PIAF - Pedagogical Scenario

(PIAF = Développement de la pensée informatique et algorithmique dans l'enseignement fondamental – Development of computational and algorithmic thinking in basic education)

Title

We are music!

Practical Information

(Ideal) Number of students: 16
Age of the students: 9 – 12 years
Duration of the scenario: 3 sessions of 40 minutes each

Main discipline of the Scenario

C 1.3 Identify the input parameters of an action sequence
C 2.3 Create a sequence of actions to reach a simple goal
C 2.4 Create a sequence of actions to reach a complex goal
C 6.1 Verify if a sequence of actions reaches a given goal
C 6.2 Notice errors in a sequence of actions

Description

Learners use the block-based visual programming language App Inventor to create music apps. They identify the input components of simple and complex action sequences required to create a musical sounds. They ensure that their music apps produce the expected musical output and make any necessary corrections in terms of the coding of the app or the parameters of the app components.

PIAF-specific competencies/goals

Specific PIAF Competencies:	
C1	Competency 1: Abstracting away / generalizing > C 1.3 Identify the input parameters of an action sequence > Given a general action sequence, identify the inputs, files, actions, and parameters needed to complete each action sequence step
C2	Competency 2: Compose/decompose a sequence of actions > C 2.3 Create a sequence of actions to reach a simple goal > From a list of tasks to do, learners create and do the sub-steps needed for developing a simple app consisting of 1 button and one sound

C2	Competency 2: Compose/decompose a sequence of actions > C 2.4 Create a sequence of actions to reach a complex goal > From a list of tasks to do, learners create and do the sub-steps needed for developing complex musical apps consisting of multiple buttons and sounds
C6	Competency 6: Build a sequence of actions iteratively > C 6.1 Verify if a sequence of actions reaches a given goal > Learners compare the sound produced from their apps with the output from other students' apps, ensuring the same sequence of tones and melodies are produced
C6	Competency 6: Build a sequence of actions iteratively > C 6.2 Notice errors in a sequence of actions > Identify when the app doesn't output the expected result (e.g. generating the wrong or no sound when tapping a button)

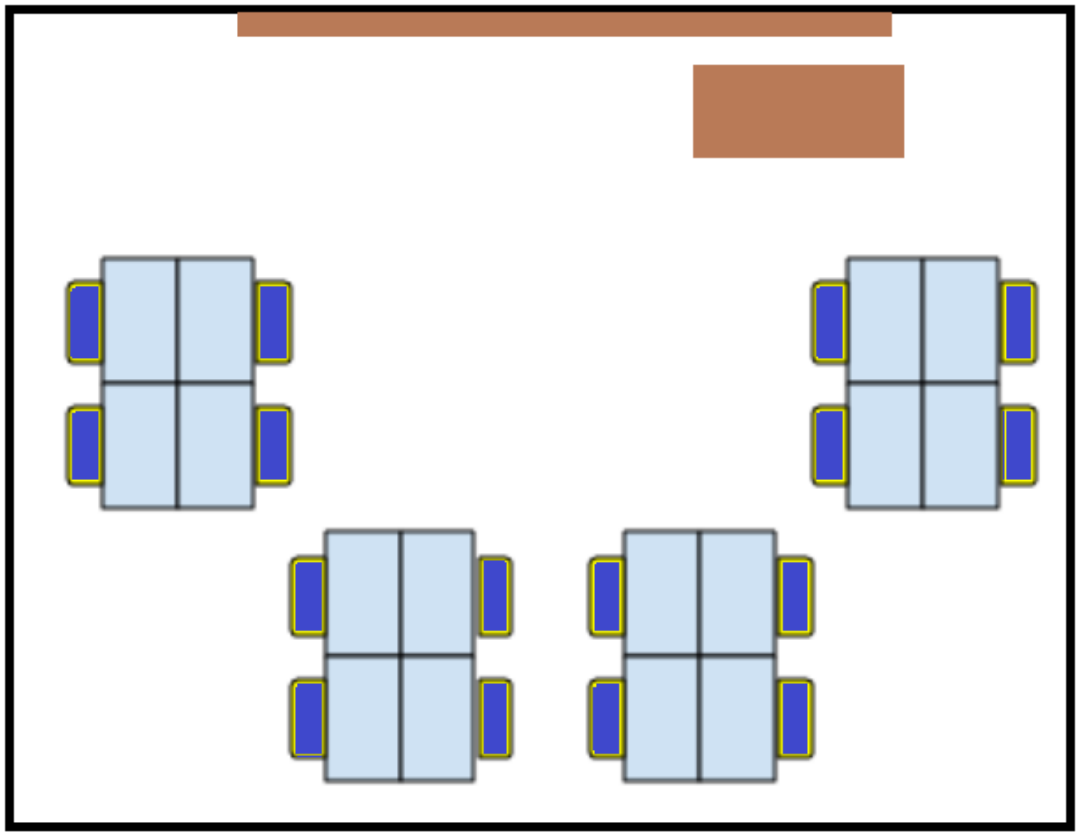
Pre-requisite for the activities

- Being able to read and follow instructions
- Basic Media competency (drag & drop, browse and upload files)

Digital Resources

Technical	**Didactic**
Laptop and Android Phone for the teacher. Per Group: one Android Phone, one tablet or PC -All tablets or PC should be able to run App Inventor: http://ai2.appinventor.mit.edu/ -All Android phones should have installed the MIT AI2 Companion app	Course notes and attachments

Organization of the classroom



Scenario (Sequence of the activities)

Activity 1- Learning Algorithms and app inventor interface		
1. Introduction (5')	Group Format: Whole class Instruction: *"How many apps you know? What are the different types of apps you know? Today you will create a phone app to make music. You will select components for your app and using the program blocks, you can define how each component behaves. As you can see, each group has a computer/ tablet with a program called App Inventor ready for you to work with."* Instructors role: Introduce the task and answer students questions	
2.Get to know App Inventor (5')	Group Format: Whole class Document: Attachment A for teacher. Instruction: *"Let' open the App Inventor!"* Students task: Follow teacher's instruction and try to create their first project. Instructors role: Introduce the main elements of the App Inventor interface.	
3. My first app (20')	Group Format: Pairs/Groups Equipment: Laptop/tablet running App Inventor, Android phones, projector. Document: Attachment B for teacher, Attachment 1 for Students Instruction: *"Now think about any app you use daily. This app for sure has buttons that you tap for something to happen: start a new level, send a message or make the phone camera take a picture. But how does that work? How can the phone know which button does what? How could we embed that function into our app?"* Students task: Students work in pairs/group to a simple one-button app that plays a sound. Instructors role: Help the students if they have questions and Intervene when needed to ensure the students are working Expected response: Being able to identify the inputs correctly for each goal	1.3 2.3
4. Trying out my first app (5')	Group Format: Whole class Equipment: Laptop/tablet running App Inventor, Android phones, projector. Document: Attachment B for teacher Instruction: *"Now let's connect App Inventor with our phones."*	6.1 6.2

