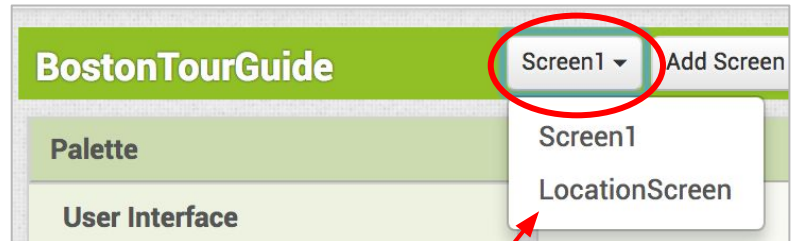


MALDEN TOUR GUIDE: LOCATION SCREEN

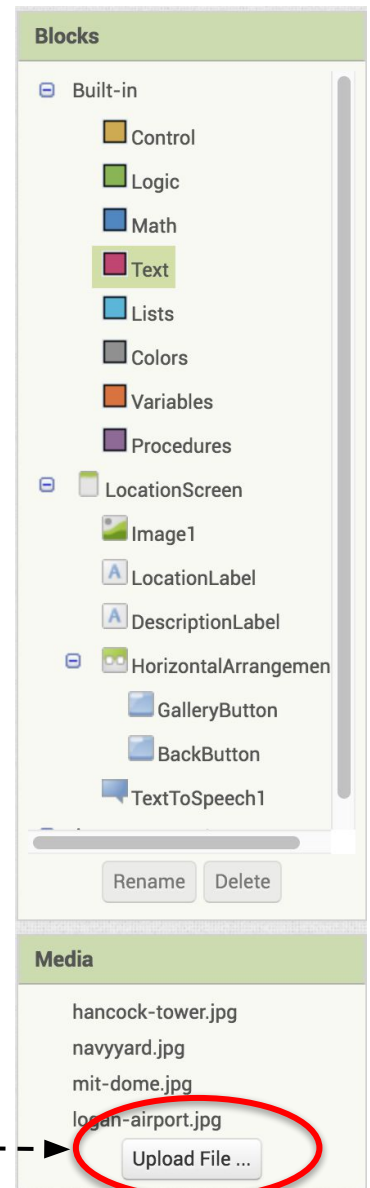
CODING THE LOCATION SCREEN

1 Switch back to the **LocationScreen**.



Let's start by uploading the images you've selected for your four landmarks.

2 Click on "**Upload File**" under Media,
and upload your four image files.



VARIABLES

We need to store all the information for our landmarks in our app.

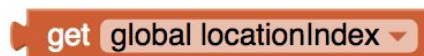
We can do that with variables. Variables are a way to store information by giving it a name, and referencing the name in our code. We can also change the value of a variable if we need to.

The Variables drawer holds the blocks used for variables.

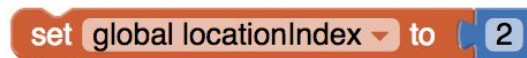
To use a variable, you must first initialize it.



You can get the value of a variable.

