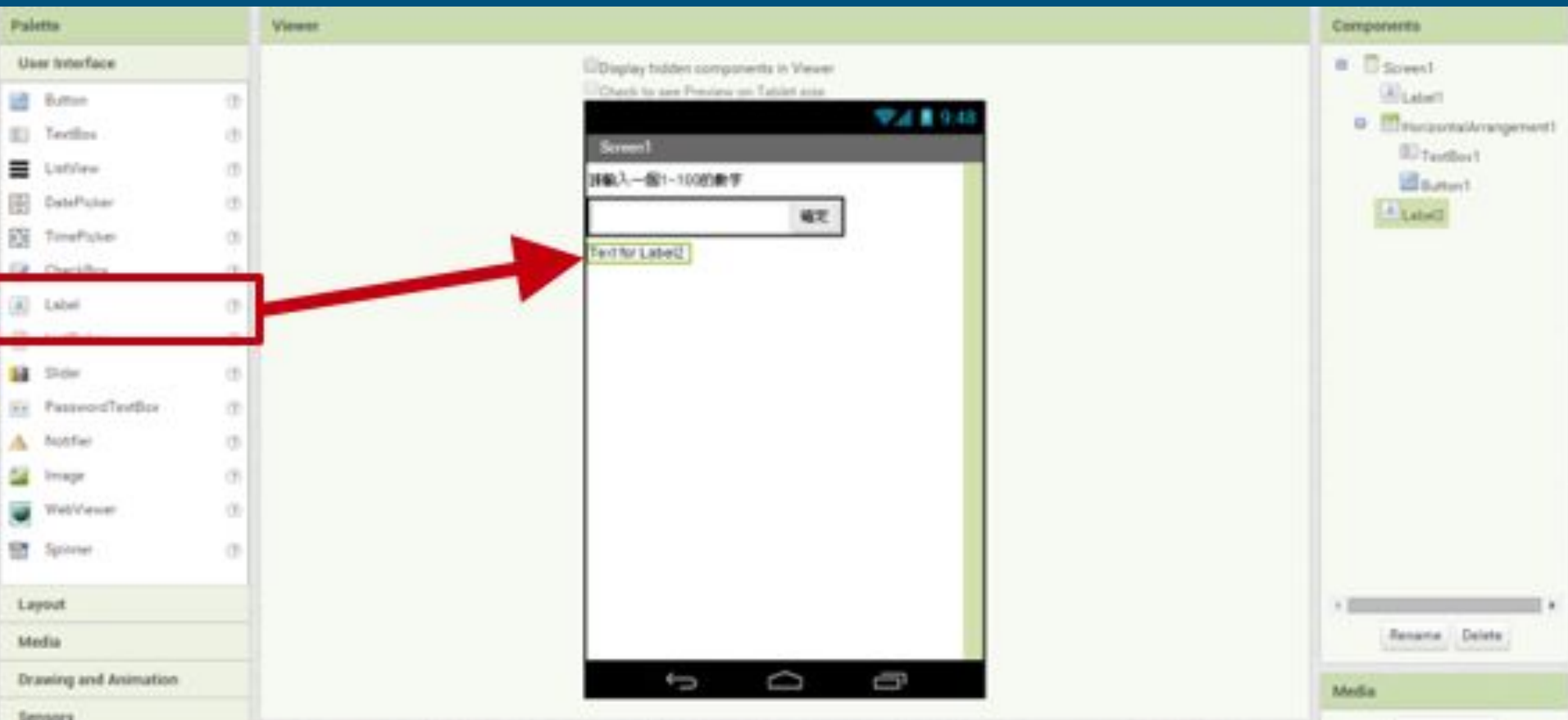


輸入顯示
文字

☐ Display hidden components in Viewer

☐ Check to see Preview on Tablet size





FontSize

14.0

FontTypeface

default ▾

HasMargins



Height

Automatic...

Width

Automatic...

Text



☐ Display hidden components in Viewer

☐ Check to see Preview on Tablet size.

9:48

Screen1

請輸入一個1-100的數字

確定



編輯程式

```
initialize global password to 0
initialize global gess to ""
initialize global min to 0
initialize global max to 100
```

```
when Screen1 - Initialize
do
  set global password to random integer from 1 to 100

when Button1 - Click
do
  set global gess to TextBox1 - Text
  set TextBox1 - Text to ""
  if
    get global password = get global gess
  then
    set Label2 - Text to "恭喜答對了!"
  else if
    get global password > get global gess
  then
    set global min to get global gess
    set Label2 - Text to 0 join get global min
    join ""
    get global max
  else if
    get global password < get global gess
  then
    set global max to get global gess
    set Label2 - Text to 0 join get global min
    join ""
    get global max
```

Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen1

Label1

HorizontalArrangement1

TextBox1

Button1

Label2

Any component

Viewer

initialize global name to

get

set to

initialize local name to

in

initialize local name to

in

initialize global password to

initialize global gess to

initialize global min to

initialize global max to

Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen1

Label1

HorizontalArrangement1

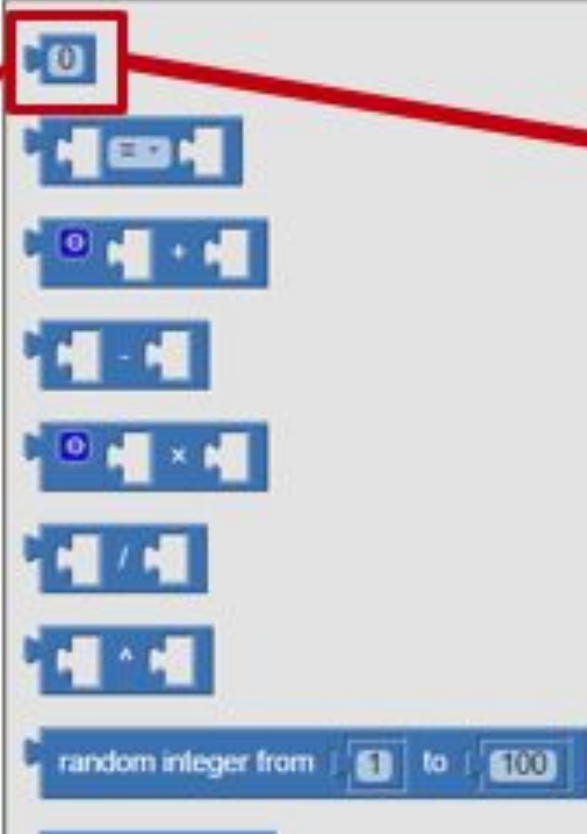
TextBox1

Button1

Label2

Any component

Viewer



The Viewer displays a sequence of math blocks in a stack:

- Block 1: A blue "Math" block with an equals sign (=).
- Block 2: A blue "Math" block with a plus sign (+).
- Block 3: A blue "Math" block with a minus sign (-).
- Block 4: A blue "Math" block with a multiply sign (×).
- Block 5: A blue "Math" block with a divide sign (/).
- Block 6: A blue "Math" block with a power sign (^).
- Block 7: A blue "Math" block labeled "random integer from" with input fields containing "1" and "100".

