Unit 4: Variables, Conditionals, and Functions

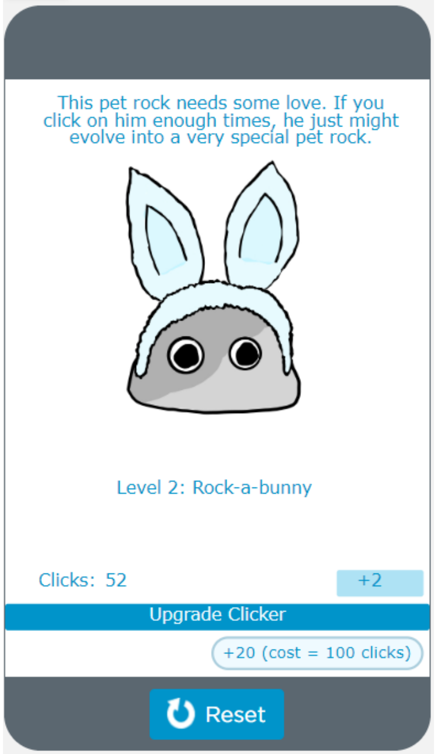
Lesson 1: Variables Explore
Lesson 2: Variables Investigate
Lesson 3: Variables Practice
Lesson 4: Variables Make
Lesson 5: Conditionals Explore
Lesson 6: Conditionals Investigate
Lesson 7: Conditionals Practice
Lesson 8: Conditionals Make
Lesson 9: Functions Explore/Investigate
Lesson 10: Functions Practice
Lesson 11: Functions Make
Lesson 12: Project - Decision Maker App Part 1
Lesson 13: Project - Decision Maker App Part 2
Lesson 14: Project - Decision Maker App Part 3
Lesson 15: Assessment Day

Unit 4 - Lesson 1
Variables Explore

Warm Up

●○○

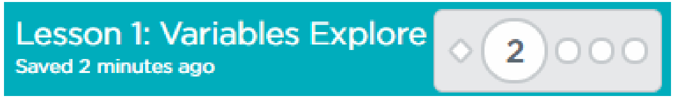
Unit 4 Lesson 1 - Warm Up



This pet rock needs some love. If you click on him enough times, he just might evolve into a very special pet rock.

Level 2: Rock-a-bunny

Clicks: 52



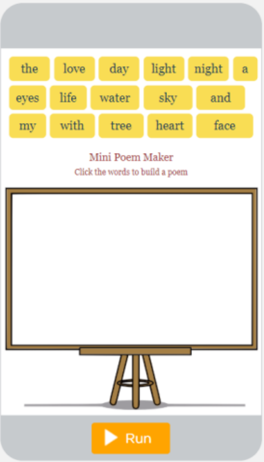
Lesson 1: Variables Explore

Saved 2 minutes ago

Do This:

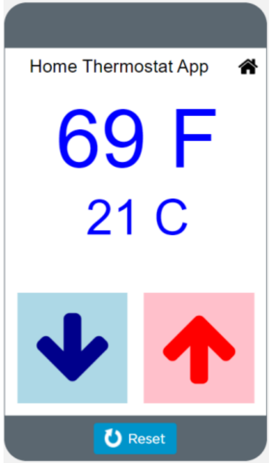
- Navigate to Lesson 1, Level 2 on Code Studio

Unit 4 Lesson 1 - Warm Up



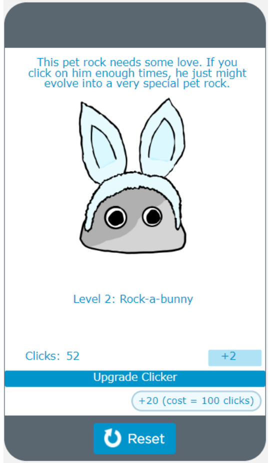
Mini Poem Maker

Click the words to build a poem



Prompt:

These are samples of the kinds of apps you'll be able to build by the end of this unit. As you go through them, write down at least two examples where the app seems to be keeping track of a piece of information or using it to make a decision.



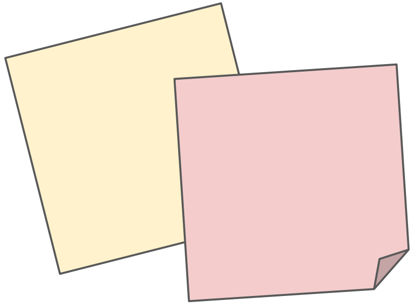
This pet rock needs some love. If you click on him enough times, he just might evolve into a very special pet rock.

Activity

● ● ○

Variables Explore

You and your partner should have:
Small stacks of red and yellow stickies
3 plastic baggies
Pen/Pencil
Dry erase marker



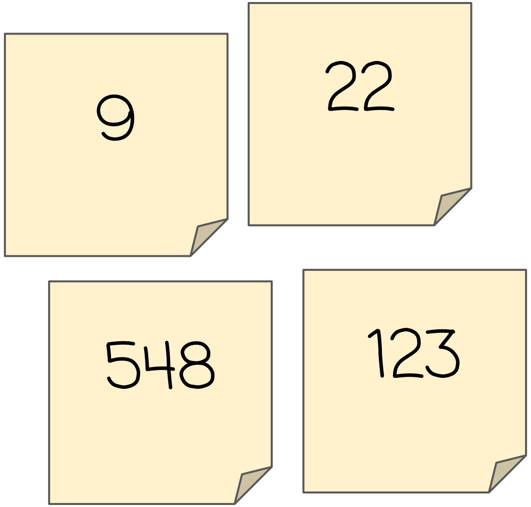
Unit 4 Lesson 1 - Activity

Value

One piece of information
Goes on a sticky

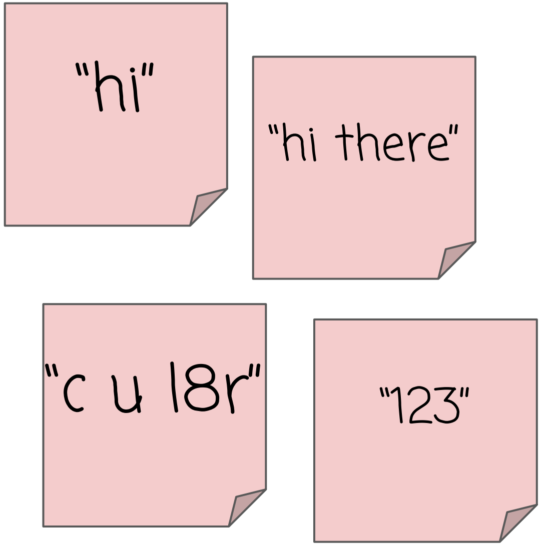
Numbers

Made of the digits 0...9
No quotes
Yellow sticky



Strings

Made of any characters
Inside double quotes
Red sticky

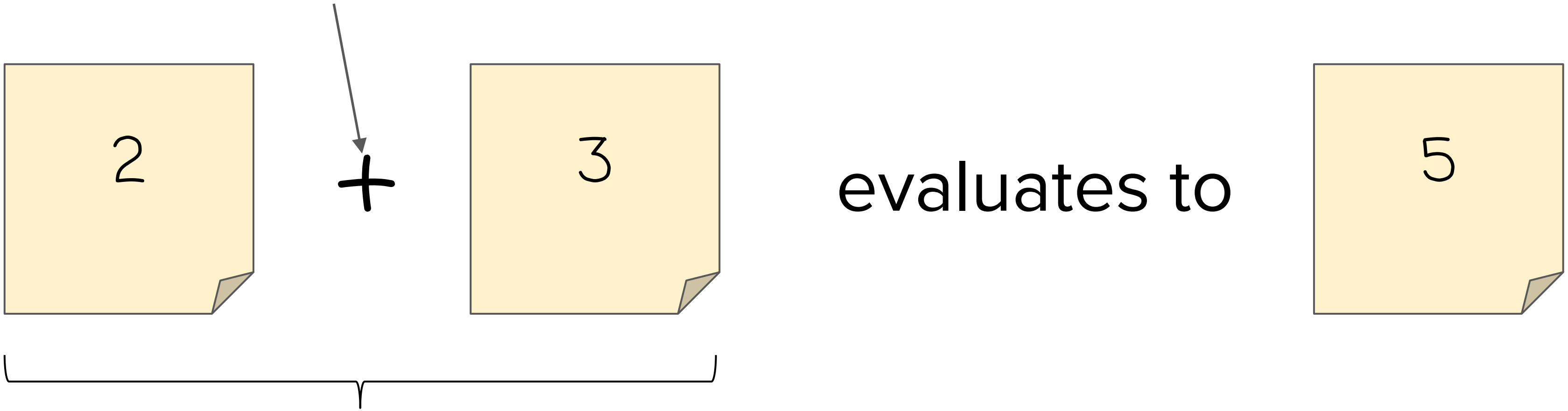


Do This:
Make one number and one string.
Share it at your table.

