

## Introduction to Design (Track 3)

### 5.2 Hands-on: Android Quiz App

---

Zilu Liang

[www.zilu-liang.net/id3](http://www.zilu-liang.net/id3)

# Important!!

**Please disinfectize your hands before entering the classroom!**

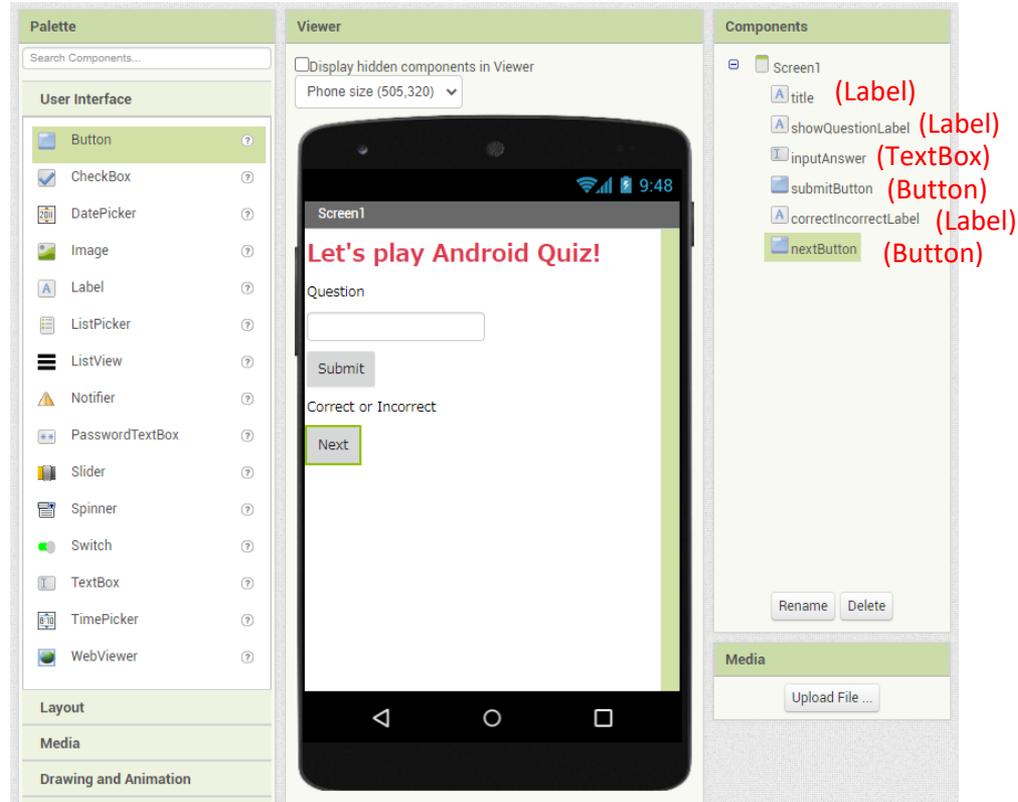
入室前にアルコールを使用して手指消毒を行ってください。

**Please disinfectize your chair and table!**

- ①ペーパーにアルコールを噴霧してください。
- ②アルコールが噴霧されたペーパーで、使用箇所（テーブル、椅子など）を拭き取ってください。
- ③使用済のペーパーは廊下のごみ箱に捨ててください。



# Build UI in designer view



# Switch to blocks view

The screenshot shows the 'counterApp' interface. At the top, there is a green header bar with the app name 'counterApp' on the left and a navigation area on the right containing buttons for 'Screen1', 'Add Screen ...', 'Remove Screen', 'Publish to Gallery', 'Designer', and 'Blocks'. The 'Blocks' button is circled in red. Below the header, the interface is split into two main sections: 'Blocks' on the left and 'Viewer' on the right. The 'Blocks' section contains a list of built-in components categorized by type: Control (orange), Logic (green), Math (blue), Text (pink), Lists (light blue), Dictionaries (dark blue), Colors (grey), Variables (orange), and Procedures (purple). Under 'Screen1', there are three components: 'click\_button' (blue), 'label' (white with a blue border), and 'Any component' (blue). The 'Viewer' section shows a large empty canvas. On the right side of the canvas, there is a vertical toolbar with icons for a blue backpack, a target, a plus sign, a minus sign, and a trash can. At the bottom of the viewer, there are two warning icons (a yellow triangle with an exclamation mark and a red circle with an X) and a 'Show Warnings' button.

