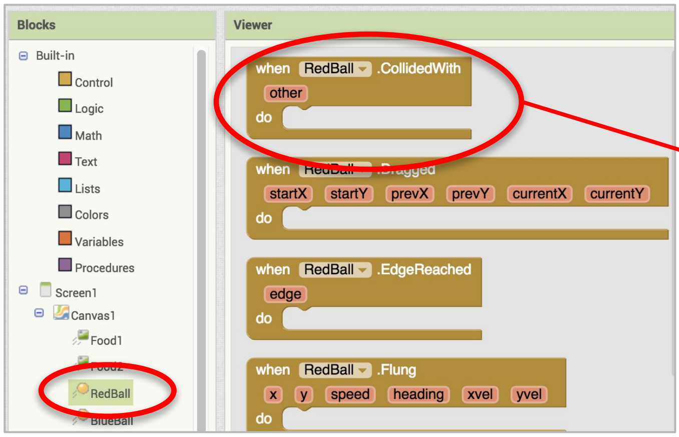
FOOD CHASE GAME: PART 2

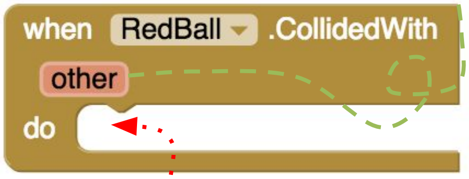
You will add code to detect collision between the RedBall and the other Sprites

COLLISION

The goal is to have **RedBall** "eat" the food and grow larger as it eats more food. However, it must avoid the **GreenBall**. If it collides with **GreenBall**, the game is over. To make it more interesting, **GreenBall** will also grow, and become more difficult to avoid.

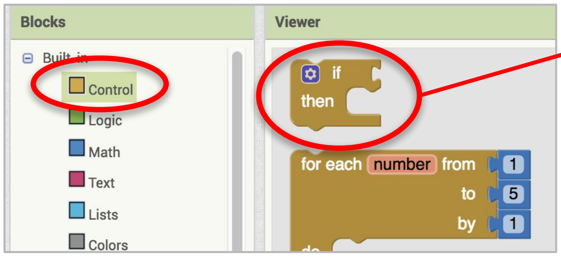
1 Drag out a **when RedBall.CollidedWith** block.

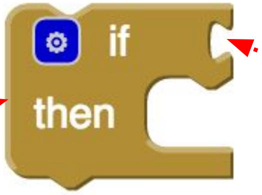




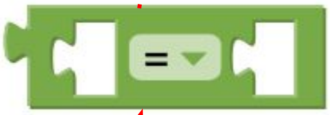
The *other* input parameter tells you what the **RedBall** has collided with

2 You need to test what **RedBall** has collided with. Drag out an **if** block from the Control drawer.

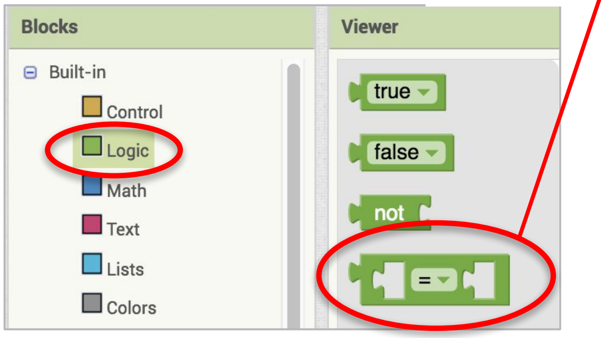




if blocks let you code different actions depending on whether a condition is true or not

