



Find It!

In this project you will make a mobile hide-and-seek game.

You will hide three objects around the room, and the player has to find them. Once they've found each item, they'll have to prove that they've found it by taking a photo of it.

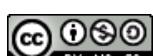
Your mobile app will be able to analyse the photo to tell which object they've found.

You'll use machine learning to train your app to be able to recognise photos of the objects that you'll hide.

The screenshot shows the MIT App Inventor interface with the 'Find It!' project open. The left side displays the blocks editor with various components like 'labelSpace1', 'button_Start', 'button_Photo', 'label_Status', 'Camera1', 'ML4K1', 'Clock1', and 'Notifier1'. The main workspace contains the following blocks:

- A 'when button_Photo .Click' event block with a 'do' loop containing a call to 'Camera1 .TakePicture'.
- A 'when Camera1 .AfterPicture' event block with a 'do' loop containing:
 - 'set label_Status .Text to [join [get confidence] [% confident that you found the] [get classification]]'
 - An 'if' block checking if 'get classification' equals 'Doggie'.
 - If true, 'set label_Doggie .BackgroundColor to [green]'.
 - If false, an 'else if' block checking if 'get classification' equals 'Monkey'.
 - If true, 'set label_Monkey .BackgroundColor to [green]'.
 - If false, an 'else if' block checking if 'get classification' equals 'Scissors'.
 - If true, 'set label_Scissors .BackgroundColor to [green]'.
 - If false, 'set label_Doggie .BackgroundColor to [red]'.
- A 'when ML4K1 .GotClassification' event block with a 'do' loop containing:
 - 'set [label_Doggie .Text] to [if [get classification] = [Doggie] then [1] else [0]]'
 - 'set [label_Monkey .Text] to [if [get classification] = [Monkey] then [1] else [0]]'
 - 'set [label_Scissors .Text] to [if [get classification] = [Scissors] then [1] else [0]]'
- A 'Show Warnings' block.
- A 'when 0 > 0' event block with a 'then' loop containing:
 - 'set [label_Doggie .BackgroundColor] to [and [label_Doggie .Text = 1] [not [label_Doggie .TextColor = red]]]'
 - 'set [label_Monkey .BackgroundColor] to [and [label_Monkey .Text = 1] [not [label_Monkey .TextColor = green]]]'
 - 'set [label_Scissors .BackgroundColor] to [and [label_Scissors .Text = 1] [not [label_Scissors .TextColor = green]]]'
 - 'set [Clock1 .TimerEnabled] to [false]'.

The right side of the interface shows a preview screen with three large green labels: '45 seconds', 'Doggie', 'Monkey', and 'Scissors'. Below these are two buttons: 'Restart' and 'Take photo .' (with a period). A status message at the bottom says '91% confident that you found the Scissors'. At the very bottom is a media control bar with icons for back, forward, and volume.



This project worksheet is licensed under a Creative Commons Attribution Non-Commercial Share-Alike License
<http://creativecommons.org/licenses/by-nc-sa/4.0/>

1. Go to <https://machinelearningforkids.co.uk/> in a web browser

2. Click on “**Get started**”

3. Click on “**Log In**” and type in your username and password

If you can't remember your username or password, ask your teacher or group leader to reset it for you.

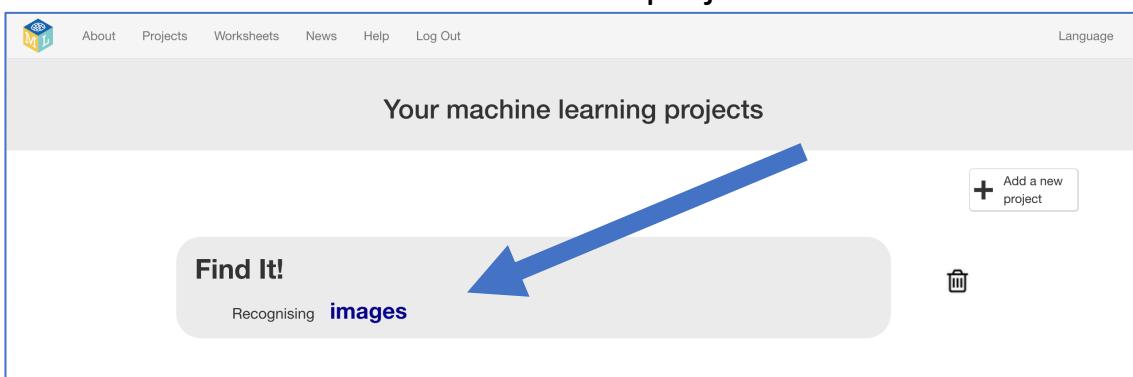
4. Click on “**Projects**” on the top menu bar