initialize global password to 0

initialize global guess to 0

initialize global min to 0

initialize global max to 100

Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Screens

Screen1

Label1

HorizontalArrangement1

TextBox1

Button1

Label2

Any component

Viewer

when Screen1 - BackPressed

do

when Screen1 - ErrorOccurred

component

functionName

errorNumber

message

do

when Screen1 - Initialize

do

when Screen1 - OtherScreenClosed

otherScreenName

result

do

when Screen1 - ScreenOrientationChanged

do

when Screen1 - Initialize

do

Blocks

Built-in

Control

Logic

Math

Text

Lists

Variables

Procedures

Screen1

Label1

HorizontalArrangement1

TextBox1

Button1

Label2

Any component

Viewer

initialize global name to

get

set to

initialize local name to

in

initialize local name to

in

set global max to 10.0

when Screen1 .Initialize

do set global password to

The image shows the Scratch development environment. On the left is the 'Blocks' palette, and on the right is the 'Viewer' area showing a script.

Blocks Palette:

- Built-in
 - Control
 - Math (highlighted with a red box)
 - Logic
 - Lists
 - Colors
 - Variables
 - Procedures
 - Screen1
 - Label1
 - HorizontalArrangement
 - Textbox1
 - Button1
 - Label2
 - Any component

Viewer Script:

```
when Screen1 Initialize
do
  set global password to random integer from 1 to 100
```

A red box highlights the 'random integer from 1 to 100' block in the Viewer. A red arrow points from the 'Math' block in the Blocks palette to this block. Another red arrow points from the red box in the Viewer to the 'set global password to' block in the script.

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

Label1

HorizontalArrangement1

TextBox1

Button1

Label2

Any component

Viewer

when Button1 Click

do

when Button1 GotFocus

do

when Button1 LongClick

do

when Button1 LostFocus

do

when Button1 TouchDown

do

when Button1 Click

do

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Calculus
- Variables**
- Procedures

Screen1

- Label1
- HorizontalArrangement1
 - TextBox1
 - Button1
 - Label2

Any component

Viewer

initialize global name to

get

set to

initialize local name to

in

initialize local name to

in

when Button1 Click

set global gess to

- ✓ global gess
- global max
- global min
- global password

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

Label1

HorizontalArrangement1

TextBox1

Label2

Any component

Viewer

TextBox1 - MultiLine -

set TextBox1 - MultiLine - to

TextBox1 - NumbersOnly -

set TextBox1 - NumbersOnly - to

TextBox1 - Text -

set TextBox1 - Text - to

TextBox1 - TextColor -

set TextBox1 - TextColor - to

TextBox1 - Visible -

set TextBox1 - Visible - to

when Button1 - Click

do set global gess - to TextBox1 - Text -

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

Label1

Management1

TextBox1

Button1

Label2

Any component

Viewer

TextBox1 - MultiLine -

set TextBox1 - MultiLine - to

TextBox1 - NumbersOnly -

set TextBox1 - NumbersOnly - to

TextBox1 - Text -

set TextBox1 - Text - to

TextBox1 - TextColor -

set TextBox1 - TextColor - to

TextBox1 - Visible -

set TextBox1 - Visible - to

when Button1 - Click

do set global gess - to TextBox1 - Text -

set TextBox1 - Text - to

The image illustrates the process of adding an 'if-then' block to a Scratch script, showing the Blocks palette, the main workspace, and a detailed view of the conditional logic blocks.

Blocks Palette: The 'Control' category is highlighted with a red box. The 'if-then' block is also highlighted with a red box.

Workspace: The 'if-then' block is being dragged from the palette into the workspace. Below it, several other control blocks are visible: 'for each number from 1 to 5 by 1', 'for each item in list', 'while test', and 'if-then-else'.

Script Area: A script is shown starting with 'when Button1 Click' followed by 'do' blocks: 'set global gess to TextBox1.Text' and 'set TextBox1.Text to []'. Below these, an 'if-then-else' block is being assembled. The 'if' block is highlighted with a red box, and the 'else if' block is also highlighted with a red box. A red arrow points from the 'if-then' block in the workspace to the 'if-then-else' block in the script area.

Conditional Logic Blocks: A detailed view of the conditional logic blocks is shown. The 'if-then-else' block is highlighted with a red box. The 'if' block is highlighted with a red box, and the 'else if' block is also highlighted with a red box. A red arrow points from the 'if-then' block in the workspace to the 'if-then-else' block in the script area.

Blocks

Built-in

Control

Math

Text

Lists

Colors

Variables

Procedures

Screen1

Label1

HorizontalArrangement1

TextBox1

Button1

Label2

Any component

Viewer

0

=

+

-

×

÷

^

random integer from 1 to 100

random fraction

when Button1 Click

do

set global gess to TextBox1.Text

set TextBox1.Text to random integer from 1 to 100

if

then

else if

then

else if

then

=

<=

<

>

>=

<=

<=

<=

<=

Blocks

Built-in

Control

Logic

Math

Text

Lists

Variables

Procedures

Screen1

Label1

HorizontalArrangement1

TextBox1

Button1

Label2

Any component

Viewer

initialize global name to

get

set to

initialize local name to

in

initialize local name to

in

when Button1 Click

do set global gess to TextBox1 Text

set TextBox1 Text to

get global password = get global gess

then

else if get global password > get global gess

then

else if get global password < get global gess

then

global gess

global max

global min

✓ global password

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - Label1
 - HorizontalArrangement1
 - TextBox1
 - Label2

Viewer

set Label2 · HasMargins · to

Label2 · Height ·

set Label2 · Height · to

set Label2 · HeightPercent · to

Label2 · Text ·

set Label2 · Text · to

Label2 · TextColor ·

set Label2 · TextColor · to

Label2 · Visible ·

set Label2 · Visible · to

when Button1 · Click

do

- set global gess · to TextBox1 · Text ·
- set TextBox1 · Text · to
- if
 - get global password · = · get global gess ·
 - set Label2 · Text · to 恭喜答對了!!!!
- else if
 - get global password · > · get global gess ·
- then
- else if
 - get global password · < · get global gess ·
- then

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists

Variables

- Screen1
 - Label1
 - HorizontalArrangement1
 - TextBox1
 - Button1
 - Label2
- Any component

Viewer

initialize global name to

get

set to

initialize local name to

in

initialize local name to

in

when Button1 Click

do

set global gess to TextBox1 . Text

set TextBox1 . Text to

if

get global password = get global gess

then

set Label2 . Text to

else if

get global password > get global gess

when

set global min to

else if

global gess

global max

global min

global password

then

< get global gess

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Variables**
 - Procedures
- Screen1
 - Label1
 - HorizontalArrangement1
 - TextBox1
 - Button1
 - Label2
 - Any component

Viewer

initialize global name to

get

set to

initialize local name to in

initialize local name to in

when Button1 Click do

- set global gess to TextBox1 Text
- set TextBox1 Text to
- if
 - get global password = get global gess
 - set Label2 Text to 恭喜答對了!!!!
- else if
 - get global password >= get global gess
 - then set global min to get global gess
- else if
 - get global password < get global gess
 - then

The image shows a block editor interface with a 'Blocks' palette on the left and a 'Viewer' area on the right. In the 'Blocks' palette, the 'Variables' category is highlighted with a red box. A red arrow points from this box to a 'get' block in the 'Viewer' area. Another red arrow points from the 'get' block to a 'get global password' block in a script. The script is a 'when Button1 Click do' block containing several 'set' and 'if' blocks. The 'if' blocks use 'get global password' and 'get global gess' blocks to compare values. One 'if' block has a true condition and sets 'Label2 Text' to '恭喜答對了!!!!'. Another 'if' block has a false condition and sets 'global min' to 'get global gess'. A third 'if' block has a false condition and is empty.