# Create Question List

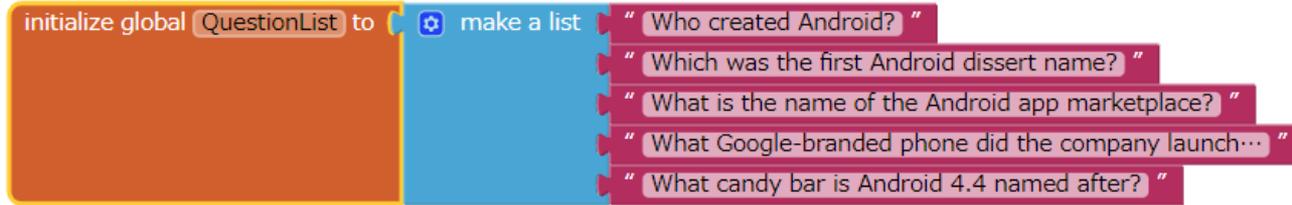


# Create Question List

Click the mutator and add more items to the list. We need 5 items.

The image shows a Scratch code block for initializing a global variable. The block is orange and contains the text "initialize global QuestionList to". Attached to the right is a blue "make a list" block with a gear icon (mutator) in its top-left corner. A callout box, outlined in blue, points to the mutator icon. Inside the callout box, there is a preview of a list: a grey box labeled "item" on the left and a blue box labeled "list" on the right. The "list" box contains five smaller blue boxes, each labeled "item", stacked vertically.

# Create Question List



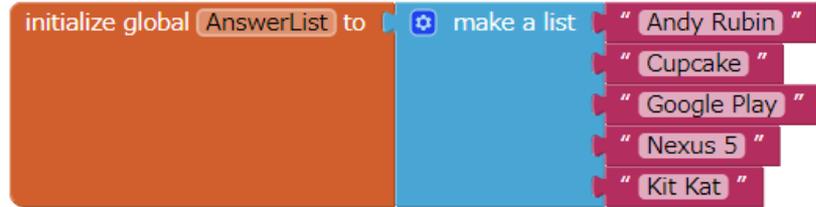
## Question list:

1. Who create Android?
2. Which was the first Android dissert name?
3. What is the name of the Android app marketplace?
4. What Google-branded phone did the company launch in 2013?
5. What candy bar is Android 4.4 named after?

# Create Answer List

## Answer list:

1. Andy Rubin
2. Cupcake
3. Google Play
4. Nexus 5
5. Kit Kat



# Show question 1 when app starts

The screenshot shows the Visual Studio Code interface with the Blocks editor. The 'Blocks' pane on the left lists various categories, with 'Screen1' selected. The 'Viewer' pane on the right displays a script with several 'when Screen1' events. The 'when Screen1 .Initialize' block is highlighted with a red circle, and a red arrow points from it to a larger, detailed view of the same block on the right.

A detailed view of the 'when Screen1 .Initialize' block. The block is highlighted with a red circle, and a red arrow points from the 'when Screen1 .Initialize' block in the viewer to this detailed view. The 'do' block is also highlighted with a red circle.

# Show question 1 when app starts

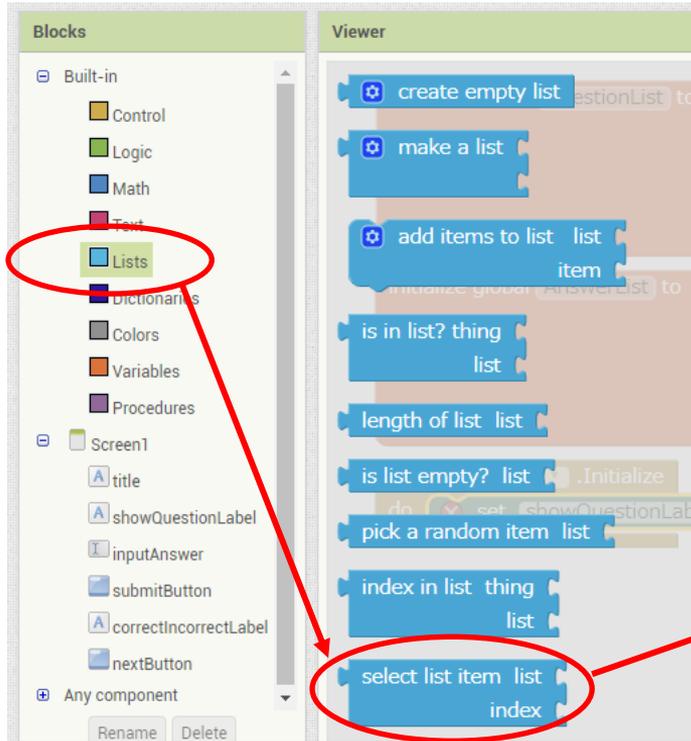
The screenshot shows the Xcode interface with the storyboard on the left and the Swift code on the right. The storyboard shows a hierarchy for 'Screen1' with components: 'title', 'showQuestionLabel', 'InputAnswer', 'submitButton', 'correctIncorrectLabel', and 'nextButton'. The 'showQuestionLabel' component is highlighted with a red circle. The Swift code shows the following lines:

```
set showQuestionLabel . HasMargins to  
when Screen1 . Initialize  
showQuestionLabel . Height  
set showQuestionLabel . Height to  
set showQuestionLabel . HeightPercent to  
showQuestionLabel . Text  
set showQuestionLabel . Text to
```

