Food Chase

Make a Game App that where the Red Ball eats food to grow, avoiding the Green Ball

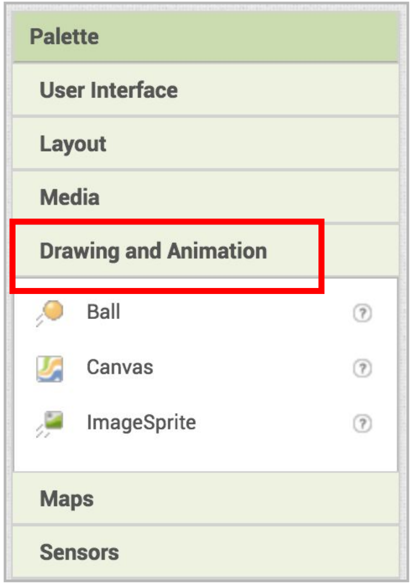
Essential Questions

- How can you control sprites or characters in a mobile game app?
- How do you animate sprites in a mobile game app?

Objectives

1. Code a game app that includes animated sprites.
2. Use conditionals to correctly check two values within a program.
3. Demonstrate abstraction with a procedure.
4. Use variables correctly to store and retrieve data.
5. Improve their computational identity by making an app that can be shared with friends and family.
6. Work collaboratively with a partner to create a mobile app.

Lesson 1: Drawing and Animation Components



Palette

User Interface

Layout

Media

Drawing and Animation

Ball

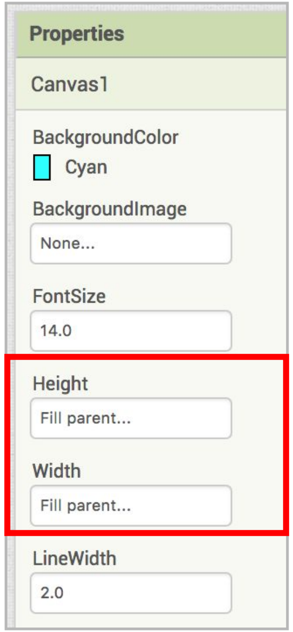
Canvas

ImageSprite

Maps

Sensors

Fill Parent for *Height* and *Width* of the **Canvas** will cause it to fill the device screen.



Properties

Canvas1

BackgroundColor
Cyan

BackgroundImage
None...

FontSize
14.0

Height
Fill parent...

Width
Fill parent...

LineWidth
2.0

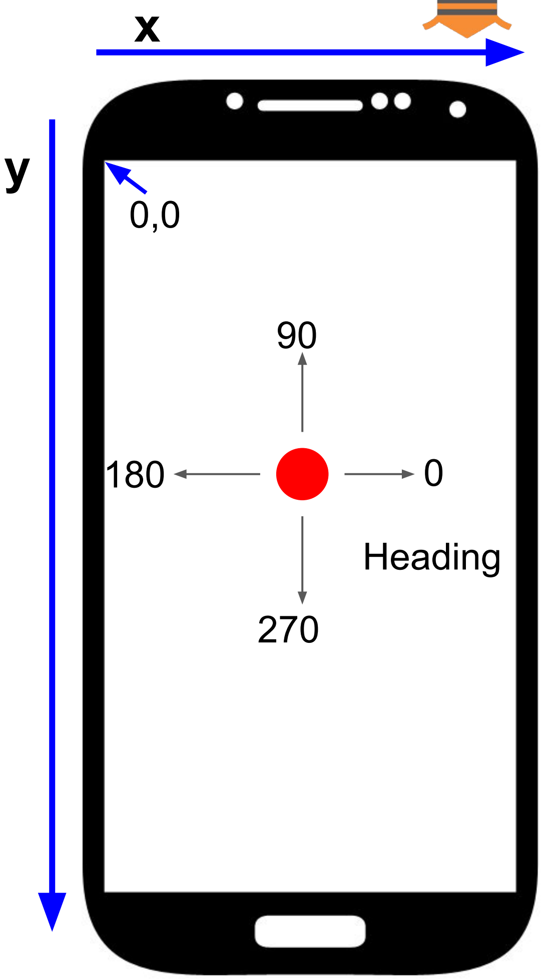
Drawing Components

- Canvas is your background for sprites to appears and move.
- Balls/ImageSprites are the elements on the canvas that can be controlled by user interaction and by coding
- The screen is considered the "parent", fill parent means fill the page.