	Students task: Students follow the oral instructions from the teacher to be able to pair their phones with their App Inventor window. Instructors role: Lead the whole class by showing in the projector how to pair App Inventor with the phone and test their apps.	
5. End of session: recap of the tasks, inputs and goals (5')	Group Format: Whole class Document: Attachment B for teacher Instruction: "*What did we learn today?"* Students task: Verbal description of what has been learned during this session Instructors role: Guide the students with questions for obtaining the expected answers Covered topics: ● Creating a basic function with algorithm on App Inventor	1.3 2.3
Activity 2 – Let's play piano!		
1. Introduction (5')	Group Format: Whole class Instruction: *"Let's quickly recap what we did and learned last session.* Instructors role: Ask students about what they did on the previous session Covered topics: ● Used App Inventor to create their first phone app ● They completed multiple tasks with steps to follow in a specific order ● They identified the files, components, and programming blocks that had to be used to complete the tasks ● They tried their apps out using the "Connect" function of App Inventor	
2. Building the Piano (25')	Group Format: Pairs/Groups Document: Attachment C for teachers, Attachment 2 for students. Instruction: *"Let's build your first piano app! You will receive a worksheet with a series of tasks that you need to do to make your piano app. Read and follow each step."* Students task: Students work in pairs/groups to build a one scale piano without minor keys. Instructors role: Facilitate to ensure that all students finalize the instruction.	1.3 2.4 6.1 6.2

	Expected response: playing a tune by pressing different buttons Anticipated Difficulties: Some groups might require more time to finalize the task. If this is not possible, you will need to assist the group to finalize it on time	
3. Play the tone (5')	Group Format: Whole class Instruction: *"Let's play the tone!"* Students task: Perform the music scale (C to B) together. Instructors role: Ensure all the tunes expected are played	
4. End of session: finalize activity and summarize (5')	Group Format: Whole class Instruction: "*What did we learn today?"* Students task: Verbal description of what has been learned during this session Instructors role: Guide the students with questions for obtaining the expected answers Covered topics: ● Creating a piano app using App Inventor ● Following tasks to create the piano app ● Identify inputs for each task	
Activity 3 – Let's play guitar!		
1. Introduction (5')	Group Format: Whole class Instruction: *"Who can remind me of what we did last session? Now that you are experts in music apps, we will create another musical instrument app."* Students task: Students interact by responding to questions Instructors role: Introduce the task and answer student questions Covered topics: ● Tasks done for creating the piano App	
2. Building the Guitar (20')	Group Format: Group of 3 Document: Attachment D for teachers, Attachment 3 for students. Instruction: *"Let's build your guitar app! You will receive a worksheet with a series of tasks. Read and follow each step."* Students task: Students build a guitar app. Instructors role: Facilitate to ensure that all students finalize the instruction. Expected response: playing a tune by pressing different buttons. In contrast to the piano app, the array of the buttons of the guitar app should be vertical.	1.3 2.4 6.1 6.2

	Anticipated Difficulties: Some groups might require more time to finalize the task. If this is not possible, you will need to assist the group to finalize it on time	
3. Perform (10')	Group Format: Whole class Document: Attachment D for teachers, Attachment 3 for students. Instruction: *"Let us see the final result for each group. Then we can play the tune as a band."* Students task: Play the given song using the created guitar app. Instructors role: Ensure all the tunes expected are played. Allow the students to play as a band.	
4. End of session: finalize activity and summarize (5')	Group Format: Whole class Instruction: *"What did we learn today?"* Students task: Verbal description of what has been learned during this session Instructors role: Guide the students with questions for obtaining the expected answers Covered topics: - Created a new musical app following a highly similar list of tasks as the ones for the piano app	

Assessment

Following a specific sorting method, ask the students to sort a set of numbers.

Competencies/ PIAF-Goals	Activities for the assessment	Assessment criteria
C 1.3 Identify the input parameters of an action sequence	To be able to create certain actions, list the inputs needed on app inventor	Understand the functionality of inputs on app inventor
C 2.3 Create a sequence of actions to reach a simple goal	Creation of a simple keyboard without minor keys based on a general sequence of tasks	Successful creation of the simple keyboard: correct amount, labeling, and placement of buttons that act as piano keys. Each key plays its corresponding note.
C 2.4 Create a sequence of actions to reach a complex goal	Creation of a complex keyboard with minor keys based on a general sequence of tasks	Successful creation of the complex keyboard: correct amount, labeling, and placement of buttons that act as piano keys. Each key plays its corresponding note.
C 6.1 Verify if a sequence of actions reaches a given goal	Playing a well-known song using the created complex piano keyboard app	The well-known song is correctly played
C 6.2 Notice errors in a sequence of actions	Debugging of the created piano app	Solution of any emerging problem while developing the piano app

Received Feedback on the created Scenario

If you have had the opportunity to experiment with the scenario presented here, suggest some feedback on it: what worked well, the obstacles encountered, the learner's feedback, your feelings, possible ways to improve it.

Bibliography

With MIT App Inventor, anyone can build apps with global impact. (n.d.). Retrieved October 01, 2020, from http://appinventor.mit.edu/

Game click sound retrieved from: https://mixkit.co/

Saraim Gebretsadik. (2018, April 29). *Playing the Piano*. https://www.youtube.com/watch?v=6MySEachpNw

My Piano. (n.d.). Retrieved November 10, 2020, from https://www.youtube.com/watch?v=UJFbuKgyw64

Attachments

Attachments Overview

Activity	Teacher Attachment	Student Attachment
1.2	A	
1.3	B	1
1.4	B	
1.5	B	
2.2	C	2
3.2	D	3
3.3	D	3

Teacher‘s Attachments

Teacher‘s Attachment: A

Used in activity:	1.2: Get to know App Inventor
Along with Student’s Attachment(s):	None

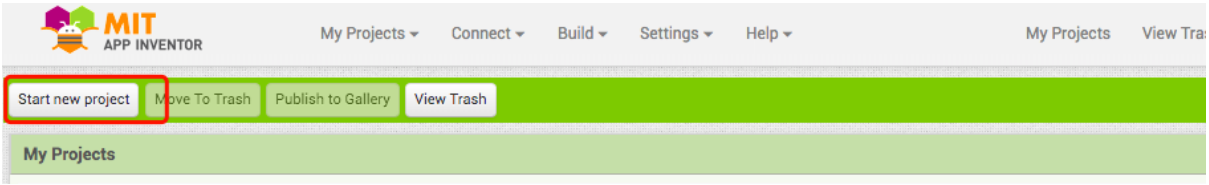
Before the session:

For programming the app, students will need to use App Inventor.