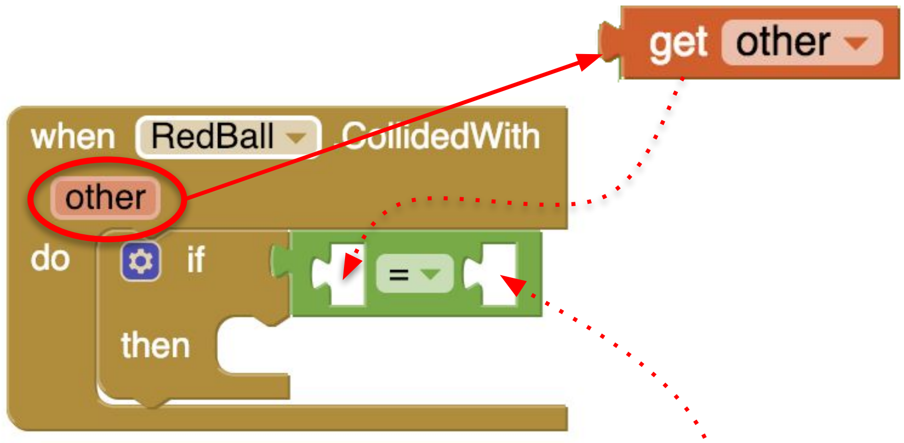


3 From the Logic drawer, drag out an **equals (=)** block.



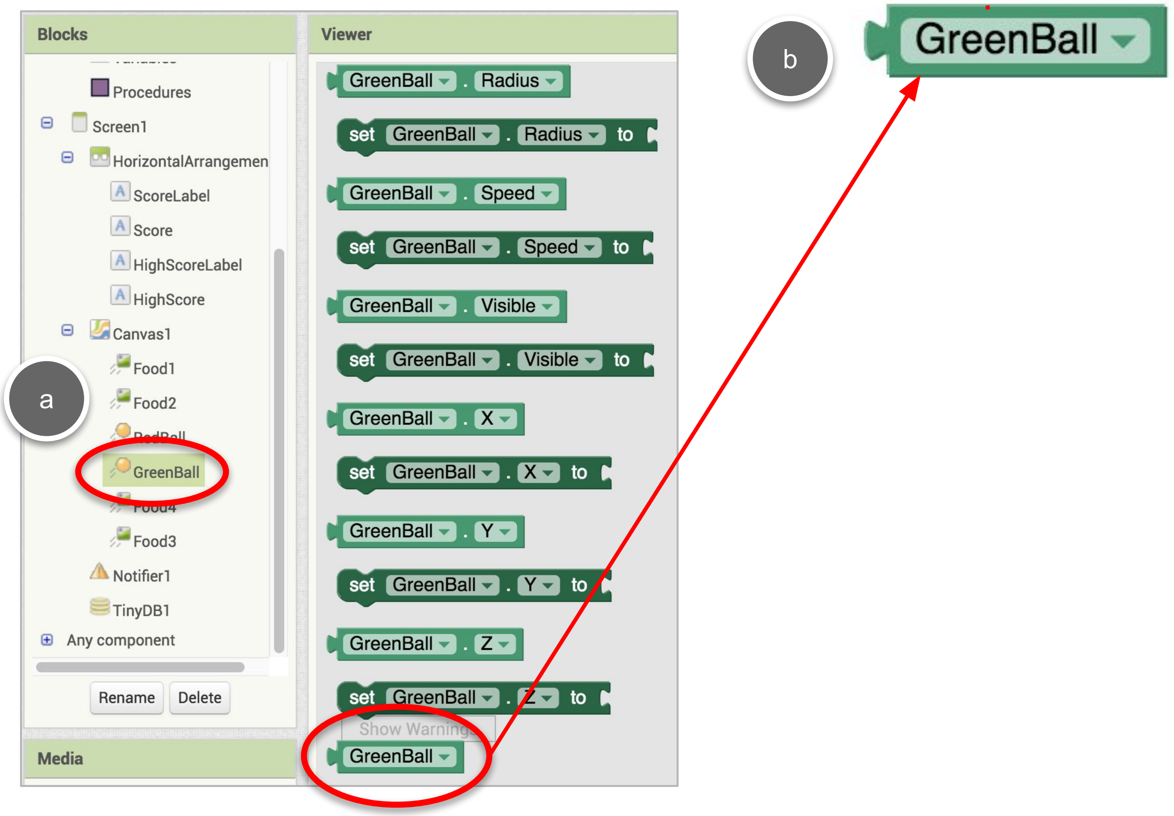
COLLISION WITH GREENBALL

4 Let’s first test if the **RedBall** collided with the **GreenBall**. Snap in other to the left side of **=**.



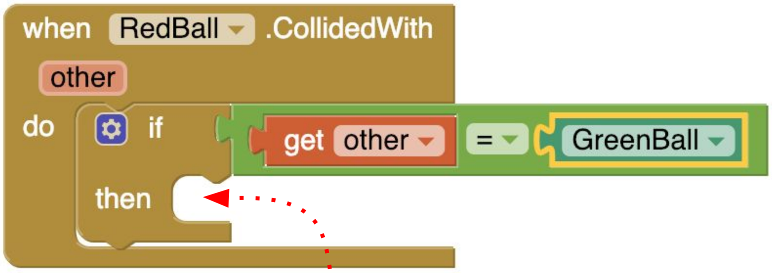
Hover over *other* to get a **get other** block.

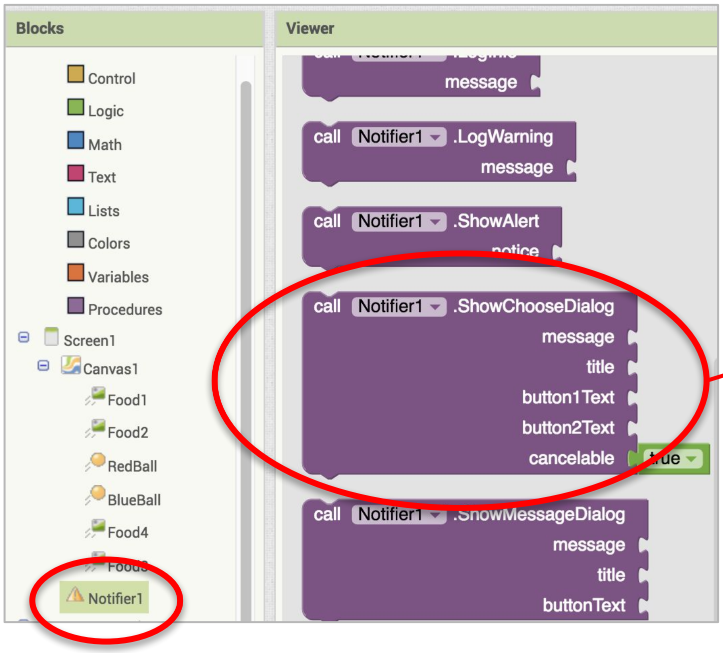
5 Snap in a **GreenBall** block to the right side of **=.**



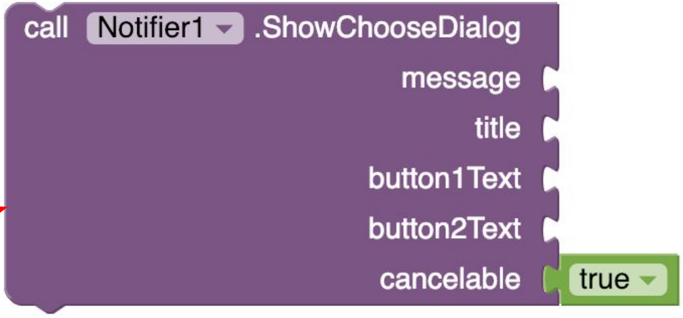
NOTIFY USER GAME OVER

6 If true, bring up a dialog box for the user by dragging a **Notifier1.ShowChooseDialog** block and snapping into the *then* slot.

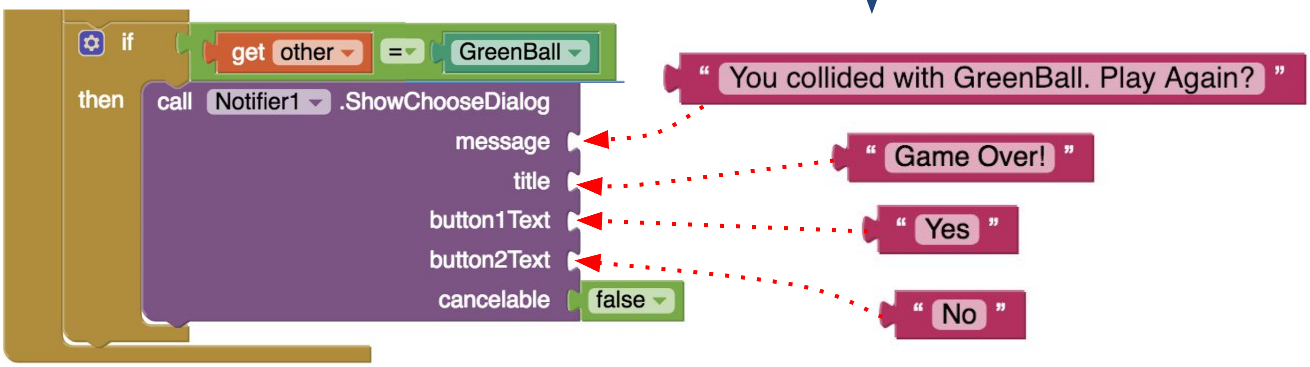




```
message
call Notifier1 .LogWarning
    message
call Notifier1 .ShowAlert
    notice
call Notifier1 .ShowChooseDialog
    message
    title
    button1Text
    button2Text
    cancelable  true
call Notifier1 .ShowMessageDialog
    message
    title
    buttonText
```