The 'showQuestionLabel . Text' property is highlighted with a red circle. A red arrow points from this property to the 'showQuestionLabel' component in the storyboard.

The close-up screenshot shows the Swift code for the 'when Screen1 Initialize' block. The 'do' block contains the line 'set showQuestionLabel . Text to', which is highlighted with a red circle. A red 'X' icon is visible next to the 'set' block, indicating an error.

# Show question 1 when app starts



The image shows the Scratch interface. On the left is the 'Blocks' palette, and on the right is the 'Viewer' area. In the 'Blocks' palette, the 'Lists' category is circled in red. In the 'Viewer' area, the 'select list item' block is circled in red. A red arrow points from the 'select list item' block in the Viewer to the 'select list item' block in the 'when Screen1.Initialize' block in the code editor.

```
when Screen1.Initialize  
do  
  set showQuestionLabel.Text to  
    select list item list  
      index
```

# Show question 1 when app starts

```
initialize global QuestionList to make a list  
  " Who created Android? "  
  " Which was the first Android dissert name? "  
  " What is the name of the Android app marketplace? "  
  " What Google-branded phone did the company launch... "  
  " What candy bar is Android 4.4 named after? "
```

```
when Screen1.Initialize  
do set showQuestionLabel.Text to select list item list  
index 1  
get global QuestionList
```

Because we want to show question 1, let's set index to 1

# Add Click Handler to 'Next' Button

initialize global `currentQuestionIndex` to `1`

- Why do we need this 'currentQuestionIndex' variable
- Can you draw a flowchart of the logic of the click handler?

# Add Click Handler to 'Next' Button

The image displays a visual programming environment with two main panels: 'Blocks' on the left and 'Viewer' on the right. In the 'Blocks' panel, a list of components is shown under the 'Built-in' category, with 'nextButton' highlighted and circled in red. In the 'Viewer' panel, a series of event handler blocks are visible, with the top one, 'when nextButton .Click', circled in red. A red arrow points from this block to a larger, detailed view of the same block on the right, which is also circled in red. This detailed view shows the 'when nextButton .Click' block with a 'do' slot below it, indicating where a click handler function would be added.

# Add Click Handler to 'Next' Button

```
when nextButton .Click
do
  set global currentQuestionIndex to (get global currentQuestionIndex + 1)
  set showQuestionLabel . Text to (select list item list (get global QuestionList) (get global currentQuestionIndex))
```

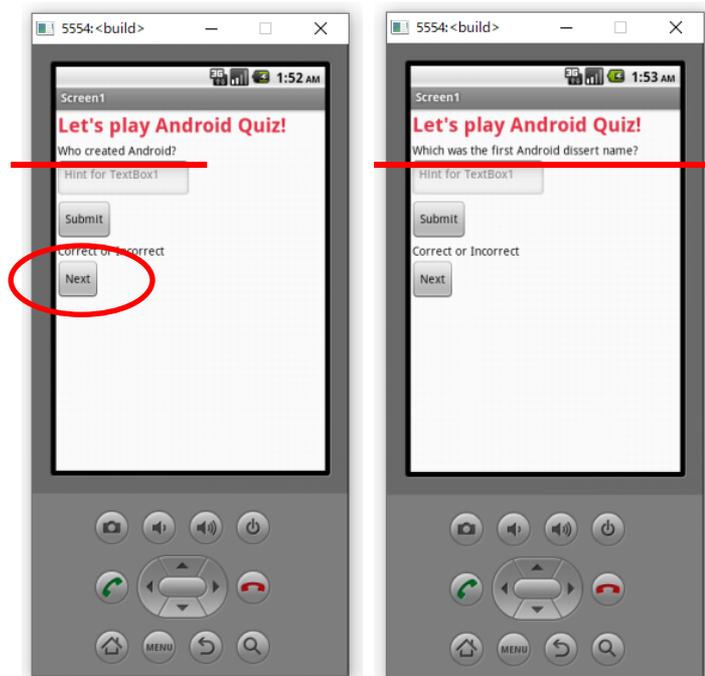
# Test App on Emulator

Click on the aiStarter on your computer, you should see the following window open.