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

Label1

HorizontalArrangement

TextBox1

Label2

Any component

Viewer

Label2 - HasMargins -

set Label2 - HasMargins - to

Label2 - Height -

set Label2 - Height - to

set Label2 - HeightPercent - to

Label2 - Text -

set Label2 - Text - to

Label2 - TextColor -

set Label2 - TextColor - to

Label2 - Visible -

when Button1 - Click

do set global gess - to TextBox1 - Text -

set TextBox1 - Text - to

if get global password - = - get global gess -

then set Label2 - Text - to 恭喜答對了!!!!

else if get global password - > - get global gess -

then set global min - to get global gess -

set Label2 - Text - to

else if get global password - < - get global gess -

then

Blocks

- Built-in
 - Control
 - Logic
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - Label1
 - HorizontalArrangement1
 - TextBox1
 - Button1
 - Label2
- Any component

Viewer

length

is empty

compare texts

trim

uppercase

starts at text piece

contains text piece

when Button1 Click

do

set global gess to TextBox1

set TextBox1 text to string

if

get global passw

then

set Label2 text to

else

get global passw

then

set global name to

set Label2 text to

else if

get global password <= get global gess

then

join

string

string

string

join

string

string

string

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Variables
- Procedures

Screen1

- Label1
- HorizontalArrangement1
 - TextBox1
 - Button1
 - Label2
- Any component

Viewer

initialize global name to

get

set to

initialize local name to

in

initialize local name to

in

when Button1 Click

do

set global gess to TextBox1 Text

set TextBox1 Text to

if

get global password = get global gess

then

set Label2 Text to 恭喜答對了!!!!

else if

get global password > get global gess

then

set global min to get global gess

set Label2 Text to

get global min

else if

get global password < get global gess

then

set global max to get global gess

set Label2 Text to

get global max

global gess

✓ global max

global min

global password

The image shows a Scratch-like programming environment. On the left, the 'Blocks' palette is visible, with the 'Variables' category highlighted. A red arrow points from the 'get' block in the script area to the 'Variables' category. The script area shows a sequence of blocks: 'initialize global name to', 'get', 'set to', 'initialize local name to', 'in', 'initialize local name to', and 'in'. The main script area shows a 'when Button1 Click' event followed by a 'do' loop. The loop contains several blocks: 'set global gess to TextBox1 Text', 'set TextBox1 Text to', 'if' (with 'get global password = get global gess'), 'then' (with 'set Label2 Text to 恭喜答對了!!!!'), 'else if' (with 'get global password > get global gess'), 'then' (with 'set global min to get global gess' and 'set Label2 Text to get global min'), 'else if' (with 'get global password < get global gess'), and 'then' (with 'set global max to get global gess' and 'set Label2 Text to get global max'). A dropdown menu is open next to the 'get global min' block, showing a list of variables: 'global gess', 'global max' (checked), 'global min', and 'global password'.

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

- Label1
- TextBox1
- Button1
- Label2

Any component

Viewer

when Button1 Click

do set global gess to TextBox1 . Text

set TextBox1 . Text to ""

if get global password = get global gess

then set Label2 . Text to "恭喜答對了!!!!"

else if get global password > get global gess

then set global min to get global gess

set Label2 . Text to join get global min

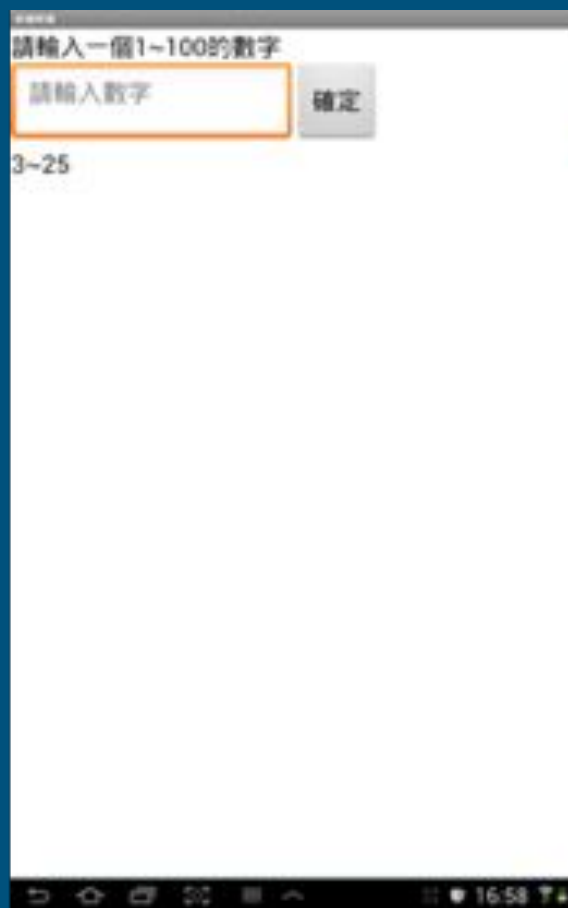
get global max

else if get global password < get global gess

then set global max to get global gess

set Label2 . Text to join get global min

get global max



基礎通訊傳輸

藍牙(Bluetooth)，是一種無線個人局域網(Wireless PAN)，最初由易立信創製，後來由藍牙技術聯盟訂定技術標準。

其發展史為1999年5月20日，索尼、易立信、國際商業機器、英特爾、諾基亞及東芝公司等業界龍頭創立「特別興趣小組」(Special Interest Group, SIG)，即藍牙技術聯盟的前身。2001年的1.1版正式列入IEEE標準，Bluetooth 1.1即為IEEE 802.15.1。同年，SIG成員公司超過2000家。過了幾年之後，採用藍牙技術的電子裝置如雨後春筍般增加，售價也大幅回落。為了擴寬藍牙的應用層面和傳輸速度，SIG先後推出了1.2、2.0版，以及其他附加新功能，例如EDR(Enhanced Data Rate，配合2.0的技術標準，將最大傳輸速度提高到3Mbps)、A2DP(Advanced Audio Distribution Profile，一個控音軌分配技術，主要應用於立體聲耳機)、ACRCP(A/V Remote Control Profile)等。