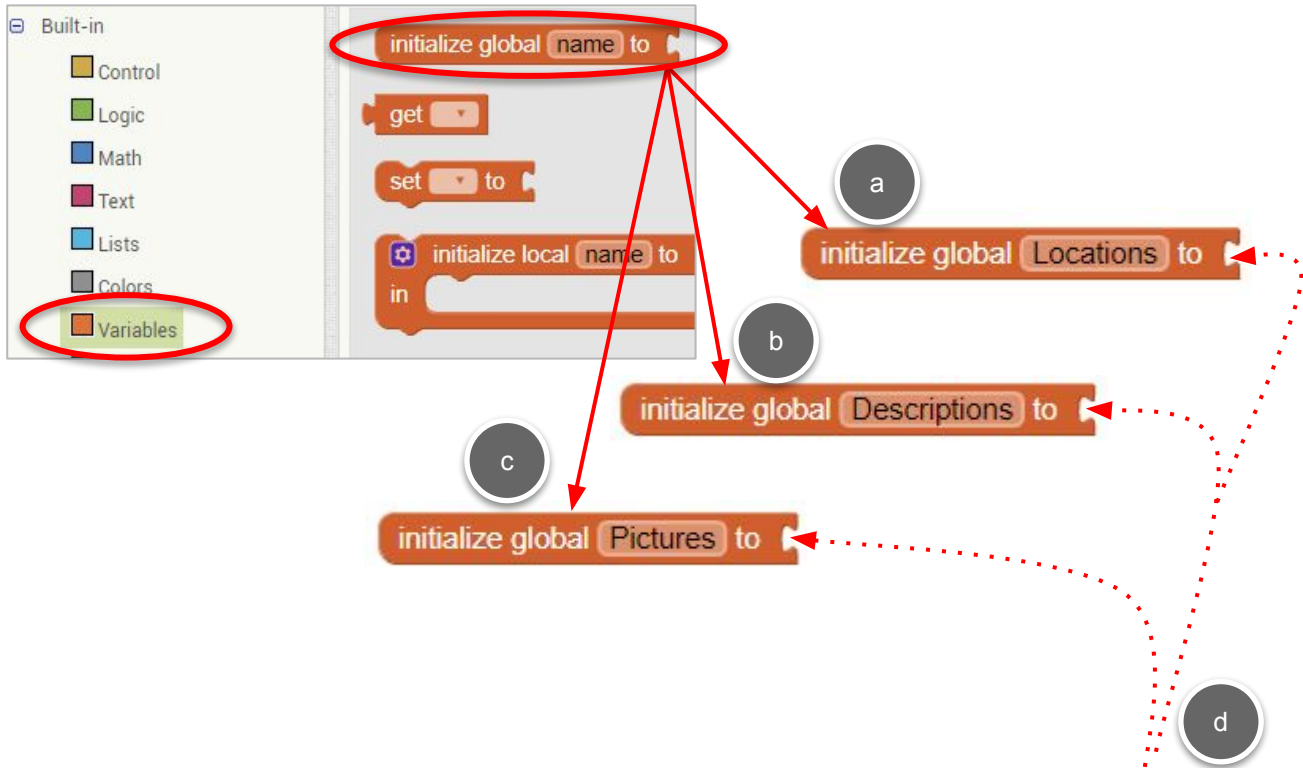


You can also set (change) the value of a variable within your app.

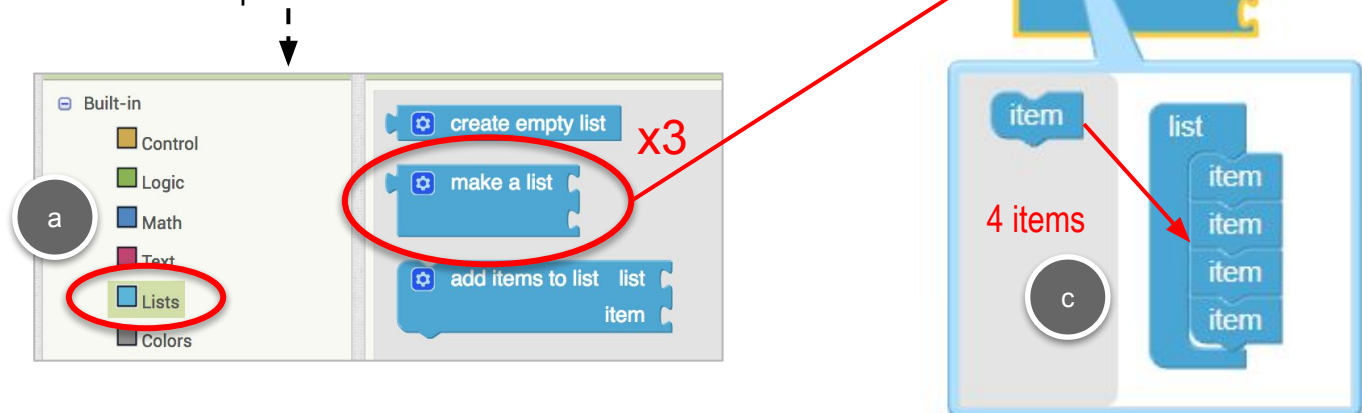


CODING THE LOCATION SCREEN

- 3 Make three new variables, and name them **Locations**, **Descriptions**, and **Pictures**.