Unit 4 Lesson 1 - Activity

Operators

Fancy name for + - * /



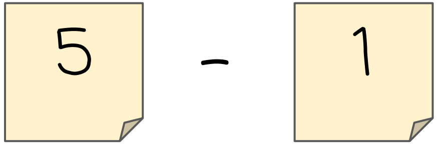
evaluates to

Expression

Combination of operators and values

Evaluates to single value

Do This: Evaluate this expression



Unit 4 Lesson 1 - Activity

3 + 4 evaluates to 7

5 - 2 evaluates to 3

11 * 2 evaluates to 22

10 / 2 evaluates to 5

"for" + "ever" evaluates to "forever"

"gr" + 8 evaluates to "gr8"

2 + "day" evaluates to "2day"

If you're using one or two strings, you can only use the + operator. The others don't make sense!

Do This: Evaluate these expressions. Pay attention to what color stickies you create and if you use quotes.

4 + 5 | 9

10 - 9 | 1

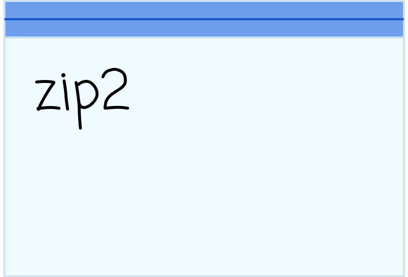
"tree" + "house" | "treehouse"

"you" + "r" | "your"

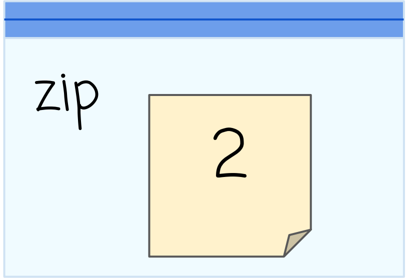
3 + "D" | "3D"

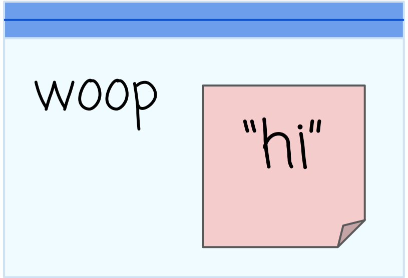
Variables

- Plastic baggies
- Can hold at most one value
- Name uses no quotes, includes no spaces, and must start with a letter



Do This:
Make one variable with any name you like. Share it with another group.

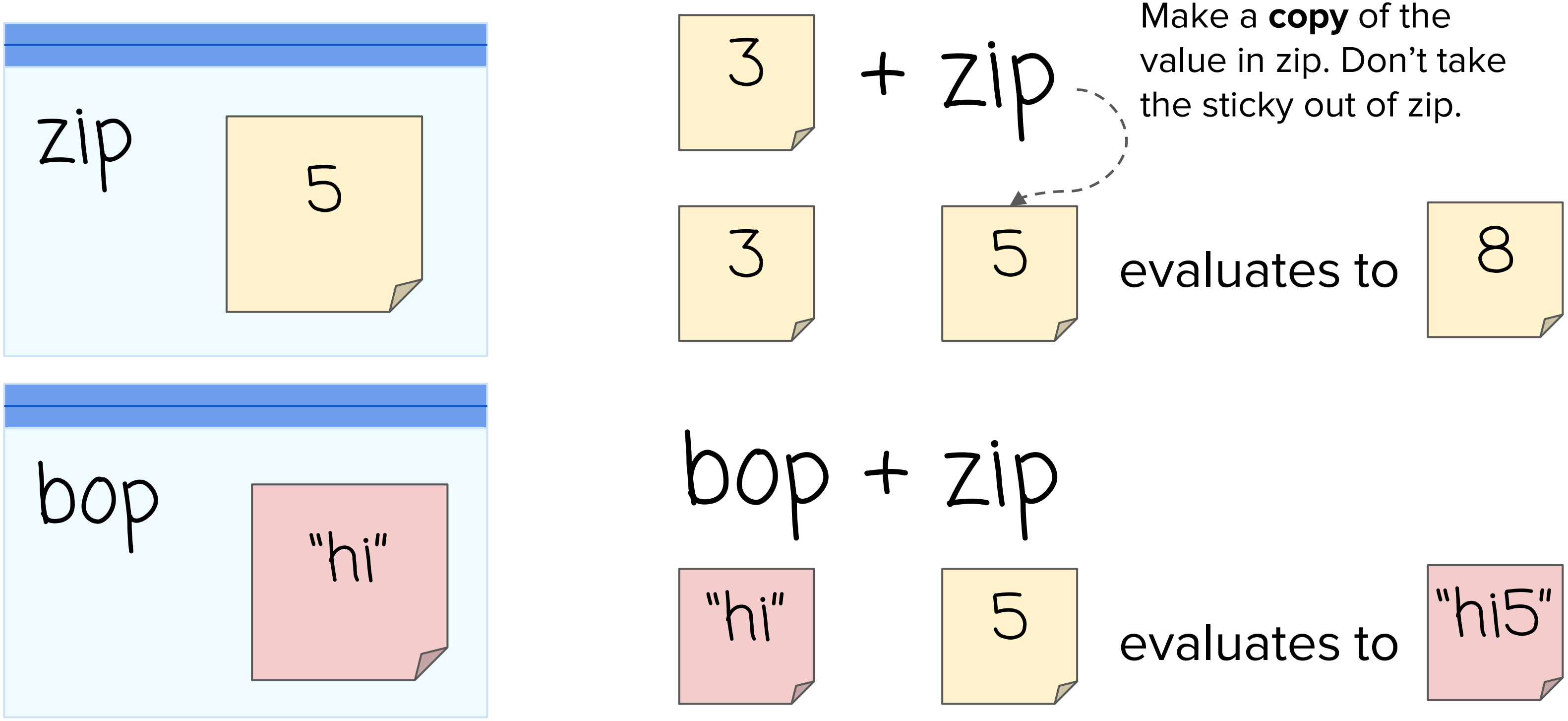




Variables and Expressions

Replace variable name with a copy of the value it holds

Evaluate the expression as normal

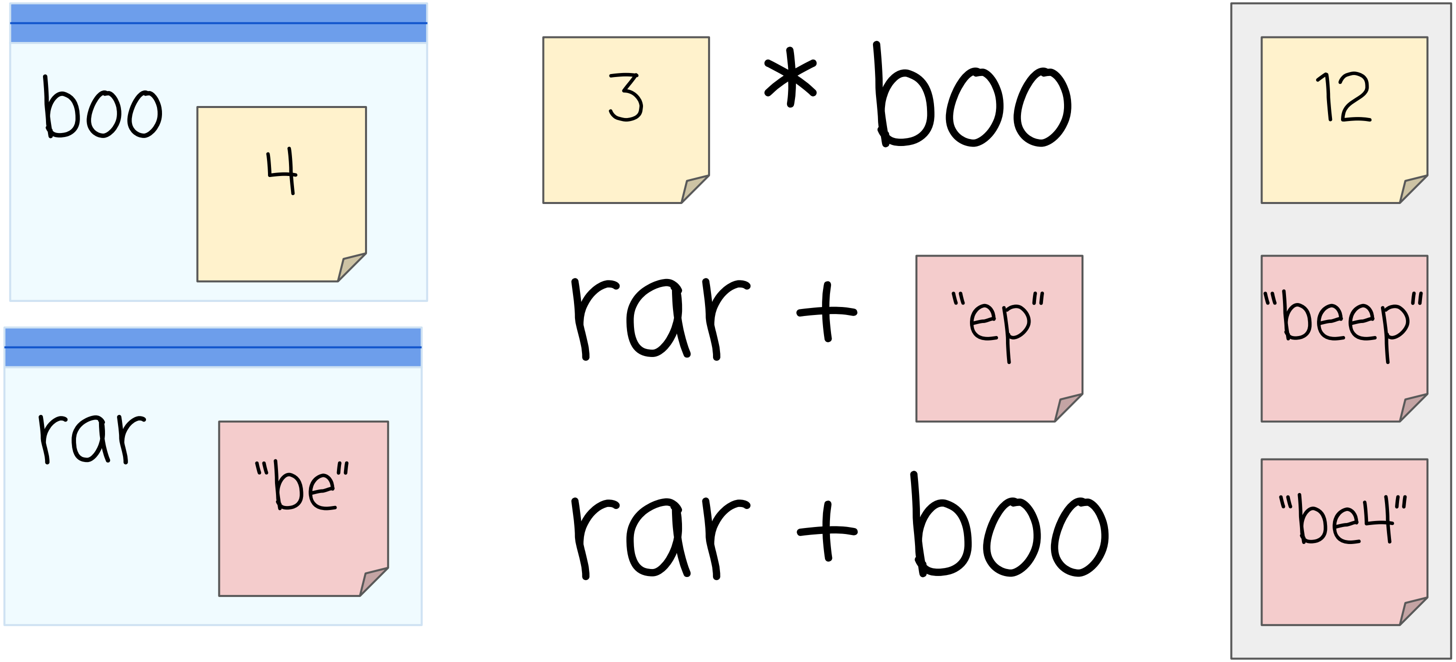


Make a **copy** of the value in zip. Don't take the sticky out of zip.

evaluates to

evaluates to

Do This: Evaluate these expressions. Make sure you pay attention to whether it evaluates to a string or a number.