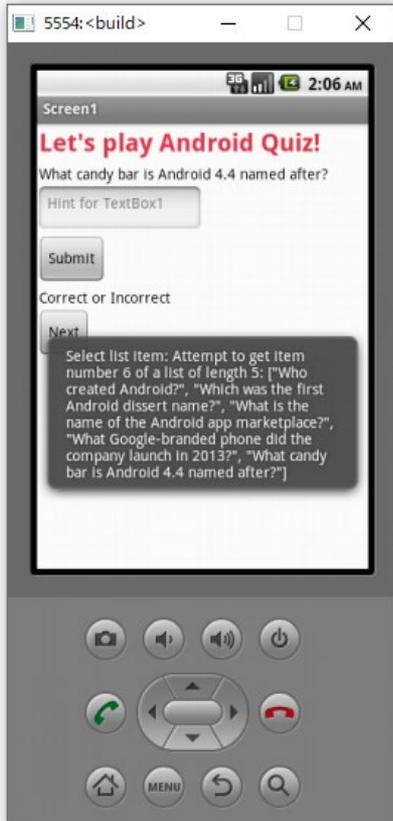
```
aiStarter
127.0.0.1 - - [21/Apr/2021 17:00:23] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:23] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:24] "GET /start/ HTTP/1.1" 200 0
127.0.0.1 - - [21/Apr/2021 17:00:24] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:24] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:24] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:24] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:24] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:24] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:25] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:26] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:27] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:28] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:29] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:30] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:31] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:32] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:33] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:34] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:35] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:36] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:37] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:38] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:39] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:40] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:41] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:42] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [21/Apr/2021 17:00:43] "GET /echeck/ HTTP/1.1" 200 67
Device = emulator-5554
127.0.0.1 - - [21/Apr/2021 17:01:09] "GET /restart/emulator-5554 HTTP/1.1" 200 0
```

# Test App on Emulator

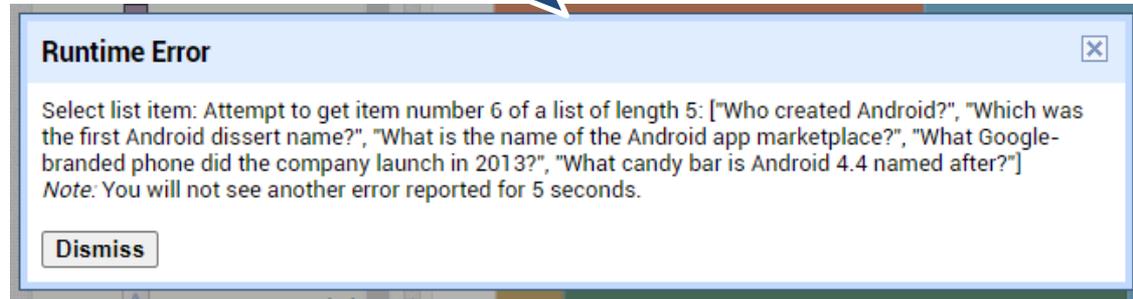
```
initialize global QuestionList to make a list " Who created Android? "  
" Which was the first Android dissert name? "  
" What is the name of the Android app marketplace? "  
" What Google-branded phone did the company launch... "  
" What candy bar is Android 4.4 named after? "  
  
initialize global AnswerList to make a list " Andy Rubin "  
" Cupcake "  
" Google Play "  
" Nexus 5 "  
" Kit Kat "  
  
when Screen1 .Initialize  
do set showQuestionLabel . Text to select list item list get global QuestionList  
index 1  
  
initialize global currentQuestionIndex to 1  
  
when nextButton .Click  
do set global currentQuestionIndex to get global currentQuestionIndex + 1  
set showQuestionLabel . Text to select list item list get global QuestionList  
index get global currentQuestionIndex
```



# Error



If we click 'Next' after the last question, we'll get an error. This is because we are trying to get an item with index number 6 from the list, but there are only 5 items in the list!