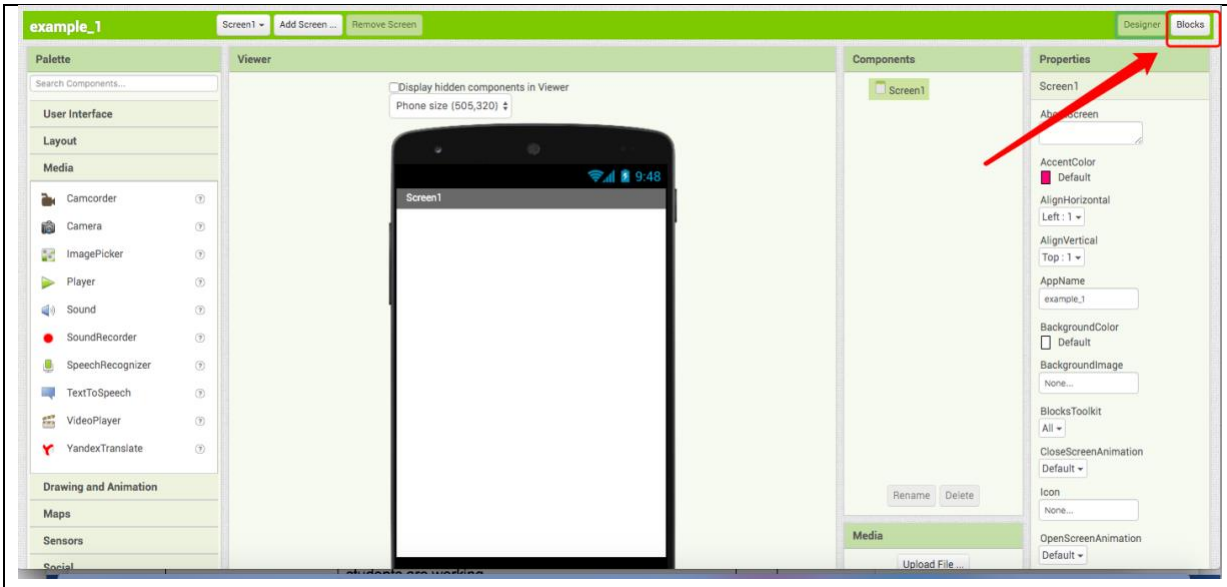
App Inventor is an online program that is free and ready to use. No installation is required, however you need a stable internet connection.

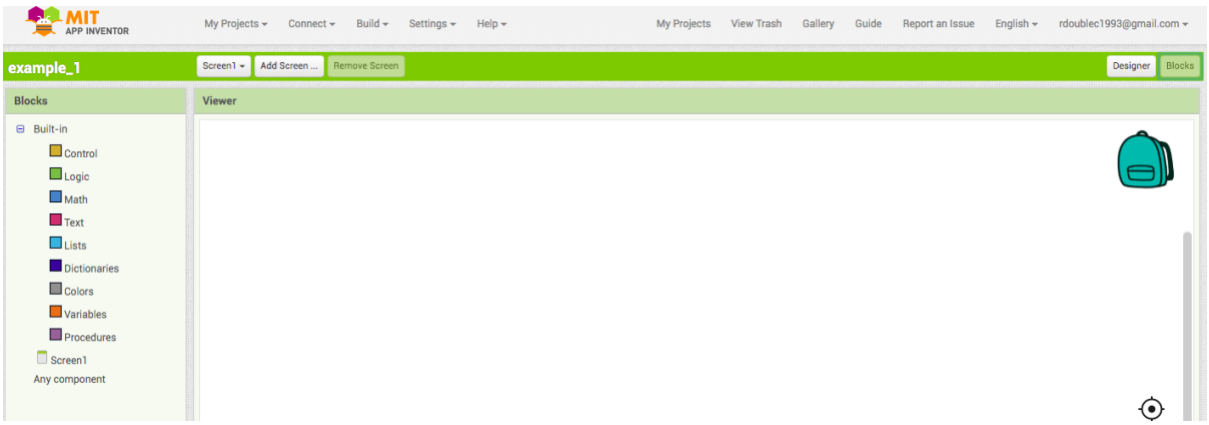
Please go to http://ai2.appinventor.mit.edu/ and log-in using a google account.

Instruction: “*Let’s have a look at the App Inventor! To get started with App Inventor, we need to create a new project”*



Instruction: *“Now we can see the Designer window. On the left side, you can see the Palette column which contains all the elements we can use in out app. On the middle we can see a smartphone where we can drag and drop the elements we need from the Palette column. Lastly, on the right side we can see the columns Components and Properties where we will see the components that we add to our app and the settings or properties of each of them.*
Now, let’s go to the Blocks window where we will be coding the components of our app. For that, look on the top right of the window and click/tap the Blocks button.”


example_1
Screen1
Add Screen ...
Remove Screen
Designer
Blocks
Palette
Search Components...
User Interface
Layout
Media
Camcorder
Camera
ImagePicker
Player
Sound
SoundRecorder
SpeechRecognizer
TextToSpeech
VideoPlayer
YandexTranslate
Drawing and Animation
Maps
Sensors
Viewer
Display hidden components in Viewer
Phone size (505,320)
9:48
Screen1
Components
Screen1
Rename
Delete
Media
Upload File ...
Properties
Screen1
AccentColor
Default
AlignHorizontal
Left : 1
AlignVertical
Top : 1
AppName
example_1
BackgroundColor
Default
BackgroundImage
None...
BlocksToolkit
All
CloseScreenAnimation
Default
Icon
None...
OpenScreenAnimation
Default


MIT APP INVENTOR
My Projects
Connect
Build
Settings
Help
My Projects
View Trash
Gallery
Guide
Report an Issue
English
rdoublec1993@gmail.com
example_1
Screen1
Add Screen ...
Remove Screen
Designer
Blocks
Blocks
Built-in
Control
Logic
Math
Text
Lists
Dictionaries
Colors
Variables
Procedures
Screen1
Any component
Viewer

Teacher's Attachment: B

Used in activity:	1.3 My first app 1.4 Trying out my first app 1.5 End of session: recap of the tasks, inputs and goals
Along with Student's Attachment(s):	1