5. Click the “**+ Add a new project**” button.

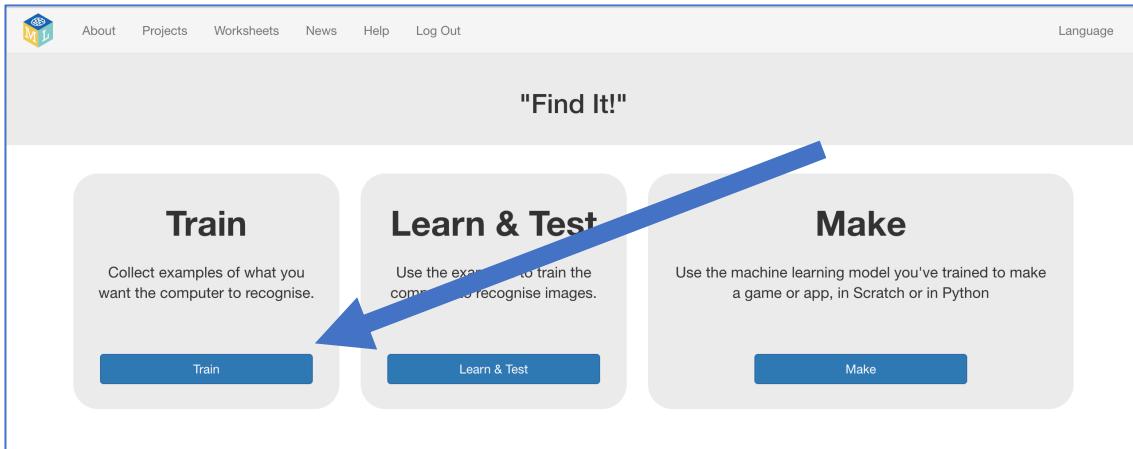
6. Name your project “Find It!” and set it to learn how to recognise “images”. Click **Create**

The screenshot shows a web page titled "Start a new machine learning project". At the top, there is a navigation bar with links for About, Projects, Worksheets, News, Help, Log Out, and Language. Below the title, the "Project Name" field contains "Find It!". Under the "Recognising" dropdown, "images" is selected. A small modal window is open, asking "What type of thing do you want to teach the computer to recognise?", with three options: "text", "images", and "numbers". The "images" option is highlighted. At the bottom right of the form are "CREATE" and "CANCEL" buttons.

7. You should see “Find It!” in the projects list. Click on it.

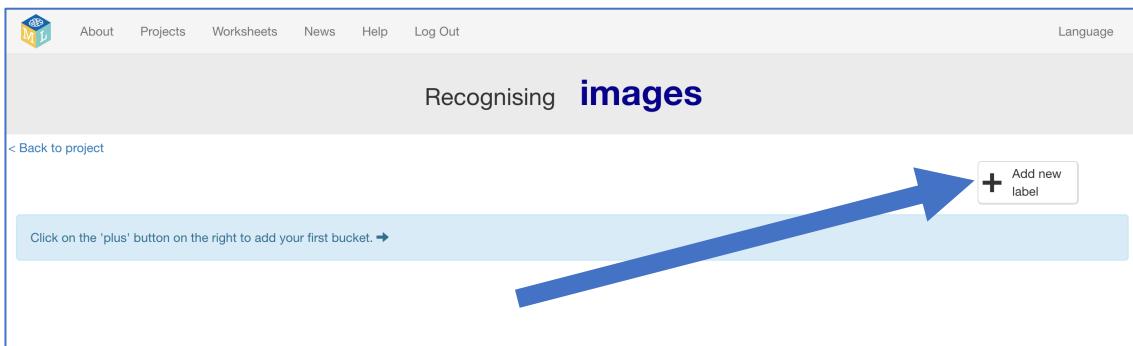


8. We'll start by getting a collecting training examples. Click "Train"

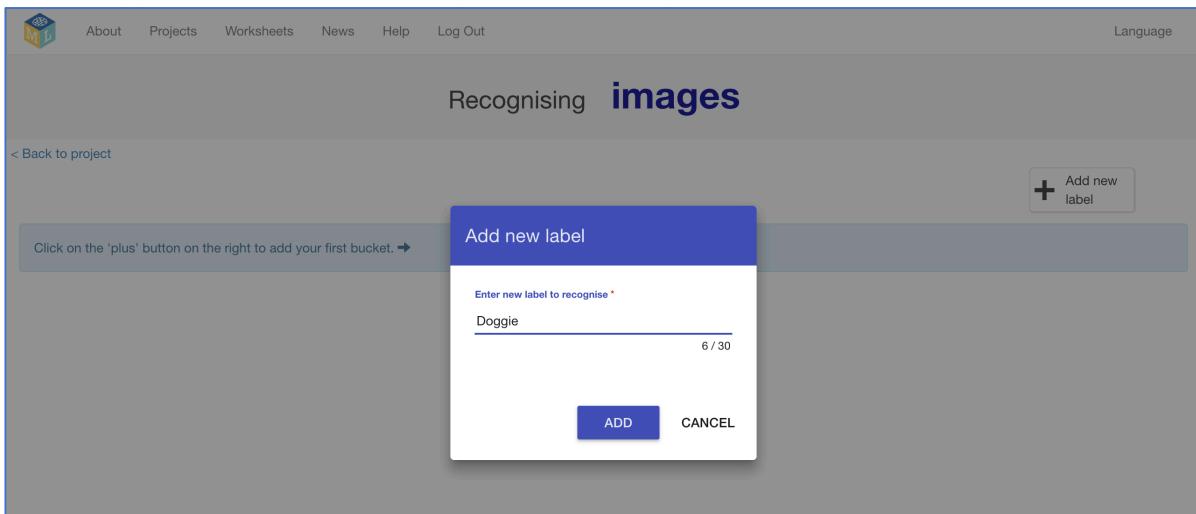


9. The game we're making is to challenge someone to find three objects that you will hide around the room. Choose your objects now. For the screenshots below, I'll be using a cuddly dog soft toy, a ceramic monkey, and a pair of scissors.