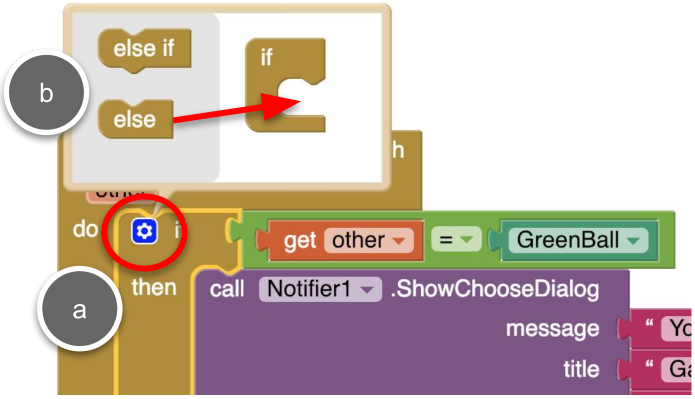
7 Add appropriate Text blocks for each slot.



FOOD COLLISION

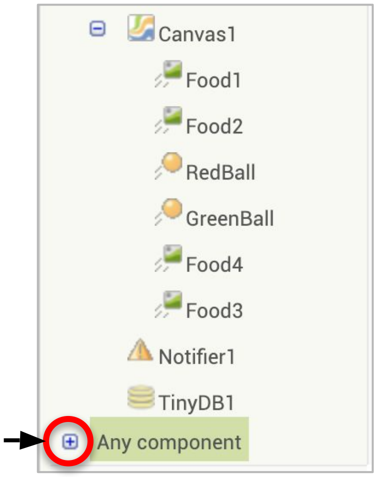
The **RedBall** can also collide with **Food** ImageSprites. To test that condition, you'll change the **if-then** block to an **if-then-else** block.

8 Click on the blue gear icon on the **if** block. A popup window appears. Drag the **else** into the **if** block in the popup to add the **else** slot.

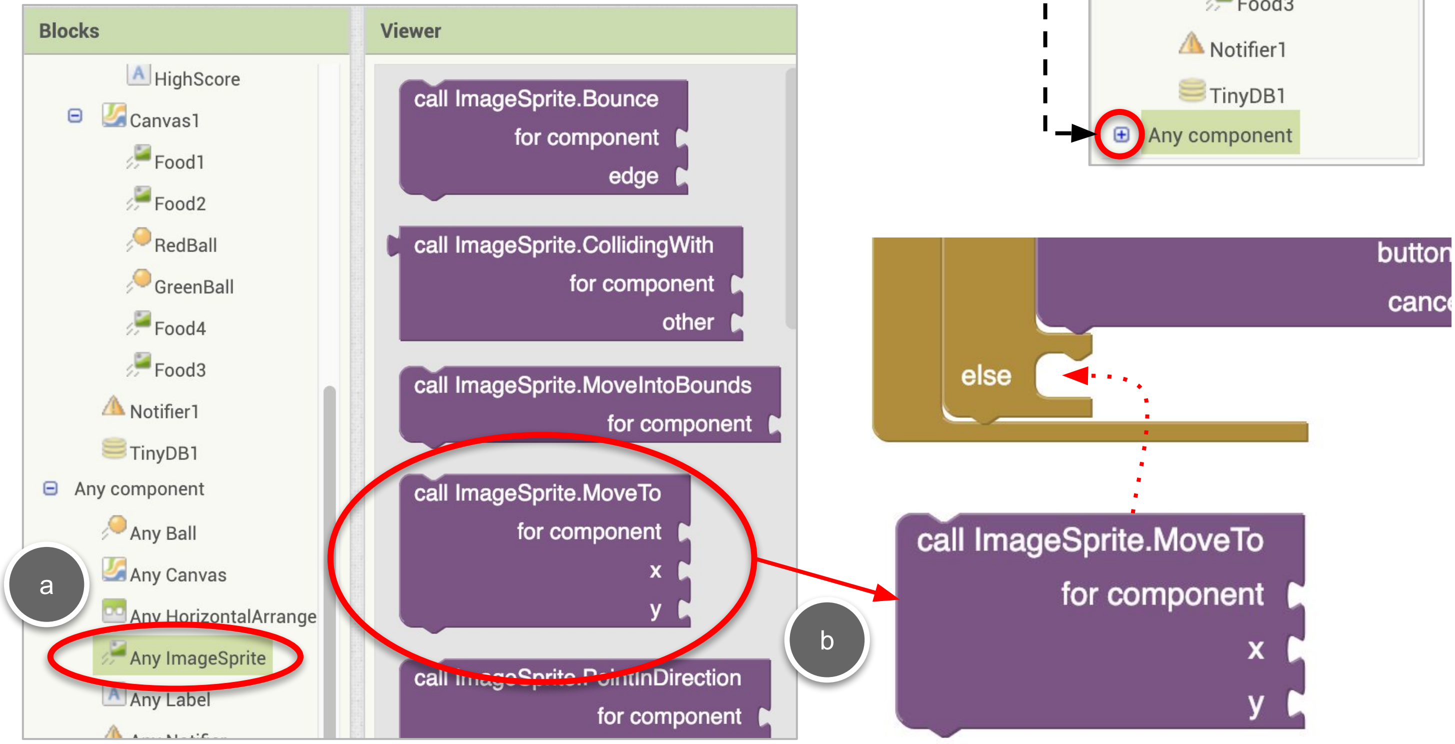


When the **RedBall** collides with any **Food ImageSprite**, you will move the **Food** randomly to another position on the screen. Because the action is the same for any of the **Food ImageSprites**, you can use the **Any Component** blocks.

