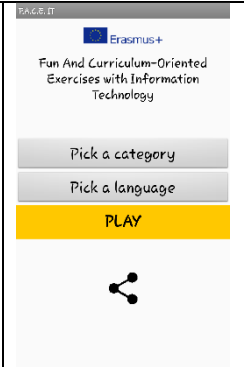






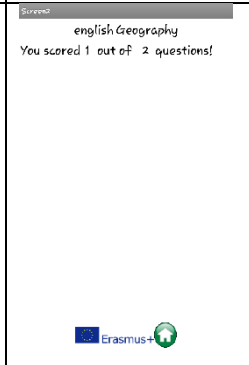


Explaining the concept

You will create the very first version (v1) of the “F.A.C.E. IT” Quiz app(lication). The user will choose a *category* and a *language* to play the Quiz. He/She will proceed through the questions by choosing an answer (correct or wrong). At the end the final score will be shown.

<ul style="list-style-type: none"> • This will be the initial user interface. • At the top there is an “Erasmus+” logo with the title of the Quiz application. • Below, a list picker for categories and one for languages. • ...followed by a “Play” button to start the game. • At the bottom, a Share Application icon. 	
<ul style="list-style-type: none"> • Since we have to keep it simple the Quiz app will be limited to 1 category (Geography) and 1 language (English). • Should someone pick a different set (category, language) a message should inform him/her about the only option available. 	<p>Demo version. Only works for category “Geography” and language “english”. Please try again. Click Home button. Thank you.</p>
<ul style="list-style-type: none"> • The user has to choose the only working combination of “Geography” and “english”. After choosing a language an icon with the country flag will appear next to it. • Pressing the “PLAY” button will start the game. 	
<ul style="list-style-type: none"> • The second screen will appear. • At the top it will inform the user about the current <i>language</i> and <i>category</i>. • The score will be set initially to zero (0). • The question will be in blue color. • The answers will be buttons to click on. • At the bottom there will be an “Erasmus+” logo and a “Home” button to start over. 	



<ul style="list-style-type: none"> • After clicking on an answer, <ul style="list-style-type: none"> ○ ...the correct answer will be highlighted in green, ○ ...the score will be updated with +1 for correct answering or nothing and ○ ...the “NEXT” button will appear at the bottom. 	
<ul style="list-style-type: none"> • We will provide <u>only</u> 2 questions • ...to keep it as simple as possible. 	
<ul style="list-style-type: none"> • At the end the final score will be displayed. • The user will have to click the “Home” button to start over again. 	

Let's start!



Worksheet 5 (Part A)

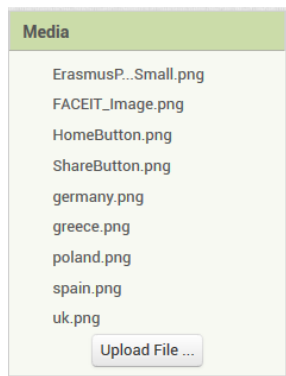
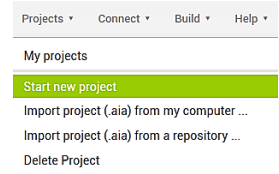
Creating the user interface of the Quiz app

Activity: You will create a simple **user interface** for the Quiz app which will allow the user to choose the **category** of the questions and the **language**.

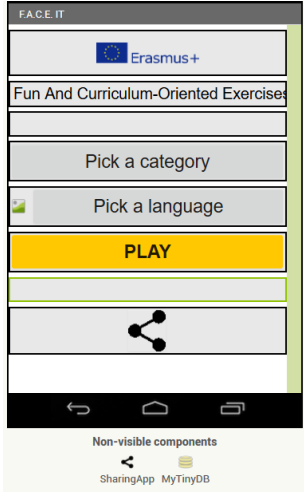
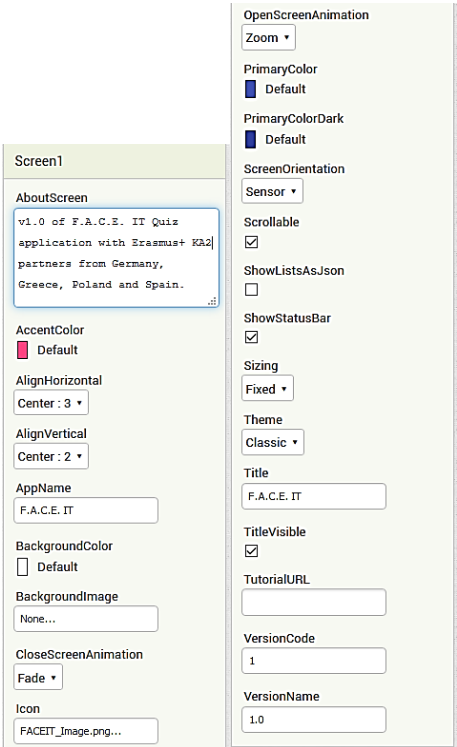
Time: 30 Minutes

*Follow your facilitator and complete the tasks below (put a mark if completed).
Don't hesitate to ask if there is something you are not sure of.*

TASK	DONE?
<p>Connecting to AppInventor</p> <ul style="list-style-type: none"> Open a browser, e.g. Firefox, Chrome, MS Edge, Safari. In the address bar of your browser type http://ai2.appinventor.mit.edu . Sign in with your Google Account. After this you're presented with the AppInventor environment. 	
<p>Starting a new project</p> <ul style="list-style-type: none"> From the menu click "Projects" and then choose "Start New Project". Enter the project name, FACE_IT_v1, and then click OK. 	
<p>Uploading images to AppInventor</p> <ul style="list-style-type: none"> You will need 9 pictures for this version of the app. The images are in a zipped file format located in the Twinspace Material folder. <ul style="list-style-type: none"> Sign in to your Twinspace account. Follow the path "Material → Files → DE-Workshops" and download the file "QuizAppImages.zip" or directly under http://twinspace.etwinning.net/files/collabspace/1/21/721/44721/files/b3f920b7.zip After downloading unzip the files at your Desktop. Upload each one of the files to your AppInventor project. NOTE: You cannot upload all files at once. After completing your Media section should look like this 	