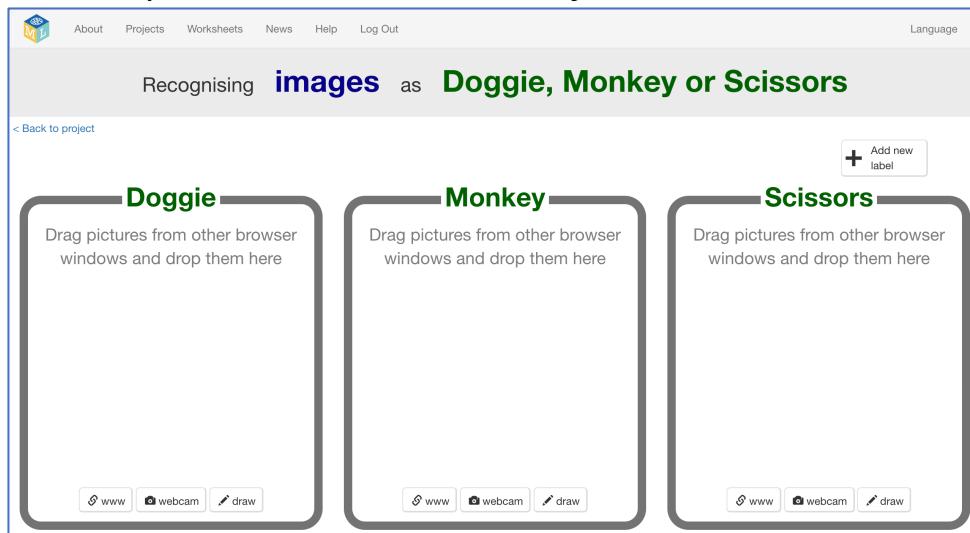
10. Click the "+ Add new label" button



11. Type in the name of your first object, and click "Add"

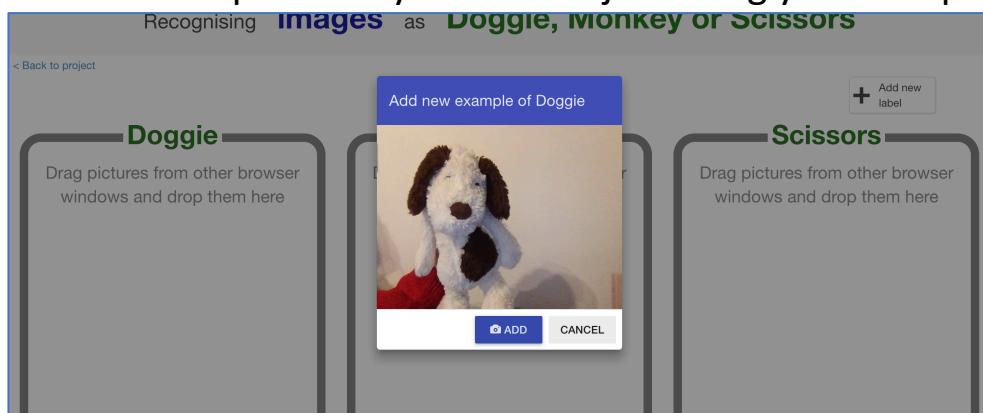


12. Repeat that for all three objects

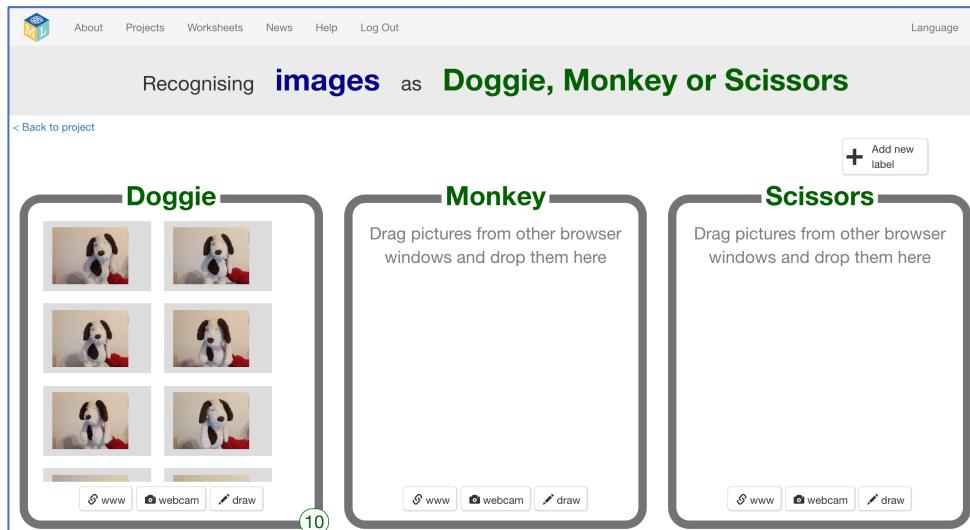


13. Click the “webcam” button in the first of your buckets

14. Take a photo of your first object using your computer webcam



15. Repeat until you've got at least ten photos of your first object



16. Repeat that for all three objects.

Use the webcam button at the bottom of each bucket to take a photo of that object. Take at least **ten** photos of **each** object.

Try to take a variety of photos of each object – from different sides and angles.

The screenshot shows a web-based application for training a machine learning model. At the top, there's a navigation bar with links for 'About', 'Projects', 'Worksheets', 'News', 'Help', 'Log Out', and 'Language'. Below the navigation is the title 'Recognising images as Doggie, Monkey or Scissors'. A 'Back to project' link is visible. On the right, there's a button to 'Add new label' with a plus sign. The main area contains three sections: 'Doggie', 'Monkey', and 'Scissors', each showing a grid of ten images. Each section has a '10' at the bottom right. Below each grid are buttons for 'www', 'webcam', and 'draw'. A large blue arrow points from the 'Learn & Test' section of the previous page to the 'Learn & Test' section here.

