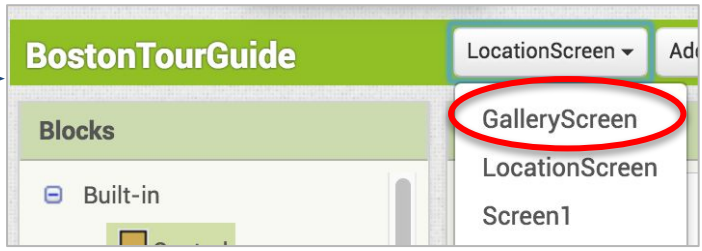
TOUR GUIDE: MULTIPLE PICTURES

You'll add the functionality for more than one picture in the Gallery!

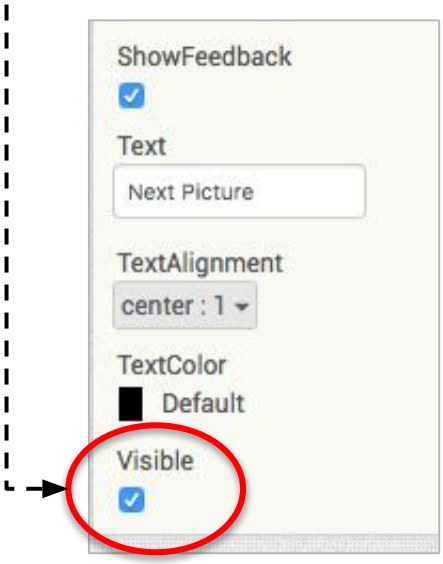
UPDATE GALLERY SCREEN

1. Go to the "**GalleryScreen**".

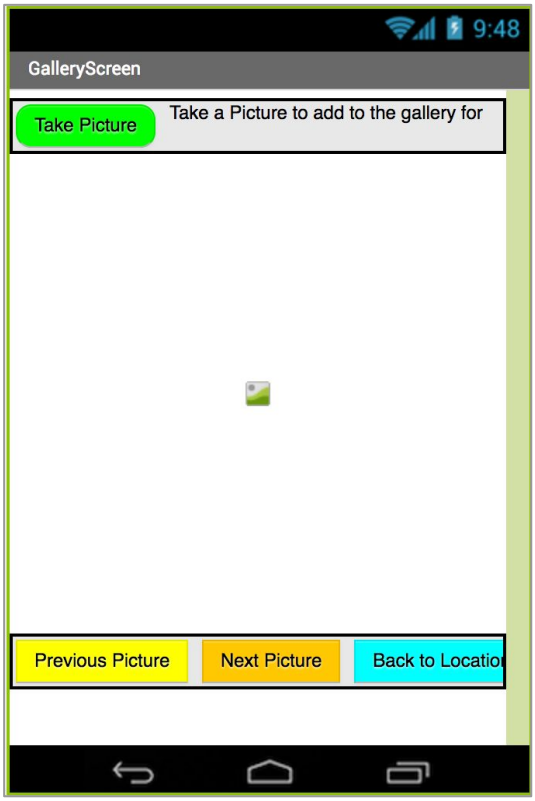


2. Switch to the Designer.

3. The template contains two invisible buttons, **PreviousButton** and **NextButton**. Find them in the Components panel and check their Visible property.



The two buttons appear next to the **BackButton**.



Copyright MIT 2021

AFTER PICTURE

Make your app more versatile, by allowing users to take many pictures instead of just one! You'll use a List called **photoList** to store the images in the app, and **TinyDB** to store the list so it can be retrieved each time the user runs the app.