Activity 1.3

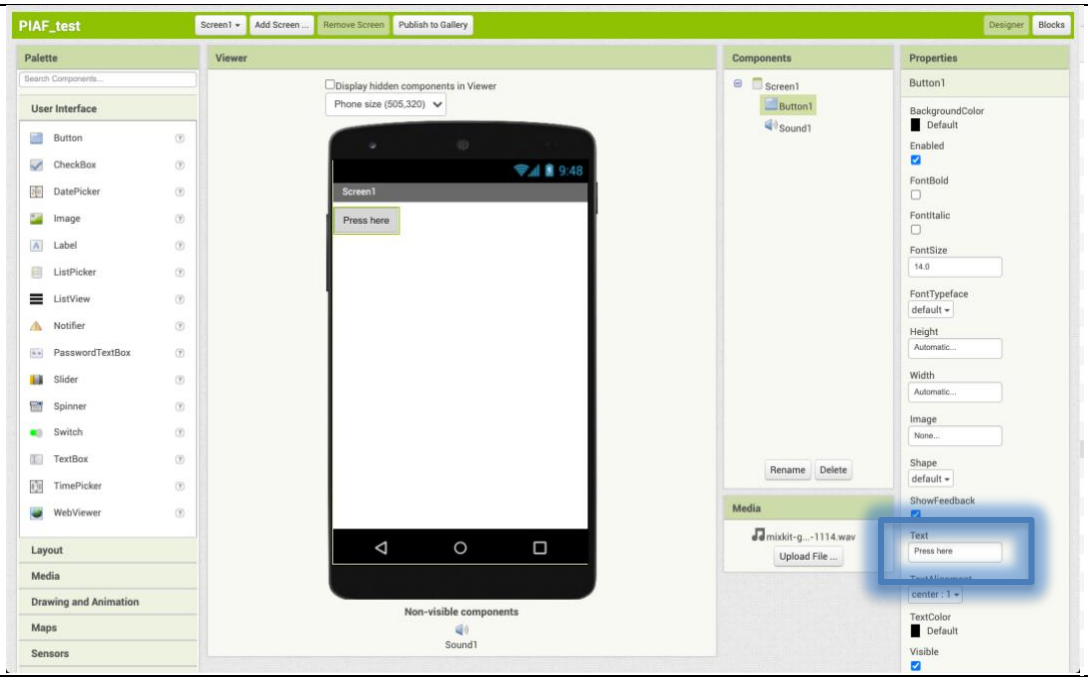
Before the session:

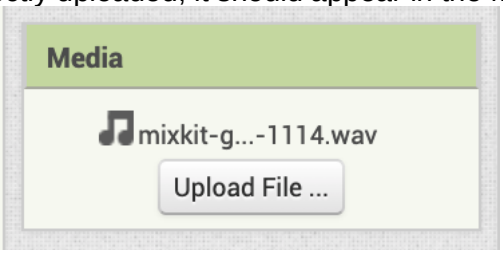
Ensure to already have the sound file (mixkit-game-click-1114.wav) pre-loaded in the computer/tablet and located in an easy to find folder such as the "Downloads" or "Desktop" folder.

Note: screenshots are originally meant only for the teacher for guidance purposes but can be shared with the students in case they need additional support.

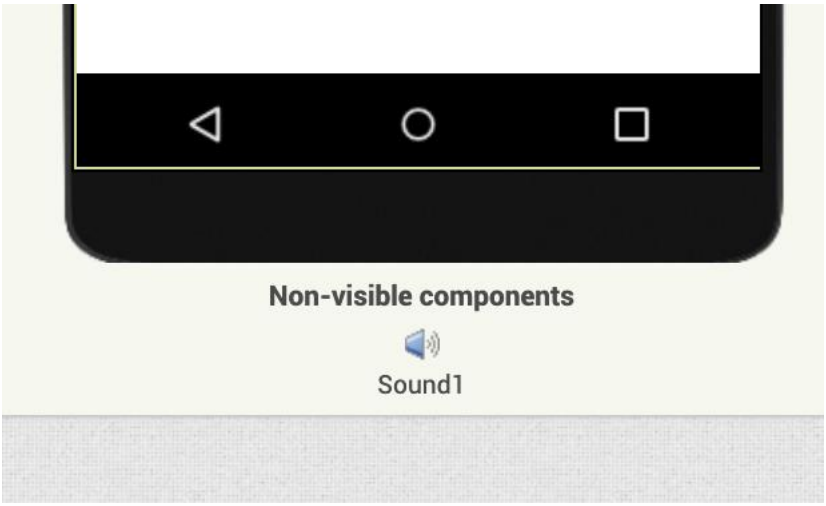
Instructions: Below you will find the three tasks required for you to create your first app. Do the tasks in order and carefully follow the steps. If you're stuck with something or something is unclear, raise your hand and ask the teacher.

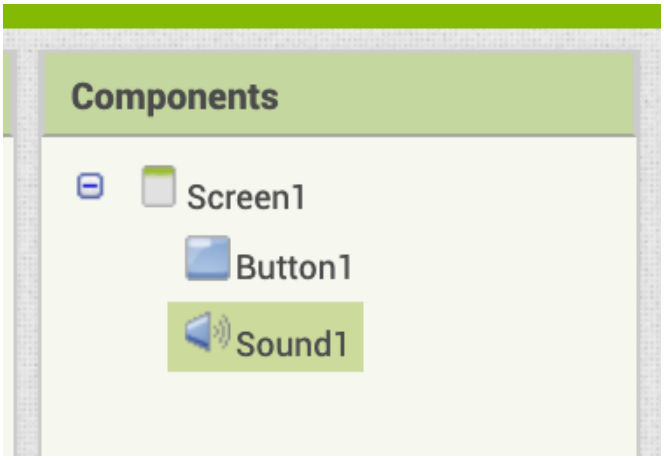
Task 1: Add a button and rename it to "press here". Change the color and size.
Step 1: In the Designer window, go to the Palette column and select the User Interface category. Step 2: In the User Interface category locate the Button component. Step 3: Select the Button component, hold it and drag it to the phone screen in the middle of the window. Step 4: In the Components column, click on the Button we just added. It should appear there named as "Button 1" or similar. Step 5: Now go to the Properties column and almost at the bottom, you will find the section "Text" along with a textbox. That's where you can change what is written in the button, Step 6: Change the text inside the button so it says "Press here"



Task 2: Add a sound to your app and a component for it to be played.
Step 1: Once again in the Designer window, now go to the Media column which is on the right and under the Components column. Step 2: In the Media column, select “Upload File” to upload the sound file “mixkit-game-click-1114.wav” from your desktop.
When the sound file is correctly uploaded, it should appear in the Media column: 
Step 3: Now we need to add to our app the component required to play the sound in our app. For that, go to the Palette column and select the Media category. Step 4: In the Media category you will see the “Sound” component that we need to add to our app. Select and hold the “Sound” component and drag it to the screen of the phone at the middle of the window.
The sound component is a non-visible component which may be hard for students to check if they added it correctly. As shown in the images below, there are two places where this

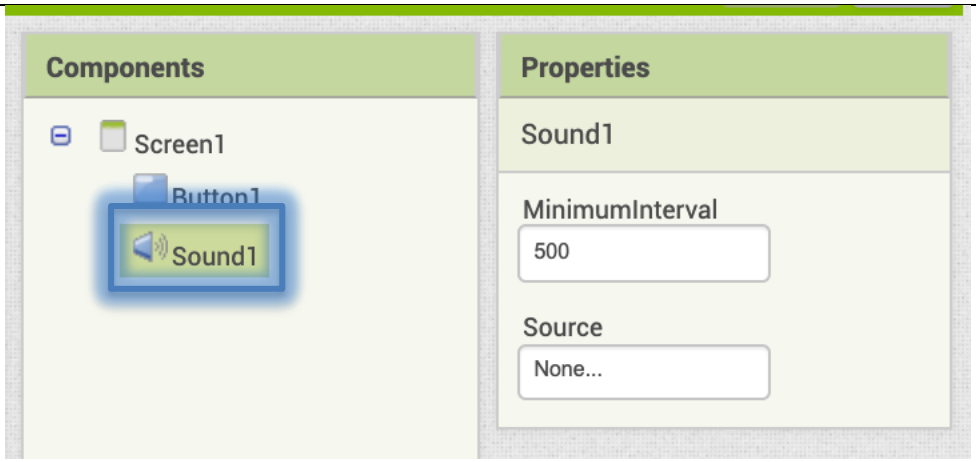
component appears where correctly added, (1) under the phone in the “Non-visible components” section and (2) listed on the Components column:



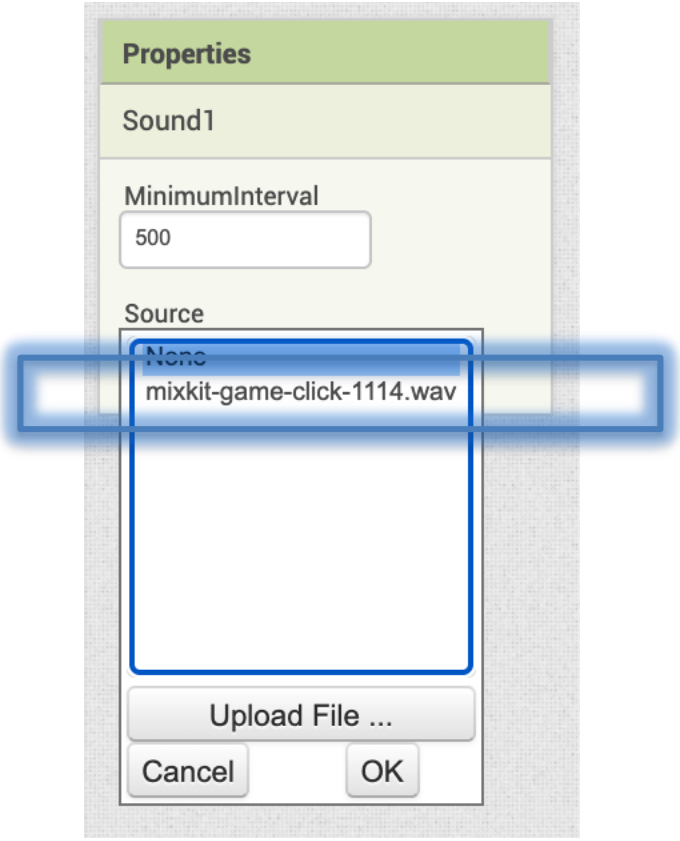


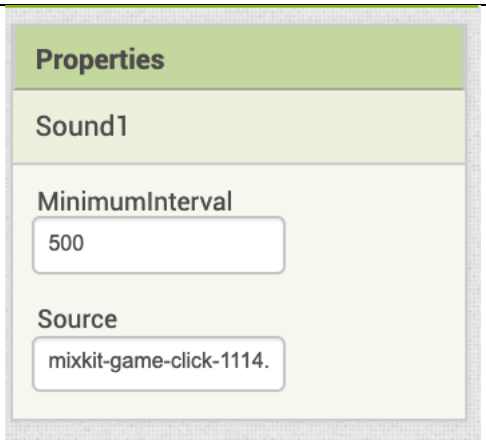
Step 5: Lastly, we need to link the sound file we uploaded with the sound component. For that, go to the Components column, click on the Sound element (it should be named “Sound1” or similar), and then look at the Properties column on the right. Look for the section “Source” and click the white box to select a source sound. There you should be able to see the filename of the sound we uploaded. Select it and then select on “OK”

To be able to select the sound source for the sound component, it’s important that the properties of the Sound1 component are shown. For that, ensure the Sound component is selected under the Components column:



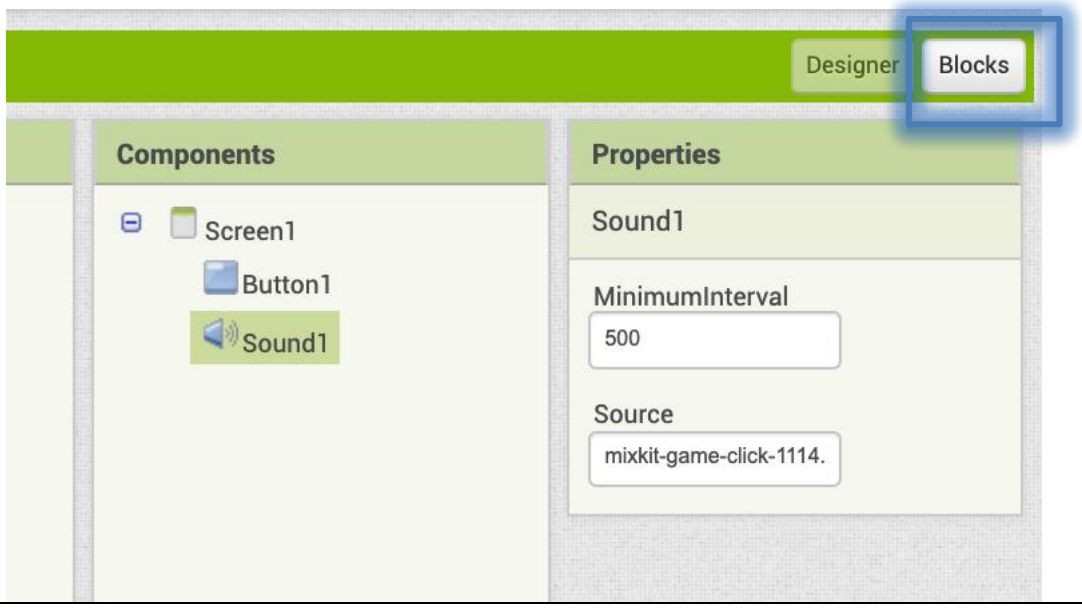
This is how it looks like after selecting the “None…” white box and after selecting the sound file we uploaded:





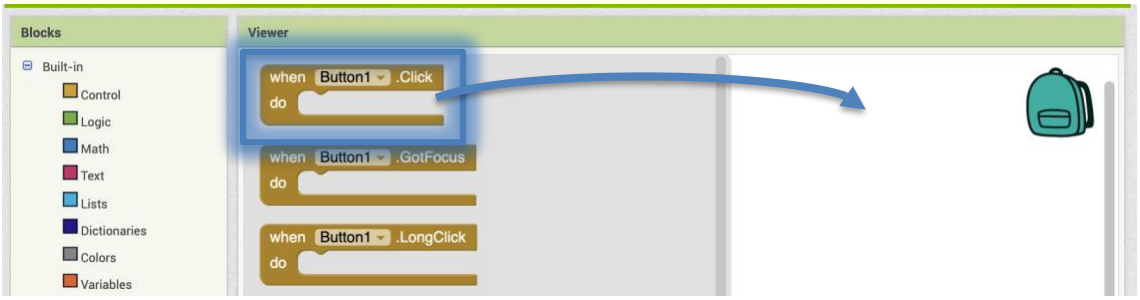
Task 3: Coding the app. We now have all the components we need for our first app. However, we need to code the instructions that our app will have for telling the phone when we want our sound to be played and which sound needs to be played

Step 1: In the main window on the right upper side, click on the “Blocks” button to see the blocks window



Step 2: This is the Blocks window. Here is where you will code your first app. There are two main sections in the Blocks window: Blocks and Viewer. On the Blocks section you will find all the blocks we can use for coding a phone app. You need to pick a type (e.g. Built-in) and then a subtype (e.g. Control), and then a side panel will appear with all the blocks from that sub-type. For using a block, you need to select and drag it to the Viewer section. You can use as many blocks for your app as needed, however they need to be correctly coded for them to work.

