

MUTLIMEDIA DEVELOPMENT

HTIW

APP INVENTOR





















MAKE QUIZ

APP INVENTOR



















FEATURES OF THE APP



Quiz Development with App Inventor

- Introduction to Database
- Database concepts in App Inventor
- Input form for entering of information
- Displaying items from multiple lists
- Sharing a web database with 2 phones



QUIZ APP WHAT YOU'LL CREATE



MakeQuiz and TakeQuiz are two apps that let students take quiz created by their teacher.

- On top of that, parents can create fun trivia apps for their children during a long road trip.
- Grade school teachers can build "Math Blaster" quizzes, and college students can build quizzes to help their study groups prepare for a final.



QUIZ &PP OVERVIEW



You'll design two apps, MakeQuiz for the "teacher" and TakeQuiz for the "student".

With MakeQuiz:

- The user enters questions and answers in an input form.
- The previously entered question-answer pairs are displayed.
- The quiz is stored persistently, in a database.

TakeQuiz will differ in that the <u>questions asked will be those</u> that were entered into the database using MakeQuiz.

QUIZ APP INITIAL STEPS



- 1. Go to "http://ai2.appinventor.mit.edu/"
- 2. Sign-in with your Gmail account. If you do not have a Gmail account, you can create a free account at "https://www.google.com.sg/".
- 3. Click 'Start new project'



- 4. Enter Application Name for project : MakeQuiz
- 5. You'll be presented with the Component Designer



COMPONENT DESIGNER



- ☐ Use the Component Designer to create the interface for MakeQuiz.
- Drag each component from the Palette into the Viewer and name it as specified in the table. Note that you can leave the header label names (Label1 Label4) as their defaults (you won't use them in the Blocks Editor anyway).

Component type	Palette group	What you'll name it	Purpose
TableArrangement	Screen Arrangement	TableArrangement1	Format the form, including the question and answer.
Label	Basic	Label1	The "Question:" prompt.
TextBox	Basic	QuestionText	The user enters questions here.
Label	Basic	Label2	The "Answer:" prompt.
TextBox	Basic	AnswerText	The user enters answers here.
Button	Basic	SubmitButton	The user clicks this to submit a QA pair.
Label	Basic	Label3	Display "Quiz Questions and Answers."
Label	Basic	QuestionsAnswersLabel	Display previously entered QA pairs.
TinyWebDB	Not ready for prime time	TinyWebDB1	Store data to and retrieve data from the database.



COMPONENT DESIGNER



er	Components	Properties
Screen Question: Answer: Submit Quiz Questions and Answers	TableArrangement1 A Label2 QuestionText AnswerText SubmitButton A Label3 A QuestionsAnswersLabel TinyWebDB1	BackgroundColor None FontBold FontItalic FontSize 14.0 FontTypeface default Question: Text Question: TextAlignment left Black Visible showing \$
Non-visible components		Width
The Mile DD1	Media	Automatic
TinyWebDB1	Upload File	Height
	opioud rife iii	Automatic



COMPONENT DESIGNER - PROPERTIES SETUP



- 1. Set the Text of Label1 to "Question", the Text of Label2 to "Answer", and the text of Label3 to "Quiz Questions and Answers".
- 2. Set the FontSize of Label3 to 18 and check the FontBold box.
- 3. Set the Hint of QuestionText to "Enter a question" and the Hint of AnswerText to "Enter an answer".
- 4. Set the Text of SubmitButton to "Submit".
- 5. Set the Text of QuestionsAnswersLabel to "Questions and Answers".
- 6. Move the QuestionText, AnswerText, and their associated labels into TableArrangement1.







- 7. You'll first define some global variables for the QuestionList and AnswerList.
- 8. The lists are defined without items in the slots. This is because with MakeQuiz and TakeQuiz, all data will be created by the app user (it is dynamic, user-generated data)

```
initialize global QuestionList to create empty list initialize global AnswerList to create empty list
```

Note: The lists are defined without items in the slots. This is because with MakeQuiz and TakeQuiz, all data will be created by the app user (it is dynamic, user-generated data).