藍牙的應用

藍牙的應用層次很廣泛且功能越來越強大，但對於感測器與控制器本身大部分時候並不需要如此多的功能，因此下面將介紹一款非常簡單容易使用的藍牙通訊模組，他可以簡單的幫我們透過藍芽做基本的資料與訊號傳輸，且十分穩定。

現在手機跟電腦都有藍芽配備，電子產品或者是互動藝術的作品，如果可以擁有藍芽通訊的能力，便可以跟手機透過藍芽結合起來做各種應用。



藍牙規格與特性

- 低單價, 低消耗功率
- 最大傳輸距離為大約1m
- 可作為Master 或是 Slave Bluetooth
- 只需要連接RX, TX 就可以直接傳資料, 不需額外設定程式 ; 使用者, 程式與原來的Serial傳輸程式一樣, 可以不用修改
- 使用UART通訊, 支援串列傳輸協定包括DTS, RTS
- 可以於其他藍牙協定的裝置相容
- 標準輸入電壓 3.3V ~ 5v (CMOS and TTL level)都可以使用。
- 尺寸: 34.41 x 45.65 x 12.51 mm

下圖是HC-0x藍芽模組的外觀，這款藍芽模組可以讓你在範圍10米內實現無線傳輸通信，而且可以界接包括Arduino, 8051, PIC, AVR, ARM, MSP430等各種MCU。

藍芽協定: Bluetooth V2.0 + EDR (Enhanced Data Rate)

工作頻率: 2.4~2.48GHz, ISM Band

傳輸距離: 空曠地有效距離 10 公尺

介面: UART

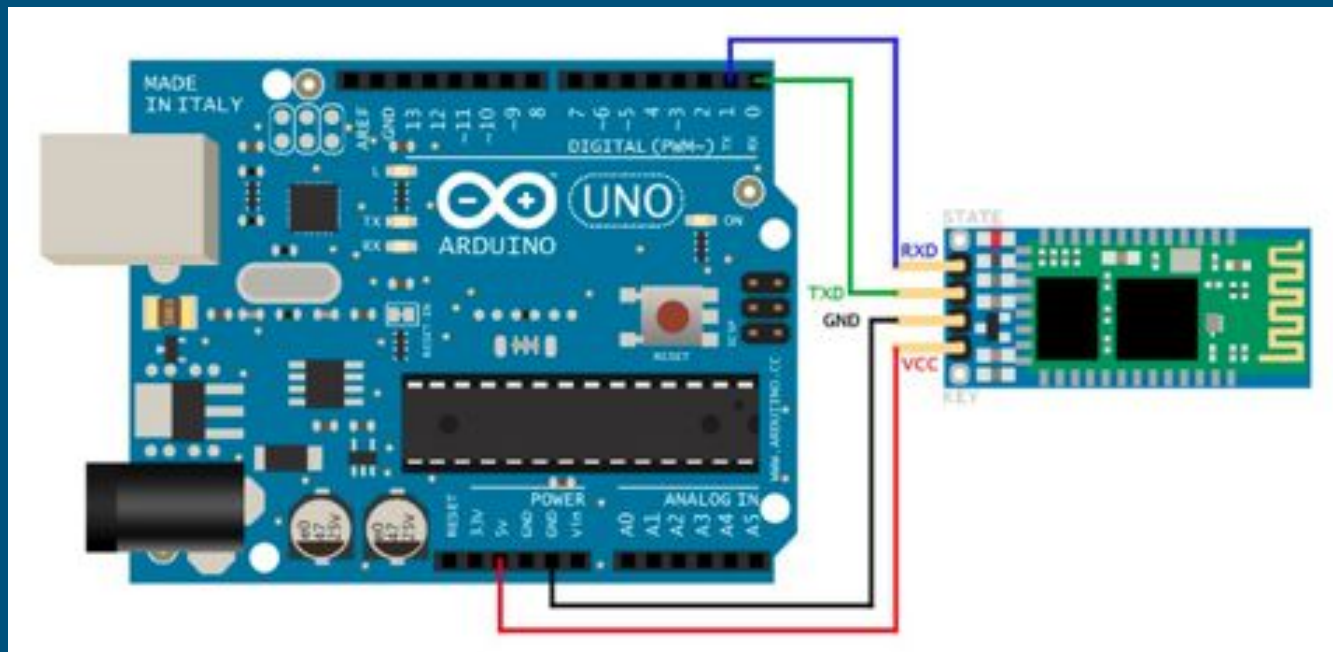
輸入電壓: 3.3V ~ 4.2V

工作溫度: -20°C ~ +75°C



原始模組使用工作電壓為 3.3V ~ 4.2V, 因為 Arduino 一般是 5V 的, 如果要跟 Arduino 連接, 得利用 LDO Regulator (Low Drop Out Regulator) 轉換電壓, 對一般使用者會有些困難。所幸市面上可以買到帶底板的模組:





下載 Bluetooth Terminal



開啟手機的藍芽設定

搜尋藍芽裝置HC-05

藍芽配對要求，輸入密碼1234，與藍芽連接

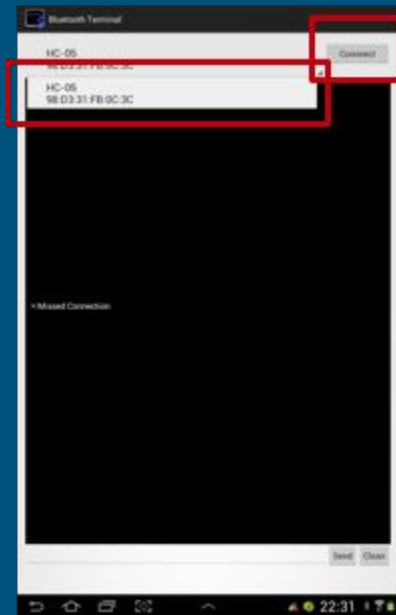


```
const int led = 13; // 定義LED腳位  
char serialA; // 儲存接收資料的變數
```

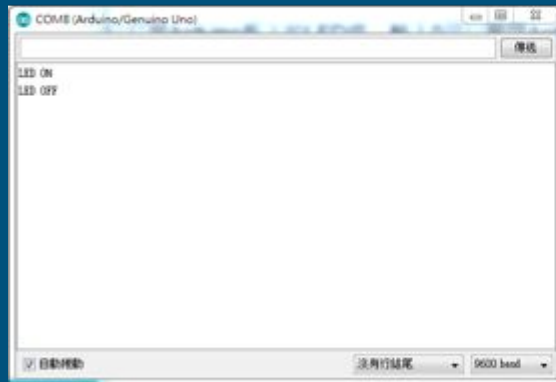
```
void setup(){  
  Serial.begin(9600);  
  pinMode(13, OUTPUT);  
}
```

```
void loop() {  
  if (Serial.available()) { // 若收到藍牙模組的資料, 則送到「序列埠監控視窗」  
    serialA = Serial.read();  
    if (serialA == '1') {  
      digitalWrite(13, HIGH); // 回應命令發送端, 告知「已開燈」  
      Serial.println("LED ON");  
      delay(1000);  
    } else if (serialA == '0') {  
      digitalWrite(13, LOW); // 回應命令發送端, 告知「已關燈」  
      Serial.println("LED OFF");  
      delay(1000);  
    }  
  }  
}
```