# Fix the Error

We need to check if the 'currentQuestionIndex' is larger than the length of the list

```
initialize global currentQuestionIndex to 0
when nextButton .Click
do
  set global currentQuestionIndex to [reset] get global currentQuestionIndex + 1
  set showQuestionLabel . Text to select list item list get global QuestionList
  index get global currentQuestionIndex
```

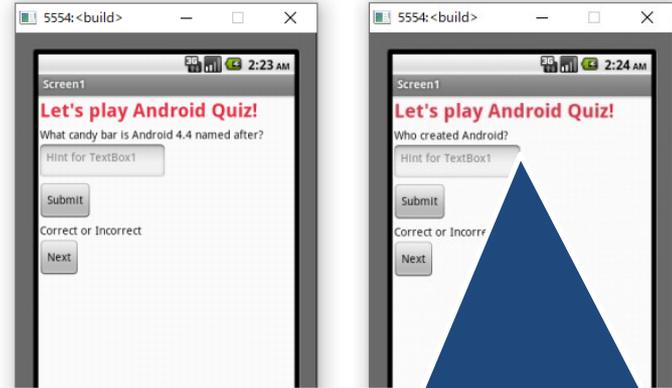
# Fix the Error

- We need to check if the 'currentQuestionIndex' is larger than the length of the list
- If 'currentQuestionIndex' is larger than list length, then reset it to 1

```
when nextButton .Click
do
  set global currentQuestionIndex to (get global currentQuestionIndex + 1)
  if (get global currentQuestionIndex > length of list list get global QuestionList)
  then set global currentQuestionIndex to 1
  set showQuestionLabel . Text to (select list item list get global QuestionList
  index get global currentQuestionIndex)
```

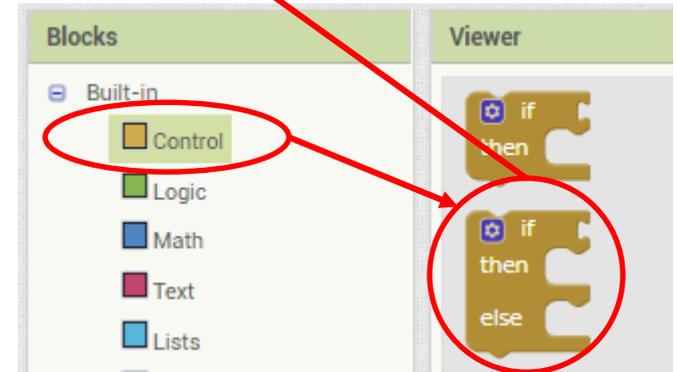
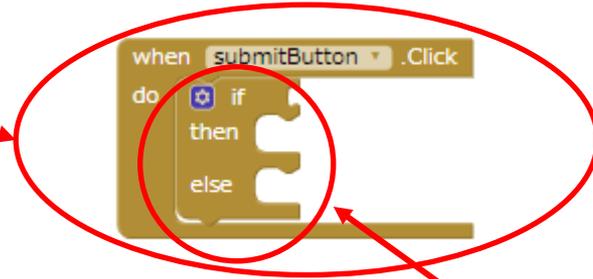
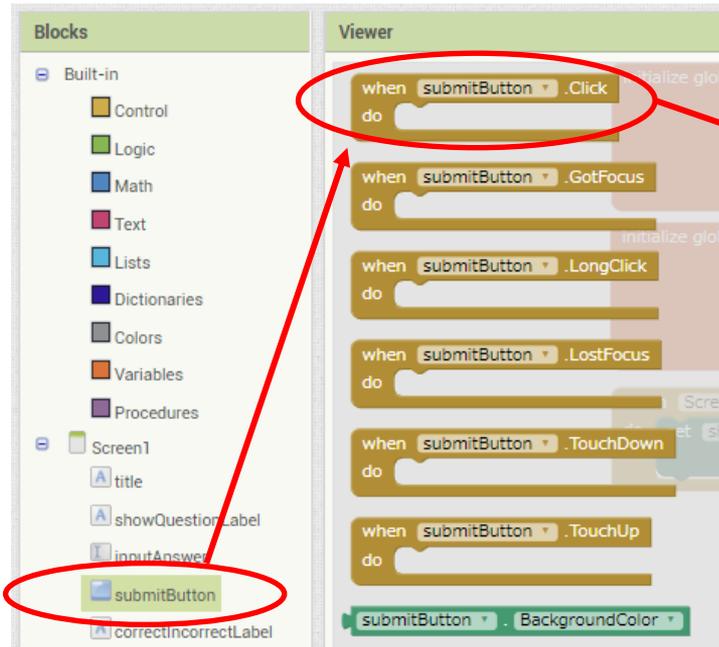
# Test App on Emulator

```
initialize global QuestionList to make a list  
  "Who created Android?"  
  "Which was the first Android dissert name?"  
  "What is the name of the Android app marketplace?"  
  "What Google-branded phone did the company launch?"  
  "What candy bar is Android 4.4 named after?"  
  
initialize global AnswerList to make a list  
  "Andy Rubin"  
  "Cupcake"  
  "Google Play"  
  "Nexus 5"  
  "Kit Kat"  
  
when Screen1.Initialize  
do set showQuestionLabel.Text to select list item list get global QuestionList  
  index  
  
initialize global currentQuestionIndex to 1  
  
when nextButton.Click  
do set global currentQuestionIndex to get global currentQuestionIndex + 1  
  if get global currentQuestionIndex > length of list list get global QuestionList  
  then set global currentQuestionIndex to 1  
  set showQuestionLabel.Text to select list item list get global QuestionList  
    index get global currentQuestionIndex
```