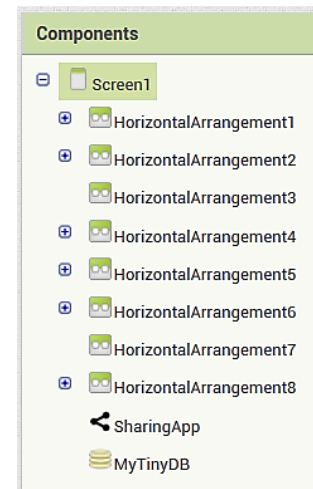




TASK	Screenshot
<ul style="list-style-type: none"> This is a screenshot of the user interface of your Screen1 which you will design. 	
<ul style="list-style-type: none"> Edit the Screen1 properties as shown. <p>NOTE: The screenshots are splitted in two halves.</p> <p>AboutScreen: “v1.0 of F.A.C.E. IT Quiz application with Erasmus+ KA2 partners from Germany, Greece, Poland and Spain.”</p>	



- The interface consists of
8 HorizontalArrangements,
1 SharingApp and
1 TinyDB.



- Check and edit the properties of all **8 HorizontalArrangements** as shown.

<p>HorizontalArrangement1</p> <p>AlignHorizontal Center : 3 ▾</p> <p>AlignVertical Center : 2 ▾</p> <p>BackgroundColor Default</p> <p>Height 50 pixels...</p> <p>Width Fill parent...</p> <p>Image None...</p> <p>Visible <input checked="" type="checkbox"/></p>	<p>HorizontalArrangement2</p> <p>AlignHorizontal Left : 1 ▾</p> <p>AlignVertical Top : 1 ▾</p> <p>BackgroundColor Default</p> <p>Height Automatic...</p> <p>Width Fill parent...</p> <p>Image None...</p> <p>Visible <input checked="" type="checkbox"/></p>	<p>HorizontalArrangement3</p> <p>AlignHorizontal Center : 3 ▾</p> <p>AlignVertical Center : 2 ▾</p> <p>BackgroundColor Default</p> <p>Height 25 pixels...</p> <p>Width Fill parent...</p> <p>Image None...</p> <p>Visible <input checked="" type="checkbox"/></p>	<p>HorizontalArrangement4</p> <p>AlignHorizontal Left : 1 ▾</p> <p>AlignVertical Top : 1 ▾</p> <p>BackgroundColor Default</p> <p>Height Automatic...</p> <p>Width Fill parent...</p> <p>Image None...</p> <p>Visible <input checked="" type="checkbox"/></p>
<p>HorizontalArrangement5</p> <p>AlignHorizontal Left : 1 ▾</p> <p>AlignVertical Top : 1 ▾</p> <p>BackgroundColor Default</p> <p>Height Automatic...</p> <p>Width Fill parent...</p> <p>Image None...</p> <p>Visible <input checked="" type="checkbox"/></p>	<p>HorizontalArrangement6</p> <p>AlignHorizontal Left : 1 ▾</p> <p>AlignVertical Top : 1 ▾</p> <p>BackgroundColor Default</p> <p>Height Automatic...</p> <p>Width Fill parent...</p> <p>Image None...</p> <p>Visible <input checked="" type="checkbox"/></p>	<p>HorizontalArrangement7</p> <p>AlignHorizontal Center : 3 ▾</p> <p>AlignVertical Center : 2 ▾</p> <p>BackgroundColor Default</p> <p>Height 25 pixels...</p> <p>Width Fill parent...</p> <p>Image None...</p> <p>Visible <input checked="" type="checkbox"/></p>	<p>HorizontalArrangement8</p> <p>AlignHorizontal Center : 3 ▾</p> <p>AlignVertical Center : 2 ▾</p> <p>BackgroundColor Default</p> <p>Height Fill parent...</p> <p>Width Fill parent...</p> <p>Image None...</p> <p>Visible <input checked="" type="checkbox"/></p>

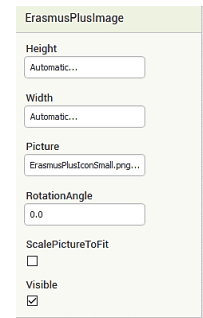


HorizontalArrangement1

- From the Palette “User Interface”, drag the **Image** component onto your **HorizontalArrangement1**.

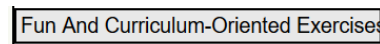


- Rename it to **ErasmusPlusImage** and
- ...edit the properties as shown:



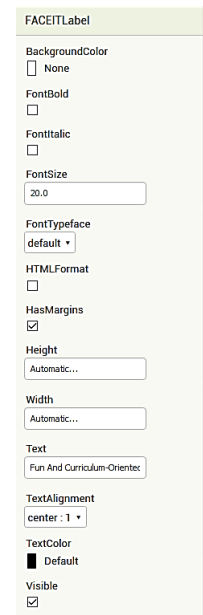
HorizontalArrangement2

- From the Palette “User Interface”, drag the **Label** component onto your **HorizontalArrangement2**.



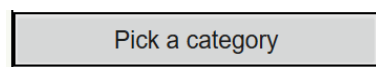
- Rename it to **FACEITLabel** and
- ...edit the properties as shown:

Text property is
“Fun And Curriculum-Oriented Exercises with Information Technology”

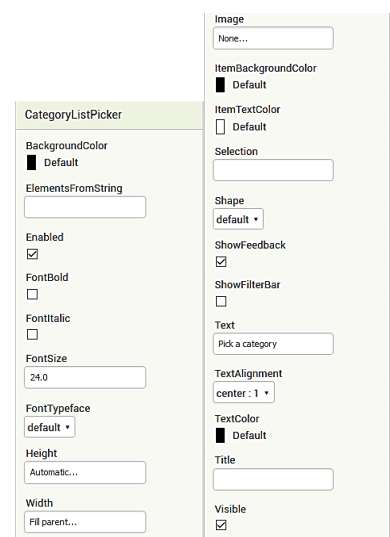


HorizontalArrangement4

- From the Palette “User Interface”, drag the **ListPicker** component onto your **HorizontalArrangement4**.