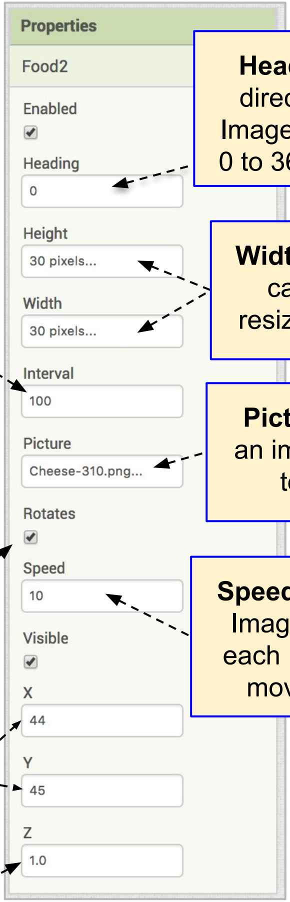
This app will use the Flung event for the Canvas and transfer the speed and heading of the fling to set the Ball's movement.

Drawing Components

- Introduction to the Canvas
  - Coordinate System
    - (0,0) in upper left corner
    - X increases to the right
    - Y increases down
  - Ball
    - Position is x,y using coordinate system
    - Speed determines how fast it moves (in pixels)
    - Heading is direction (0-360 degrees)
    - Interval is how often the ball moves by its speed
  - ImageSprite
    - Works the same as Ball but can attach an image



Properties



Properties

Food2

Enabled

☑

Heading

0

Height

30 pixels...

Width

30 pixels...

Interval

100

Picture

Cheese-310.png...

Rotates

☑

Speed

10

Visible

☑

X

44

Y

45

Z

1.0

Heading is the direction of the ImageSprite (from 0 to 360 degrees).

Width and **Height** can be set to resize your sprite.

Interval is how often the ImageSprite moves. 100 means every 1/10th of a second. 1000 means every second!

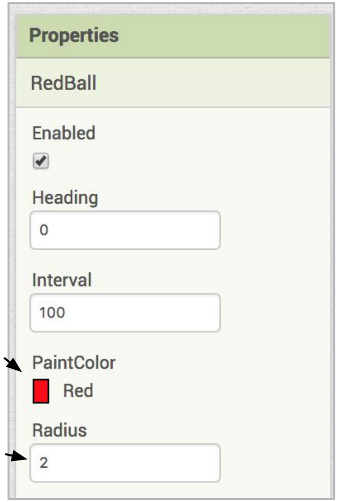
Picture can be set to an image file uploaded to your project.

Rotates here is checked, meaning the ImageSprite rotates according to its heading.

Speed is how fast the ImageSprite moves each Interval. Here it moves 10 pixels.

X and **Y** are the positions of the ImageSprite (before it starts moving).

Z is not used in this app.