9 Click the **+** by **Any Component** in the Blocks window on the left.



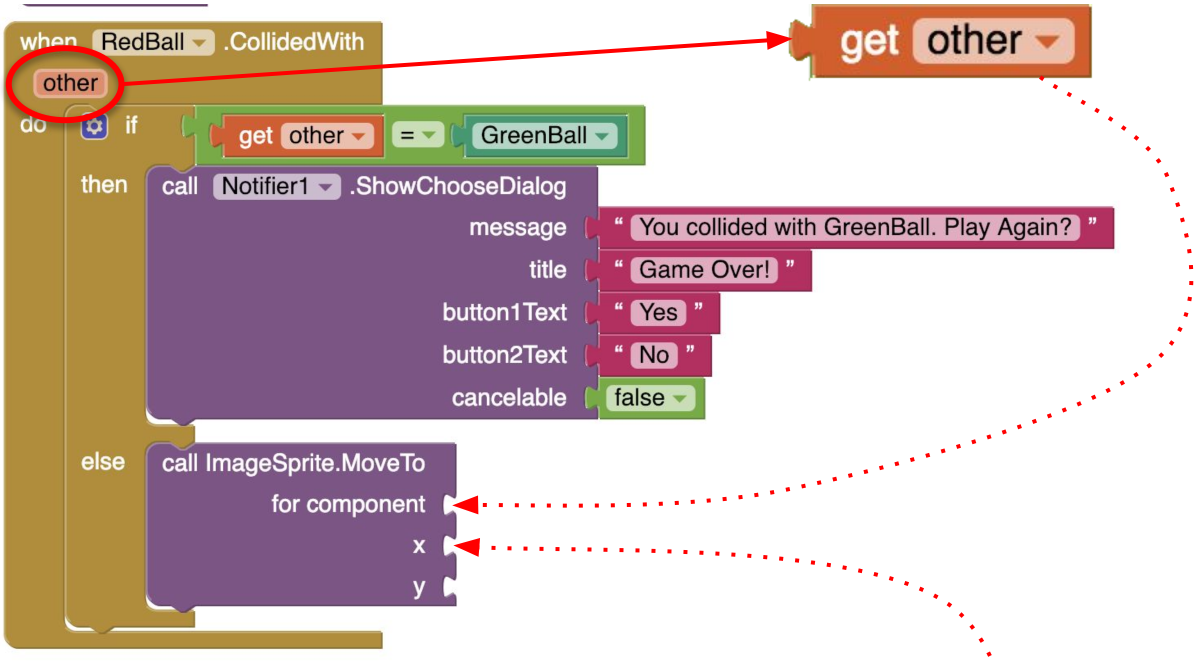
10 Click on **Any ImageSprite** and drag out an **ImageSprite.MoveTo** block and snap it into the **else** slot of the **if-then-else** block.



FOOD COLLISION

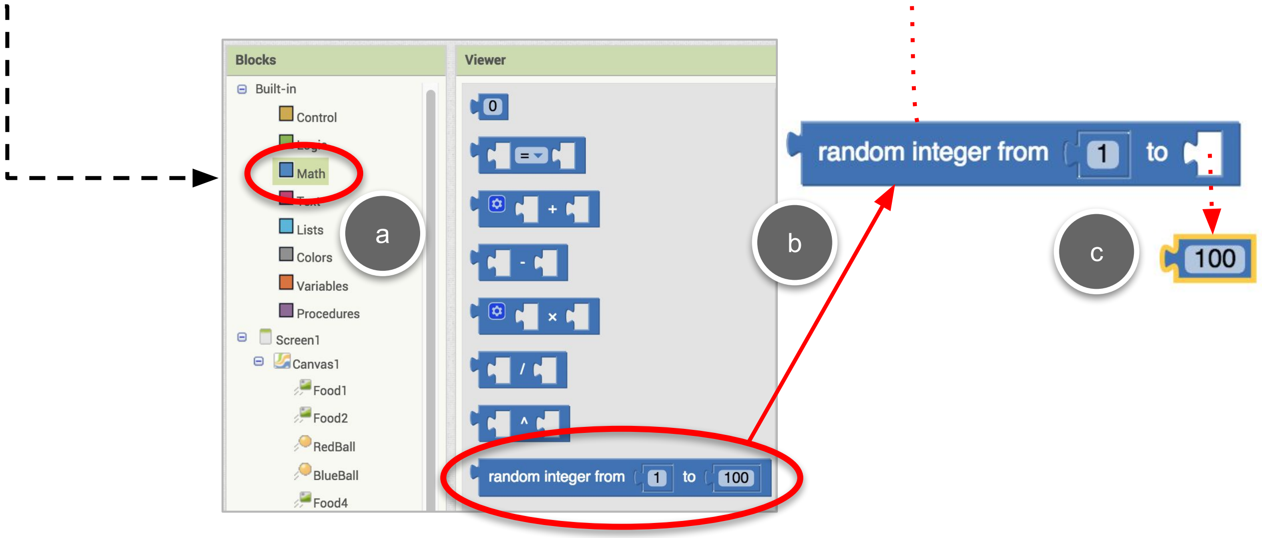
With Any Component blocks, there is an additional slot for component, because it needs to know which ImageSprite is being moved. For this event, other identifies the ImageSprite that is colliding with the RedBall, so that is the ImageSprite to be moved.