17. Click the “< Back to project” link

18. Next, we'll use your examples to train a machine learning model. Click “Learn & Test”

The screenshot shows the 'Find It!' project interface. At the top, there's a navigation bar with links for 'About', 'Projects', 'Worksheets', 'News', 'Help', 'Log Out', and 'Language'. Below the navigation is the title 'Find It!'. The interface is divided into three main sections: 'Train', 'Learn & Test', and 'Make'. A large blue arrow points from the 'Learn & Test' section of the previous page to the 'Learn & Test' section here. Each section has a corresponding button below it: 'Train', 'Learn & Test', and 'Make'.

19. Click the “Train new machine learning model” button

This will take a few minutes, but you can carry on and start building your mobile app while this is happening.

The screenshot shows a web-based application interface for training machine learning models. At the top, there is a navigation bar with links for About, Projects, Worksheets, News, Help, Log Out, and Language selection. The main title is "Machine learning models". Below the title, there are two main sections: "What have you done?" and "What's next?". The "What have you done?" section contains text about collecting images for Doggie, Monkey, and Scissors, followed by a bulleted list: "• 10 examples of Doggie, • 10 examples of Monkey, • 10 examples of Scissors". The "What's next?" section asks if the user is ready to start training and provides a button to do so. A large blue arrow points from the "What's next?" section down to the "Train new machine learning model" button. At the bottom of the page, there is a box labeled "Info from training computer:" which contains the button.

[Train new machine learning model](#)

20. Click the “< Back to project” link

21. Finally, we'll use your machine learning model to make a game. Click “Make”

The screenshot shows a "Find It!" page with three main options: "Train", "Learn & Test", and "Make". Each option has a brief description and a corresponding button. A large blue arrow points from the "Learn & Test" section down to the "Make" button.

[Train](#)

[Learn & Test](#)

[Make](#)

22. Click on the “App Inventor” button

23. You will need the URL shown in red to set up your App Inventor project.

That is the unique address for your machine learning model.

You can use [App Inventor](#) to make mobile apps that run on your Android phones and tablets.

It runs in a web browser, like Scratch. Like Scratch, you code the app by dragging, dropping and snapping together blocks.

If you use [App Inventor](#) together with [Machine Learning for Kids](#) you can make an app for your phone, powered by artificial intelligence, using your own machine learning models.

For more detailed instructions on how to use the App Inventor extension, see github.com/kylecorry31/ML4K-AI-Extension

Support for App Inventor was created by [Kyle Corry](#) and [Joe Mazzone](#)

To use [App Inventor](#), go to <http://ai2.appinventor.mit.edu>

To add your machine learning model to your App Inventor project:

1. Click on **Import extension**
2. Click on **URL**
3. Fill in the URL for your project:
<https://machinelearningforkids.co.uk/api/appinventor/b3c23199-eaea-11e8-8237-273d108bf05e26de1138-6a7c-226c-6a19-e5ca823d79c6/extension>

24. Go to [App Inventor](http://ai2.appinventor.mit.edu) at <http://ai2.appinventor.mit.edu>

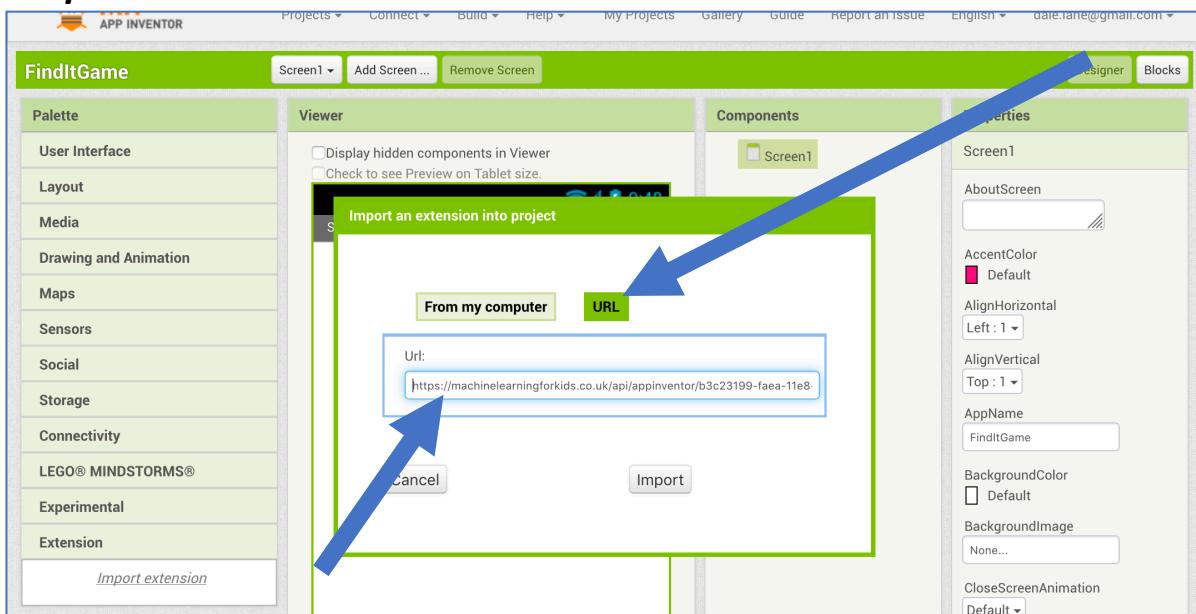
25. Start a new App Inventor project

26. Click on “Import Extension”

The screenshot shows the MIT App Inventor Designer interface. The top navigation bar includes 'Projects', 'Connect', 'Build', 'Help', 'My Projects', 'Gallery', 'Guide', 'Report an Issue', 'English', and a user account. The main workspace is titled 'FindItGame' and contains a single screen labeled 'Screen1'. The 'Palette' panel on the left lists categories like User Interface, Layout, Media, Drawing and Animation, Maps, Sensors, Storage, Connectivity, LEGO® MINDSTORMS®, Experimental, and Extension. The 'Extension' category is highlighted, and the 'Import extension' button is visible. The 'Components' panel shows 'Screen1' selected. The 'Properties' panel on the right displays properties for 'Screen1', including 'AboutScreen', 'AccentColor', 'AlignHorizontal', 'AlignVertical', 'AppName', 'BackgroundColor', 'BackgroundColor', 'BackgroundImage', 'CloseScreenAnimation', and 'Icon'.