Properties

RedBall

Enabled

☑

Heading

0

Interval

100

PaintColor

🟥 Red

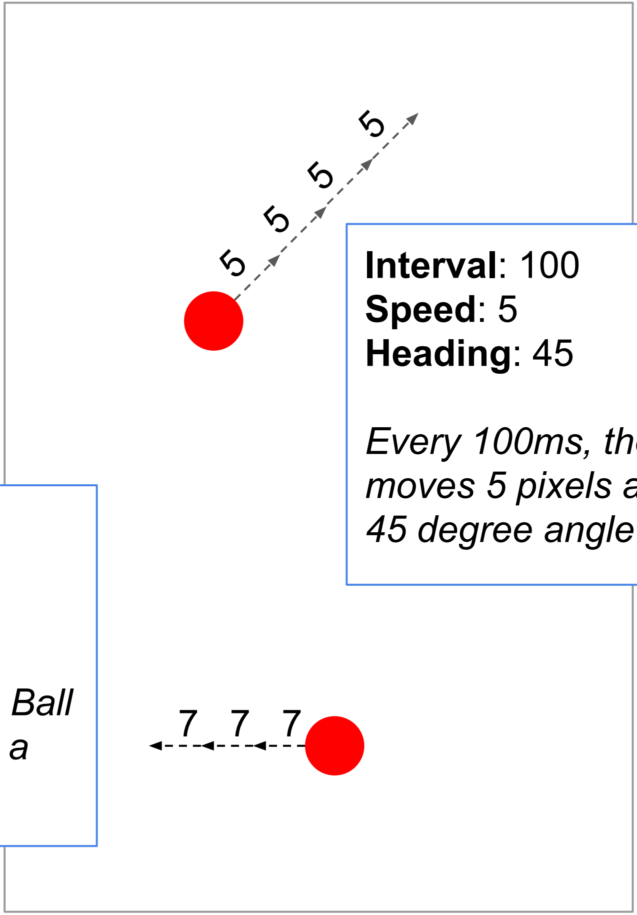
Radius

2

PaintColor lets you change the Ball's color.

Radius lets you change the size of the Ball.

Animation



Interval: 100
Speed: 5
Heading: 45

Every 100ms, the Ball moves 5 pixels at a 45 degree angle

Interval: 500
Speed: 7
Heading: 180

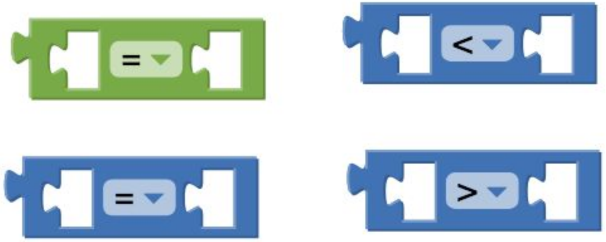
Every 500ms, the Ball moves 7 pixels at a 180 degree angle

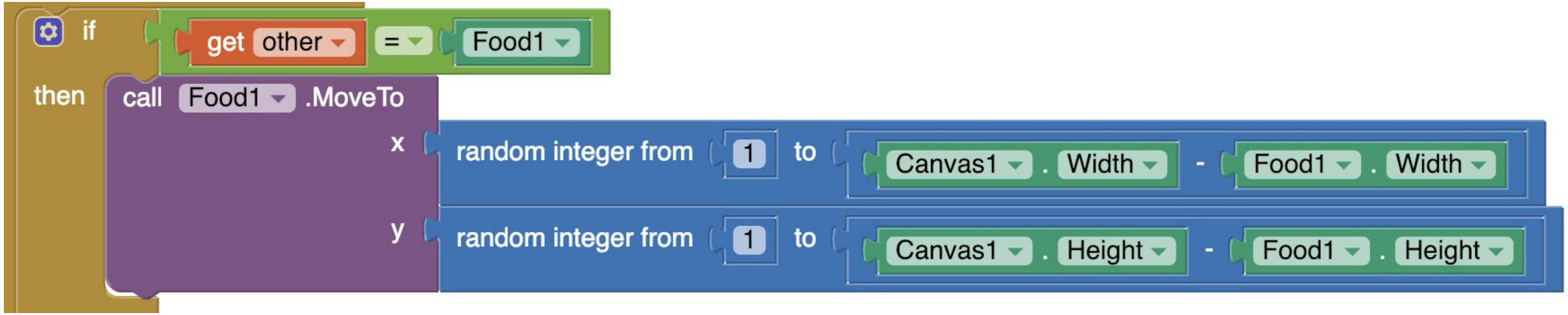
Lesson 1:

Complete Student Guide Part 1:

Lesson 2: Conditionals

- **If** blocks allow you to only execute code blocks when certain conditions are true.
- Logic and Math blocks are use to test if something is true or false.