11 Hover over other and snap get other into the component slot in ImageSprite.MoveTo.



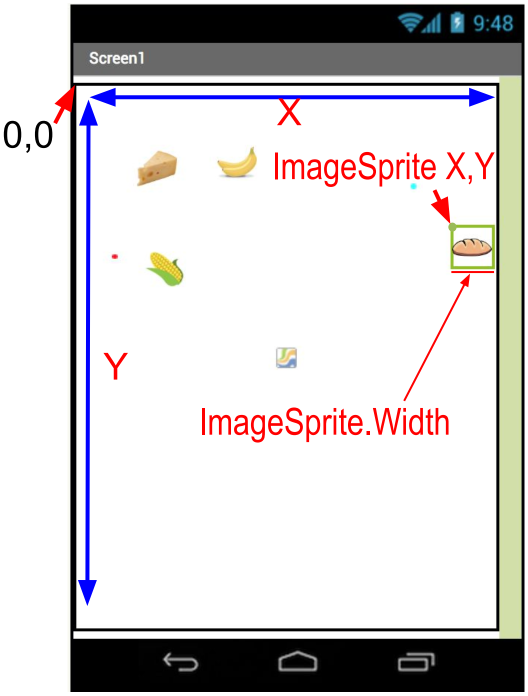
```
when RedBall .CollidedWith
  other
do  if  get other = GreenBall
    then  call Notifier1 .ShowChooseDialog
                message  " You collided with GreenBall. Play Again? "
                  title  " Game Over! "
            button1Text  " Yes "
            button2Text  " No "
             cancelable  false
    else  call ImageSprite.MoveTo
          for component
                      x
                      y
```

12 From the Math Drawer, drag out a **random integer from** block. Keep the first number, **1**, but remove the **100** block from the second puzzle slot. Snap it into the **x** slot.



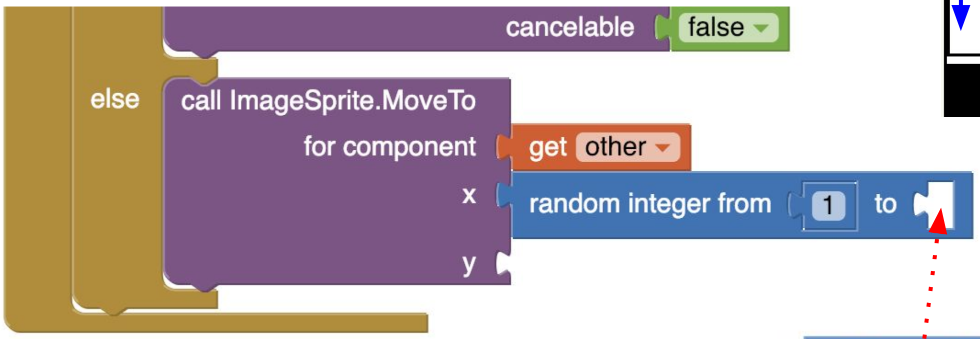
MIT APP INVENTOR

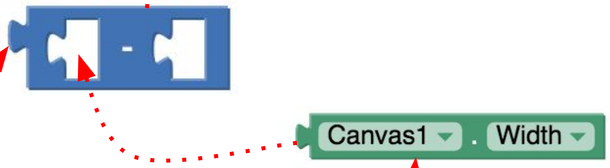
MOVE FOOD RANDOMLY



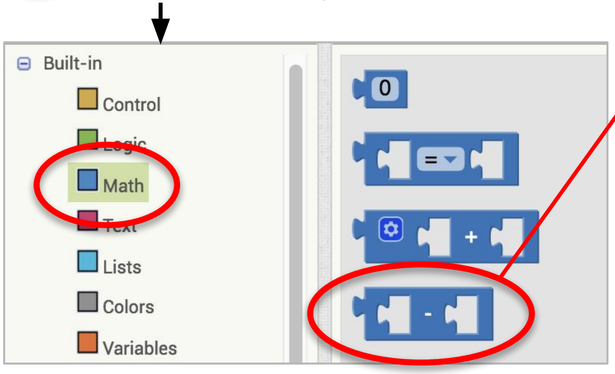
The layout of the app's screen is based on the Cartesian coordinate system, but with the origin in the upper left corner. The upper left corner of the **ImageSprite's** Picture is its **X,Y**.

The value range for a random X will be from 1 to the *Canvas.Width*. However, if it is placed just at *Canvas.Width*, the **ImageSprite** would appear to the right of the **Canvas**, which is off the screen. So you need to set the range for possible X values from 1 to the *Canvas.Width* minus the **ImageSprite's** *Width.*