Step 3: Go to the Blocks section, then Screen1 type and select the Button1 subtype. From there, select the “when Button1.Click” yellow block and drag it to the Viewer area

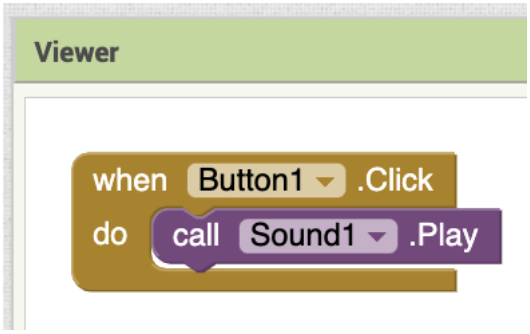


IMPORTANT: component blocks such as the Button and Sound blocks only appear **AFTER** adding such component to the phone/ app. If such blocks don’t appear, repeat tasks 1 and 2.

Step 4: Now, click the Sound1 subtype, locate the “call Sound1 .Play” block and drag it to the Viewer. Once in the viewer, select again the “call Sound1 .Play” block and move it inside the yellow “when Button1.Click” block. You will hear a clicking sound when the to blocks are successfully connected.

Your first app is ready!

These are the two blocks that should be added and how they need to be connected



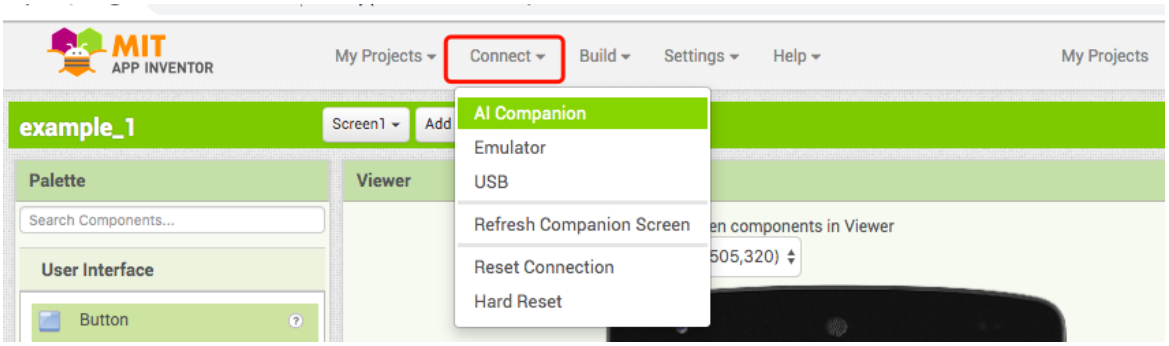
Activity 1.4

Preparation for this activity: Ensure the Android phones that the groups will use:

1. are fully charged (if possible, have them connected to an electric plug),
2. are connected to the internet,
3. have installed the Companion app. The app is available on the Google Play app and is named “MIT AI2 Companion”.

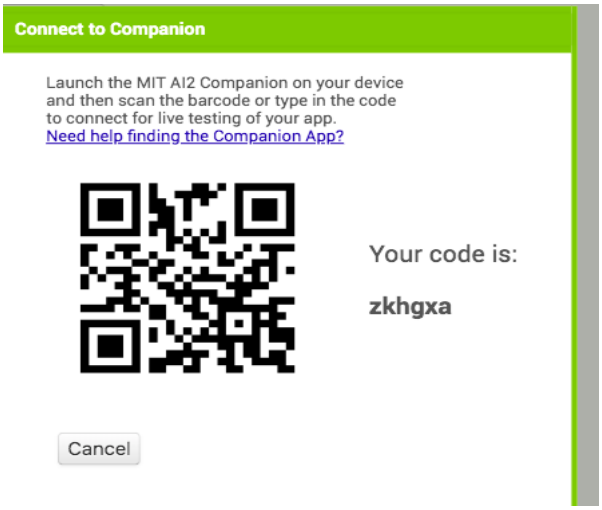
4. Similarly, have ready your computer/tablet with App Inventor and your Android phone so you can project the screen from your computer/tablet showing how to pair App Inventor with the phone.
5. **Advice:** there are 3 ways to connect App Inventor with a phone. Due to stability issues, we recommend using the "AI Companion" option which will be described below.
6. **Advice:** once connected, the Companion App can disconnect when the screen locks. Therefore, if possible, it is recommendable to set the screens of the phones to not auto-lock.

Instruction: "Good job! Now that our app is ready, we can try it out on our phones. For doing that, we need to connect our Android phone with App Inventor. We will do this altogether with me showing how to do it".
"In the main window of App Inventor, look for the Connect option in the gray top bar. Click the option and select "AI Companion"".



“After selecting “AI Companion”, a pop-up window appears with a QR code and a letter code. Keep that window open”

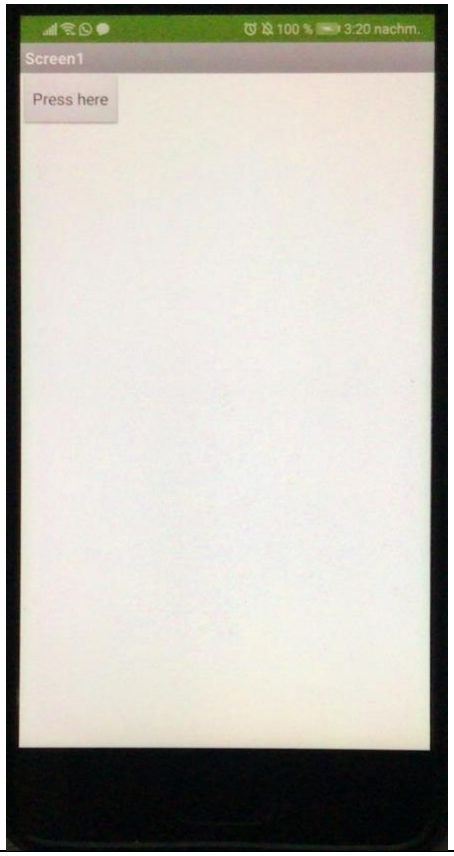
Please note that the QR and letter codes are different every time.



“With the window open, take your Android phone and open the “MIT AI2 Companion” app. Select the “scan QR code” button which will open the phone camera and use that to scan the QR code in your computer/tablet. The code should be detected automatically. It may take a couple of seconds for the app to load.”

“Now you should be able to see the button that you added on App Inventor in your phone. Press the button and hear what it does!”