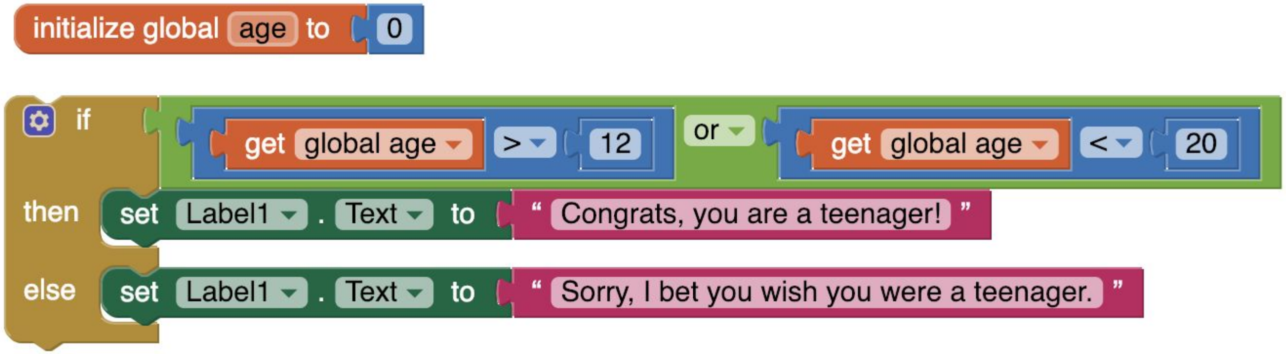




```
if   get other = Food1
then call Food1 .MoveTo
                   x  random integer from 1 to Canvas1 . Width - Food1 . Width
                   y  random integer from 1 to Canvas1 . Height - Food1 . Height
```

if-then-else

- Do one thing if the condition is true
- Do another thing if it is false

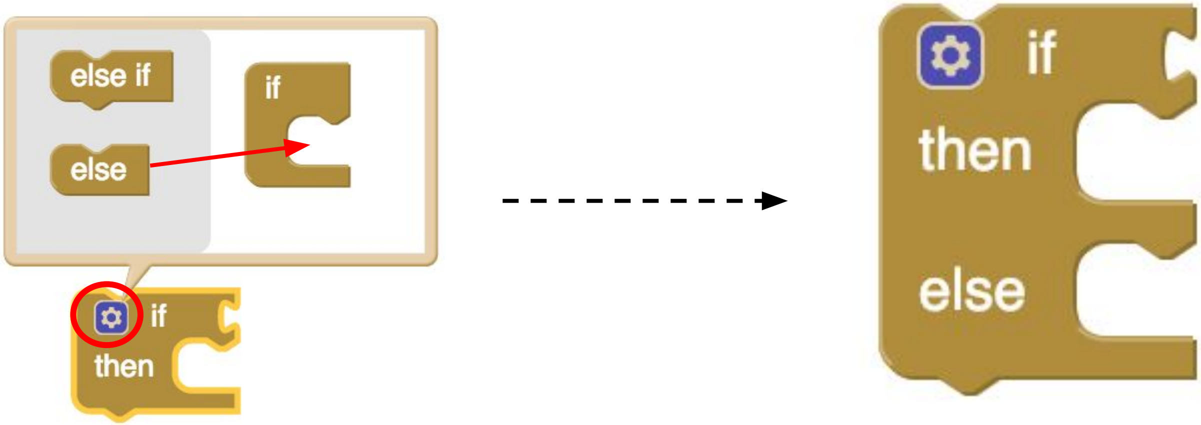


```
initialize global age to 0

if   get global age > 12  or  get global age < 20
then set Label1 . Text to " Congrats, you are a teenager! "
else set Label1 . Text to " Sorry, I bet you wish you were a teenager. "
```