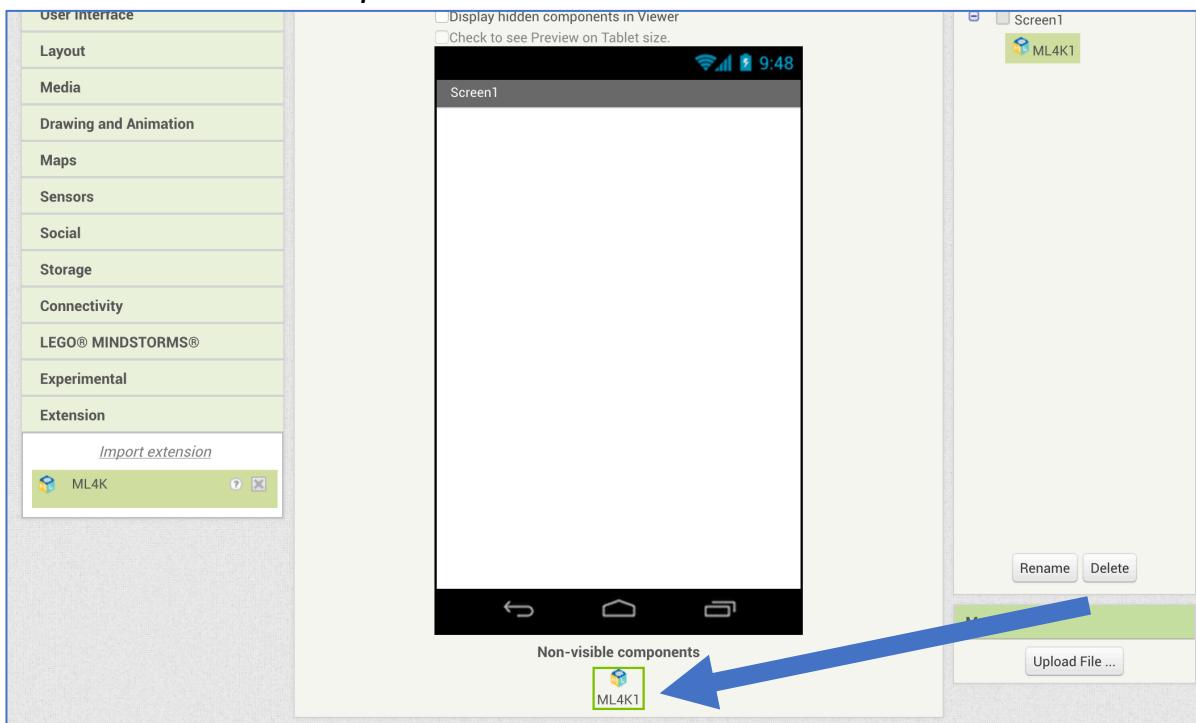
27. Import your machine learning model into the project.

Click on “URL”, then enter the URL that you got in step 23, and finally click “Import”.



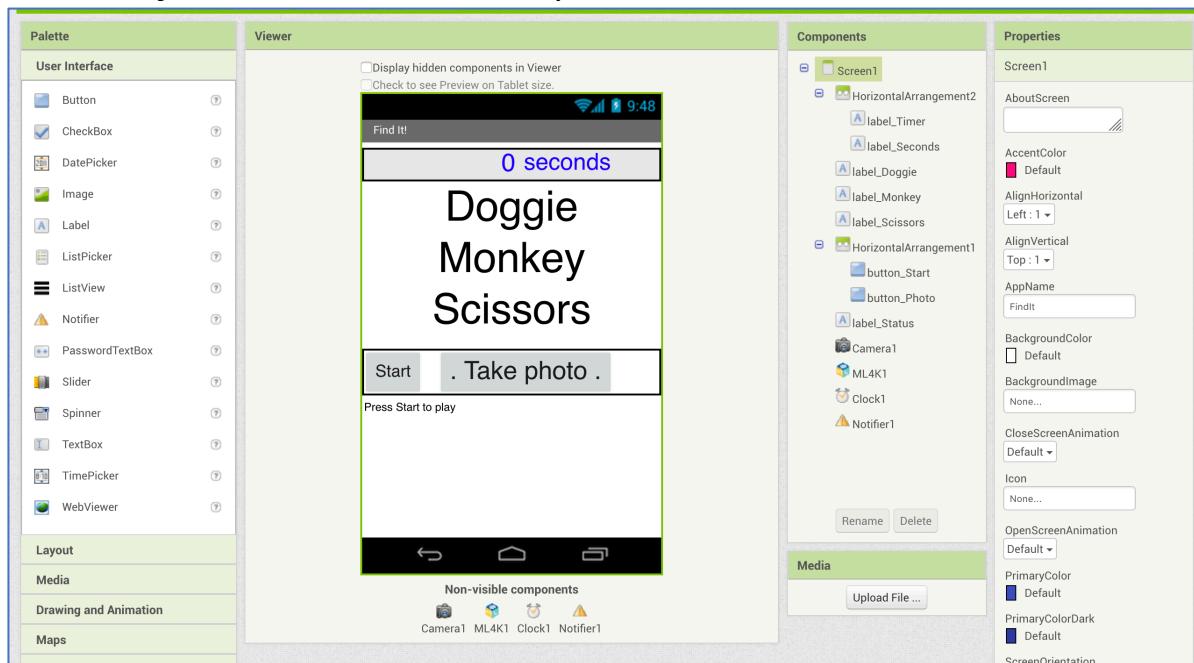
28. Drag the Machine Learning for Kids extension (“ML4K”) to the “Viewer”.