COMPONENT DESIGNER - RECORDING THE USER'S ENTRIES.

9. The first behavior you'll build is for handling the user's input. Specifically, when the user enters a question and answer and clicks Submit, you'll use add item to list blocks to update the QuestionList and AnswerList.

```
The user clicking on submit
                      triggers the processing.
when SubmitButton . Click
                                                             Take user input from textboxes
                                                              and add to lists.
     ? add items to list list
                                 get global QuestionList -
                                 QuestionText - Text -
  add items to list list get global AnswerList
                             AnswerText - Text
 ? set QuestionsAnswersLabel . Text to
                                                      0
                                                          join
                                                                 get global QuestionList
                                                                  global AnswerList
                                                                display as two lists separated by
                                                                 a colon (for now)
```



MULTIMEDIA APP

BLOCK EDITOR - BLANKING OUT QUESTION AND ANSWER

In this app, when a user submits a question-answer pair, you'll want to clear the QuestionText and AnswerText text boxes so that they're ready for a new entry instead of showing the previous one. The blocks should appear as those shown in Figure



BLOCK EDITOR - DISPLAYING QUESTION-ANSWER PAIRS ON MULTIPLE LINES

The task here is a bit more complicated, as you're dealing with two lists. Because of its complexity, you'll put the blocks for it in a procedure named displayQAs, and call that procedure from the SubmitButton.Click event handler.

To display question-answer pairs on separate lines, you'll need to do the following:

- Use a foreach block to iterate through each question in the QuestionList.
- Use a variable answerIndex so that you can grab each answer as you iterate through the questions.
- Use make text to build a text object with each question and answer pair, and a newline character (\n) separating each pair.



QUIZ APP BLOCK EDITOR -



The blocks should appear as:

```
initialize global (answerIndex) to
                                                    for each can only walk through one list at a time
initialize global answer to
                                                    (QuestionList) so we use an index (answerIndex)
                                                    to simultaneously walk through the answers.
    to displayQAs
                                                                    get the answer corresponding to
     ? set QuestionsAm
                                           Text ▼
                                __apel -
                                                                    each question.
     set global ... werIndex - to
                                     get global CostionList
         for each question in list
                                     ? select list item list get global AnswerList
           set global answer - to
                                                                     global answerIndex
                                                                                                   increment the index to
                                                                                                   get to the next answer
           ? set global answerIndex - to [ 0]
                                                    get (global answerIndex 🕶
           set QuestionsAnswersLabel
                                           . Text ▼
                                                            QuestionsAnswersLabel
                                                                           " (n
                                                                                                    show QA pairs, one on
                                                                           get question
                                                                                                     each line (\n is newline)
                                                                           get global answer
```







You now have a procedure for displaying the question-answer pairs, but it won't help unless you call it when you need it. Modify the SubmitButton.Click event handler by calling displayQAs instead of displaying the lists with the simple set QuestionsAnswersLabel.Text to block. The updated blocks should appear as shown in Figure

```
when SubmitButton . Click
    ? add items to list list
                               get global QuestionList
                               QuestionText - Text -
 add items to list list |
                          get global AnswerList 🗸
                                                         The call to displayQAs replaces
                              global AnswerText
                                                         the blocks below
 ? call displayQAs -
                                                                                                                 get global QuestionList
                                                       set QuestionsAnswersLabel ▼ . Text ▼ to
                                                                                                      o join
     QuestionText - . Text - to
 set AnswerText - . Text -
                                                                                                                 get global AnswerList
```



BLOCK EDITOR - DATABASE STORAGE



So far, you've created an app that puts the entered questions and answers into a list. But what happens if the quiz maker closes the app?

- Storing the data persistently will allow the quiz maker to view or edit the latest update of the quiz each time the app is opened.
- Persistent storage is also necessary because the TakeQuiz app needs access to the data as well.
- For this purpose you'll be using the TinyWebDB component to store and retrieve data in a database. TinyWebDB stores data in databases that live on the Web.
- Because **TinyWebDB** stores data on the Web, the quiz taker can access the quiz questions and answers on a different device than the quiz maker's. (Online data storage is often referred to as the cloud.)