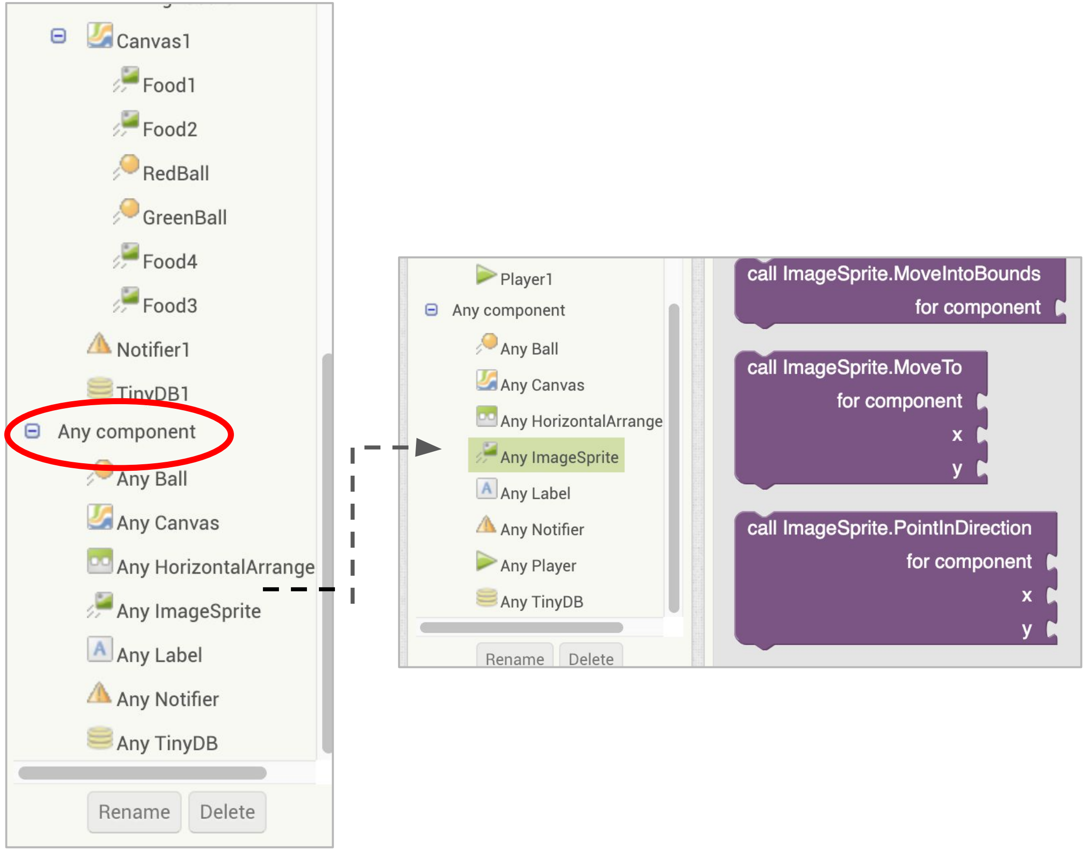
if-then-else

- Click blue gear icon on if block
- Drag else into if block



Lesson 2: Any Component

- Each type of Component in the app has its own set of Any Component blocks



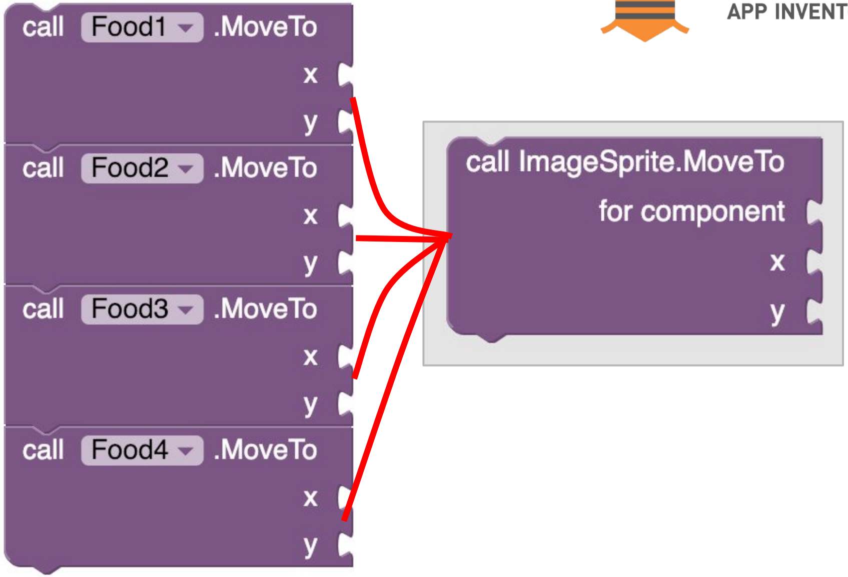
```
call ImageSprite.MoveIntoBounds
                  for component

call ImageSprite.MoveTo
       for component
                   x
                   y

call ImageSprite.PointInDirection
                  for component
                              x
                              y
```

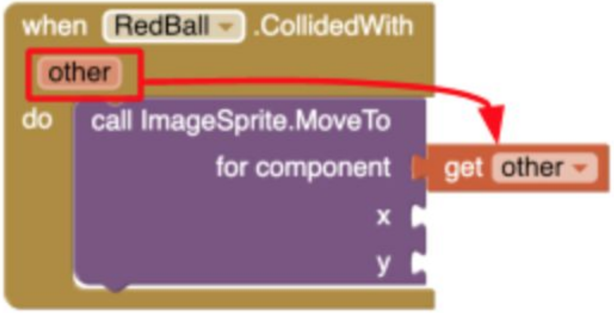
Any Component

- You can use this block and apply it to *any* ImageSprite.
- Generalizes the movement of an ImageSprite (*example of **Abstraction***)
- Allows you to condense your code to a single block (instead of 4)
- Use the **other** parameter to specify which **ImageSprite** in **RedBall.CollidedWith**.



```
call Food1 .MoveTo
    x
    y
call Food2 .MoveTo
    x