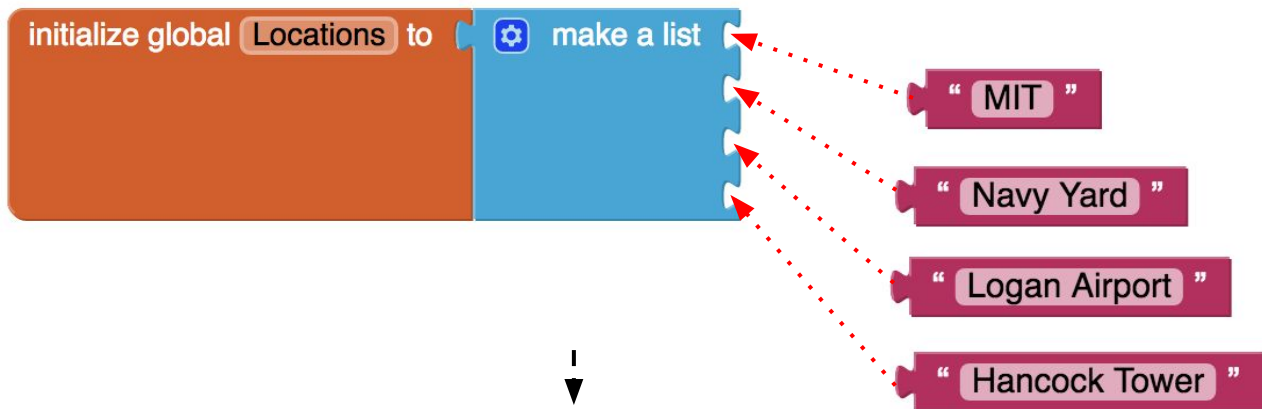


- 4 Drag out 3 **make a list** blocks from the Lists drawer. Add items so they are each 4 elements long, and snap them into the 3 variables.

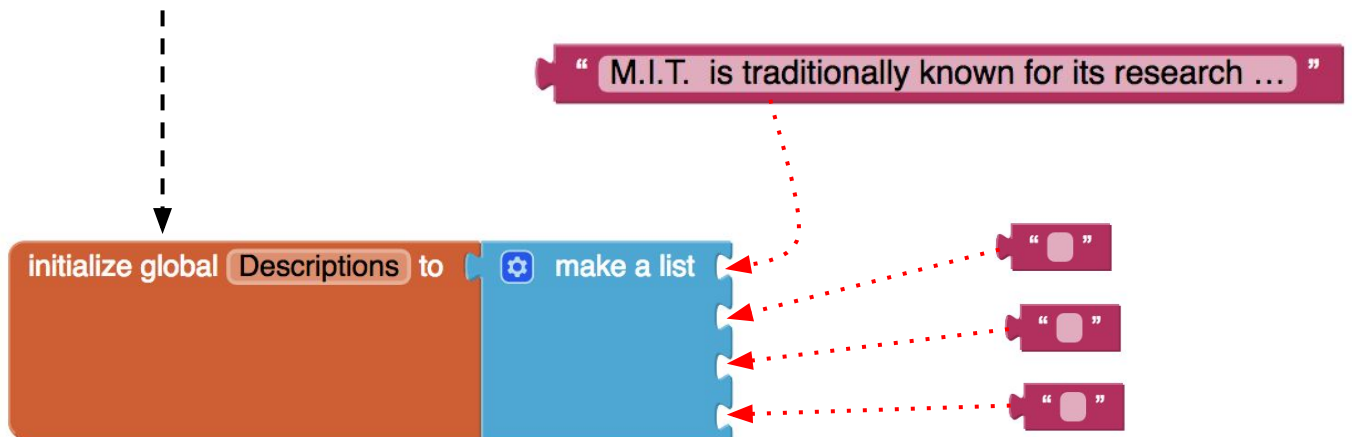


LET'S CONTINUE

- 5 To **Locations**, add 4 **text** blocks with the names of your four landmarks. These should match exactly the names you used as **startValue** in **Screen1**.

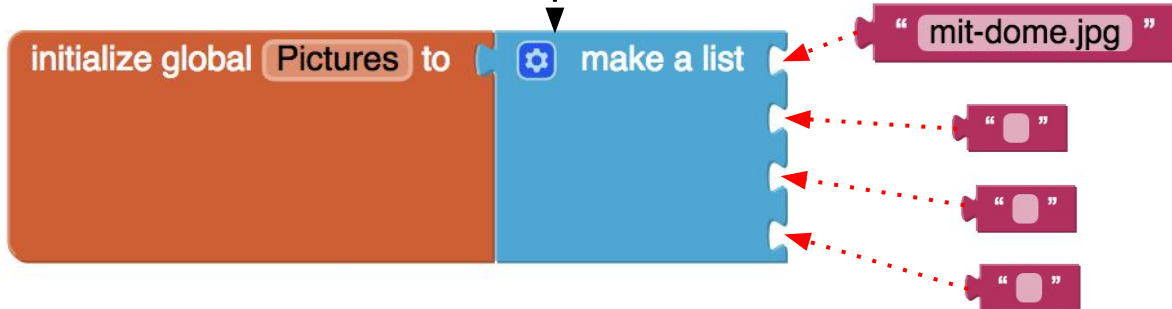


- 6 Add text descriptions (the longer descriptions from the Landmarks worksheet) for your four landmarks to the **Descriptions** list, making sure they are in the same order as the **Locations**!