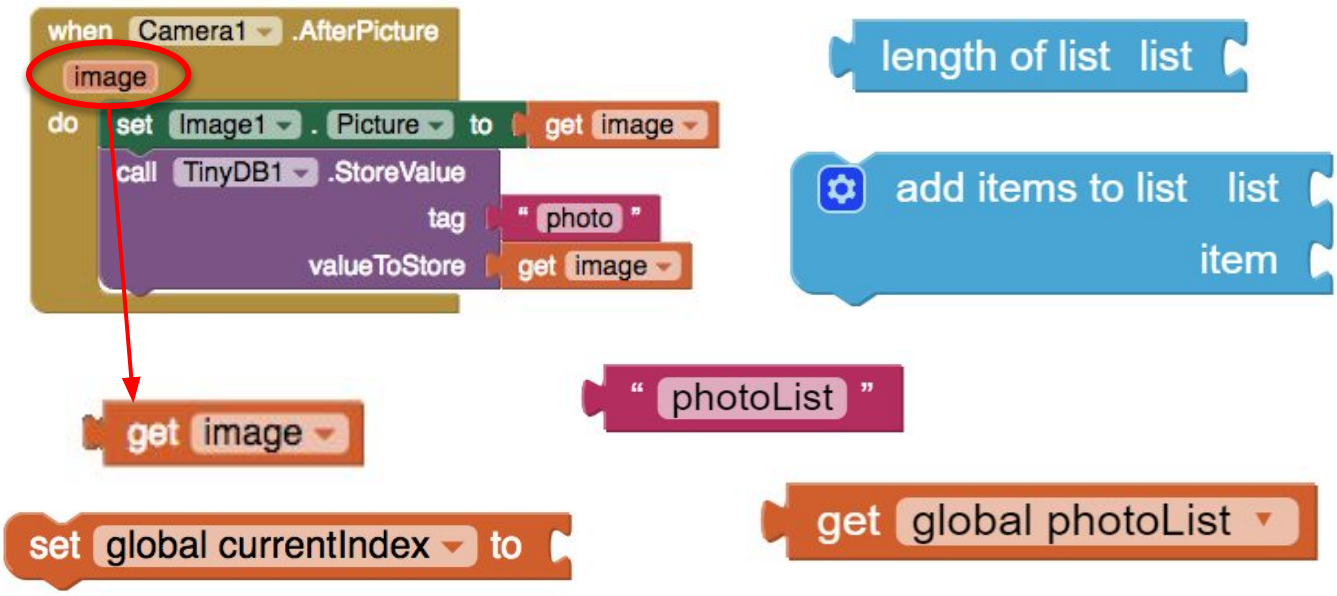
4 Initialize two new variables. Name them and set them as seen below.

- **currentIndex** points to the index of the current picture in **photoList**.
- **photoList** is the list of photos.

5 When a picture is taken:

- Add the **image** to the **photoList**.
- Save **photolist** to **TinyDB** with the tag "photolist". *(replace the current TinyDB1.StoreValue block with new tag/value)*
- Set **currentIndex** to point to the end of the list (HINT: use length of list block)

Use the following blocks.

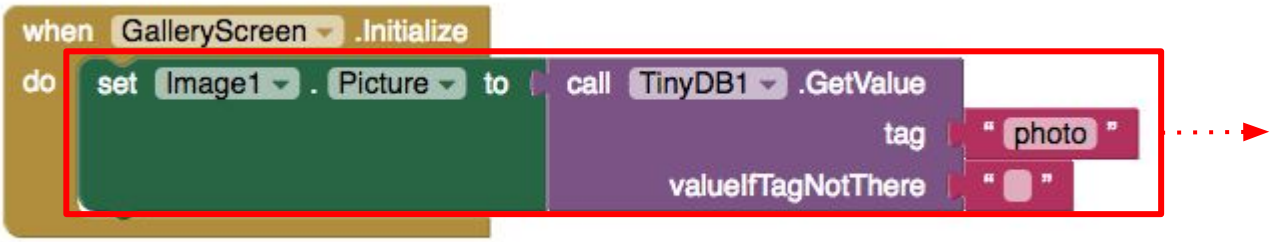


Copyright MIT 2021

INITIALIZE SCREEN

Because you are swtiching from a single photo to a list, you need to update the **GalleryScreen.Initialize** event.