    y
call Food3 .MoveTo
    x
    y
call Food4 .MoveTo
    x
    y
```

```
call ImageSprite.MoveTo
    for component
    x
    y
```



```
when RedBall .CollidedWith
 other
do  call ImageSprite.MoveTo
        for component  get other
        x
        y
```

Lesson 2:

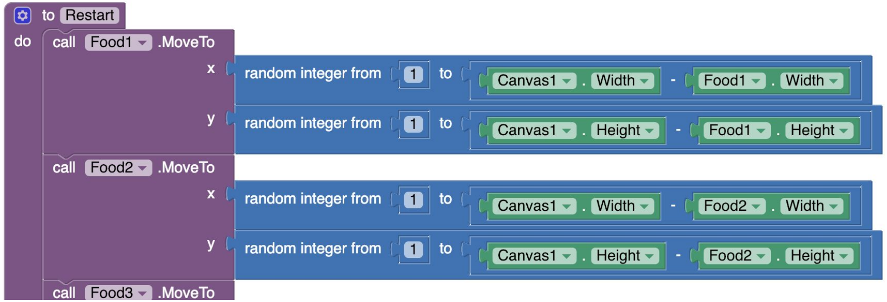
Complete Student Guide Part 2:

Lesson 3: Procedures

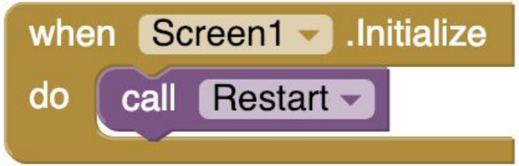
- **Separates** out a particular "task" or action that may involve several code blocks
- If you have similar code blocks in multiple parts of your program, it is easier and better practice to have **one set of code blocks**, organized as a procedure.
- Helpful with **testing and debugging**. Once you have tested a procedure and confirmed that it works correctly, then it does not have to be tested again.
- Procedures **make updating code easier**. If a procedure's blocks need to be changed to accommodate a new feature, it only has to be changed in one place.