BLOCK EDITOR - GENERAL SCHEME FOR MAKING THE LIST DATA PERSISTENT:

Here's the general scheme for making list data—like the questions and answers—persistent:

- Store a list to the database each time a new item is added to it.
- When the app launches, load the list from the database into a variable.

Start by storing the QuestionList and AnswerList in the database each time the user enters a new pair.





BLOCK EDITOR

The TinyWebDB1.StoreValue blocks store data in a web database. StoreValue has two arguments: the tag that identifies the data and the value that is the actual data you want to store. As shown in Figure , the QuestionList is stored with a tag of "questions" while the AnswerList is stored with a tag of "answers." However, for your app, you should use tags that are more distinctive than "questions" and "answers" (e.g., "DavesQuestions" and "DavesAnswers").

```
? when SubmitButton . Click
    2 add items to list list get global QuestionList
                                QuestionText - Text -
    o add items to list list 🍃 get global AnswerList 🔻
                            AnswerText -
                                          Text
                      item |
       call displayQAs -
        QuestionText - . Text - to
                                                        Copy the QuestionList
        AnswerText ▼ Text
                                                        data to the database
                                                        (and AnswerList below)
    ? cail TinyWebDB1 .StoreValue
                                          questions
                         valueToStore
                                        get global QuestionList
        answers
                      valueToStore
                                     get global AnswerList
```







- Let's program the blocks for loading the lists back into the app from the web database each time the app is restarted.
- In this case, the app needs to request two lists from the TinyWebDB web
 database—the questions and the answers—so the Screen1.Initialize will make
 two calls to TinyWebDB.GetValue. The blocks should appear as:

TinyWebDB.GetValue only requests the data from the web database; it doesn't immediately receive a value. Instead, when the data arrives from the web database, a **TinyWebDB.GotValue** event is triggered. You must also program another event handler to process the data that is returned.

BLOCK EDITOR - LOADING DATA FROM DATABASE



When the TinyWebDB.GotValue event occurs, the data requested is contained in an argument named valueFromWebDB. The tag you requested is contained in the argument tagFromWebDB.

In this app, since two different requests are made for the questions and answers, GotValue will be triggered twice. To avoid putting questions in your AnswerList or vice versa, your app needs to check the tag to see which request has arrived, and then put the value returned from the database into the corresponding list (QuestionList or AnswerList). Now you're probably realizing how useful those tags really are!



BLOCK EDITOR - PROCESSING THE LIST



```
Make sure the data returned is a
                                                                      list. The first time the app runs
 tagFromWebDB
                   valueFromWebDR
                                                                      there is no data and the empty
                                                                      string (not a list) is returned.
                ? is a list? thing get valueFromWebDB -
    then
                       get (tagFromWebDB -
                                                        " questions
                                                = -
                  set global QuestionList - to
           then
                                               get valueFromWebDB
                  set global AnswerList to get valueFromWebDB
           else
           o if
                      ? | length of list list | get global QuestionList |
                                                                                  length of list list get global AnswerList
                                                                         call displayQAs
           then
                                               You know both lists have arrived when their
                                               lengths are the same, and you only want to
                                               display them when they're both in.
```



BLOCK EDITOR - HOW THE BLOCKS WORK



The app calls TinyWebDB1.GetValue twice: once to request the stored QuestionList and once to request the stored AnswerList. When the data arrives from the web database from either request, the TinyWebDB1.GotValue event is triggered, as shown in Figure.

The valueFromWebDB argument of GotValue holds the data returned from the data- base request. We need the outer if block in the event handler because the database will return an empty text ("") in valueFromWebDB if it's the first time the app has been used and there aren't yet questions and answers. By asking if the valueFromWebDB is a list?, you're making sure there is some data actually returned. If there isn't any data, you'll bypass the blocks for processing it.