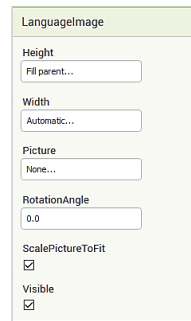
- Rename it to **CategoryListPicker** and
- ...edit the properties as shown:



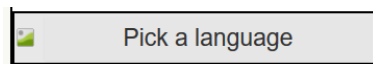


HorizontalArrangement5

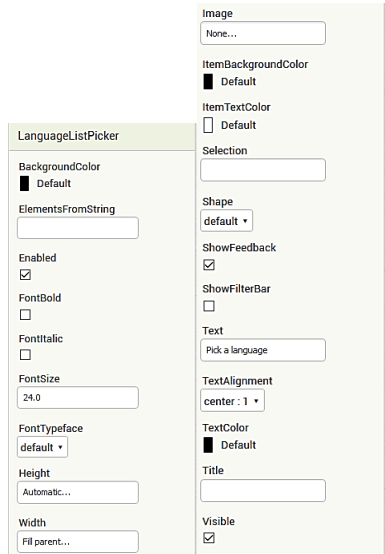
- From the Palette “User Interface”, drag the **Image** component onto your **HorizontalArrangement5**.
- Rename the **Image** to **LanguageImage** and
- ...edit the properties as shown.



- From the Palette “User Interface”, drag the **ListPicker** component onto your **HorizontalArrangement5**.



- Rename the **ListPicker** to **LanguageListPicker** and
- ...edit the properties as shown.

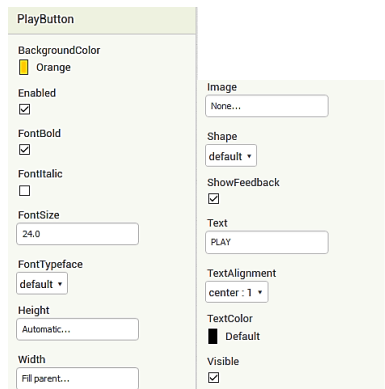


HorizontalArrangement6

- From the Palette “User Interface”, drag the **Button** component onto your **HorizontalArrangement6**.

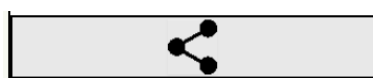


- Rename the **Button** to **PlayButton** and
- ...edit the properties as shown.

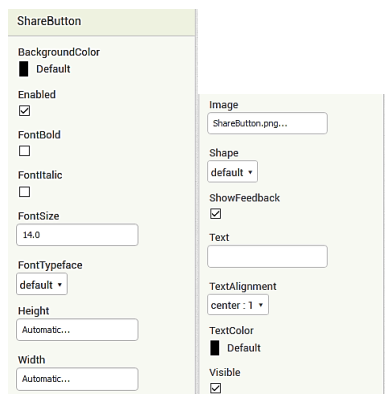


HorizontalArrangement8

- From the Palette “User Interface”, drag the **Button** component onto your **HorizontalArrangement8**.



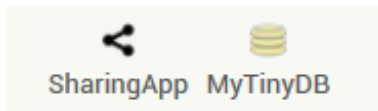
- Rename the **Button** to **ShareButton** and
- ...edit the properties as shown.





Non-visible components

- From the Palette “Social” choose the component **SharingApp** and drag it to the interface.
- From the Palette “Storage” choose the component **TinyDB** and drag it to the interface. Rename to **MyTinyDB**.

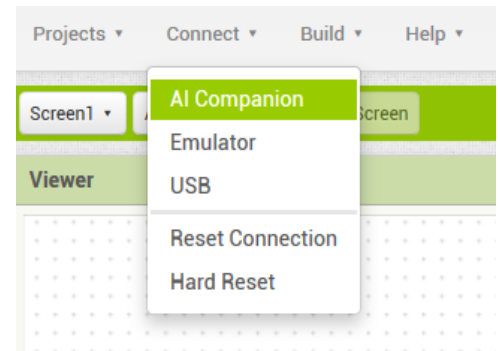


Testing the app

TIP: Before testing the app you should download the “MIT AI2 Companion” from Google Play.

Connect your smartphone or tablet from the Connect menu at the top of your screen.

1. Connect your smartphone on the Wi-Fi network.
2. Choose AI Companion from the Connect menu
3. A unique app code appears in both QR and text form.
4. On your phone start the MIT AI2 Companion app.
5. Either type the app code and click “Connect with Code” or click “Scan the QR Code” and point your phone at the QR code on your computer screen.



Try your app by pressing the buttons.



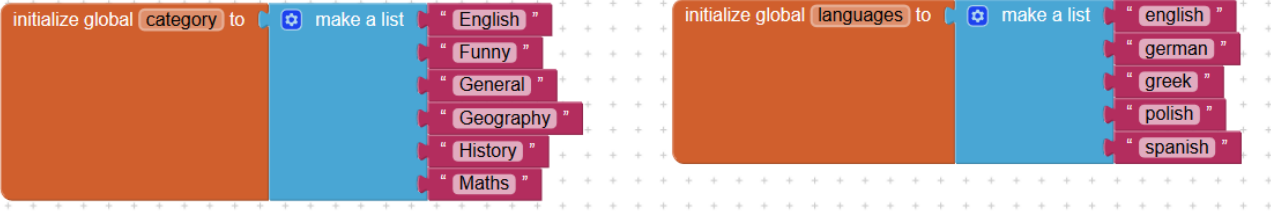
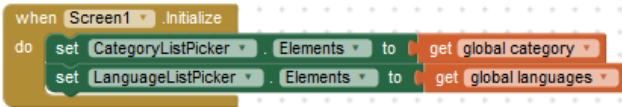
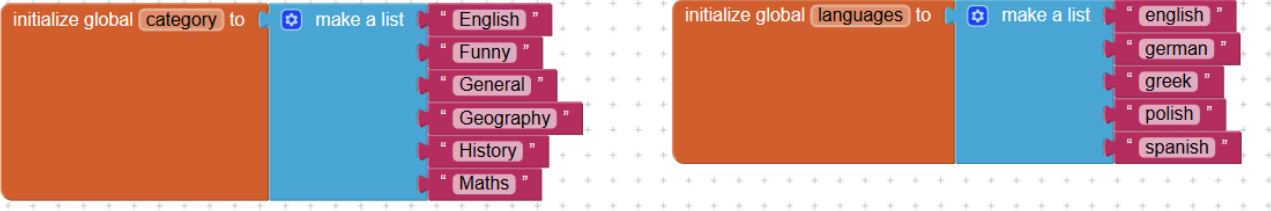
Worksheet 5 (Part B)

Programming the user interface (Screen1)

Activity: You will program the components of the user interface in order to add interactivity to your application.