Let’s start writing programs
that control our variables.

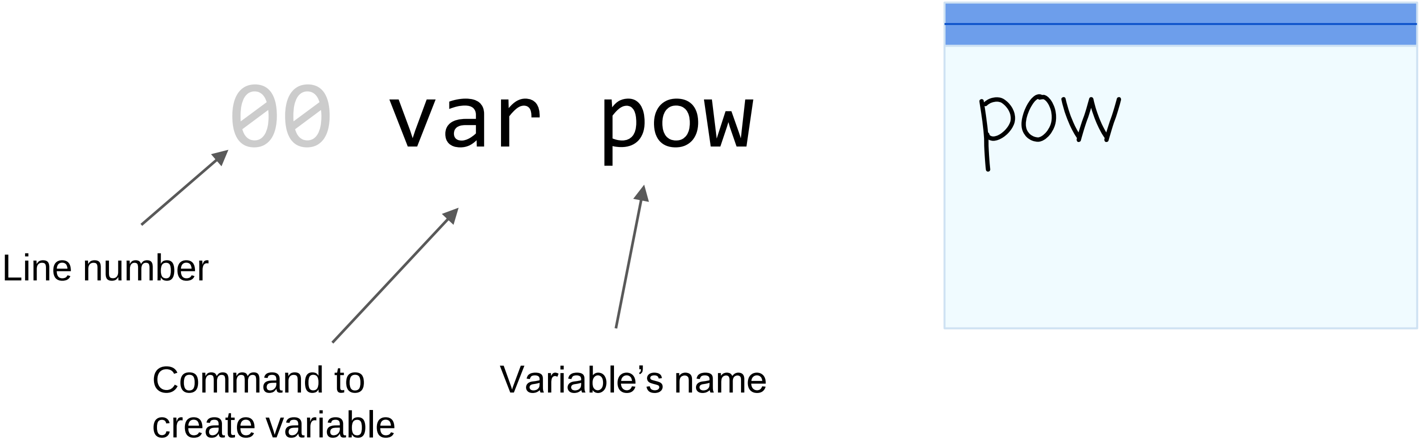
We’re going to stop using stickies but will highlight strings and numbers to help you remember the difference.

var

Creates a new variable

Grab a new baggie

Write the variable's name on the baggie



```
00 var pow
```

Line number

Command to create variable

Variable's name

Do This: Run this program

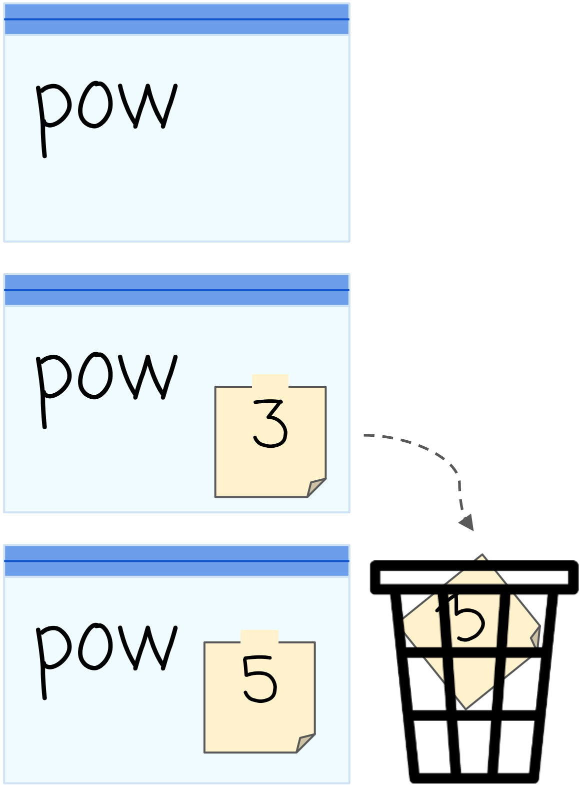
“Assignment operator”

“Assign”: a fancy name for putting a value inside the baggie.

Variables can only hold one stickie. If there’s already a sticky note in there, throw it away.

“pow gets 3” and “pow gets 5”

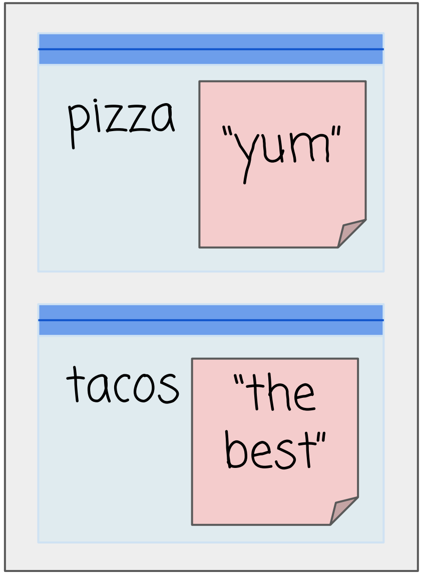
```
var pow
pow ← 3
pow ← 5
```



Do This:

Run this program. Compare your result with another group.

```
var pizza
pizza ← 3
var tacos
pizza ← “yum”
tacos ← “the best”
```



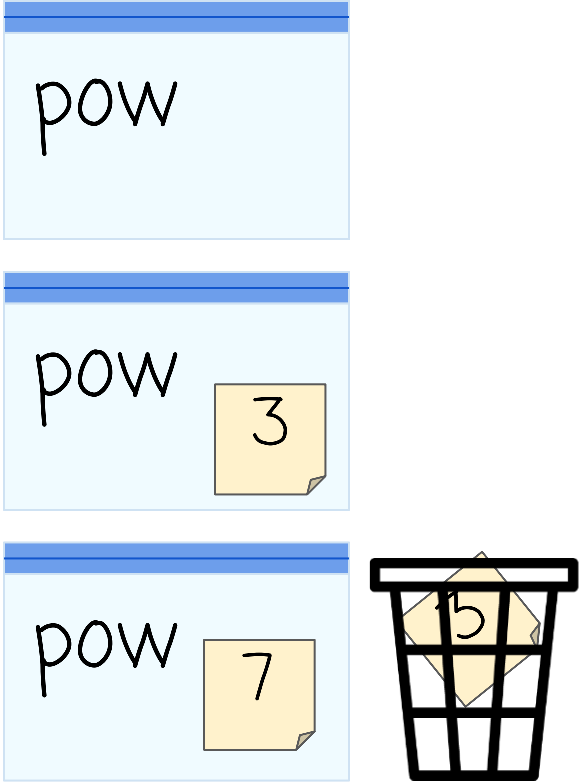
Assign a Variable with Expression

Evaluate the expression first to get one value.