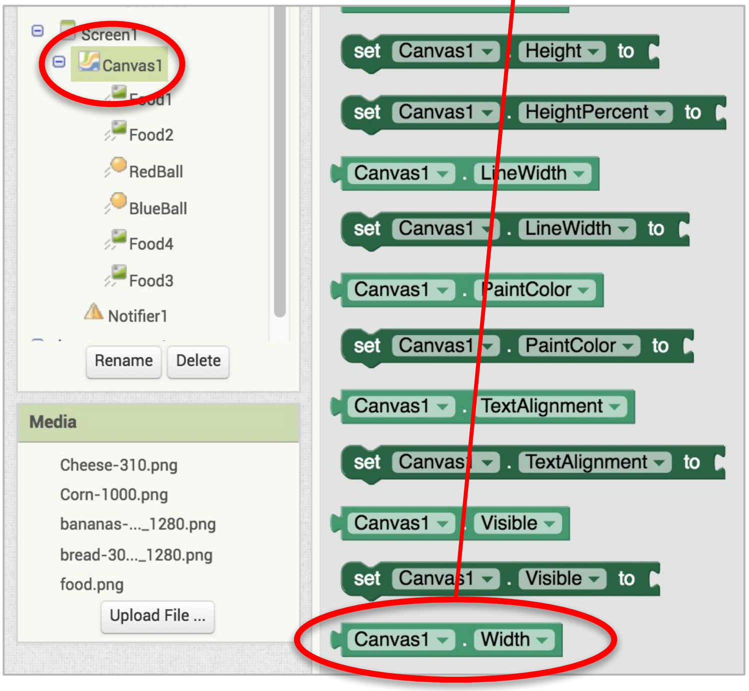




13 Drag a **subtract (-)** block from the Math drawer and snap into the random block.

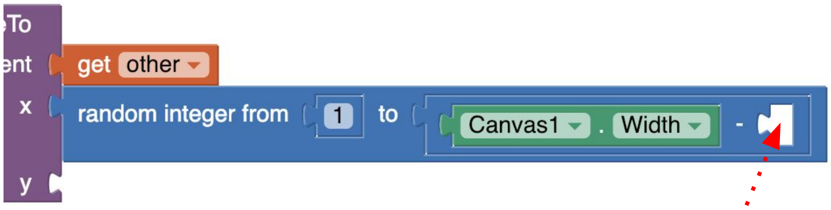


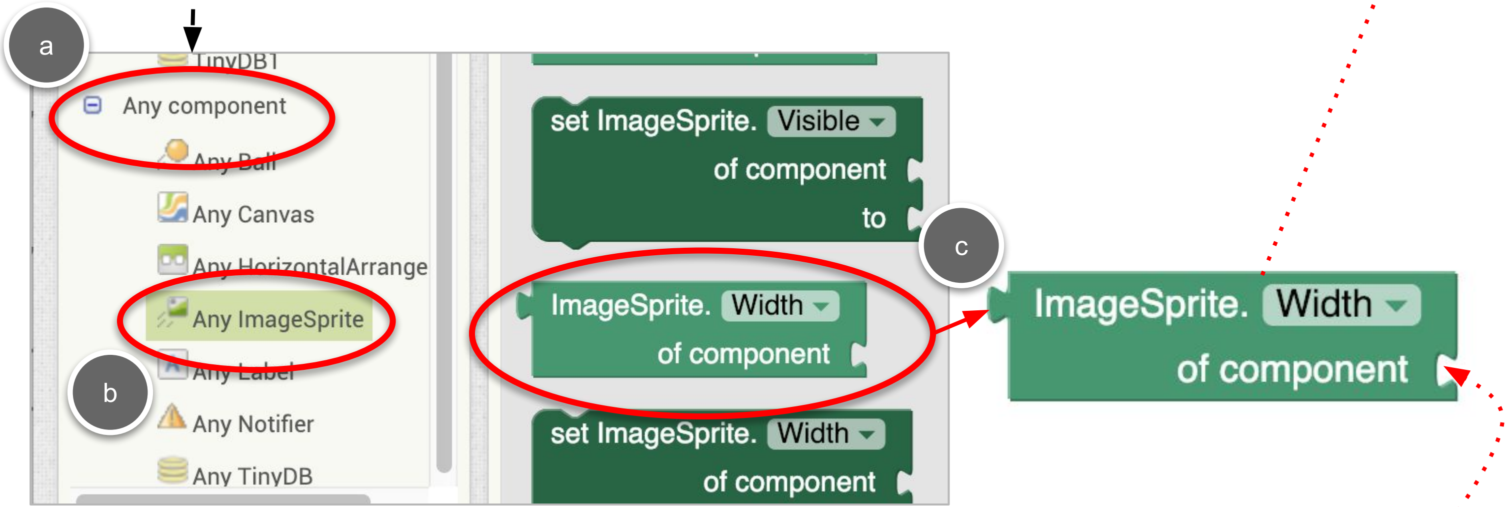
14 From the Canvas1 drawer, drag a **Canvas1.Width** block and snap it to the left side of the **subtract (-)** block.



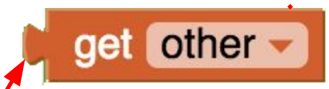
MOVE FOOD RANDOMLY

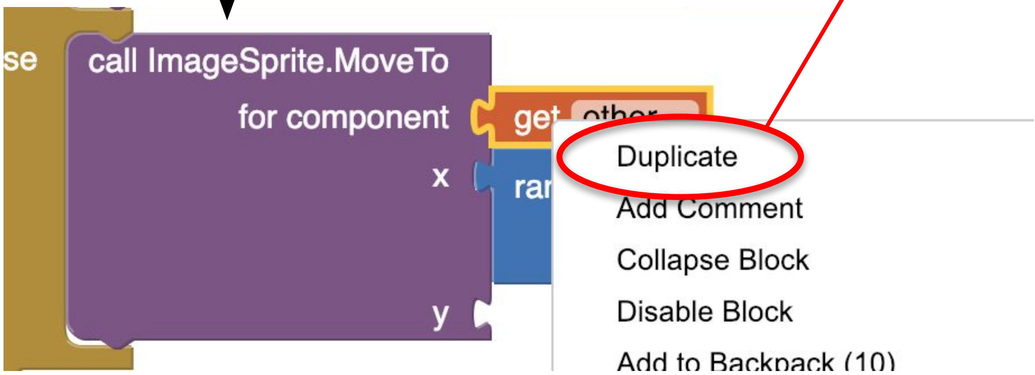
15 You need to subtract the *Width* to make sure the Food image fits on the Canvas. Drag out an **ImageSprite.Width** block from the **Any ImageSprite** drawer.





16 Duplicate **get other** and snap it into the component slot of **ImageSprite.Width**.

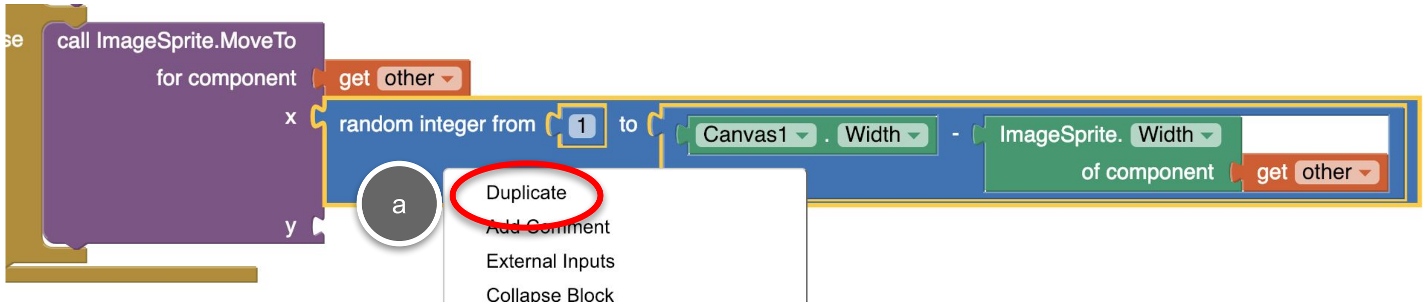


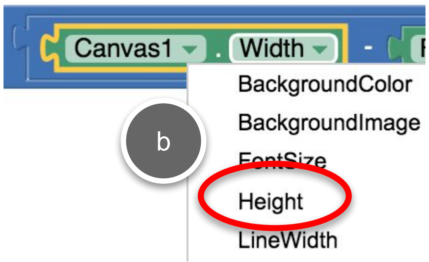


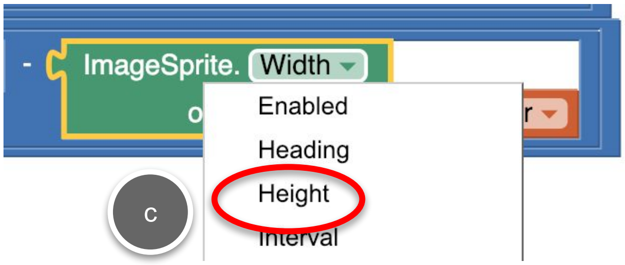
MOVE FOOD RANDOMLY (continued)

Can you guess what goes in the **y** slot?