打開手機藍芽程式，選取HC-05，再按下連線。



輸入1可以打開Arduino板上的LED燈，輸入0可以關閉Arduino板上的LED燈



Palette

User Interface

- Button
- CheckBox
- Clock
- Image
- Label
- ListPicker
- Notifier
- PasswordTextBox
- Slider
- TextBox
- WebView

Layout

Media

Drawing and Animation

Sensors

Social

Storage

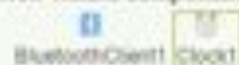
Connectivity

Viewer

Display hidden components in Viewer



Non-visible components



Components

- Screen1
 - HorizontalArrangement
 - BTconnect
 - BTdisconnect
 - HorizontalArrangement
 - Label1
 - TableArrangement1
 - LED1_ON
 - LED1_OFF
 - LED2_ON
 - LED2_OFF
 - BluetoothClient1
 - Clock1

Rename

Delete

Media

Upload File...

Palette

User Interface

Layout

 HorizontalArrangement ⓘ

 TableArrangement ⓘ

 VerticalArrangement ⓘ

Media

Drawing and Animation

Sensors

Social

Storage

Connectivity

LEGO® MINDSTORMS®

Viewer

1.點選Layout

☐ Display hidden components in Viewer

Screen1

2.選擇水平排列後向右拖曳

☐ Display hidden components in Viewer



Components

Screen1
HorizontalArrangement1

1. 點選水平方塊

2. 修改成適當大小

Properties

HorizontalArrangement1

AlignHorizontal

Left

AlignVertical

Top

Visible

showing

Width

Fill parent...

Height

40 pixels...

Palette

Viewer

User Interface

- Button
- CheckBox
- Clock
- Image
- Label
- ListPicker
- Notifier
- PasswordTextbo

☐ Display hidden component

Screen1

Text for ListPicker1

1.點選User Interface

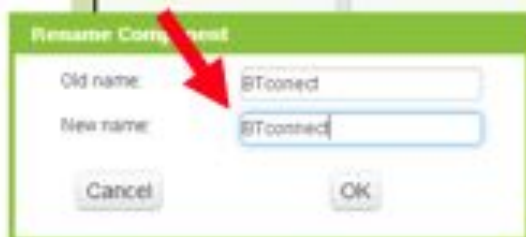
2.選擇ListPicker後向右拖曳至水平排列方塊中



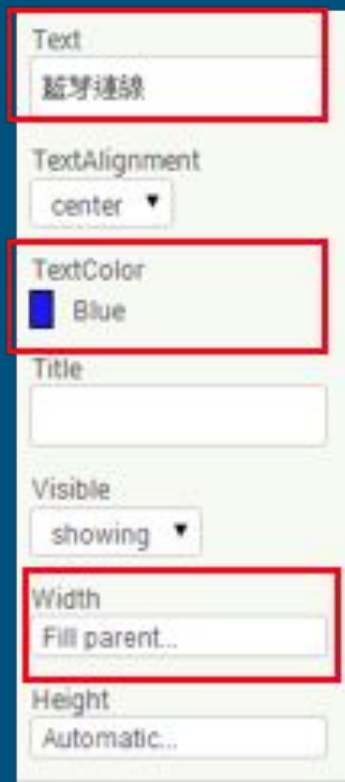
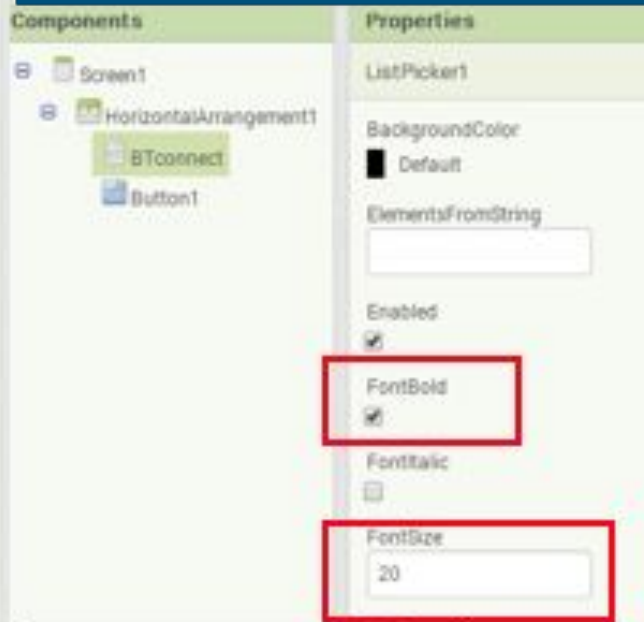
1. 點選ListPicker

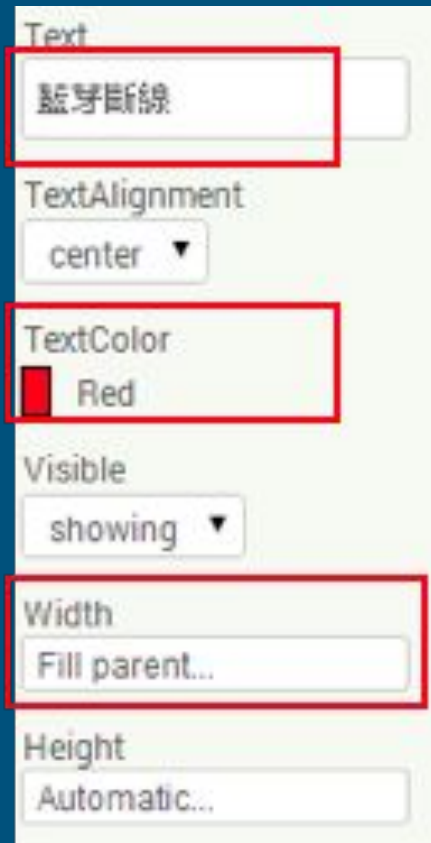
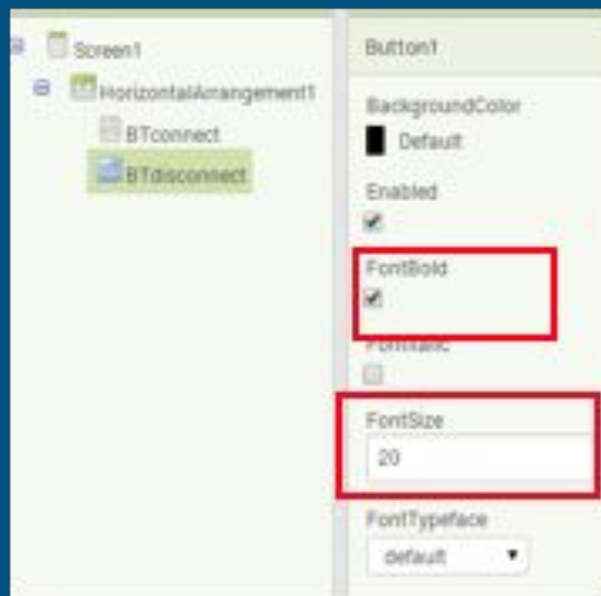