Time: 20 Minutes

Follow your facilitator and complete the tasks below (put a mark if completed).
 Don't hesitate to ask if there is something you are not sure of.

TASK	Screenshot
<p>Blocks editor</p> <p>Switch to the BLOCKS editor (upper right corner).</p> <p>Initializing</p> <p>We initialize two variables as lists</p> <ul style="list-style-type: none"> • category to accommodate a list with the choices “English”, “Funny”, “General”, “Geography”, “History”, and “Maths” and • language to accommodate a list with the choices “english”, “german”, “greek”, “polish” and “spanish”.  <p>...and when Screen1 is loaded both lists get their elements from their respective variables.</p> 	



Picking elements from lists

- After the user picks a category we change the Text property of **CategoryListPicker** to reflect the user selection.
- Likewise, after picking a language we change the Text property of **LanguageListPicker** to reflect the user selection. Also, we set a picture of the respective country flag by changing the Picture property of **LanguageImage** as shown.

```

when CategoryListPicker .AfterPicking
do
  set CategoryListPicker .Text to CategoryListPicker .Selection

when LanguageListPicker .AfterPicking
do
  set LanguageListPicker .Text to LanguageListPicker .Selection
  if LanguageListPicker .Selection = "english"
  then set LanguageImage .Picture to "uk.png"
  if LanguageListPicker .Selection = "german"
  then set LanguageImage .Picture to "germany.png"
  if LanguageListPicker .Selection = "greek"
  then set LanguageImage .Picture to "greece.png"
  if LanguageListPicker .Selection = "polish"
  then set LanguageImage .Picture to "poland.png"
  if LanguageListPicker .Selection = "spanish"
  then set LanguageImage .Picture to "spain.png"
  
```

Playing

- The user has made a choice of category and language.
- Those values have to be passed on to a new screen to load the appropriate questions.
- For this purpose, we take advantage of the Database component **TinyDB** to share those values between two screens.

```

when PlayButton .Click
do
  call MyTinyDB .StoreValue
  tag "categoryID"
  valueToStore CategoryListPicker .Selection
  call MyTinyDB .StoreValue
  tag "languageID"
  valueToStore LanguageListPicker .Selection
  open another screen screenName "Screen2"
  
```

Sharing the application

- Last, we want to offer the opportunity to share the application through social media.
- At this point, we haven't yet any functional link to Google Play.
- That's why we'll show only a message with the text *"A link will be provided in future to download from Google Play."*

```

when ShareButton .Click
do
  call SharingApp .ShareMessage
  message "A link will be provided in future to download fr..."
  
```

Testing the app

- Up until this point you've created the first screen (**Screen1**) and added functionality to its components. Before proceeding with the next task you should test your app.



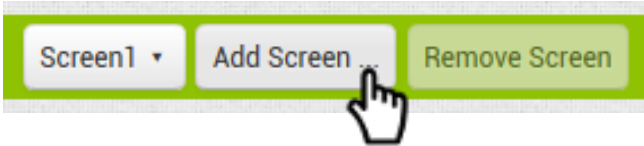
Worksheet 5 (Part C)

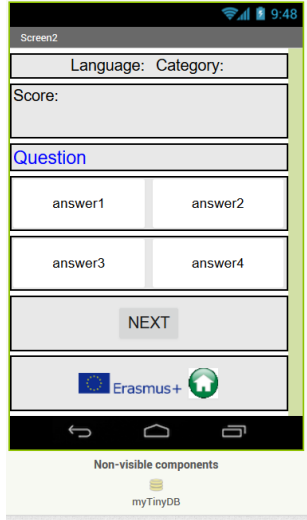
Creating the Question interface

Activity: You will create a second screen for the interface of the questions.

Time: 30 Minutes

Follow your facilitator and complete the tasks below (put a mark if completed).
Don't hesitate to ask if there is something you are not sure of.

TASK	DONE?
<p>Adding a Screen</p> <p>Click on the Add Screen button and add a new screen with the name Screen2.</p> 	

TASK	Screenshot
<ul style="list-style-type: none"> This is a screenshot of the interface of your Screen2 which you will design. 	



- Edit the **Screen2** properties as shown.

Screen2

AboutScreen

AlignHorizontal
Center : 3 ▾

AlignVertical
Center : 2 ▾

BackgroundColor
 Default

BackgroundImage

CloseScreenAnimation
Zoom ▾

OpenScreenAnimation
Zoom ▾

ScreenOrientation
Sensor ▾

Scrollable

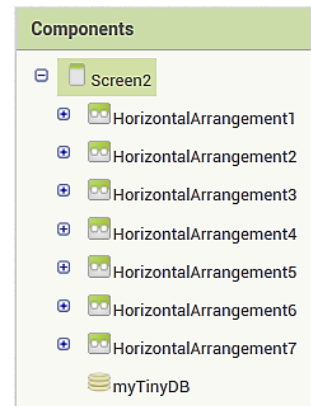
ShowStatusBar

Title

TitleVisible



- The interface consists of **7 HorizontalArrangements** and **1 TinyDB**.



- Check and edit the properties of all **7 HorizontalArrangements** as shown.