Assign the value as normal

```
var pow

pow ← 1 + 2

pow ← 3 + 4
```

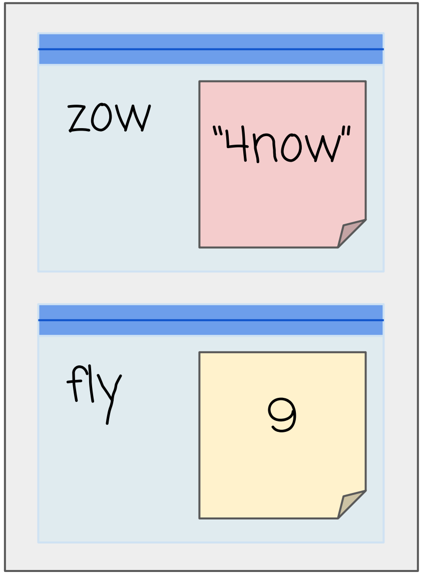
Evaluate expression first



Do This:

Run this program. Compare your result with another group.

```
var zow
var fly
fly ← “to” + “day”
zow ← 4 - 1
fly ← 3 * 3
zow ← 4 + “now”
```

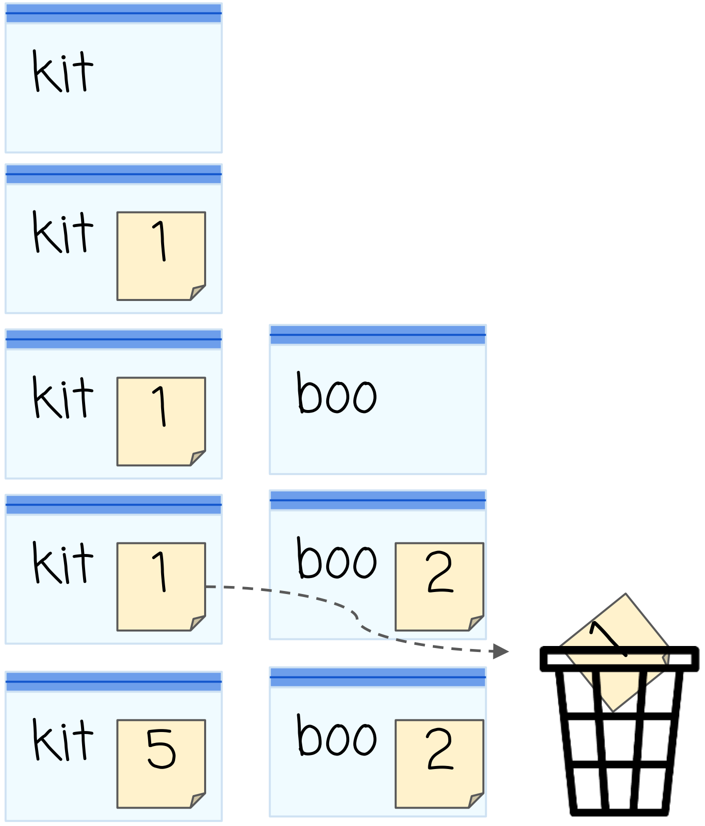


We’re not going to highlight our strings and numbers anymore. We can just use double quotes around the strings to tell the difference.

Assign a Variable: Expressions with Variables

Evaluate the expression on the right first to get one value.
Assign the value as normal

```
var kit
kit ← 1
var boo
boo ← kit + 1
kit ← 5
```

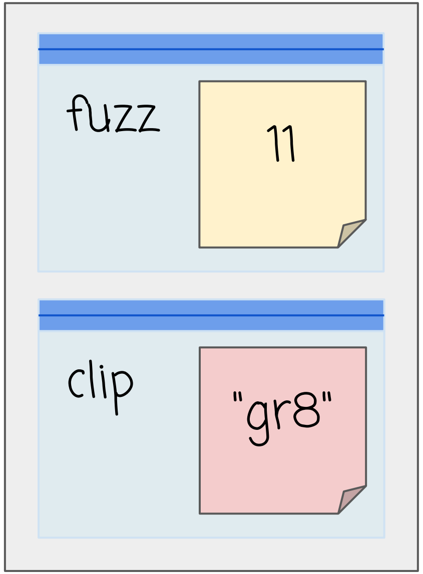


Note: Variables aren’t “connected”. Changing kit doesn’t change boo.

Do This:

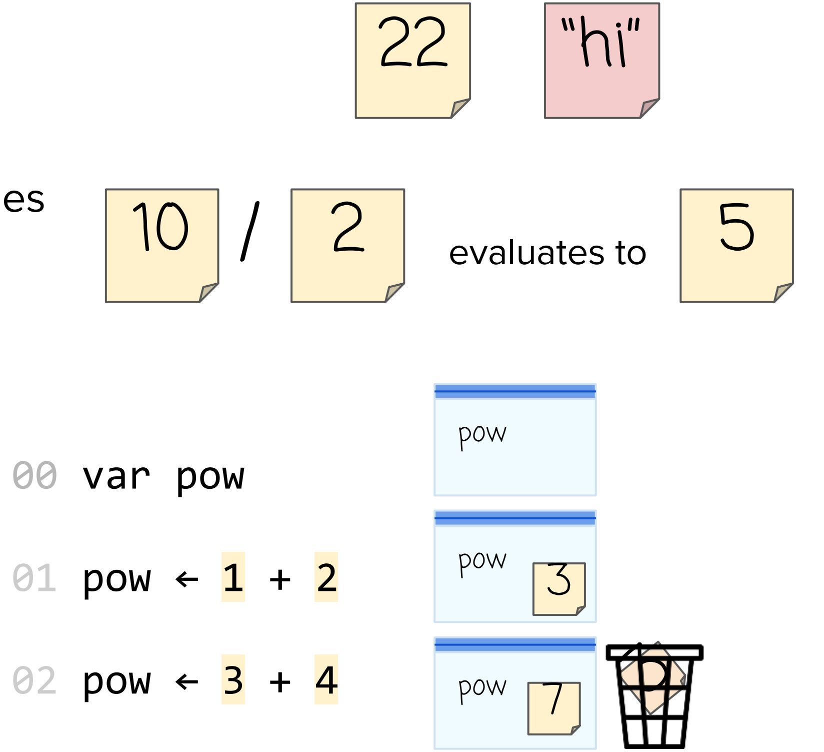
Run this program. Compare your result with another group.

```
var fuzz
var clip
fuzz ← 5
clip ← fuzz + 2
fuzz ← clip + 1
clip ← “gr” + fuzz
fuzz ← fuzz + 1
fuzz ← fuzz + 1
fuzz ← fuzz + 1
```



Key Takeaways

- Numbers and strings are two different types of values
- Expressions evaluate to a single new value
- When variables are in the expression just make a copy, don't change the actual variable.
- Variables are "assigned" a new value
- Evaluate first, then assign
- Old values are deleted forever.
- Assignment just moves information around. It does not "connect" variables.



```
var pow
pow ← 1 + 2
pow ← 3 + 4
```

In some languages (including Javascript) the assignment operator is not written

←

it is written as

=

So the command

```
fuzz ← fuzz + 1
```

it is written as

```
fuzz = fuzz + 1
```

In math = means “are equal forever”
In programming = means “put this value in this variable”

We’ll see this more next time.

Wrap Up

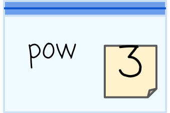
●●●

Unit 4 Lesson 1 - Wrap Up

Expression: a combination of operators and values that evaluates to a single value

```
pow ← 1 + 2
```

Variable: holds one value at a time



Assignment Operator: allows a program to change the value represented by a variable

← OR =

Unit 4 - Lesson 2
Variables Investigate

Warm Up

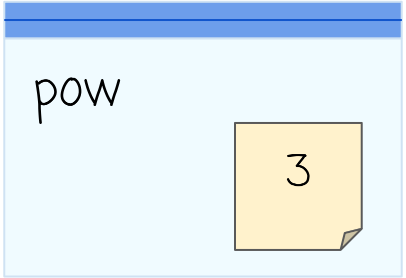
●○○

Unit 4 Lesson 2 - Warm Up

Prompt:

Let's do a quick review.

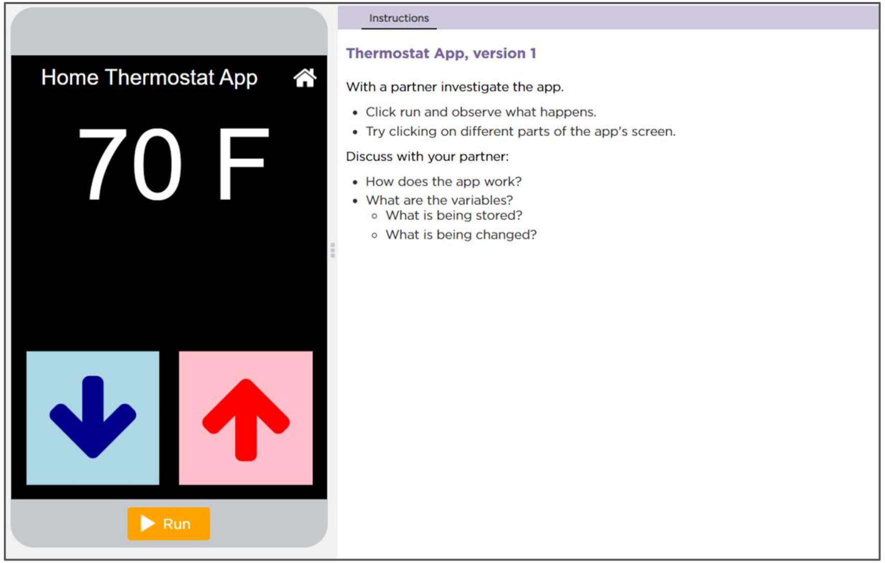
How does a baggy represent a variable?