Restart procedure

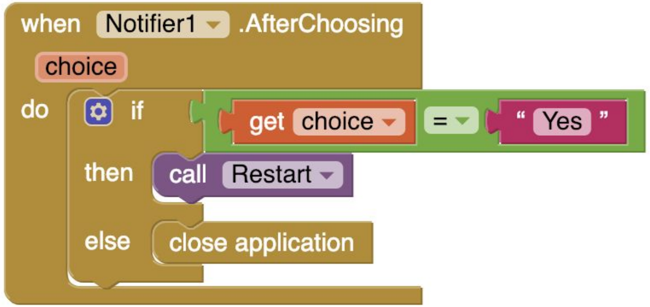
- Sets positions of all the ImageSprites
- Sets radii of RedBall and GreenBall to 2
- Called when app starts and when user chooses to play again



```
to Restart
do  call Food1.MoveTo
        x  random integer from 1 to Canvas1.Width - Food1.Width
        y  random integer from 1 to Canvas1.Height - Food1.Height
    call Food2.MoveTo
        x  random integer from 1 to Canvas1.Width - Food2.Width
        y  random integer from 1 to Canvas1.Height - Food2.Height
    call Food3.MoveTo
```



```
when Screen1.Initialize
do  call Restart
```



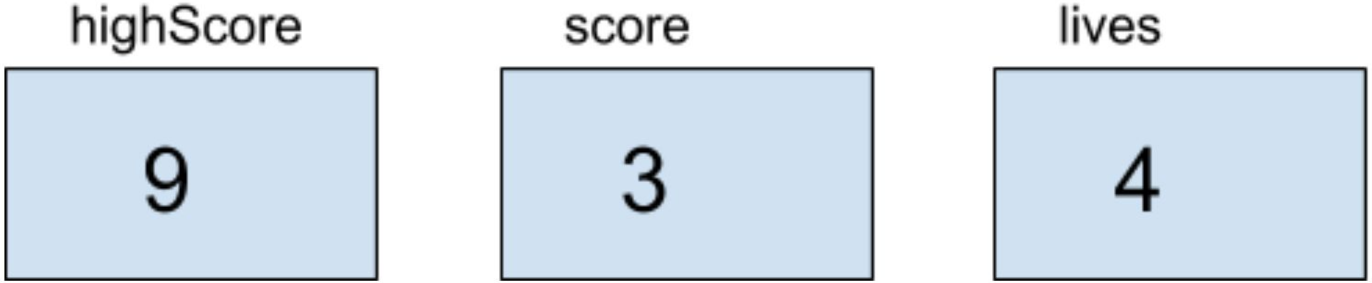
```
when Notifier1.AfterChoosing
  choice
do  if    get choice = " Yes "
    then  call Restart
    else  close application
```

Lesson 3:

Complete Student Guide Part 3:

Lesson 4: Variables

- Placeholders to store values in an app
- Can update the value
- Can get the value by variable name



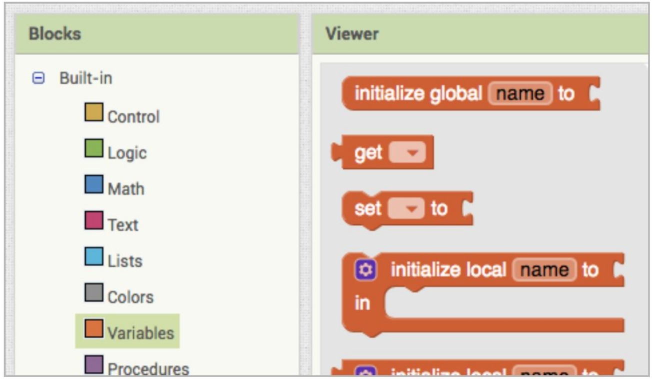
highScore 9

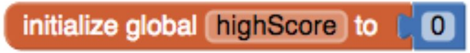
score 3

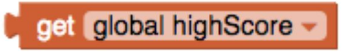
lives 4

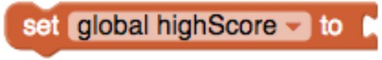
Variables

- Initialize sets the variable to its starting value
- Set changes the value
- Get accesses the value