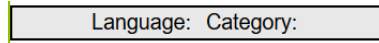
HorizontalArrangement1	HorizontalArrangement2	HorizontalArrangement3	HorizontalArrangement4
AlignHorizontal Center : 3 ▾	AlignHorizontal Left : 1 ▾	AlignHorizontal Left : 1 ▾	AlignHorizontal Left : 1 ▾
AlignVertical Center : 2 ▾	AlignVertical Top : 1 ▾	AlignVertical Top : 1 ▾	AlignVertical Top : 1 ▾
BackgroundColor ■ Default	BackgroundColor ■ Default	BackgroundColor ■ Default	BackgroundColor ■ Default
Height Automatic...	Height Fill parent...	Height Automatic...	Height Fill parent...
Width Fill parent...	Width Fill parent...	Width Fill parent...	Width Fill parent...
Image None...	Image None...	Image None...	Image None...
Visible <input checked="" type="checkbox"/>	Visible <input checked="" type="checkbox"/>	Visible <input checked="" type="checkbox"/>	Visible <input checked="" type="checkbox"/>

HorizontalArrangement5	HorizontalArrangement6	HorizontalArrangement7
AlignHorizontal Left : 1 ▾	AlignHorizontal Center : 3 ▾	AlignHorizontal Center : 3 ▾
AlignVertical Top : 1 ▾	AlignVertical Center : 2 ▾	AlignVertical Center : 2 ▾
BackgroundColor ■ Default	BackgroundColor ■ Default	BackgroundColor ■ Default
Height Fill parent...	Height Fill parent...	Height Fill parent...
Width Fill parent...	Width Fill parent...	Width Fill parent...
Image None...	Image None...	Image None...
Visible <input checked="" type="checkbox"/>	Visible <input checked="" type="checkbox"/>	Visible <input checked="" type="checkbox"/>



HorizontalArrangement1

- From the Palette “User Interface”, drag two **Label** components onto your **HorizontalArrangement1**.



- Rename the first label to **LanguageLabel** and
- ...edit the properties as shown.
- Rename the second label to **CategoryLabel** and
- ...edit the properties as shown.

LanguageLabel	CategoryLabel
BackgroundColor <input type="checkbox"/> None FontBold <input type="checkbox"/> FontItalic <input type="checkbox"/> FontSize <input type="text" value="20.0"/> FontTypeface default ▾ HTMLFormat <input type="checkbox"/> HasMargins <input checked="" type="checkbox"/> Height <input type="text" value="Automatic..."/> Width <input type="text" value="Automatic..."/> Text <input type="text" value="Language:"/> TextAlignment left : 0 ▾ TextColor <input checked="" type="checkbox"/> Default Visible <input checked="" type="checkbox"/>	BackgroundColor <input type="checkbox"/> None FontBold <input type="checkbox"/> FontItalic <input type="checkbox"/> FontSize <input type="text" value="20.0"/> FontTypeface default ▾ HTMLFormat <input type="checkbox"/> HasMargins <input checked="" type="checkbox"/> Height <input type="text" value="Automatic..."/> Width <input type="text" value="Automatic..."/> Text <input type="text" value="Category:"/> TextAlignment left : 0 ▾ TextColor <input checked="" type="checkbox"/> Default Visible <input checked="" type="checkbox"/>

HorizontalArrangement2

- From the Palette “User Interface”, drag a **Label** component onto your **HorizontalArrangement2**.

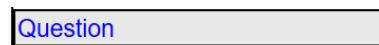


- Rename the label to **ScoreLabel** and
- ...edit the properties as shown.

ScoreLabel	Properties
BackgroundColor <input type="checkbox"/> None FontBold <input type="checkbox"/> FontItalic <input type="checkbox"/> FontSize <input type="text" value="20.0"/> FontTypeface default ▾ HTMLFormat <input type="checkbox"/>	HasMargins <input checked="" type="checkbox"/> Height <input type="text" value="Automatic..."/> Width <input type="text" value="Automatic..."/> Text <input type="text" value="Score:"/> TextAlignment left : 0 ▾ TextColor <input checked="" type="checkbox"/> Default Visible <input checked="" type="checkbox"/>

HorizontalArrangement3

- From the Palette “User Interface”, drag a **Label** component onto your **HorizontalArrangement3**.



- Rename the label to **QuestionLabel** and
- ...edit the properties as shown.

QuestionLabel	Properties
BackgroundColor <input type="checkbox"/> None FontBold <input type="checkbox"/> FontItalic <input type="checkbox"/> FontSize <input type="text" value="24.0"/> FontTypeface default ▾ HTMLFormat <input type="checkbox"/>	HasMargins <input checked="" type="checkbox"/> Height <input type="text" value="Automatic..."/> Width <input type="text" value="Automatic..."/> Text <input type="text" value="Question"/> TextAlignment left : 0 ▾ TextColor <input checked="" type="checkbox"/> Blue Visible <input checked="" type="checkbox"/>

HorizontalArrangement4

- From the Palette “User Interface”, drag two **Button** components onto your **HorizontalArrangement4**.



- Rename the first button to **AnswerButton1** and
- ...the second button to **AnswerButton2**.
- Edit the properties as shown.

AnswerButton1	Properties
BackgroundColor <input type="checkbox"/> White Enabled <input checked="" type="checkbox"/> FontBold <input type="checkbox"/> FontItalic <input type="checkbox"/> FontSize <input type="text" value="18.0"/> FontTypeface default ▾ Height <input type="text" value="Fill parent..."/> Width <input type="text" value="Fill parent..."/>	Image <input type="text" value="None..."/> Shape default ▾ ShowFeedback <input checked="" type="checkbox"/> Text <input type="text" value="answer1"/> TextAlignment center : 1 ▾ TextColor <input checked="" type="checkbox"/> Default Visible <input checked="" type="checkbox"/>

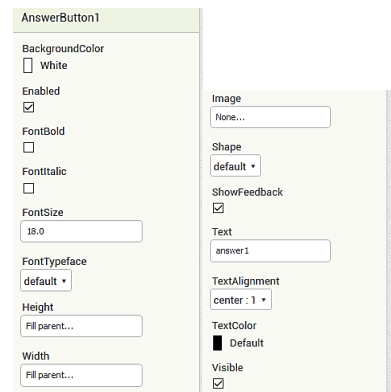


HorizontalArrangement5

- From the Palette "User Interface", drag two **Button** components onto your **HorizontalArrangement5**.

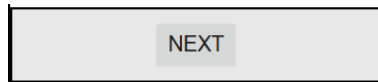


- Rename the first button to **AnswerButton3** and
- ...the second button to **AnswerButton4**.
- Edit the properties as shown.