The icon will be added under the mobile screen once you’ve done this, in the “Non-visible components” list.

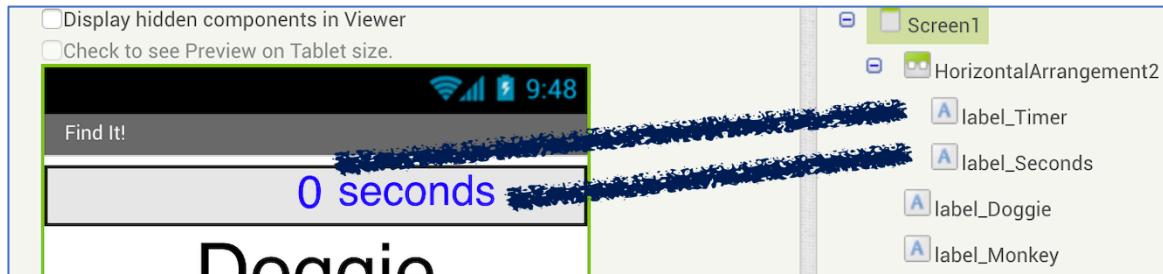


29. Create this mobile game user interface.

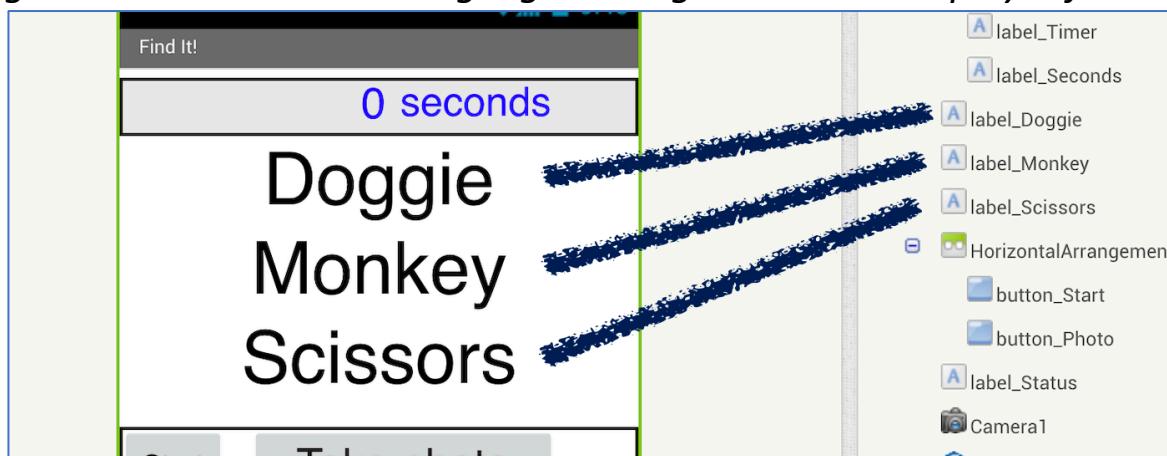
The next few screenshots will explain the main elements here.



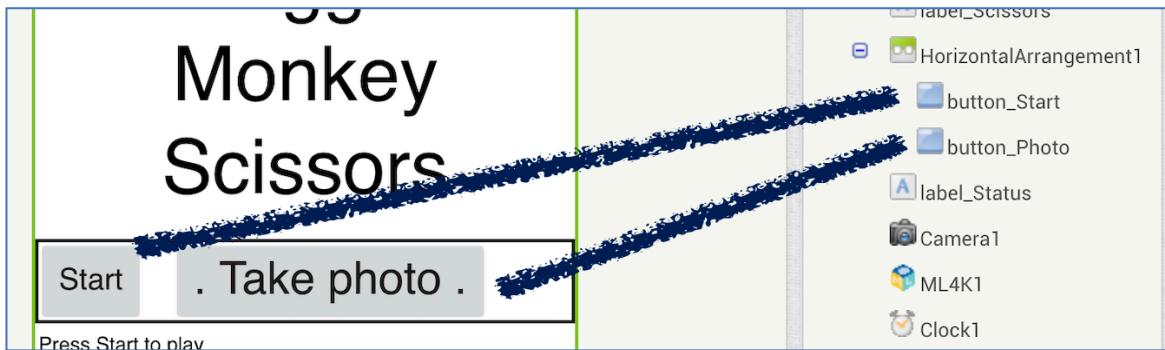
The game timer will keep track of how long it takes the player to find the objects. `label_Timer` will display the current time, and `label_Seconds` just says “seconds”.