Photo of how the app should look once App Inventor and the phone (via the AI Companion) are successfully connected:



Activity 1.5

"Let's end this session by describing what we did today. How many tasks did we do? What files, inputs, or things did we need for those tasks? Why were each of those tasks important?"

Task	Inputs used	Why important?
1	main input was adding the button component	because without button to press, the phone would never know that we want to play a sound and the sound would never be played
2	-Sound (non-visible) component -Sound file (.wav) for the "Press here" button	w/o the sound component the app can't play any sound, w/o the sound file theres no sound to play even if the sound component is added
3	"when .Click" and "call .Play" blocks	without the blocks, the app cannot tell the phone when we want the sound to be played

Teacher's Attachment: C

Used in activity:	2.2: Building the Piano
Along with Student's Attachment(s):	2

Before this activity:

- Ensure all mp3 files from the folder "PIAF piano notes" are available on the computers / tablets of each group
- Create in advance a new project for the activity

Creating a piano

Instruction: "*Who plays piano? How does piano works? How could we make a piano app?"*

Step 1: let's observe this piano below. Who could tell me which tone is middle C? Where to start a scale?

Step 2: So how could we make a piano app? What should happen if I clicked the button? Expected answers: The piano app should look like the real piano. And I click the key of piano, it will play the proper tones.

Build a simple piano using App Inventor

Instructions: Using the same method that we did last time for creating an app with a button that plays a sound, we will create this time a piano app for our phone.

Below, you can see the photo of a piano. We only need 1 set of tones/notes/keys so for this time we will only try to replicate the ones inside the blue square.

Below you can find a table with the 9 main tasks that you will need to do. Write down the inputs (e.g. files, components, blocks) that you will need for completing each task.
Once you fill the table, complete the tasks in the shown order.

In case of questions, raise your hand to let know the teacher.
Once you have completed the tasks, inform the teacher.

Order	Task	Inputs needed
1	Have the required files in your computer/tablet	-Piano key sounds
2	Add a HorizontalArrangement component and set its properties to: Height: 40% Width: 100%	-HorizontalArrangement component
3	Inside the HorizontalArrangement component, add one button component for each piano key and change the shown text of each button to show a note (Properties > Text)	-Seven Button components -Change Text property
4	Inside the HorizontalArrangement component, organize the buttons so they appear on the following order: C, D, E, F, G, A, B	
5	Set the properties of each button to: BackgroundColor: Cyan Height: Fill parent… Width: Fill parent…	-adjusting the properties of each button
6	Add one Sound component for each button	-Seven Sound components
7	Upload and link each note file to each sound component	-Seven note files -adjusting the Source property of each sound component (7 times total)
8	Go to the Blocks view, to create the code for each button: When (button) is clicked, play (note sound)	-14 blocks in total: 7 “when.Click” blocks and 7 “call.Play” blocks
9	Connect App Inventor with your phone and try your app	Do the “connect with AI Companion” procedure

Possible problems:

- All note files are on mp3 format and can be uploaded and played on App Inventor without any problem

- If students can’t find a certain component, remind them of the Palette column and to open all categories inside it
- If students can’t find a certain block, check first if they have added the component. All added components appear listed in the Components column of the Designer view
- If students can’t find the properties of a component, ensure they have it selected. This can be checked in the “Components” column and selecting the desired component
- Import the project “PIAF_piano_simple.aia” to your own App Inventor account to see how the app should look like. All components have been named accordingly

Teacher's Attachment: D

Used in activity:	3.2: Building the Guitar 3.3. Perform
Along with Student's Attachment(s):	3

Before this activity:

- Ensure all mp3 files from the folder "PIAF guitar notes" are available on the computers / tablets of each group.
- Create in advance a new project for the activity

Instructions:

Below you can find a table with the 9 main tasks that you will need to do. Write down the inputs (e.g. files, components, blocks) that you will need for completing each task.
Once you fill the table, complete the tasks in the shown order.

In case of questions, raise your hand to let know the teacher.
Once you have completed the tasks, inform the teacher.

Order	Task	Inputs needed
1	Have the required files in your computer/tablet	-Guitar key sounds
2	Add a VerticalArrangement component and set its properties to: Height: Fill parent… Width: 80%	-VerticalArrangement component
3	Inside the VerticalArrangement component, add one button component for each guitar key and change the shown text of each button to show a note (Properties > Text)	-Eight Button components -Change Text property
4	Inside the VerticalArrangement component, organize the buttons so they appear on the following vertical order. From top to bottom:a, C, D, E, F, G, A, B	
5	Set the properties of each button to: BackgroundColor: White Height: Fill parent… Width: 60%	-adjusting the properties of each button
6	Add one Sound component for each button	-Eight Sound components

7	Upload and link each note file to each sound component	-Eight note files -adjusting the Source property of each sound component (8 times total)
8	Go to the Blocks view, to create the code for each button: When (button) is clicked, play (note sound)	-16 blocks in total: 8 “when.Click” blocks and 8 “call.Play” blocks
9	Connect App Inventor with your phone and try your app	Do the “connect with AI Companion” procedure

Possible problems:

- All note files are on mp3 format and can be uploaded and played on App Inventor without any problem
- If students can’t find a certain component, remind them of the Palette column and to open all categories inside it
- If students can’t find a certain block, check first if they have added the component. All added components appear listed in the Components column of the Designer view
- If students can’t find the properties of a component, ensure they have it selected. This can be checked in the “Components” column and selecting the desired component
- Import the project “PIAF_guitar.aia” to your own App Inventor account to see how the app should look like. All components have been named accordingly

Below is the music sheet for the song we will play altogether. If you’re already done, you can start practicing.

Bruder Jakob

C D E C | C D E C

E F G | E F G

G A G F E C | G A G F E C

C G C | C G C

Student’s Attachments

Attachment: 1

Instructions: Below you will find the three tasks required for you to create your first app. Do the tasks in order and carefully follow the steps. If you're stuck with something or something is unclear, raise your hand and ask the teacher.

Task 1: Add a button and rename it to "press here". Change the color and size.
Step 1: In the Designer window, go to the Palette column and select the User Interface category. Step 2: In the User Interface category locate the Button component. Step 3: Select the Button component, hold it and drag it to the phone screen in the middle of the window. Step 4: In the Components column, click on the Button we just added. It should appear there named as "Button 1" or similar. Step 5: Now go to the Properties column and almost at the bottom, you will find the section "Text" along with a textbox. That's where you can change what is written in the button, Step 6: Change the text inside the button so it says "Press here"

Task 2: Add a sound to your app and a component for it to be played.
Step 1: Once again in the Designer window, now go to the Media column which is on the right and under the Components column. Step 2: In the Media column, select "Upload File" to upload the sound file "mixkit-game-click-1114.wav" from your desktop.

Step 3: Now we need to add to our app the component required to play the sound in our app. For that, go to the Palette column and select the Media category.

Step 4: In the Media category you will see the “Sound” component that we need to add to our app. Select and hold the “Sound” component and drag it to the screen of the phone at the middle of the window.