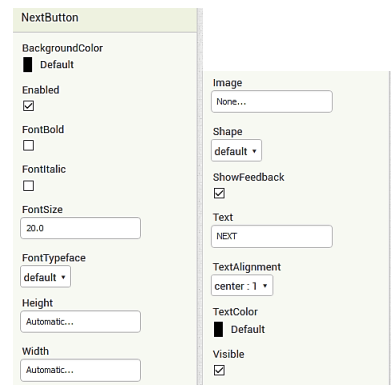


HorizontalArrangement6

- From the Palette "User Interface", drag a **Button** component onto your **HorizontalArrangement6**.

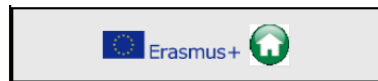


- Rename the button to **NextButton** and
- ...edit the properties as shown.

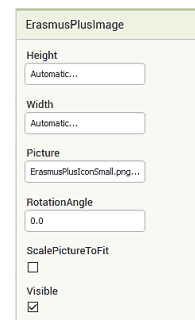


HorizontalArrangement7

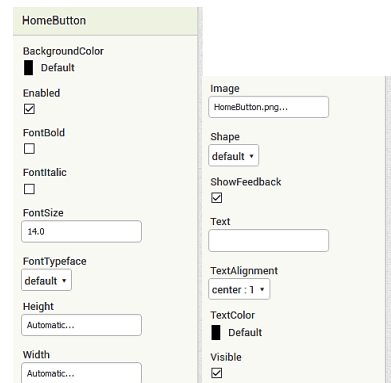
- From the Palette "User Interface", drag an **Image** and a **Button** component onto your **HorizontalArrangement7**.



- Rename the image to **ErasmusPlusImage** and
- ...edit the properties as shown.

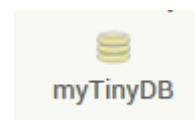


- Rename the button to **HomeButton** and
- ...edit the properties as shown.



Non-visible components

- From the Palette "Storage" choose the component **TinyDB** and drag it to the interface. Rename to **MyTinyDB**.





Testing the app

- Up until this point you've created successfully the design of the second screen (**Screen2**). Before proceeding with the next task you should test your app.




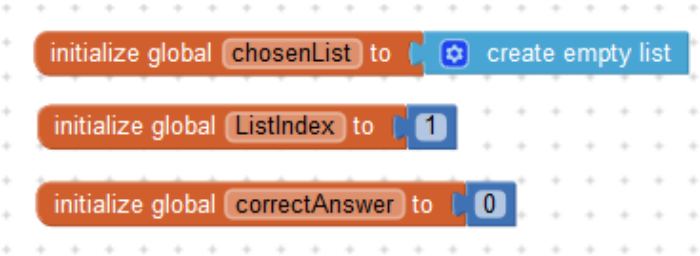

Worksheet 5 (Part D)

Programming the Question interface (Screen2)

Activity: You will program the components of the Question interface in order to pick the right list of user's choice of category and language. Then to proceed through the questions, to evaluate the user's answer, to update the score and finally to show a message of the final score.

Time: 45 Minutes

Follow your facilitator and complete the tasks below (put a mark if completed).
 Don't hesitate to ask if there is something you are not sure of.

TASK	Screenshot
<p>Blocks editor</p> <p>Switch to the BLOCKS editor (upper right corner).</p> <p>Initializing</p> <p>We will use a set of variables which have to be initialized with a proper value before using.</p> <ul style="list-style-type: none"> Variables category and language will hold the shared values of the user's choice of category and language from Screen1. Initially they will hold an empty text string.  <ul style="list-style-type: none"> The variable chosenList will be used to load the appropriate Question-Answers of the user's choice. Initially it will hold an empty list. The variable ListIndex will point to the elements in the list and the variable correctAnswer will hold the correct answer for every question.  <ul style="list-style-type: none"> The variable score will hold the current score of the user. Initially it is set to 0. The variable userAnswer will be used to compare user's choice with the correctAnswer. Both of them are initially set to 0. 	



Initializing the Question interface

- The Question interface (**Screen2**) gets the values of the **category** and the **language** from the shared Database **myTinyDB**.
- Also, sets the labels of **CategoryLabel** and **LanguageLabel** respectively to inform the user during the game.

IMPORTANT: In order to keep it simple we will create a quiz with only 2 questions and for 1 category in 1 language. For this purpose we have chosen *Geography* and *english*.

- If the criteria of “Geography” and “english” are met, the **chosenList** is loaded with the appropriate list from a csv (comma separated value) table.
- You have to fill the data of the csv table in the text field as follows
*What is the capital of Germany?, Hamburg, Munich, Berlin, Hannover, 3\n
Which is the largest river in Germany?, Elbe, Danube, Rhine, Amazon, 2\n*

Pay attention to the punctuation, such as the comma (,) and the question mark (?). Each question ends with a new line character “\n”.

IMPORTANT: At this point the procedure **initializeQA** must be called. Since you haven’t created it yet you will add this programming block later on once you’ve built it.

- If the criteria are not met, a message will inform the user to choose “Geography” and “english”. The message will be
Demo version.\n Only works for category "Geography" and language "english".\nPlease try again. Click Home button.\nThank you.

```

when Screen2.Initialize
do
  set global category to call myTinyDB.GetValue
                        tag "categoryID"
                        valueIfTagNotThere ""
  set global language to call myTinyDB.GetValue
                        tag "languageID"
                        valueIfTagNotThere ""
  set CategoryLabel.Text to get global category
  set LanguageLabel.Text to get global language
  if
    get global category = "Geography" and
    get global language = "english"
  then
    set global chosenList to list from csv table text "What is the capital of Germany?, Hamburg, Munich..."
    call initializeQA
  else
    set QuestionLabel.Text to "Demo version.\n Only works for category "Geograp..."
    set NextButton.Visible to false