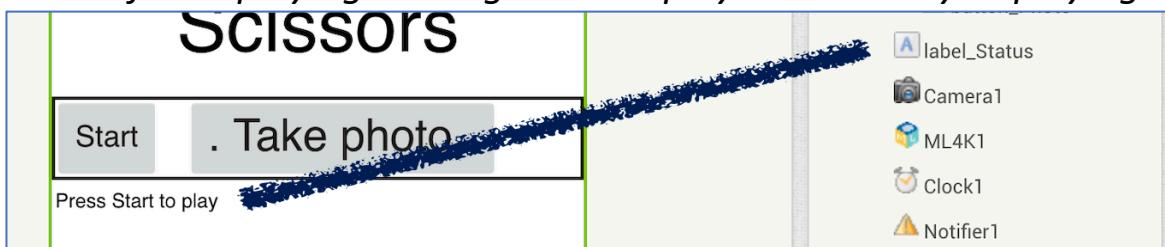
Three labels display the objects that you want the player to find in this game. Each item will be highlighted in green when the player finds them.



A couple of buttons to control the game – one to start, and one to take a photo of the object that the player has found.



A label for displaying messages to the player while they're playing.

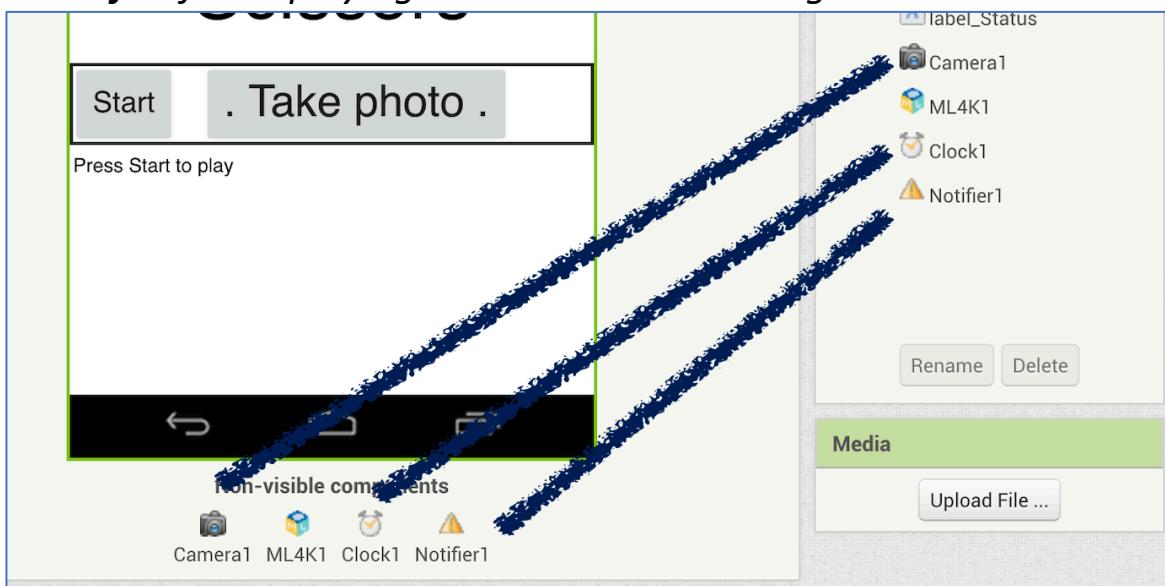


A collection of non-visible components needed in the game:

A Camera for taking pictures of objects.

*A Clock for the game timer. Set it to **disabled** so the timer won't start yet.*

A Notifier for displaying the Game Over message.



You can change the look of the app to make it your own, but you need to include these basic elements.

- 30.** Click on the “**Blocks**” button to start creating your script
- 31.** Create a script to start the game when the player clicks Start

```

when button_Start .Click
do
  set label_Doggie .BackgroundColor to [white]
  set label_Monkey .BackgroundColor to [white]
  set label_Scissors .BackgroundColor to [white]
  set label_Status .Text to "Try to find all the things!"
  set label_Timer .Text to "0"
  set Clock1 .TimerEnabled to false
  set Clock1 .TimerEnabled to true
  set button_Start .Text to "Restart"
  set button_Photo .Enabled to true

```

- 32.** Create a script to display the game timer on the screen
- 33.** Create a script to take a photo when the player clicks the button

```

when button_Photo .Click
do call Camera1 .TakePicture

```

- 34.** Create a script to send photos to your machine learning model

```

when Camera1 .AfterPicture
  image
do
  set label_Status .Text to "(Thinking...)"
  set button_Photo .Enabled to false
  call ML4K1 .ClassifyImage
    path get image

```

35. Create a script to display error messages if something goes wrong

```
when ML4K1 .GotError
  data error
  do set label_Status . Text to join " Something went wrong: " get error
    set button_Photo . Enabled to true
```

36. Create a script to update the game screen to confirm the object that they took a picture of.

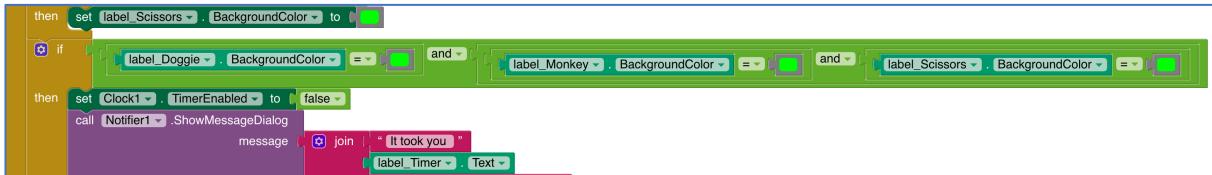
The line that is cut off in the picture is shown on the next page.

```
when ML4K1 .GotClassification
  data classification confidence
  do set label_Status . Text to join get confidence
    " % confident that you found the "
    get classification
    if get classification = " Doggie "
      then set label_Doggie . BackgroundColor to green
    else if get classification = " Monkey "
      then set label_Monkey . BackgroundColor to green
    else if get classification = " Scissors "
      then set label_Scissors . BackgroundColor to green
    if label_Doggie . BackgroundColor = and label_Monkey .
      then set Clock1 . TimerEnabled to false
        call Notifier1 . ShowMessageDialog
          message join " It took you "
          label_Timer . Text
          " seconds to find all the things. "
          title " You finished! "
          buttonText " OK "
    else set button_Photo . Enabled to true
```

37. The if line needs to check if every object is now green.

*If label_A.BackgroundColor = Green and label_B.BackgroundColor = Green
and label_C.BackgroundColor = Green*

If every object is green, it means the player has found all three objects and the game ends.