3. 修改元件名為BTconnect



2. 點下Rename









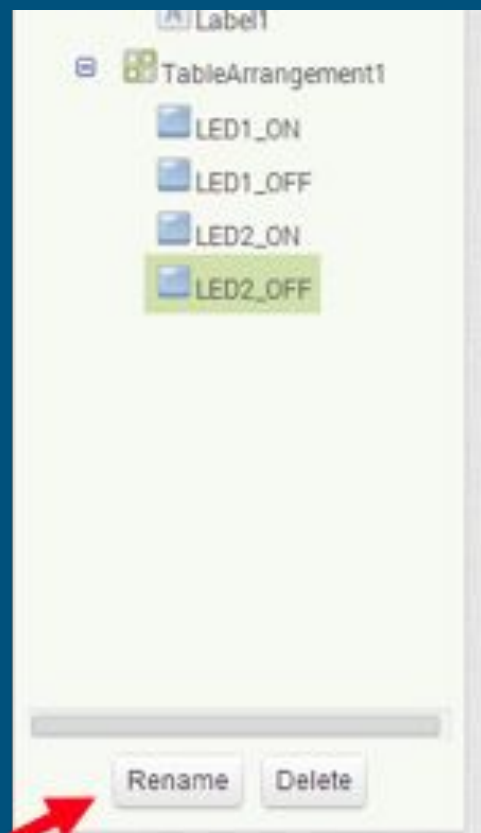
將Label拖曳至方塊
內








1. 拖曳四個
Button
至方塊內



2. 點選Rename更換名字

BackgroundColor
 Green

Enabled
☒

FontBold
☒

FontItalic
☐

FontSize

FontTypeface


Image


Shape

ShowFeedback
☒

Text

TextAlignment

TextColor
 Dark Gray

BackgroundColor
 Green

Enabled
☒

FontBold
☒

FontItalic
☐

FontSize

FontTypeface


Image

Shape


ShowFeedback
☒

Text

TextAlignment

TextColor
 Dark Gray



BackgroundColor
 Dark Gray

Enabled
☒

FontBold
☒

FontItalic
☐

FontSize

FontTypeface


Image


Shape

ShowFeedback
☒

Text

TextAlignment

TextColor
 White

BackgroundColor
 Dark Gray

Enabled
☒

FontBold
☒

FontItalic
☐

FontSize

FontTypeface


Image

Shape

ShowFeedback
☒

Text

TextAlignment

TextColor
 White

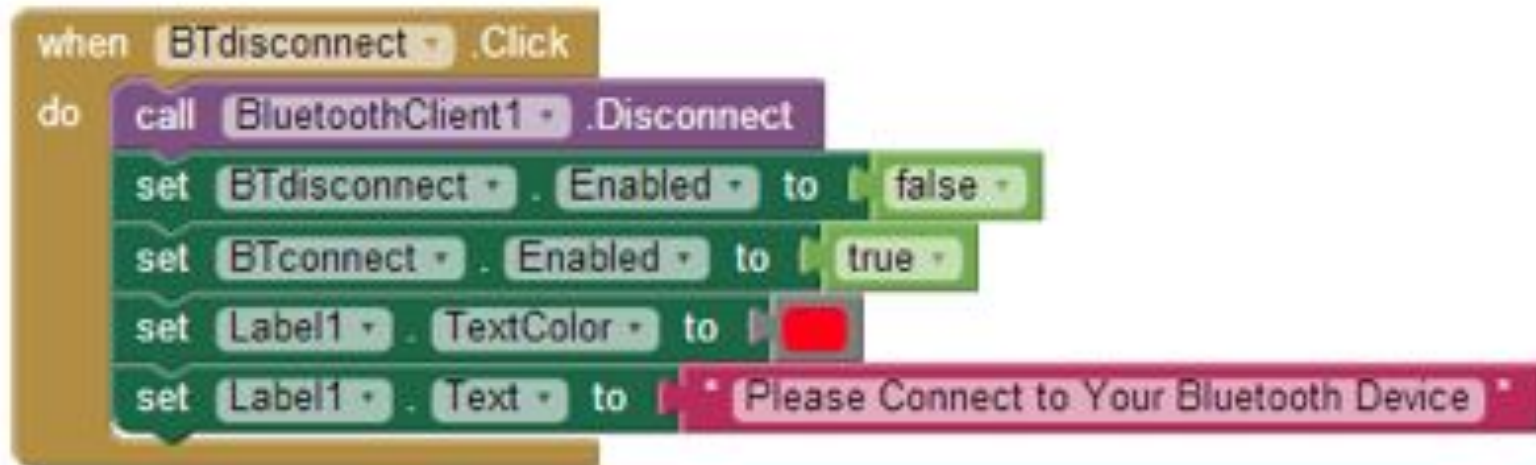




設定"Screen1"起始畫面



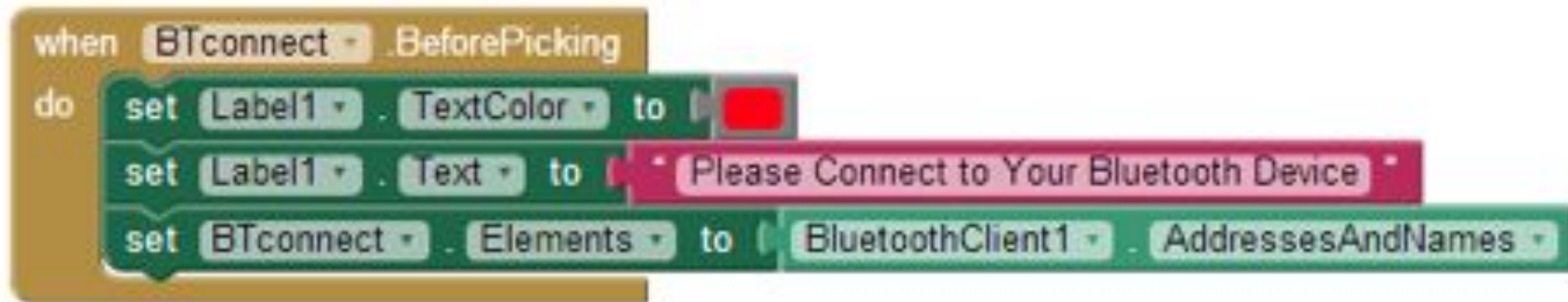
編輯“BTdisconnect”程式



```
when BTdisconnect .Click
do
  call BluetoothClient1 .Disconnect
  set BTdisconnect . Enabled to false
  set BTconnect . Enabled to true
  set Label1 . TextColor to red
  set Label1 . Text to "Please Connect to Your Bluetooth Device"
```

The image shows a Scratch script for a button click event. The script is contained within a 'when clicked' block. It performs the following actions in sequence: calls the 'Disconnect' method on 'BluetoothClient1', sets the 'Enabled' property of 'BTdisconnect' to 'false', sets the 'Enabled' property of 'BTconnect' to 'true', sets the 'TextColor' of 'Label1' to 'red', and sets the 'Text' of 'Label1' to 'Please Connect to Your Bluetooth Device'.