Activity

●●○

Variables Investigate



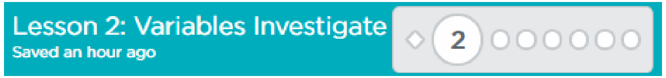
Thermostat App, version 1

With a partner investigate the app.

- Click run and observe what happens.
- Try clicking on different parts of the app's screen.

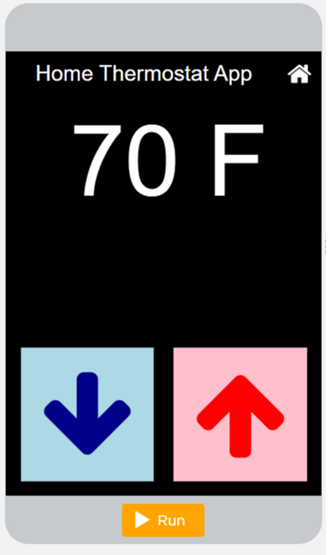
Discuss with your partner:

- How does the app work?
- What are the variables?
  - What is being stored?
  - What is being changed?



Lesson 2: Variables Investigate
Saved an hour ago

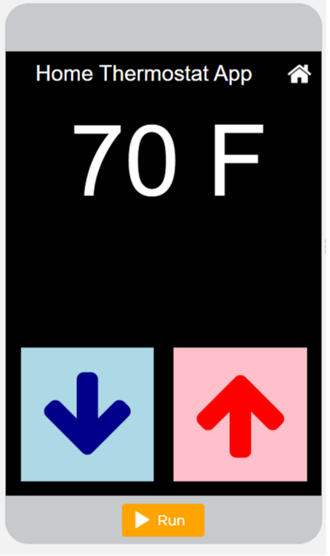
Level 2



Do This:

- Run the app
- Predict the information that is being stored in variables

Level 3

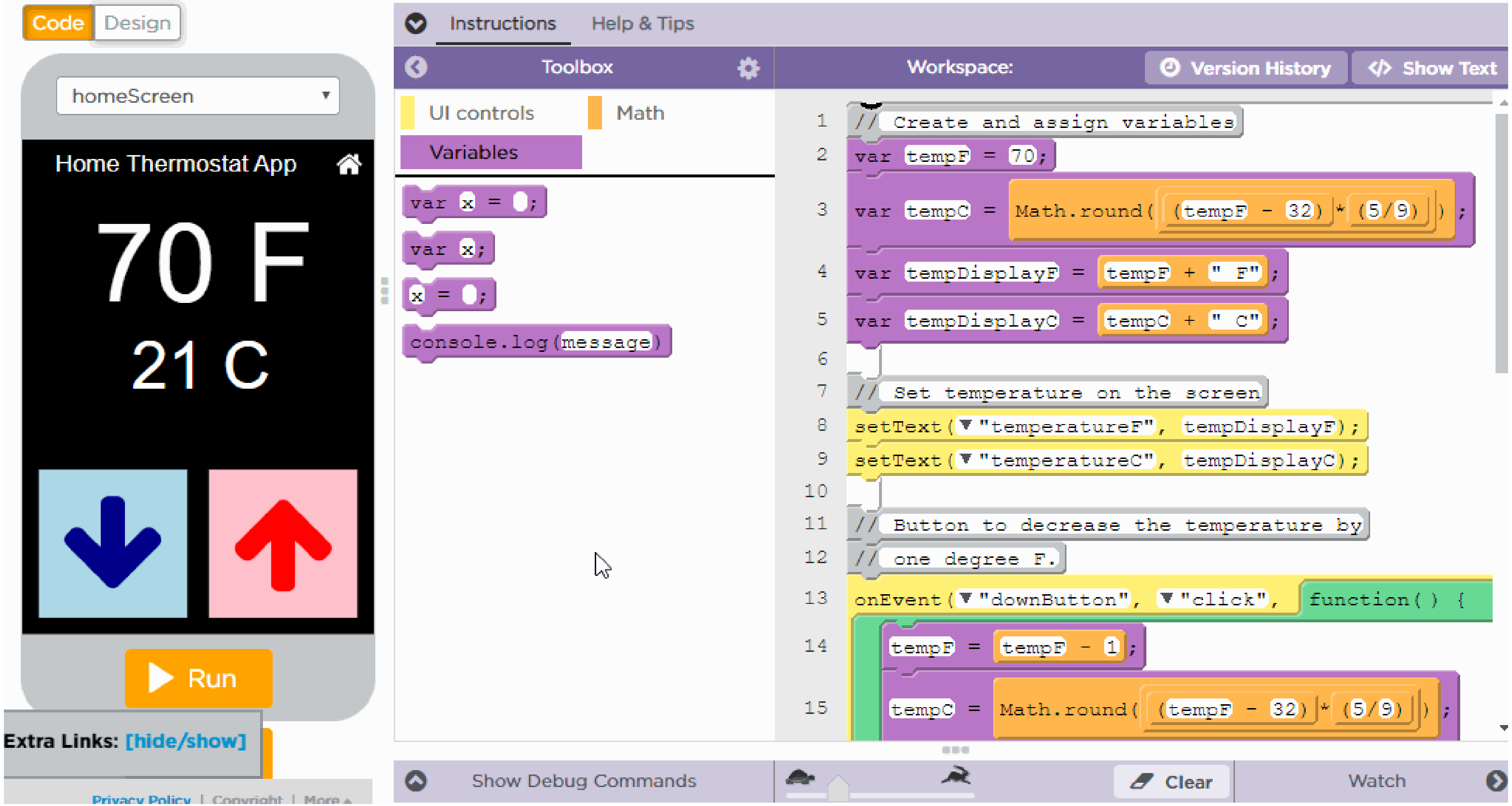


Do This:

- **Read Code:** Read the code for your assigned section making sure you understand how it works.

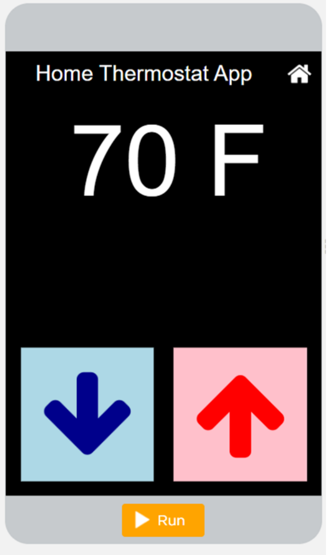
- **Explain Your Section:** Make a group with partners who investigated the other section. Carefully explain how your section works line by line.

Adding a Watcher



```
// Create and assign variables
var tempF = 70;
var tempC = Math.round((tempF - 32) * (5/9));
var tempDisplayF = tempF + " F";
var tempDisplayC = tempC + " C";

// Set temperature on the screen
setText("temperatureF", tempDisplayF);
setText("temperatureC", tempDisplayC);

// Button to decrease the temperature by
// one degree F.
onEvent("downButton", "click", function() {
  tempF = tempF - 1;
  tempC = Math.round((tempF - 32) * (5/9));
```

Level 3

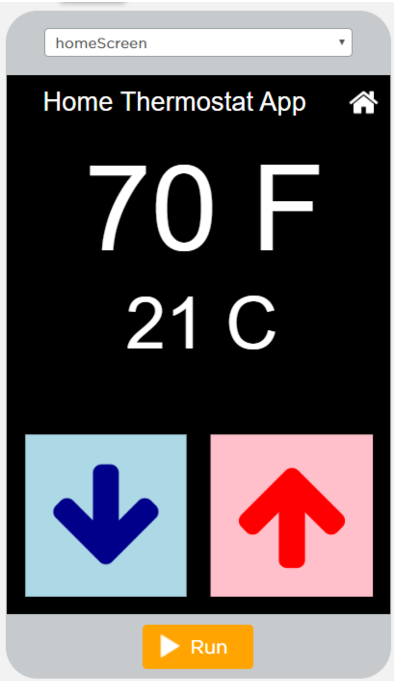


Do This:

- **Modify:** Change the degrees by two when the up and down arrow are clicked.

Unit 4 Lesson 2 - Activity

Level 4

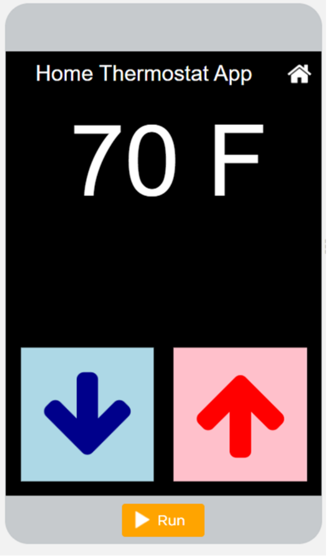


Do This:

- Run the app.
- Discuss changes.
- Find the Math.round command. Discuss with a partner how this command might work.