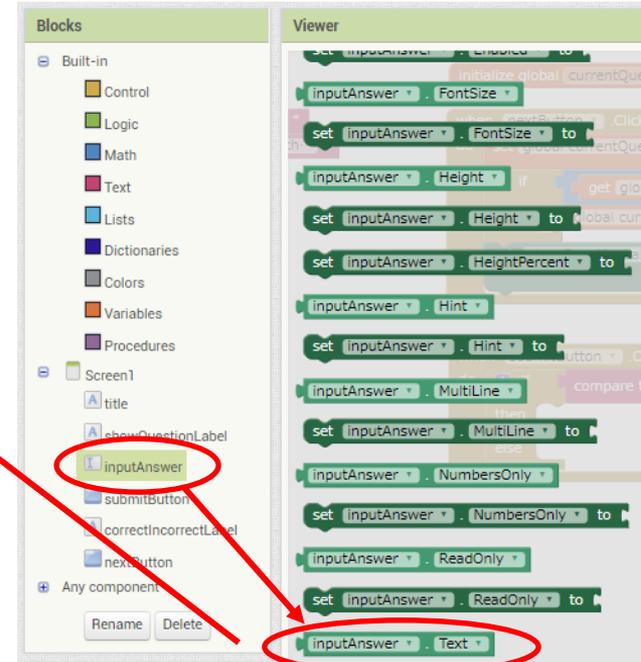
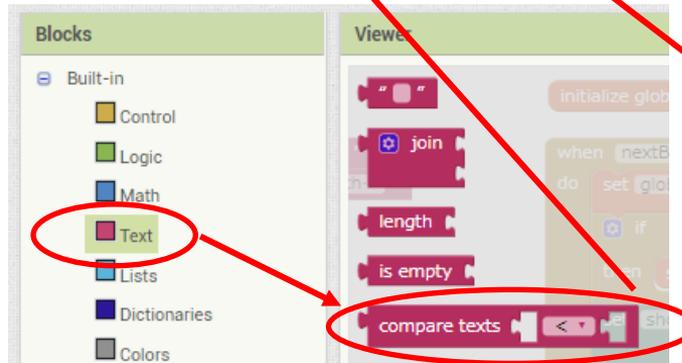
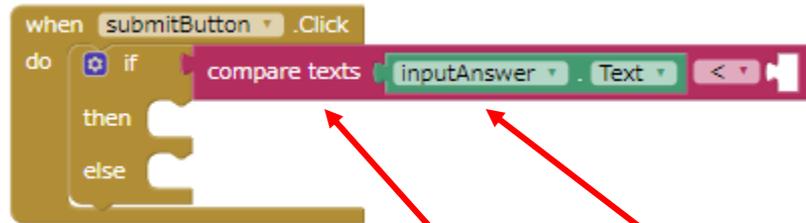


Now if we click 'Next' after the last question, the index will be reset to 1 and the first question will be shown

# Add Click Handler to 'Submit' Button



# Add Click Handler to 'Submit' Button



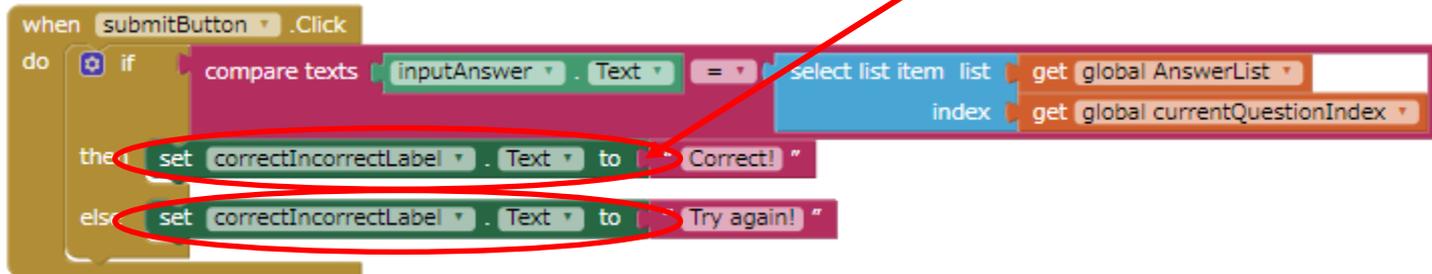
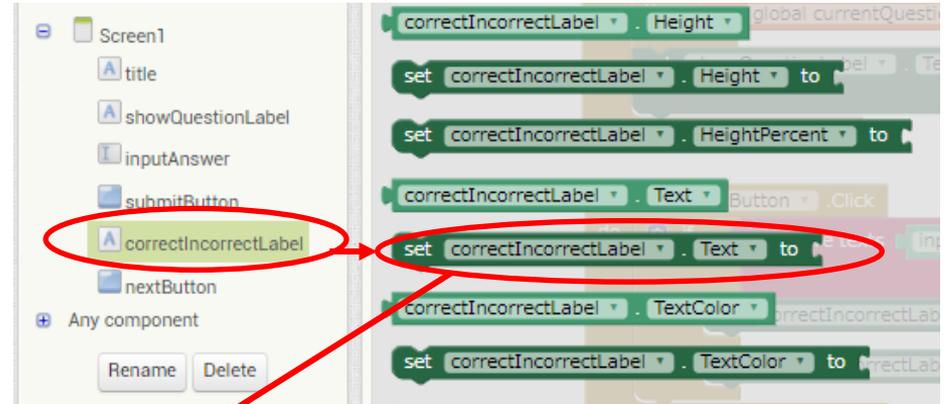
# Add Click Handler to 'Submit' Button