17 You can Duplicate the **random integer** block and change **Canvas1.Width** to **Canvas1.Height** and **ImageSprite.Width** to **ImageSprite.Height**.

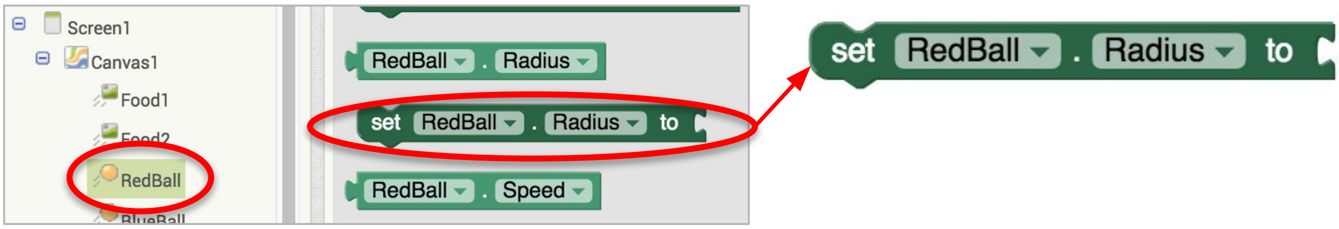






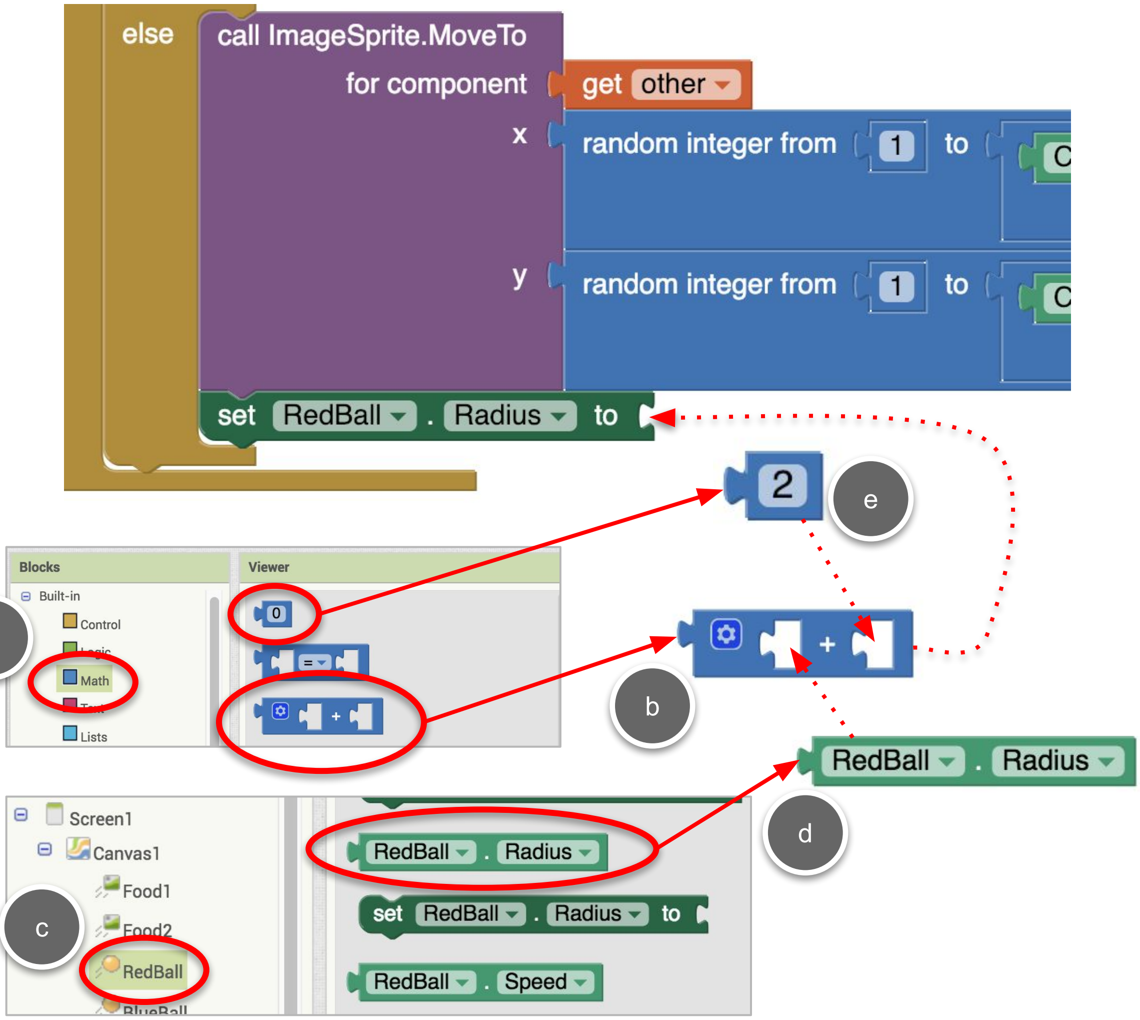
Last thing to do is "grow" the **RedBall** when it eats Food, increasing its radius by 2.