- **Modify:** Change the code so that no space displays between the temperature and the unit descriptions ("F" or "C").

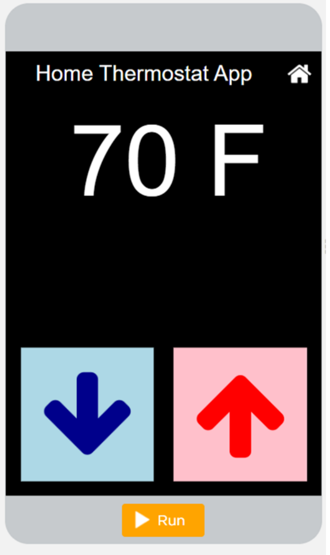
Level 5



Do This:

- Run the app
- Predict the information that is being stored in variables

Level 6

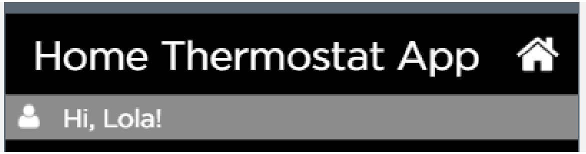


Do This:

- **Read Code:** Read the code and discuss with your partner what has changed and what has stayed the same.

- **Class Discussion:** How does `getText()` work?

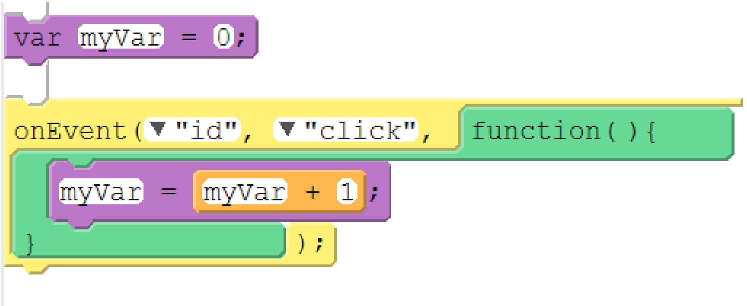
- **Modify:** Add an explanation point `"!"` to the end of the string stored in userName so the following is displayed on the screen:


Home Thermostat App

Hi, Lola!

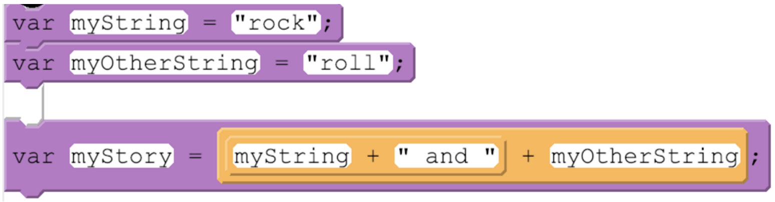
Counter Pattern with Event

Name	Code (Block)	Code (Text)

Counter Pattern with Event	
	var myVar = 0;
	onEvent("id", "click", function(){
	myVar = myVar + 1;
	});
		var myVar = 0;
		onEvent("id", "click", function() {
		  myVar = myVar + 1;
		});

Variable with String Concatenation

Name	Code (Block)	Code (Text)

Variable with String Concatenation	
	var myString = "rock";
	var myOtherString = "roll";
	var myStory = myString + " and " + myOtherString;
		var myString = "rock";
		var myOtherString = "roll";
		
		var myStory = myString + " and " + myOtherString;

Wrap Up

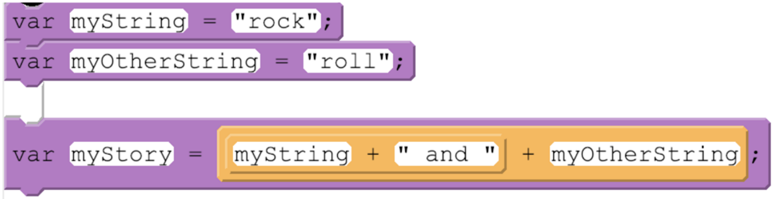
Unit 4 Lesson 2 - Wrap Up

Prompt:

Let's review. What can be stored in a variable? Why is using a meaningful name for the variable important?

Unit 4 Lesson 2 - Wrap Up

Variable: a reference to a value or expression that can be used repeatedly throughout a program.



```
var myString = "rock";
var myOtherString = "roll";

var myStory = myString + " and " + myOtherString;
```

Unit 4 - Lesson 3
Variables Practice

Warm Up

● ○ ○

Activity

●●○

Unit 4 Lesson 3 - Activity

Debugging: the process of finding and fixing problems in code

Describe

The Problem

What do you expect it to do?

What does it actually do?

Does it always happen?

Hunt

For Bugs

Are there warnings or errors?

What did you change most recently?

Explain your code to someone else

Look for code related to the problem

Try

Solutions

Make a small change

Document

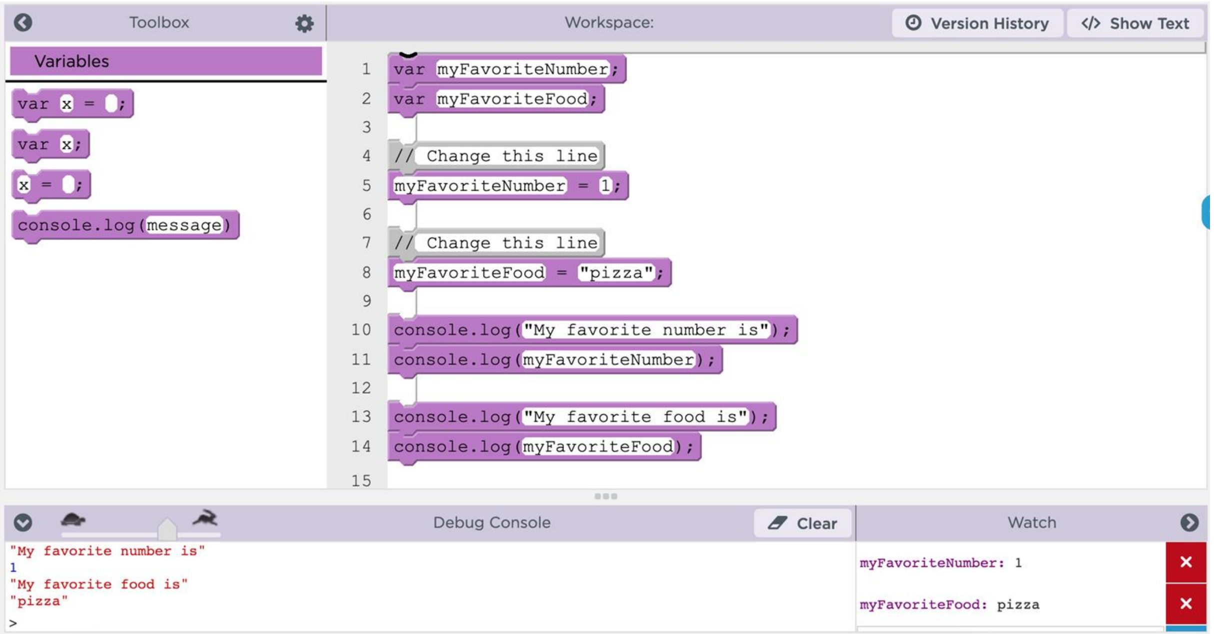
As You Go

What have you learned?

What strategies did you use?

What questions do you have?