LET'S CONTINUE

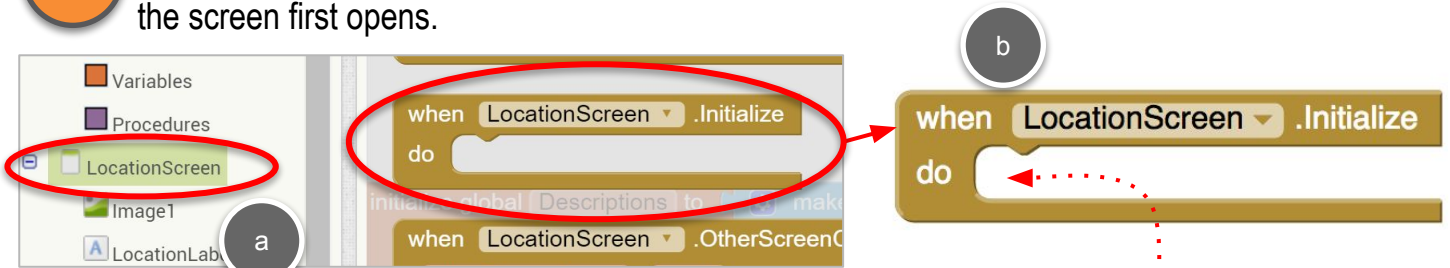
- 7 And to **Pictures**, add **text** blocks that contain the exact filenames for the uploaded pictures.



- 8 Let's make a new variable, **locationIndex**, to keep track of which Location we want to display. Initialize it to 0.



- 9 Set the value of **locationIndex** in **LocationScreen.Initialize** event, the event is triggered when the screen first opens.



LET'S CONTINUE

- 10 Set the value of **locationIndex** based on the **start value** passed from **Screen1**. Look for the location passed in start value in the **Locations** list, and set **locationIndex** to whatever that index is.

when **LocationScreen** .Initialize
do **set global locationIndex** to

index in list thing
list

index in list finds the thing (location) and tells us its index in the **Locations** list.

get start value

get global Locations

Control
Logic
Math
Text
Lists
Colors
Variables
Procedures

Control
Logic
Math
Text
Lists
Colors
Variables
Procedures

Control
Logic
Math
Text
Lists
Colors
Variables
Procedures

initialize global name to
get
set to
initialize local name to in

a

b

c

d

e

f

LET'S CONTINUE

Now we use **locationIndex** to point to the correct Location, Description, and Picture items in our three lists!

- 11 Start with the **set LocationLabel.Text** block, snapping it in below **set global locationIndex**.

select list item pulls out an item from a list, when you tell it the index.

d The list is **Locations**.

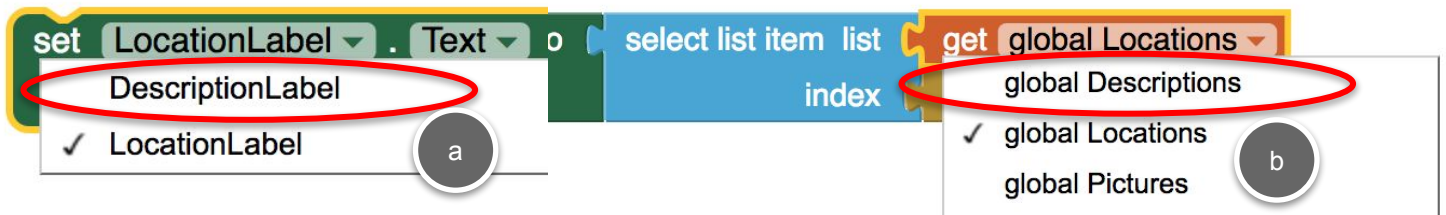
e and index is **locationIndex**.

Let's do the same for the **Description**. Let's make it easier by duplicating the block we just made.