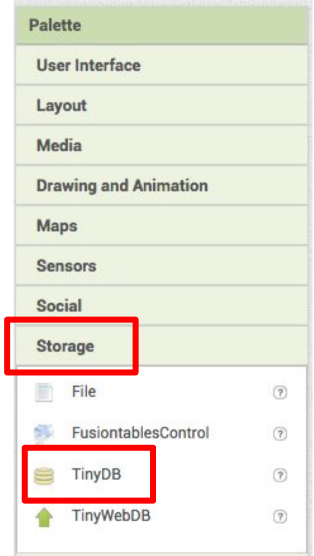






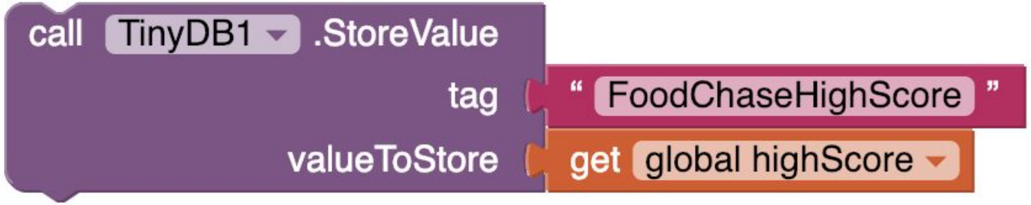
Lesson 4: TinyDB

- Variable values disappear when you close the app.
- If you want to store values persistently (between executions of the app), use the TinyDB component.
- Stores tags and their values on the mobile device.

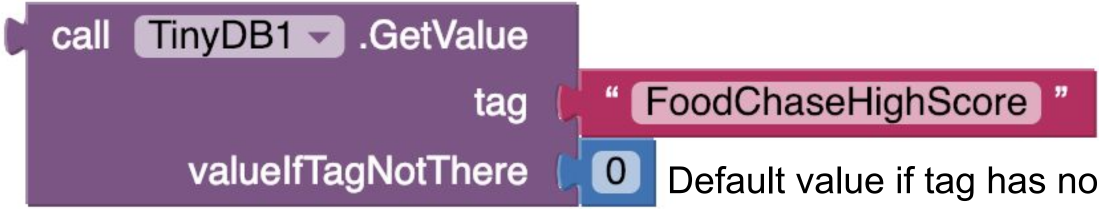


TinyDB

- **StoreValue** stores the value in TinyDB
- **GetValue** gets the value from TinyDB
- ***Tag*** is like the variable name
- ***Value*** is the value being stored (and gotten)



```
call TinyDB1 .StoreValue
                    tag  " FoodChaseHighScore "
           valueToStore  get global highScore
```



```
call TinyDB1 .GetValue
                    tag  " FoodChaseHighScore "
       valueIfTagNotThere  0
```

Default value if tag has not been stored yet

Variables and TinyDB

	Variable	**TinyDB**
Persistence	Anything stored in a variable is erased when the app closes	Anything stored in TinyDB with a tag can be retrieved at any point, even after the app is closed and reopened
Storing data	**set name** to value	**TinyDB.StoreValue** (tag, value)
Retrieving stored data	**get name**	**TinyDB.GetValue**(tag, valueIfTagNotThere)

Lesson 4:

Complete Student Guide Part 4:

Vocabulary Words

conditional

if-then

if-then-else

Variable

Persistent data

Tag

Value

TinyDB

Any component

generalization

abstraction