The image shows the Scratch IDE interface. On the left, the 'Blocks' palette has the 'Lists' category highlighted with a red circle. A red arrow points from this circle to the 'select list item' block in the 'Viewer' area, which is also circled in red. Below, a script area shows a 'when submitButton .Click' event with an 'if' block. The 'if' block's condition is 'compare texts' with 'inputAnswer . Text' and an equals sign. A red circle highlights the equals sign, with a callout box saying 'Set the condition to '='. The 'select list item' block is also highlighted with a red circle, with a callout box listing two steps: 'Set list to AnswerList' and 'Set index to currentQuestionIndex'. The 'select list item' block's 'list' field is set to 'get global AnswerList' and the 'index' field is set to 'get global currentQuestionIndex'.

- Set list to 'AnswerList'
- Set index to 'currentQuestionIndex'

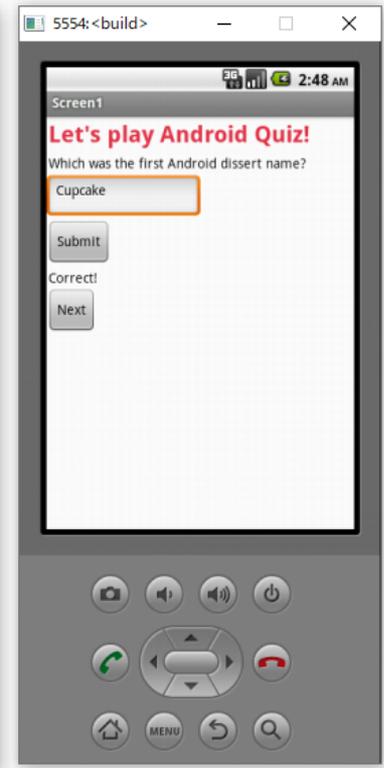
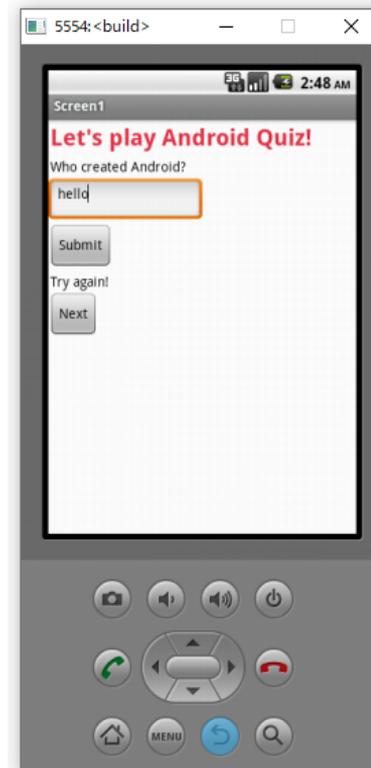
Set the condition to '='