Step 5: Lastly, we need to link the sound file we uploaded with the sound component. For that, go to the Components column, click on the Sound element (it should be named “Sound1” or similar), and then look at the Properties column on the right. Look for the section “Source” and click the white box to select a source sound. There you should be able to see the filename of the sound we uploaded. Select it and then select on “OK”

Task 3: Coding the app. We now have all the components we need for our first app. However, we need to code the instructions that our app will have for telling the phone when we want our sound to be played and which sound needs to be played

Step 1: In the main window on the right upper side, click on the “Blocks” button to see the blocks window

Step 2: This is the Blocks window. Here is where you will code your first app. There are two main sections in the Blocks window: Blocks and Viewer. On the Blocks section you will find all the blocks we can use for coding a phone app. You need to pick a type (e.g. Built-in) and then a subtype (e.g. Control), and then a side panel will appear with all the blocks from that sub-type. For using a block, you need to select and drag it to the Viewer section. You can use as many blocks for your app as needed, however they need to be correctly coded for them to work.

Step 3: Go to the Blocks section, then Screen1 type and select the Button1 subtype. From there, select the “when Button1.Click” yellow block and drag it to the Viewer area.

Step 4: Now, click the Sound1 subtype, locate the “call Sound1 .Play” block and drag it to the Viewer. Once in the viewer, select again the “call Sound1 .Play” block and move it inside the yellow “when Button1.Click” block. You will hear a clicking sound when the to blocks are successfully connected.

Your first app is ready!

For teacher only (either at the end of this activity or on the next one or as recap)

QQ for later?: Verbally describe what they did for: (1) adding a button, (2) adding a sound, and (3) coding the app. Ask students to explain which elements they had to use and why was the purpose these 3 tasks. Ask if they could have done, for instance, coding the app first and then adding the sound later

Task	Inputs used	Why important?
1	main input was adding the button component	because without button to press, the phone would never know that we want to play a sound and the sound would never be played
2	-Sound (non-visible) component -Sound file (.wav) for the “Press here” button	w/o the sound component the app can’t play any sound, w/o the sound file theres no sound to play even if the sound component is added
3	“when .Click” and “call .Play” blocks	without the blocks, the app cannot tell the phone when we want the sound to be played

Attachment: 2

Build a simple piano using App Inventor

Instructions: Using the same method that we did last time for creating an app with a button that plays a sound, we will create this time a piano app for our phone.
Below, you can see the photo of a piano. We only need 1 set of tones/notes/keys so for this time we will only try to replicate the ones inside the blue square.

Below you can find a table with the 9 main tasks that you will need to do. Write down the inputs (e.g. files, components, blocks) that you will need for completing each task.
Once you fill the table, complete the tasks in the shown order.

In case of questions, raise your hand to let know the teacher.
Once you have completed the tasks, inform the teacher.

Order	Task	Inputs needed
1	Have the required files in your computer/tablet	
2	Add a HorizontalArrangement component and set its properties to: Height: 40% Width: 100%	
3	Inside the HorizontalArrangement component, add one button component for each piano key and change the shown text of each button to show a note (Properties > Text)	
4	Inside the HorizontalArrangement component, organize the buttons so they appear on the following order: C, D, E, F, G, A, B	
5	Set the properties of each button to: BackgroundColor: Cyan Height: Fill parent… Width: Fill parent…	
6	Add one Sound component for each button	

7	Upload and link each note file to each sound component	
8	Go to the Blocks view, to create the code for each button: When (button) is clicked, play (note sound)	
9	Connect App Inventor with your phone and try your app	

Attachment: 3

Instructions:

Below you can find a table with the 9 main tasks that you will need to do.
Write down the inputs (e.g. files, components, blocks) that you will need for completing each task.
Once you fill the table, complete the tasks in the shown order.

In case of questions, raise your hand to let know the teacher.
Once you have completed the tasks, inform the teacher.

Order	Task	Inputs needed
1	Have the required files in your computer/tablet	
2	Add a VerticalArrangement component and set its properties to: Height: Fill parent… Width: 80%	
3	Inside the VerticalArrangement component, add one button component for each guitar key and change the shown text of each button to show a note (Properties > Text)	
4	Inside the VerticalArrangement component, organize the buttons so they appear on the following vertical order. From top to bottom: a, C, D, E, F, G, A, B	
5	Set the properties of each button to: BackgroundColor: White Height: Fill parent… Width: 60%	

6	Add one Sound component for each button	
7	Upload and link each note file to each sound component	
8	Go to the Blocks view, to create the code for each button: When (button) is clicked, play (note sound)	
9	Connect App Inventor with your phone and try your app	

Below is the music sheet for the song we will play altogether. If you're already done, you can start practicing.

Bruder Jakob

C D E C | C D E C

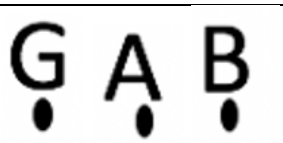
E F G | E F G

G A G F E C | G A G F E C

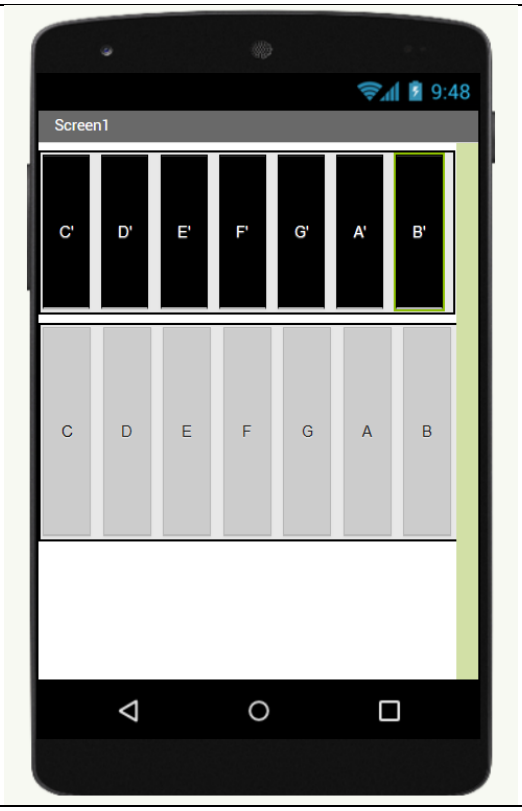
C G C | C G C

Evaluation

Task:
Now you have a new song to play with! For playing this song you will need to create a new piano app that has a second row of keys in a lower octave. With this, you will be able to play the following low notes (low notes are indicated with a mark under the note):



Using the provided sound files, create a piano app that looks like this:



You can use the tasks used for the first piano app but beware that you may need to repeat some steps to be able to have two rows of keys. Once you're done, inform the teacher.

Happy Birthday

G G A G C B

G G A G D C

G G G E C B A

F F E C D C

Evaluation (ANSWERS)

Answers:

To see how the app looks like, please open the App Inventor project “PIAF_piano_full.aia”