12 Right-click on the **set LocationLabel.Text** block and **Duplicate** it.

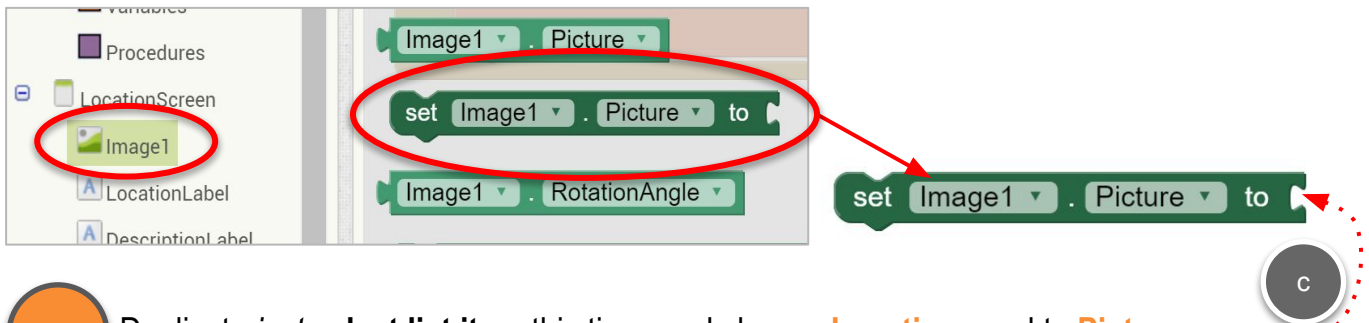
LET'S CONTINUE

- 13 Change **LocationLabel** to **DescriptionLabel**, and **Locations** to **Descriptions**.

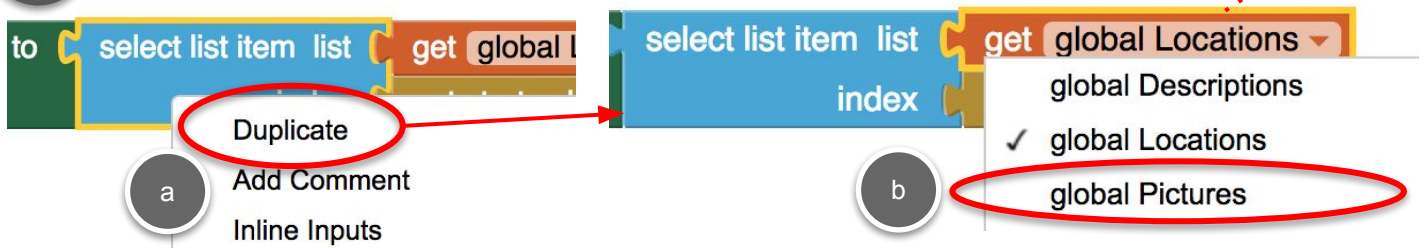


- 14 And snap it into **LocationScreen.Initialize** event, below **set LocationLabel.Text**.

- 15 Let's also set the **Image.Picture** to the correct Picture.
We can't Duplicate the entire block so drag out out **set Image1.Picture**.



- 16 Duplicate *just* **select list item** this time, and change **Locations** and to **Pictures**.



- 17 And snap the block into **LocationScreen.Initialize** event, below **set DescriptionLabel.Text**.

LET'S CONTINUE

18 Let's use the **TextToSpeech** component to have the app "speak" the description..

The screenshot shows the Scratch IDE with the 'LocationScreen' project. In the top-left pane, the 'TextToSpeech1' component is highlighted with a red circle (a). In the main workspace, a 'call TextToSpeech1 .Speak message' block is connected to the 'Text' property of the 'DescriptionLabel' component (b). The 'Text' property is also connected to the 'Country' variable (c). In the bottom-left pane, the 'DescriptionLabel' component is highlighted with a red circle (d). In the bottom-right pane, the 'Text' property of the 'DescriptionLabel' component is highlighted with a red circle (e). A red arrow points from the 'Text' property to the 'DescriptionLabel' component, and another red arrow points from the 'Text' property to the 'TextToSpeech1' component. A red dotted arrow points from the 'Text' property to the 'TextToSpeech1' component.

And snap it in to **LocationScreen.Initialize** event, below **set Image1.Picture**.

One more button and we're finished with this screen!

19 Complete the **when BackButton.Click** block.

The screenshot shows the Scratch IDE with the 'when BackButton.Click' block. A 'do' block is attached to it, and a red dotted arrow points from the 'do' block to the 'close screen' block in the next screenshot.

Screen1 is still open under this screen, so if we close this screen, the map will appear again.

The screenshot shows the Scratch IDE with the 'when BackButton.Click' block. A 'do' block is attached to it, and a 'close screen' block is being added to it. The 'close screen' block is highlighted with a red circle (a). In the bottom-left pane, the 'close screen' block is highlighted with a red circle (b). In the bottom-right pane, the 'close screen' block is highlighted with a red circle (c). A red arrow points from the 'close screen' block to the 'do' block, and another red arrow points from the 'close screen' block to the 'close screen' block in the bottom-right pane.

TESTING

20

Test your app with the MIT AI2 companion.

- Long click on each of your markers and see that the **LocationScreen** opens, displays the correct description and picture, and the correct description is read aloud.
- Try the **Back to Maps** button to make sure you can go back and forth between the first two screens.