```



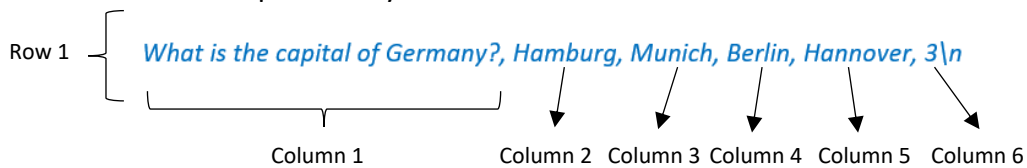
Procedure initializeQA

- This procedure will be called each time to set the labels of **Screen2** with the content of the next question, the possible answers, and the score respectively.
- In order to avoid an “empty” response from the user the **NextButton** is hidden (set to false). It will only be shown once the user makes a choice, i.e. clicks on an **AnswerButton**.

```

to initializeQA
do
  set NextButton . Visible to false
  set ScoreLabel . Text to join " Score: "
  get global score
  set QuestionLabel . Text to select list item list
  select list item list index 1
  get global chosenList
  get global ListIndex
  set AnswerButton1 . Text to select list item list
  select list item list index 2
  get global chosenList
  get global ListIndex
  set AnswerButton2 . Text to select list item list
  select list item list index 3
  get global chosenList
  get global ListIndex
  set AnswerButton3 . Text to select list item list
  select list item list index 4
  get global chosenList
  get global ListIndex
  set AnswerButton4 . Text to select list item list
  select list item list index 5
  get global chosenList
  get global ListIndex
  set global correctAnswer to select list item list
  select list item list index 6
  get global chosenList
  get global ListIndex
  
```

NOTE: We use a table for storing our questions, answers and an index to the correct answer. The table consists of rows and columns separated by commas. Each row ends with a new line character “\n”.



Column1 holds the question. Columns 2 to 5 the four possible answers. Column 6 the number for the correct answer which for the above example is 3, i.e. “Berlin”.

```

select list item list index 1
select list item list index
get global chosenList
get global ListIndex
  
```

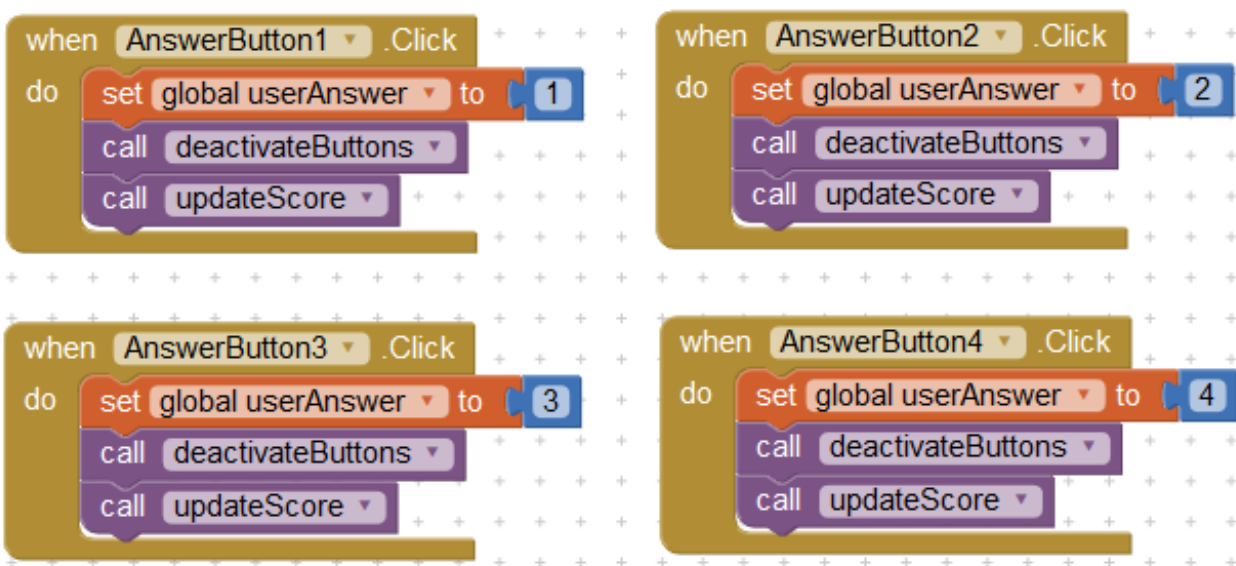
The above programming blocks translate to “Choose from the current list (variable chosenList) and the row (variable ListIndex) the value of the first column (number 1)”, i.e. the question.



Adding functionality to the AnswerButtonX

- Once the user clicks on an **AnswerButton** his/her choice is marked (**userAnswer** set to 1, 2, 3 or 4),
- ...all the other buttons are deactivated (procedure **deactivateButtons**) to prevent the user from trying many times, and
- ...the score is updated accordingly (procedure **updateScore**) with +1 if the user chose correctly.

IMPORTANT: At this point the procedures **deactivateButtons** and **updateScore** must be called. Since you haven't created them yet you will add those programming block later on once you've built them.



```

when AnswerButton1 .Click
do
  set global userAnswer to 1
  call deactivateButtons
  call updateScore

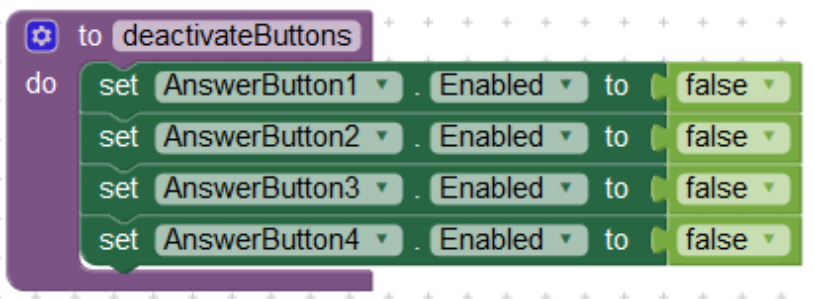
when AnswerButton2 .Click
do
  set global userAnswer to 2
  call deactivateButtons
  call updateScore

when AnswerButton3 .Click
do
  set global userAnswer to 3
  call deactivateButtons
  call updateScore

when AnswerButton4 .Click
do
  set global userAnswer to 4
  call deactivateButtons
  call updateScore
  