BLOCK EDITOR - HOW THE BLOCKS WORK



If data is returned (is a list? is true), the blocks go on to check which request has arrived. The tag identifying the data is in tagFromWebDB: it will be either "questions" or "answers." If the tag is "questions," the valueFromWebDB is put into the variable QuestionList. Otherwise (else), it is placed in the AnswerList. (If you used tags other than "questions" and "answers," check for those instead.)

We only want to display the lists after both have arrived (GotValue has been triggered twice). Can you think of how you'd know for sure that you have both lists loaded in from the database? These blocks use an if test to check if the lengths of the lists are the same, as this can only be true if both have been returned. If they are, the handy displayQAs procedure you wrote earlier is called to display the loaded data.



BLOCK EDITOR - MAKEQUIZ



```
initialize global answerIndex to [1]
                                                         initialize global AnswerList to Concreate empty list
initialize global QuestionList to Concreate empty list
                                                                                                                   initialize
                                   Take user input from textboxes
? when SubmitButton -
                                                                           to displayQAs
                                   and add to lists.
                                                                                                                                                display as two lists separated by
    2 add items to list list get global QuestionList
                                                                           do ? set QuestionsAnswersLabel - . Text - to
                                                                                                                                                a colon (for now)
                                   QuestionText - Text -
                                                                                set global answerIndex - to-
                                                                                                                                           for each can only walk through one list at a
                                                                                                                                           time (QuestionList) so we use an index
     add items to list list
                               get global AnswerList -
                                                                                ? for each question in list / get global QuestionList
                                                                                                                                           (answerIndex) to simultaneously walk through
                               AnswerText - Text -
                                                                                do set global answer to 7 select list item list
                                                                                                                                         get global AnswerList
     7 caii displayQAs -
                                                  The call to displayQAs replaces
                                                                                                                                         get global answerIndex
                                                  the blocks below
     set QuestionText . Text to ( ?
                                                                                     ? set global answerIndex - to [ 0 ]
                                                                                                                               get global answerIndex -
     set AnswerText . Text -
                                                      Copy the QuestionList
                                                                                     set QuestionsAnswersLabel - . Text - to
                                                                                                                                    o poin QuestionsAnswersLabel - Text -
                                                      data to the database
     ? caii TinyWebDB1 .StoreValue
                                                      (and AnswerList below)
                                                                                                                                                   " (m "
                                              questions
                                                                                                                                                   get question -
                            valueToStore
                                             get global QuestionList -
     call TinyWebDB1 . StoreValue
                                                                                                                                                   get global answer
                                           answers "
                                           🚺 global AnswerList 🕶
                                                                                                                                                GetValue only
  when TinyWebDB1 .GotValue
                                                                                                            when Screen1 - .Initialia
                                                                          Make sure the data returned is a
                                                                                                                                                requests the data
                                                                          list. The first time the app runs
    tagFromWebDB
                      valueFromWebDB
                                                                                                            do 7 call TinyWebDB1 .GetValue
                                                                          there is no data and the empty
                                                                          string (not a list) is returned.
  do 👩 if
                  ? is a list? thing
                                       get valueFromWebDB -
                                                                                                                                                       questions
       then
             (0) i
                            get tagFromWebDB
                                                           questions
                                                                                                                 call TinyWebDB1 ▼ .GetValue
                                                                                                                                                   answers
                                                                                                                                            tag
                    set global QuestionList - to
                                                    get valueFromWebDB
                    set global AnswerList - to get valueFromWebDB
                                                   get global QuestionList - = -
                                                                                     length of list list get global AnswerList
              then call displayQAs
                                                   You know both lists have arrived when their
                                                   lengths are the same, and you only want to
                                                  display them when they're both in.
```





TAKE QUIZ

APP INVENTOR



















QUIZ &PP INITIAL STEPS



- 1. Go to "http://ai2.appinventor.mit.edu/"
- 2. Sign-in with your Gmail account. If you do not have a Gmail account, you can create a free account at "https://www.google.com.sg/".
- 3. Click 'Start new project'



- 4. Enter Application Name for project : TakeQuiz
- 5. You'll be presented with the Component Designer



COMPONENT DESIGNER-TAKEQUIZ:



Viewer	Components	Properties
Display hidden components in Viewer Check to see Preview on Tablet size.	B ☐ Screen1	Screen1
Check to see Preview on Tablet size.	A QuestionLabel	AboutScreen
Take Quiz	□ AnswerText	
Question	AnswerButton	AlignHorizontal
Submit	A RightWrongLabel	Left •
	NextButton	AlignVertical Top •
Next	TinyWebDB1	AppName
		TakeQuiz
		BackgroundColor
		White
		BackgroundImage
		None
		CloseScreenAnimation
		Default •
		Icon
		None
▼	Davis Dalas	OpenScreenAnimation Default •
4 b	Rename Delete	
	Media	ScreenOrientation Unspecified •
Non-visible components	Upload File	Scrollable
TinyWebDB1	Oproductive	
,		



COMPONENT DESIGNER-TAKEQUIZ:



Add the components listed in Table :

Component type	Palette group	What you'll name it	Purpose
Label	User Interface	QuestionLabel	Display the current question.
HorizontalArrangement	Layout	HorizontalArrangement1	Put the answer text and button in a row.
TextBox	User Interface	AnswerText	The user will enter his answer here.
Button	User Interface	AnswerButton	The user clicks this to submit an answer.
Label	User Interface	RightWrongLabel	Display "correct!" or "incorrect!"
Button	User Interface	NextButton	The user clicks this to proceed to the next question.



COMPONENT DESIGNER-TAKEQUIZ:



- 1. Set QuestionLabel.Text to "Question..." (you'll input the first question in the Blocks Editor).
- 2. Set AnswerText.Hint to "Enter an answer". Set its Text property to blank. Move it into HorizontalArrangement1.
- 3. Change AnswerButton. Text to "Submit" and move it into Horizontal Arrangement 1.
- 4. Change NextButton.Text to "Next".
- 5. Change RightWrongLabel.Text to blank.
- 6. Because TakeQuiz will work with database data that resides on the Web, drag a TinyWebDB component into the app.
- 7. Because you don't want the user to answer or click the NextButton until the questions are loaded, uncheck the Enabled property of the AnswerButton and NextButton.



BLOCK EDITOR - TAKEQUIZ: AN APPLICATION FOR TAKING THE QUIZ IN THE DATABASE

8. Now, add the blocks so that the quiz given to the user is loaded from the database. The resulting blocks should appear as shown in Figure

```
initialize global questionList to ( create empty list initialize global answerList to ( create empty list
```

9. Now, modify your Screen1.Initialize so that it calls TinyWebDB.GetValue twice to load the lists, just as you did in MakeQuiz. The blocks should look as they do in Figure

```
when Screen1 .Initialize

do call TinyWebDB1 .GetValue

tag " questions "

call TinyWebDB1 .GetValue

tag " answers "
```



BLOCK EDITOR - TAKEQUIZ: AN APPLICATION FOR TAKING THE QUIZ IN THE DATABASE

10. Finally, drag out a TinyWebDB.GotValue event handler. This event handler should look similar to the one used in MakeQuiz, but here you want to show only the first question and none of the answers. Try making these changes yourself first, and then take a look at the blocks in Figure to see if they match your solution.

```
when TinyWebDB1 GotValue
                   valueFromWebDB
 tagFromWebDB
                                                                          Show the first question once the
               is a list? thing get valueFromWebDB
                                                                          question list is loaded.
    then
                         get tagFromWebDB -
                                                         questions
                  set global questionList - to
           then
                                               get valueFrom UDB
                                                      ? select list item list get global questionList
                     QuestionLabel - Text - to
                  set global answerList - to get valueFromWebDR -
                                                                                Enable the buttons once
                     sei AnswerButton - . Enabled -
                                                          true -
                                                                                the answers have arrived
                                                                                from the web.
                     NextButton - Enabled -
```

BLOCK EDITOR - HOW THE BLOCK WORKS



When the app starts, Screen1.Initialize is triggered and the app requests the questions and answers from the web database. When each request arrives, the Tiny WebDB.GotValue event handler is triggered. The app first checks if there is indeed data in valueFromWebDB using is a list? If it finds data, the app asks which request has come in, using tagFromWebDB, and places the valueFromWebDB into the appropriate list.