# Add Click Handler to 'Submit' Button



# Test App on Emulator

```
initialize global QuestionList to make a list  
  "Who created Android?"  
  "Which was the first Android dissert name?"  
  "What is the name of the Android app marketplace?"  
  "What Google-branded phone did the company launch..."  
  "What candy bar is Android 4.4 named after?"  
  
initialize global AnswerList to make a list  
  "Andy Rubin"  
  "Cupcake"  
  "Google Play"  
  "Nexus 5"  
  "Kit Kat"  
  
when Screen1 .Initialize  
do set showQuestionLabel . Text to select list item list get global QuestionList  
  index 1  
  
initialize global currentQuestionIndex to 1  
  
when nextButton .Click  
do set global currentQuestionIndex to get global currentQuestionIndex + 1  
  if get global currentQuestionIndex > length of list list get global QuestionList  
  then set global currentQuestionIndex to 1  
  set showQuestionLabel . Text to select list item list get global QuestionList  
  index get global currentQuestionIndex  
  
when submitButton .Click  
do if compare texts inputAnswer . Text = select list item list get global AnswerList  
  index get global currentQuestionIndex  
  then set correctIncorrectLabel . Text to "Correct!"  
  else set correctIncorrectLabel . Text to "Try again!"
```



# Code Anatomy: Android Quiz App

initialize global `QuestionList` to `make a list` `"Who created Android?"` `"Which was the first Android dissert name?"` `"What is the name of the Android app marketplace?"` `"What Google-branded phone did the company launch..."` `"What candy bar is Android 4.4 named after?"` → Initialize a global variable with five questions

initialize global `AnswerList` to `make a list` `"Andy Rubin"` `"Cupcake"` `"Google Play"` `"Nexus 5"` `"Kit Kat"` → Initialize a global variable with the corresponding answers

when `Screen1` `.Initialize`  
do `set` `showQuestionLabel` `.Text` to `select list item` `list` `get global QuestionList` `index` `1` → Show the first question when app starts

initialize global `currentQuestionIndex` to `1` → Initialize a global variable to store the index of the current question

when `nextButton` `.Click`  
do `set` `global currentQuestionIndex` to `get global currentQuestionIndex` `+` `1` → If 'Next' button is clicked, increase the index by 1  
`if` `get global currentQuestionIndex` `>` `length of list` `list` `get global QuestionList` → If it's larger than the length of the question list, reset it to 1.  
then `set` `global currentQuestionIndex` to `1`  
`set` `showQuestionLabel` `.Text` to `select list item` `list` `get global QuestionList` `index` `get global currentQuestionIndex`

when `submitButton` `.Click`  
do `if` `compare texts` `inputAnswer` `.Text` `=` `select list item` `list` `get global AnswerList` `index` `get global currentQuestionIndex` → Get user's answer and compare with the correct answer  
then `set` `correctIncorrectLabel` `.Text` to `"Correct!"`  
else `set` `correctIncorrectLabel` `.Text` to `"Try again!"`

# Assignment

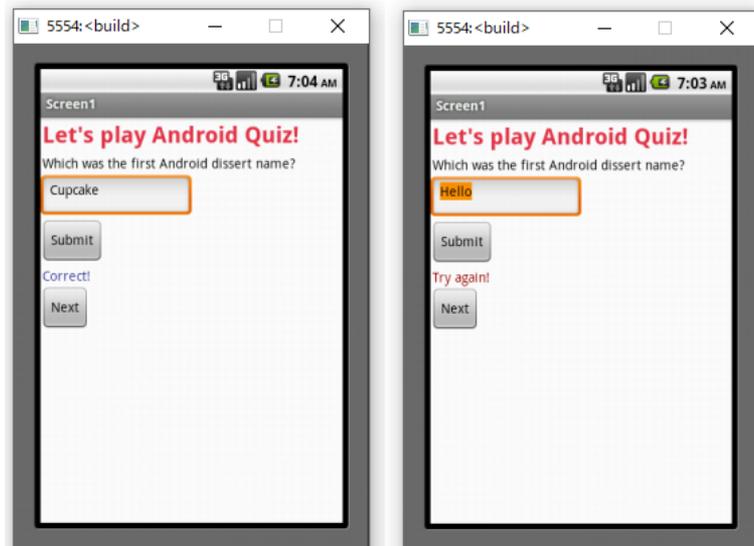
## 5.1 Complete the hands-on tasks in the tutorial

- ✓ If you finish all the steps in class, show your Android Quiz App to one of the instructors before you leave
- ✓ If you cannot finish all the steps, you can work on them after class and show your app to one of the instructors in the class next week

# Assignment (Optional)

If you have time, why not challenging these advanced tasks?

- 5.2 Add more questions and answers to the pool. E.g., Who Dream of Electric Sheep? What is the Answer to the Ultimate Question of Life, the Universe, and Everything?
- 5.3 When user's answer is correct, show the message 'Correct!' in blue color; when user's answer is incorrect, show the message 'Try again!' in red color.
- 5.4 Remove the text in 'inputAnswer' and 'correctIncorrectLabel' from last answer when user click the 'Next' button.



KUAS

KYOTO UNIVERSITY of ADVANCED SCIENCE

京都先端科学大学