```

Procedure deactivateButtons

To deactivate the buttons we set the **Enabled** property to **false**.



```

to deactivateButtons
do
  set AnswerButton1 . Enabled to false
  set AnswerButton2 . Enabled to false
  set AnswerButton3 . Enabled to false
  set AnswerButton4 . Enabled to false
  
```



Procedure updateScore

- The user's answer is checked against the correct answer. If it is the same (true) the score is updated +1, otherwise not.
- The **BackgroundColor** of the correct answer is highlighted in green (procedure **highlightCorrectAnswer**).
- The value of the new score is stored in our Database **TinyDB**.
- The **NextButton** is shown, i.e. set **Visible** to **true**.

```

to updateScore
do
  if (get global userAnswer = get global correctAnswer)
  then
    set global score to (get global score + 1)
  call highlightCorrectAnswer
  set ScoreLabel . Text to (join " Score: " (get global score))
  call myTinyDB .StoreValue
  tag "userScore"
  valueToStore (get global score)
  set NextButton . Visible to true
  
```

Procedure highlightCorrectAnswer

The **BackgroundColor** of the correct **AnswerButton** is set to green.

```

to highlightCorrectAnswer
do
  if (get global correctAnswer = 1)
  then
    set AnswerButton1 . BackgroundColor to green
  else if (get global correctAnswer = 2)
  then
    set AnswerButton2 . BackgroundColor to green
  else if (get global correctAnswer = 3)
  then
    set AnswerButton3 . BackgroundColor to green
  else
    set AnswerButton4 . BackgroundColor to green
  
```



Moving to next question

- When the user clicks on the **NextButton** the **ListIndex** points to the next question,
- ...and if it hasn't reached the end of the question list,
- ...activates all the buttons (procedure **activateButtons**),
- ...changes the **BackgroundColor** of all the buttons (procedure **whiteButtons**),
- ...and sets the text of the new question with its respective choices of answers.
- Otherwise, the final score is shown (procedure **showFinalScore**).

```

when NextButton .Click
do
  set global ListIndex to (get global ListIndex + 1)
  if (get global ListIndex ≤ length of list list) (get global chosenList)
  then
    call activateButtons
    call whiteButtons
    call initializeQA
  else
    call showFinalScore
  
```

```

to activateButtons
do
  set AnswerButton1 . Enabled to true
  set AnswerButton2 . Enabled to true
  set AnswerButton3 . Enabled to true
  set AnswerButton4 . Enabled to true
  
```

```

to whiteButtons
do
  set AnswerButton1 . BackgroundColor to white
  set AnswerButton2 . BackgroundColor to white
  set AnswerButton3 . BackgroundColor to white
  set AnswerButton4 . BackgroundColor to white
  
```



Show final score

- After completing the quiz challenge for a certain category the user is shown the final score.
- For this purpose, we hide some of the components such as the **QuestionLabel**, the **AnswerButtonsX** and the **NextButton** as shown below (procedure **hideComponents**).
- The message is “You scored X out of Y questions!” where X is the user’s score and Y the total number of questions.

```

to showFinalScore
do
  call hideComponents
  set ScoreLabel . Text to join
    " You scored "
    get global score
    " out of "
    length of list list get global chosenList
    " questions! "

```

```

to hideComponents
do
  set QuestionLabel . Visible to false
  set AnswerButton1 . Visible to false
  set AnswerButton2 . Visible to false
  set AnswerButton3 . Visible to false
  set AnswerButton4 . Visible to false
  set NextButton . Visible to false

```

Testing the app

- At this point you’ve created two screens and added proper functionality to its componets. It’s time to test your app check its behaviour. Make adjustments if necessary.



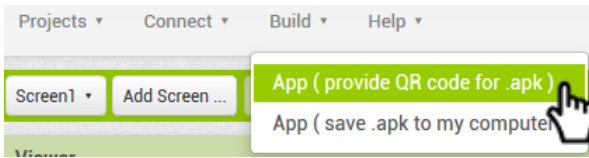
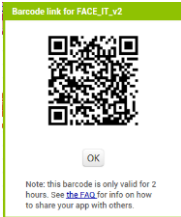
Worksheet 5 (Part E)

Downloading and installing the Application

Activity: You will download and install your recently-created Quiz App to your smartphone.

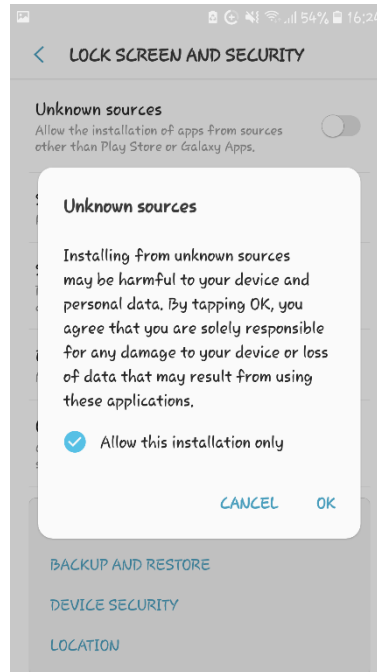
Time: 10 Minutes

*Follow your facilitator and complete the tasks below (put a mark if completed).
Don't hesitate to ask if there is something you are not sure of.*

TASK	DONE?
<p>Before downloading</p> <ul style="list-style-type: none"> AppInventor will bundle all your components and programming blocks of your Quiz app into an APK file (Android Package Kit). This package file format is used by the Android operating system for distribution and installation of mobile apps. Just like Windows systems use an .exe file for installing software. There are two options to install the app to your smartphone: ...either through a download link provided by a QR code ...or by saving it first on the computer desktop. We will go for the first option (QR code). Follow the screenshot instructions below <div style="text-align: center; margin: 10px 0;">  </div> <ul style="list-style-type: none"> After selecting the above choice, AppInventor will compile all parts, build the APK file (you will notice the progress bars) and provide the QR Code to download the file like shown below. <div style="text-align: center; margin: 10px 0;">  </div> <p>NOTE: In order to read the above QR code you have to have a “QR Reader”. If you don't have any on your smartphone or tablet download one of your choice from Google Play.</p>	



- Your security settings of your smartphone may prevent you from “installing from unknown sources”. For this time only, allow the installation since it’s your app you’ve built.



- After installation you are ready to try your application.