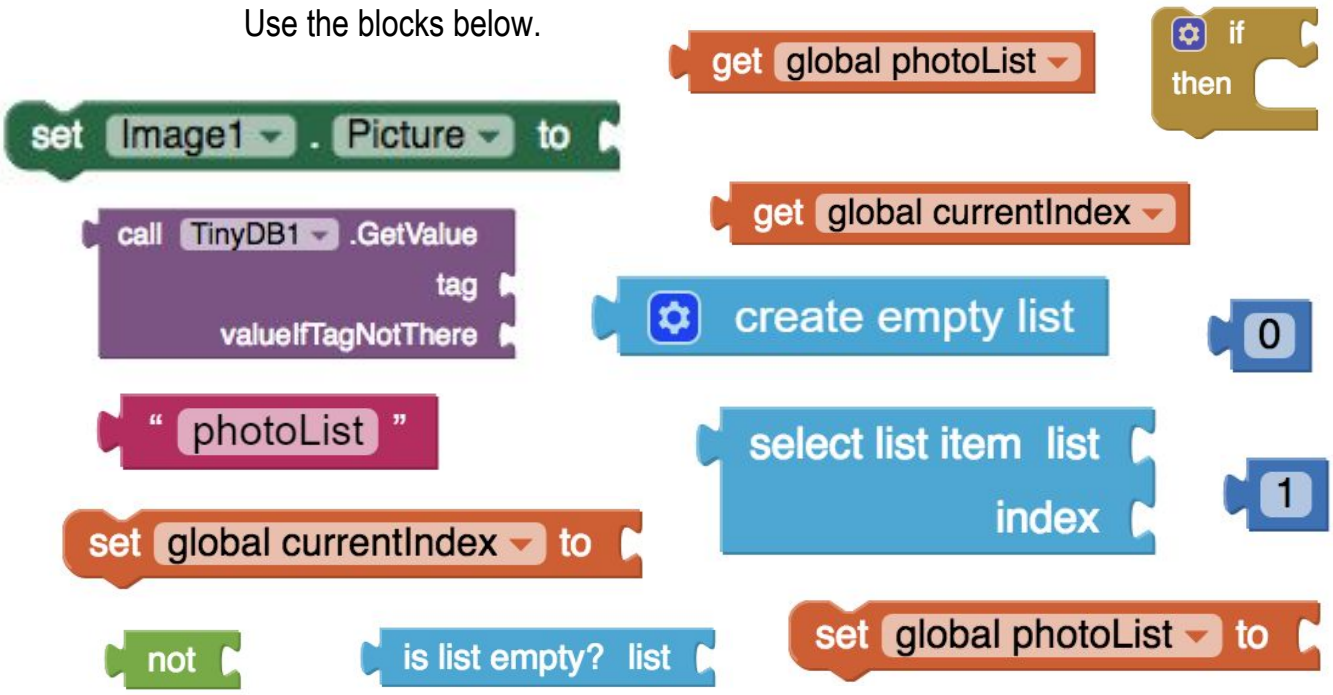
6 Remove the **set Image1.Picture** block, and set it aside. Don't delete it because you can reuse the blocks.



7 Here are the following steps needed in **GalleryScreen.Initialize**:

- Set variable **currentIndex** to 0.
- Set variable **photoList** to an empty list.
- Get the "photoList" tag from **TinyDB1** and store it in the **photoList** variable.
- If the list is not empty, there are pictures to see, so
  - set **Image1.PIcture** to the first item in the list
  - set **currentIndex** to 1, since it points to the first item in the list

Use the blocks below.



Copyright MIT 2021

PREVIOUS AND NEXT BUTTONS

Now code the **PreviousButton** and **NextButtons** to allow the user to scroll through the images in the gallery. You will need to update **currentIndex**, by either adding or subtracting to go back or forward in our list of photos.

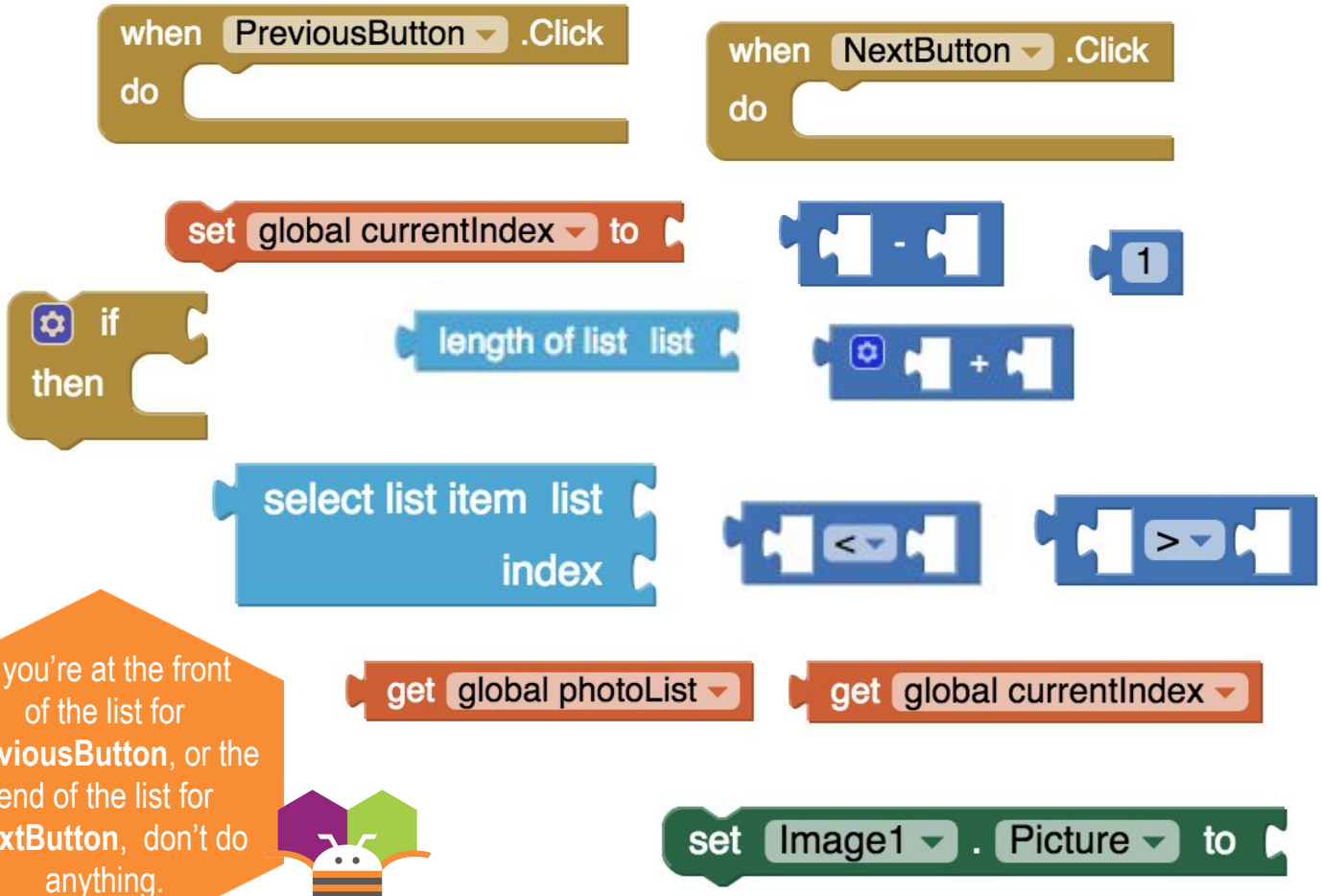
8 Use the blocks below to code the logic for **PreviousButton** and **NextButton**.