Specific Debugging Skills for the Day



```
var x = ;
var x;
x = ;
console.log(message)
```

```
var myFavoriteNumber;
var myFavoriteFood;

// Change this line
myFavoriteNumber = 1;

// Change this line
myFavoriteFood = "pizza";

console.log("My favorite number is");
console.log(myFavoriteNumber);

console.log("My favorite food is");
console.log(myFavoriteFood);
```

```
"My favorite number is"
1
"My favorite food is"
"pizza"
>
```

```
myFavoriteNumber: 1
myFavoriteFood: pizza
```

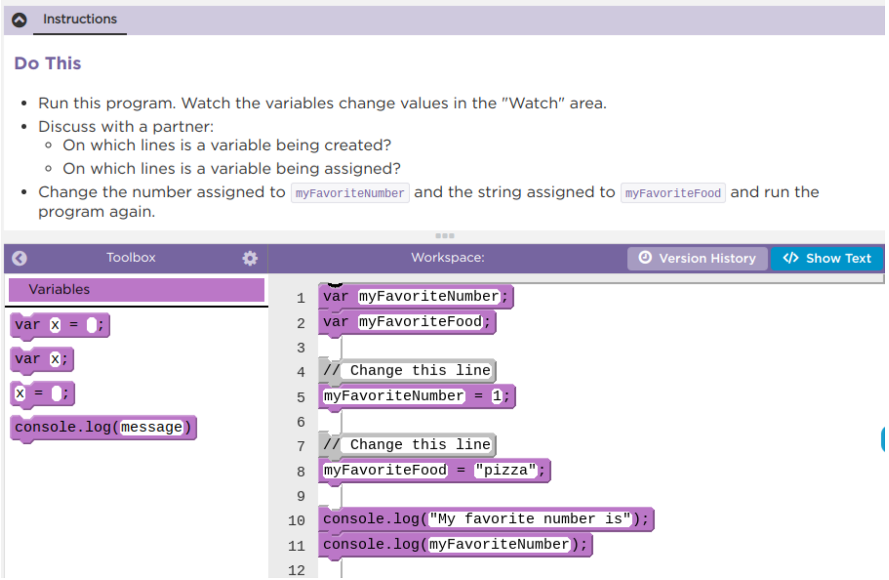
1. Slow down code with the speed slider

2. Use console.log to get output

3. Use Watchers to see your variables change values

Unit 4 Lesson 3 - Activity

Variables Practice

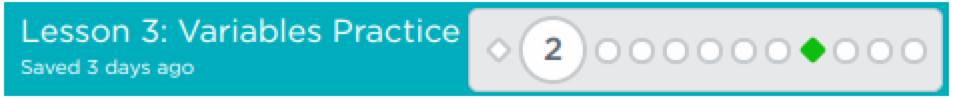


Do This

- Run this program. Watch the variables change values in the "Watch" area.
- Discuss with a partner:
  - On which lines is a variable being created?
  - On which lines is a variable being assigned?
- Change the number assigned to `myFavoriteNumber` and the string assigned to `myFavoriteFood` and run the program again.

```
var x = ;
var x;
x = ;
console.log(message)
```

```
1   var myFavoriteNumber;
2   var myFavoriteFood;
3
4   // Change this line
5   myFavoriteNumber = 1;
6
7   // Change this line
8   myFavoriteFood = "pizza";
9
10  console.log("My favorite number is");
11  console.log(myFavoriteNumber);
12
```



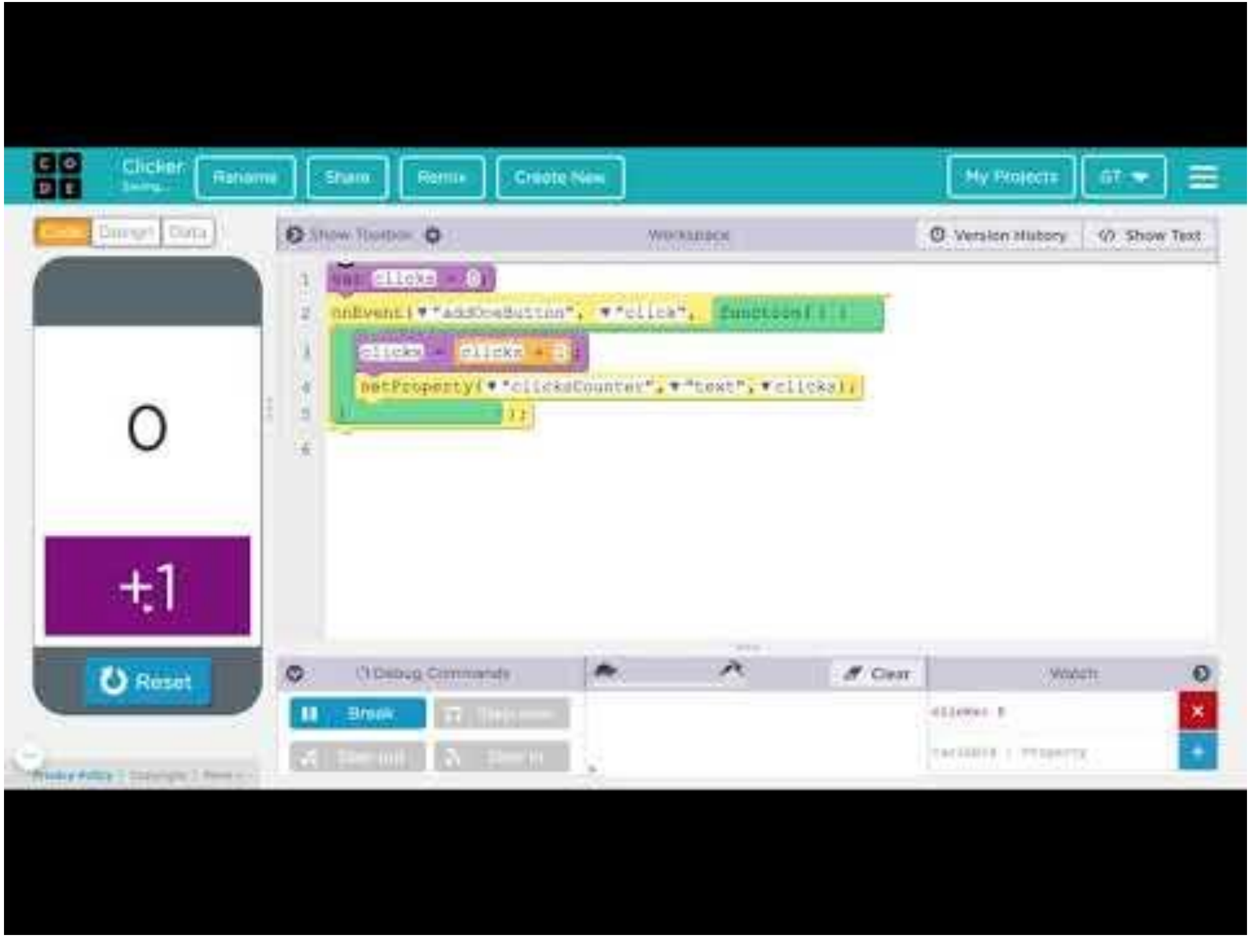
Lesson 3: Variables Practice

Saved 3 days ago

Do This:

- Navigate to Lesson 3, Level 2 on Code Studio
- Complete Levels 2-8

Unit 4 Lesson 3 - Activity



```
var clicks = 0;
onEvent("addOneButton", "click", function( ) {
  clicks = clicks + 1;
  setProperty("clicksCounter", "text", clicks);
});
```

Video
Download Link

Create Variables Once, At the Top, Outside Functions or onEvent()

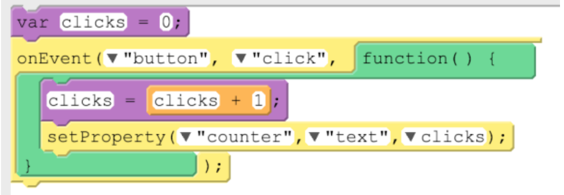
When you create variables you should:

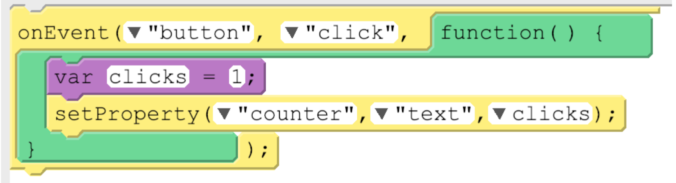
- **Use `var` only once.** You don't need to create variables twice and this can cause errors.
- **Create your variables at the top of your program.** This keeps your code organized and easier to read for you and others.
- **Create your variables outside any `function` or `onEvent()` bocks.**

Global vs. Local Variables

There's two types of variables, global and local, and so far we've only used global variables. Here's the main difference between global and local variables.