If the QuestionList is being loaded, the first question is selected from QuestionList and displayed. If the AnswerList is being loaded, the AnswerButton and NextButton are enabled so that the user can begin taking the test. These are all the changes you need for TakeQuiz. If you've added some questions and answers with MakeQuiz and you run TakeQuiz, the questions that appear should be the ones you input.



BLOCK EDITOR - THE COMPLETE APP: TAKE QUIZ



Figure 10-15 shows the blocks for the entire TakeQuiz app.

```
initialize global (AnswerList) to 🕻 🧿 create empty list
                                                          when Screen1 ▼ .Initialize
                                                               call TinyWebDB1 ▼ .GetValue
questions
                                                                                      taq
initialize global index to 11
                                                               call TinyWebDB1 ▼ .GetValue
                                                                                             answers
when TinyWebDB1 ▼ .GotValue
 tagFromWebDB
                 valueFromWebDB
           is a list? thing / get valueFromWebDB v
    if 🔯
    then
                  get (tagFromWebDB ▼)
                                               " questions
                set global QuestionList ▼ to
                                           get valueFromWebDB
          then
                set QuestionLabel ▼
                                   Text ▼
                                                select list item list get global QuestionList v
                                          to 🔝
                                                               index
                set global AnswerList v to get valueFromWebDB v
          else
                   set AnswerButton v . Enabled v to true v
                   NextButton ▼ . Enabled ▼ to  true
```



BLOCK EDITOR - THE COMPLETE APP - CONTD



```
when NextButton ▼ .Click
    ? set global index ▼ to 
do
                                  🏮 get (global index 🔻
    Ø if
                  get global index ▼
                                       length of list list 👂 get global QuestionList 🔻
          set global index v to 1

    select list item list  
    get global QuestionList ▼

    set QuestionLabel ▼ . Text ▼ to
                                                              🕽 get (global index 🔻
                                                       index
        set AnswerText ▼ . Text ▼ to
        RightWrongLabel V . Text V
      AnswerButton v
                       Click
                       upcase ▼
                                    AnswerText ▼
                                                  Text ▼
 do
                                    select list item list | get global AnswerList v
                       upcase v
                                                index 🖟 🕜 get global index 🔻
            set RightWrongLabel v
                                   . Text v to
                                                  Great You're Awesome!
     then
            set RightWrongLabel . Text to
                                                  Incorrect Sorry.
      else
```



VARIATIONS - TO EXPLORE



- Allow the quiz maker to specify an image for each question. This is a little
 complicated because TinyWebDB doesn't allow you to store images. Therefore,
 the images will need to be URLs to pictures on the Web, and the quiz maker
 will need to enter these URLs as a third item in the MakeQuiz form. Note that
 you can set the Picture property of an Image component to a URL.
- Allow the quiz maker to delete items from the questions and answers. You can
 let the user choose a question by using the ListPicker component, and you can
 remove an item with the remove list item
- Let the quiz maker name the quiz. You'll need to store the quiz name under a different tag in the database, and you'll need to load the name along with the quiz in TakeQuiz. After you've loaded the name, use it to set the Screen. Title property so that it appears when the user takes a quiz.
- Allow multiple named quizzes to be created. You'll need a list of quizzes, and you can use each quiz name as (part of) the tag for storing its questions and answers.





Here are some of the concepts we covered in this chapter:

- Dynamic data is information input by the app's user or loaded in from a database. A program that works with dynamic data is more abstract.
- You can store data persistently in a web database with the TinyWebDB component.
- You retrieve data from a TinyWebDB database by requesting it with TinyWebDB.GetValue. When the web database returns the data, the TinyWebDB.GotValue event is triggered. In the TinyWebDB.GotValue event handler, you can put the data in a list or process it in some way.
- TinyWebDB data can be shared among multiple phones and apps





THANK YOU

SEE YOU &GAIN

