The algorithm for **PreviousButton** is:

- If currentIndex is not pointing to the beginning of the list
  - Subtract 1 from **currentIndex**.
  - Set **Image1.Picture** to the item pointed to by the new **currentIndex** value.

The algorithm for **NextButton** is:

- If **currentIndex** is not pointing to the end of the list
  - Add 1 to **currentIndex**.
  - Set **Image1.Picture** to the item pointed to by the new **currentIndex** value.



If you're at the front of the list for **PreviousButton**, or the end of the list for **NextButton**, don't do anything.

Copyright MIT 2021

BACKBUTTON AND TESTING

9 The **BackButton** works the same as it did in the **LocationScreen.**

10 Here you go! Test out your app with the MIT AI2 Companion.

- Go to the **GalleryScreen**.
- Take some pictures.
- Scroll through your pictures with the Previous and Next buttons.
- Close the app and reopen it. The pictures should still be there!

Copyright MIT 2021

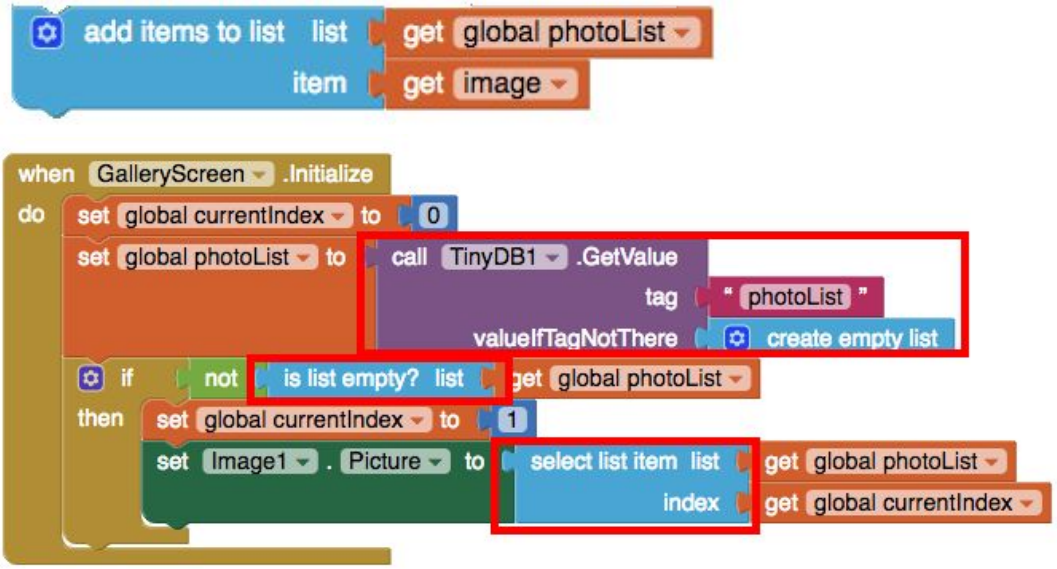
COMPUTATIONAL THINKING CONCEPTS

The following are the Computational Thinking Concepts used in Multiple Pictures.

Tour Guide

1. Naming/Variables

2. Manipulation of data and elementary data structures



3. Conditionals

4. Operators

Copyright MIT 2021