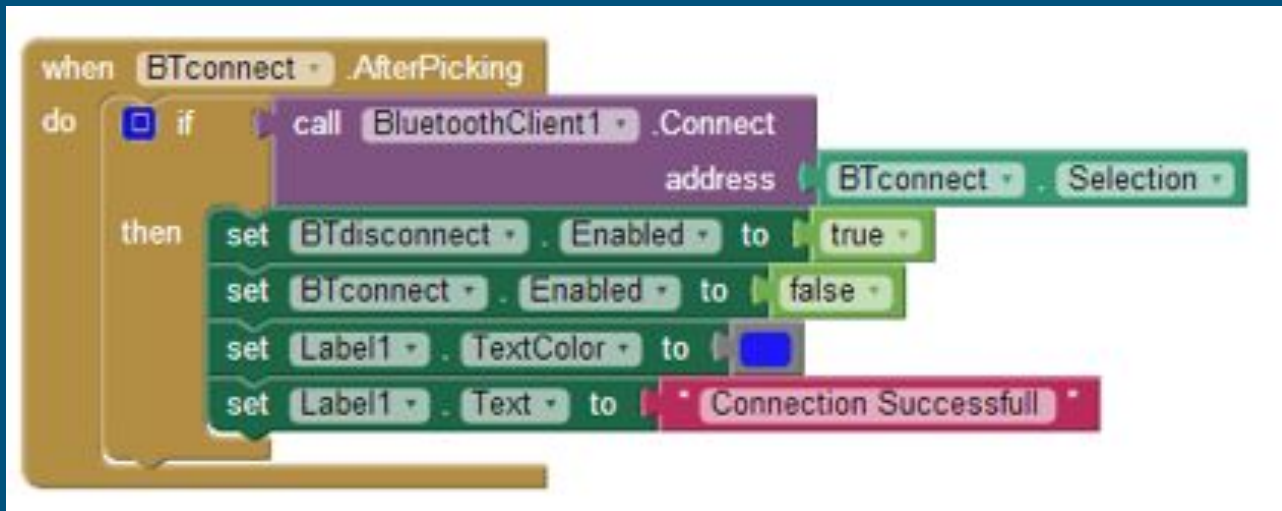
編輯"BTconnect"程式(被選擇前)



```
when BTconnect .BeforePicking
do
  set Label1 . TextColor to red
  set Label1 . Text to "Please Connect to Your Bluetooth Device"
  set BTconnect . Elements to BluetoothClient1 . AddressesAndNames
```

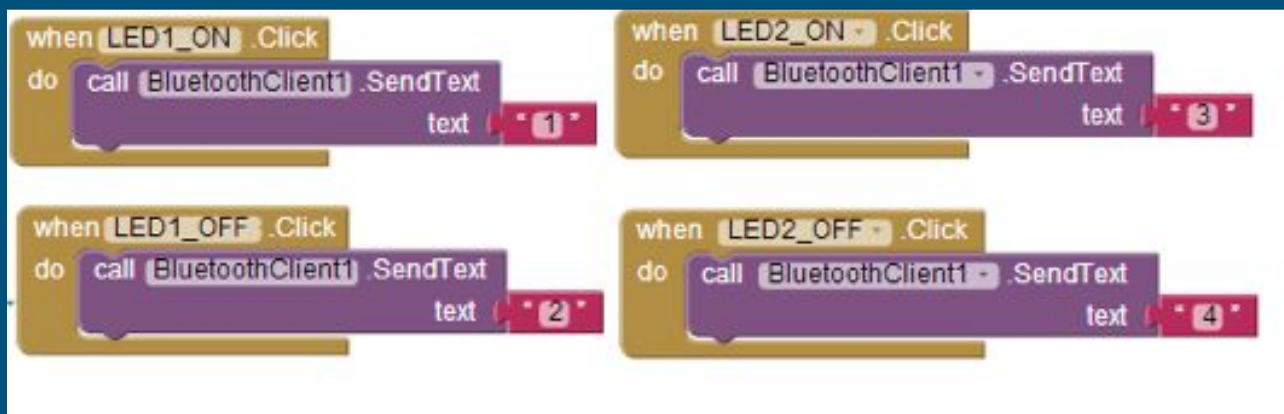
The image shows a Scratch code block for the "when BTconnect .BeforePicking" event. The code block is a "do" block containing three "set" blocks. The first "set" block sets "Label1 . TextColor" to "red". The second "set" block sets "Label1 . Text" to "Please Connect to Your Bluetooth Device". The third "set" block sets "BTconnect . Elements" to "BluetoothClient1 . AddressesAndNames".

編輯"BTconnect"程式(被選擇後)



編輯各LED按鈕的程式

| Arduino命令對照表 | | | | |
|--------------|---|---|---|---|
| 實際符號 | 1 | 2 | 3 | 4 |
| LED1_動作 | H | L | | |
| LED2_動作 | | | H | L |



when Screen1.Initialize

do
 set [LED1] . [ENABLED] to [true]
 set [Label1] . [text] to [Please Connect to Your Bluetooth Device]

when [BTdisconnected] . Click

do
 call [BluetoothManager] . Disconnected
 set [BTdisconnected] . [ENABLED] to [false]
 set [BTdisconnected] . [ENABLED] to [true]
 set [LED1] . [ENABLED] to [true]
 set [Label1] . [text] to [Please Connect to Your Bluetooth Device]

when [BTconnected] . BeforePicking

do
 set [LED1] . [ENABLED] to [true]
 set [Label1] . [text] to [Please Connect to Your Bluetooth Device]
 set [BTconnected] . [ENABLED] to [BluetoothClient] . AddressesAndName

when [BTconnected] . AfterPicking

do
 if [call [BluetoothManager] . Connected
 address: [BTconnected] . Selection]
 then
 set [BTdisconnected] . [ENABLED] to [true]
 set [BTconnected] . [ENABLED] to [false]
 set [LED1] . [ENABLED] to [false]
 set [Label1] . [text] to [Connection Success!]

when [LED1_ON] . Click

do
 call [BluetoothManager] . SendText
 text: [1]

when [LED2_ON] . Click

do
 call [BluetoothManager] . SendText
 text: [2]

when [LED1_OFF] . Click

do
 call [BluetoothManager] . SendText
 text: [3]