18 Drag a **set RedBall.Radius** block and snap it in after the **ImageSprite.MoveTo** block.



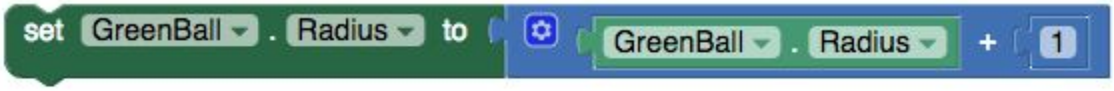
COLLISION continued

19 Snap in an addition (**+**) block from the Math drawer, and add 2 to the current value of **RedBall.Radius**.

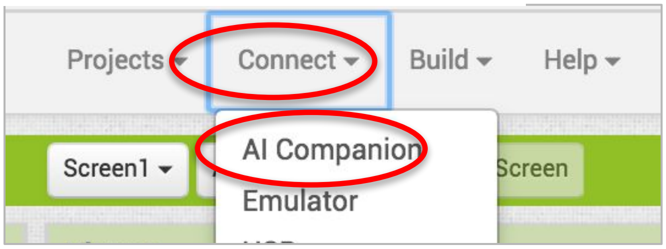


Also increase **GreenBall**'s radius, but just by 1.

20 Duplicate the **set RedBall.Radius** block and change **RedBall** to **GreenBall**. Change **2** to **1.**



21 Test your game now! **RedBall** should grow each time it touches Food, and a message should appear if it touches **GreenBall**.

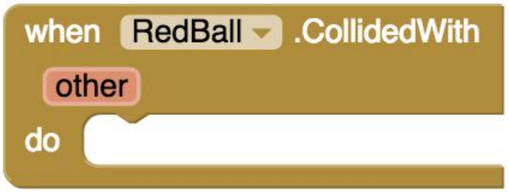


COMPUTATIONAL THINKING CONCEPTS

The following are the Computational Thinking Concepts learned in Part 2.

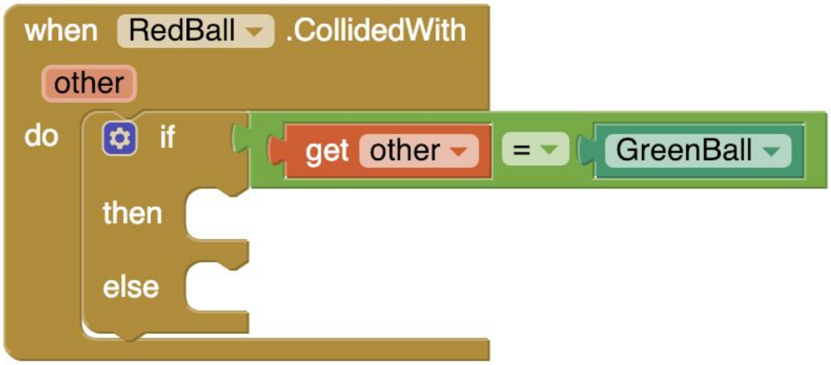
Food Chase Game

1. Events



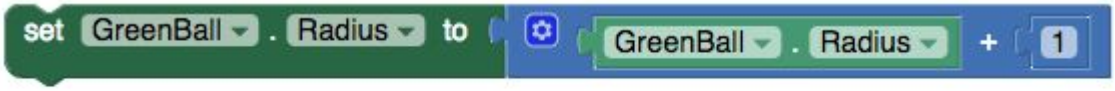
```
when RedBall .CollidedWith
  other
do
```

2. Conditionals

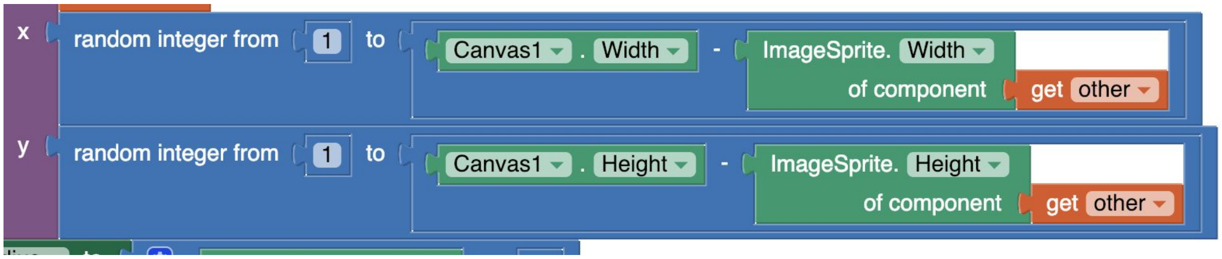


```
when RedBall .CollidedWith
  other
do  if    get other = GreenBall
    then
    else
```

3. Operators



```
set GreenBall . Radius to   GreenBall . Radius + 1
```



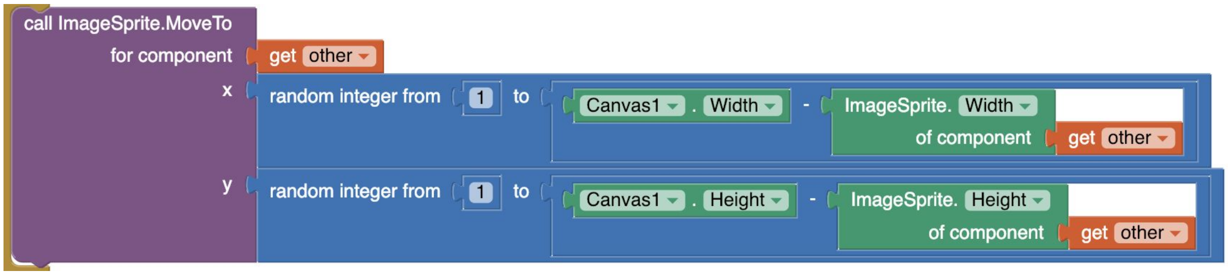
```
x  random integer from 1 to   Canvas1 . Width  -  ImageSprite. Width
                                                     of component  get other
y  random integer from 1 to   Canvas1 . Height -  ImageSprite. Height
                                                     of component  get other
```

COMPUTATIONAL THINKING PRACTICES

The following are the Computational Thinking Practices learned in Part 2.

Food Chase Game

1. Abstraction and Modularization



```
call ImageSprite.MoveTo
       for component  get other
                   x  random integer from 1 to Canvas1 . Width - ImageSprite. Width
                                                                  of component  get other
                   y  random integer from 1 to Canvas1 . Height - ImageSprite. Height
                                                                   of component  get other
```