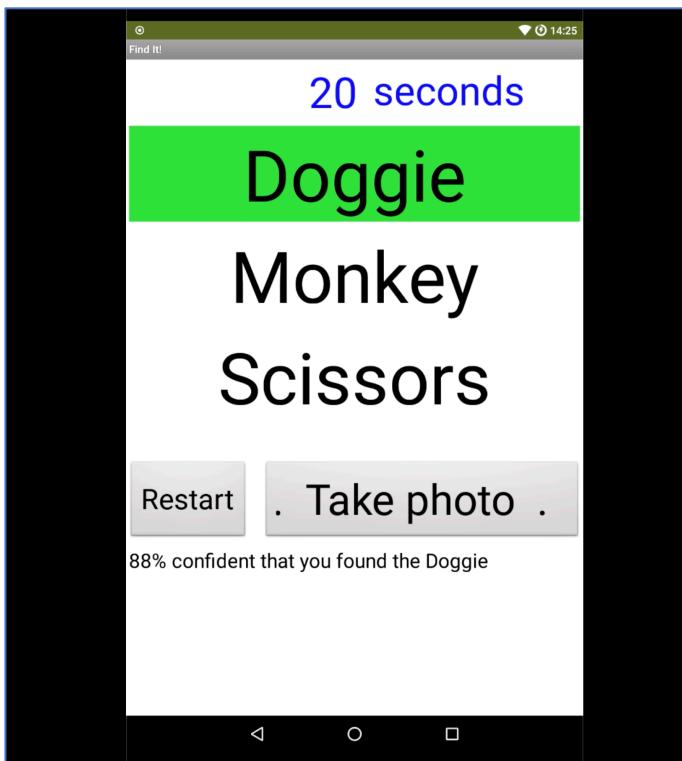


38. You're finished – test your game!

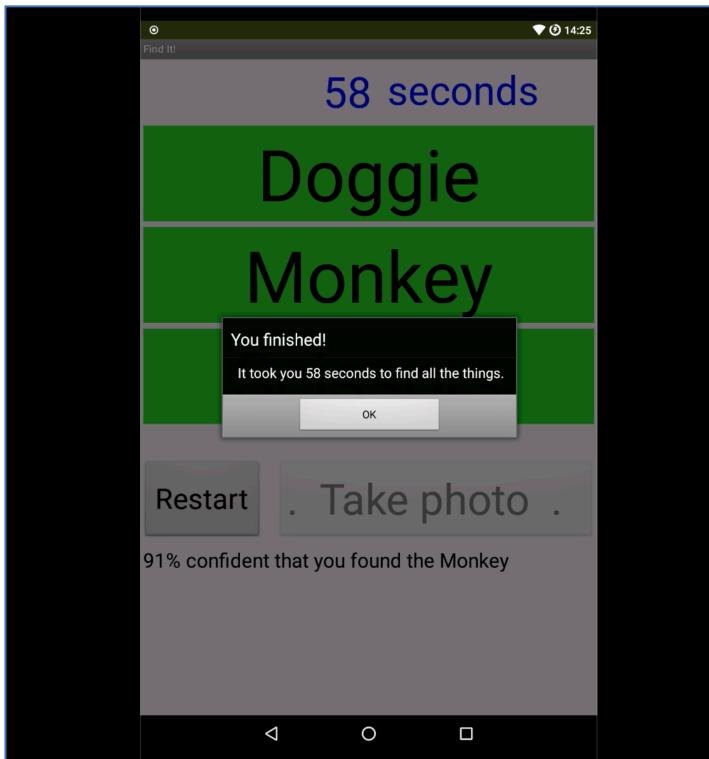
Press Start then hunt for the three objects.

When you find an object, use the “Take photo” button to take a photo.

If your machine learning model verifies that you've found the object, it will be marked on your list.



When your machine learning model has verified that you've found all three objects, the timer will stop and report how long you took.



What have you done?

You've trained a machine learning model to recognise photos of objects.

You did this by collecting examples of photos of these objects, and “labelling” them – telling the computer what is in each photo. The computer uses these labelled training examples to learn how to recognise new photos, by looking for patterns in the colours and shapes for examples.

This is called “supervised learning” because of the way you had to tell the computer what each training example was, rather than leave it to learn for itself from a big group of photos of all three objects.

Tips, ideas and Extensions

Now that you've finished, why not give one of these ideas a try?

Background

If the background of photos taken while playing the game are very different from the background of photos you took with the webcam to collect training examples, you might find that your machine learning model gets confused between objects and makes mistakes.

If that happens, try adding training examples with different backgrounds, to train the machine learning model to cope with a variety of backgrounds.

Mix things up with your examples

Take photos of the objects close to the camera and far away. Take photos of the objects from every side, upside down, from the top and from the bottom.

The more variety you can get in your training examples, the better your machine learning model will perform.

Try using confidence limits

The App Inventor block will return the confidence score for how certain your machine learning model is that it has recognised the photo.

What should your game do if the confidence score is very low?

Can you modify the script to use the score?