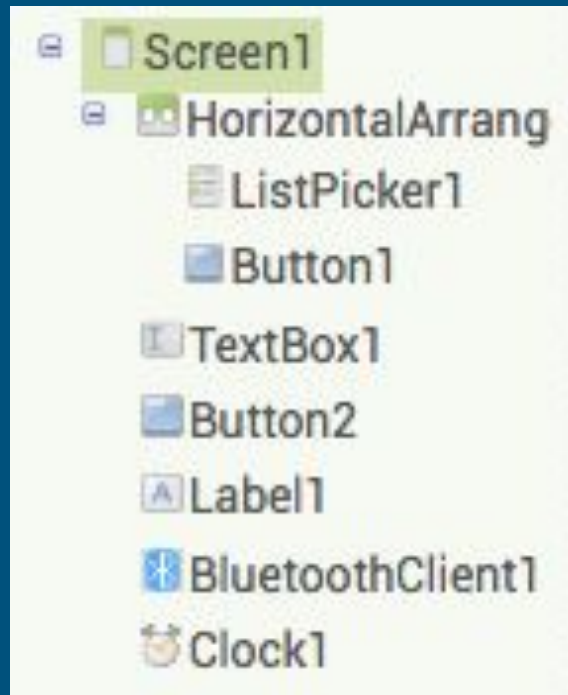
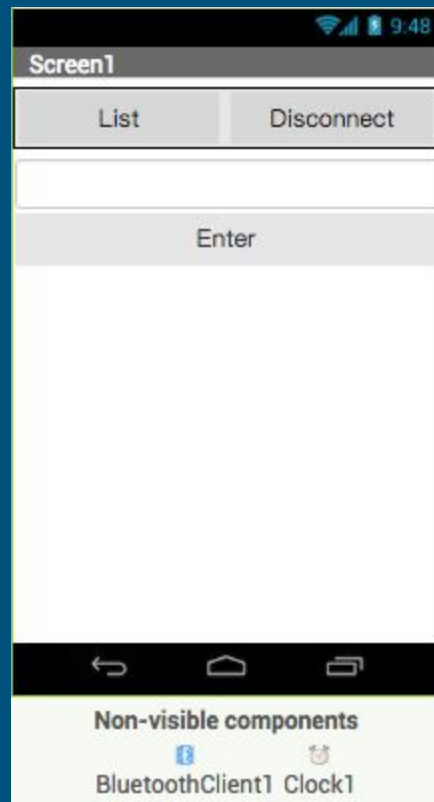
when [LED2_OFF] . Click

do
 call [BluetoothManager] . SendText
 text: [4]

練習

連接兩個LED並使用app控制

藍牙通訊



when Screen1.Initialize

do
 set Button1.Enabled to false
 set Button2.Enabled to false
 set Label1.Text to " "

Initialize global C to " "

Initialize global S to " "

when ListPicker1.BeforePicking

do set ListPicker1.Elements to BluetoothClient1.AddressesAndNames

when ListPicker1.AfterPicking

do
 set global C to ListPicker1.Selection
 if call BluetoothClient1.Connect address get global C
 then
 set ListPicker1.Enabled to false
 set Button1.Enabled to true
 set Button2.Enabled to true

when Clock1.Timer

do
 if BluetoothClient1.IsConnected and call BluetoothClient1.BytesAvailableToReceive ≠ 0
 then
 set global S to Label1.Text
 set Label1.Text to join
 get global S
 call BluetoothClient1.ReceiveText
 numberOfBytes call BluetoothClient1.BytesAvailableToReceive

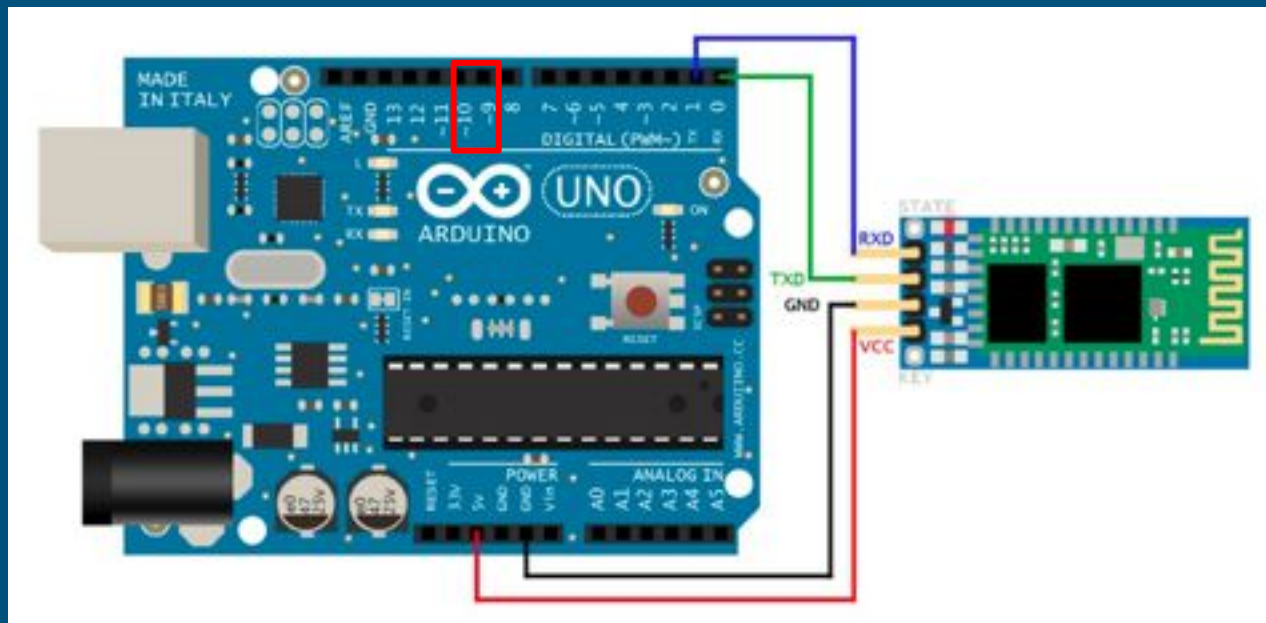
when Button1.Click

do
 call BluetoothClient1.Disconnect
 set ListPicker1.Enabled to true
 set Button1.Enabled to false
 set Button2.Enabled to false

when Button2.Click

do
 call BluetoothClient1.SendText text TextBox1.Text
 set TextBox1.Text to " "

將arduuno原本 0 跟 1 腳位
改成10及11腳位



```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11); // RX, TX
void setup() {
  Serial.begin(9600);
  while (!Serial);
  Serial.println("Goodnight moon!");
  mySerial.begin(9600);
  mySerial.println("Hello, world?");
}
```

```
void loop() {  
  if (mySerial.available()) {  
    Serial.write(mySerial.read());  
    //Serial.println(">end!");  
  }  
  if (Serial.available()) {  
    mySerial.write(Serial.read());  
    //Serial.println("Send!");  
  }  
}
```