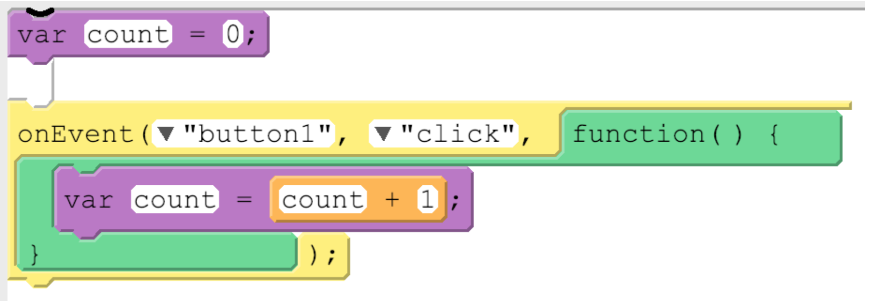
Type of Variable	How It Works	How Created	Picture

Global	Permanent. Can be used anywhere in your code.	var used outside an onEvent()	
			`var clicks = 0;`
			`onEvent(▼"button", ▼"click", function() {`
			`  clicks = clicks + 1;`
			`  setProperty(▼"counter", ▼"text", ▼clicks);`
			`});`

Local	Temporary. Can be used only in the part of the code where it was created, like inside an onEvent(). Deleted once the onEvent() is done running.	var used inside an onEvent()	
			`onEvent(▼"button", ▼"click", function() {`
			`  var clicks = 1;`
			`  setProperty(▼"counter", ▼"text", ▼clicks);`
			`});`

Avoiding Local Variables and Debugging

Local variables will eventually be useful but for now they're most likely to just be confusing. The biggest issue you'll run into right now with local variables is accidentally using `var` inside of an `onEvent()` or `function`. Here's what the code usually looks like:



```
var count = 0;

onEvent(▼"button1", ▼"click", function( ) {
  var count = count + 1;
});
```

This code is pretty confusing. While it looks like there's only one variable being used, it actually has two variables, one local and one global, and they're both named count! Changing the value on one will have no impact on the other. This can cause unexpected behavior in your code and it can get tricky to debug.

The best way to avoid these issues is to **make sure for now that you're not using** `var` **inside of an** `onEvent()` **or** `function`. If you run into a tricky debugging problem, check if you're accidentally creating a local variable.

Do This:

- Review Lesson 9
- Finish Levels 10-11

Wrap Up

●●●

Unit 4 Lesson 3 - Wrap Up

Prompt:

What aspects of working with variables do you feel like clicked today? What do you still feel like you have trouble with?

Unit 4 - Lesson 4
Variables Make

Warm Up

● ○ ○

Activity

●●○

Variables Make: Photo Liker

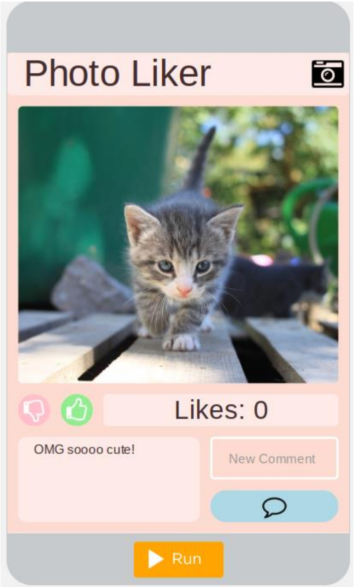
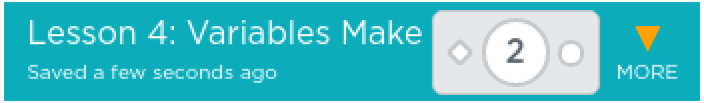


Photo Liker

Likes: 0

OMG soooo cute!



Lesson 4: Variables Make

Saved a few seconds ago

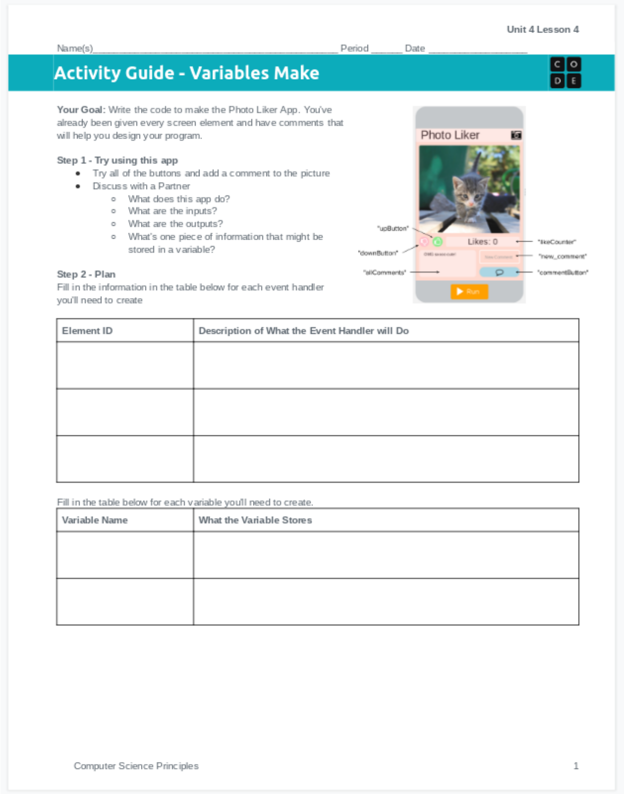
Do This:

- Navigate to Lesson 4, Level 2 on Code Studio

Prompt: Try all of the buttons and add a comment to the picture

- What does this app do?
- What are the inputs?
- What are the outputs?
- What's one piece of information that might be stored in a variable?

Do This: Make the Photo Liker!



Unit 4 Lesson 4

Name(s)_______________________________________ Period ______ Date __________________

Activity Guide - Variables Make

Your Goal: Write the code to make the Photo Liker App. You've already been given every screen element and have comments that will help you design your program.

Step 1 - Try using this app

- Try all of the buttons and add a comment to the picture
- Discuss with a Partner
  - What does this app do?
  - What are the inputs?
  - What are the outputs?
  - What's one piece of information that might be stored in a variable?

Step 2 - Plan

Fill in the information in the table below for each event handler you'll need to create

Element ID	Description of What the Event Handler will Do

Fill in the table below for each variable you'll need to create.

Variable Name	What the Variable Stores

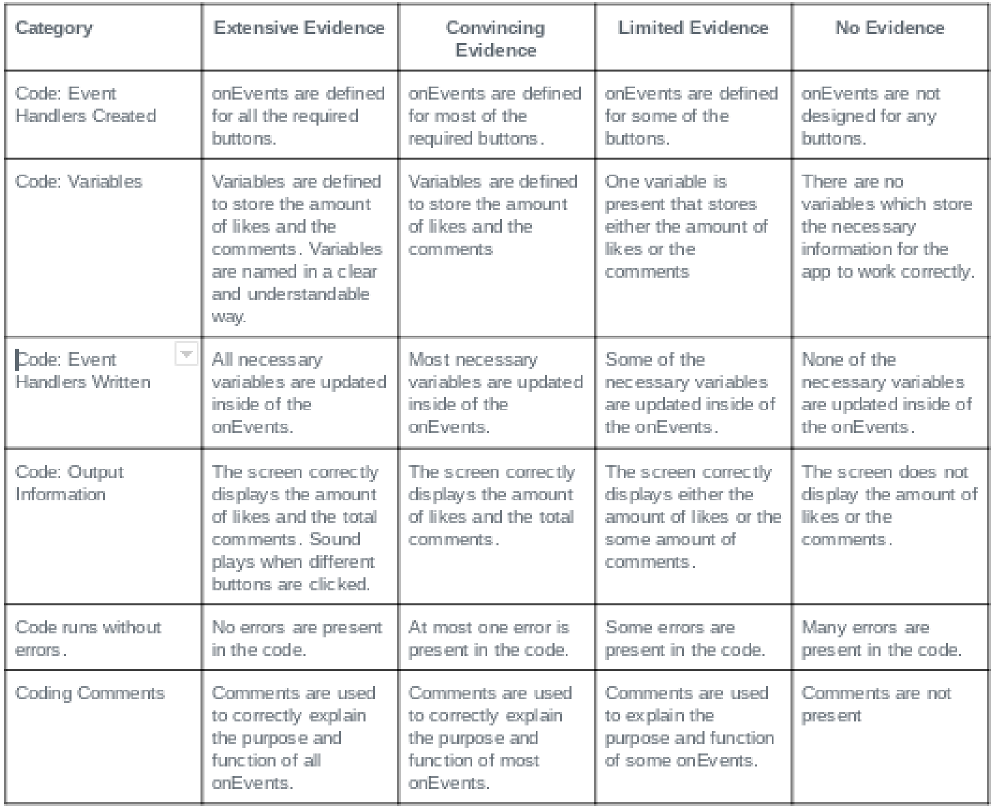
Computer Science Principles 1

Use the activity guide to plan out your code, including the event handlers and variables you'll need to create.

Step 3 includes steps you can follow to build the app, or you can use your own process.

Unit 4 Lesson 11 - Activity

Don't forget to check the rubric before hitting submit!



Category	Extensive Evidence	Convincing Evidence	Limited Evidence	No Evidence
Code: Event Handlers Created	onEvents are defined for all the required buttons.	onEvents are defined for most of the required buttons.	onEvents are defined for some of the buttons.	onEvents are not designed for any buttons.
Code: Variables	Variables are defined to store the amount of likes and the comments. Variables are named in a clear and understandable way.	Variables are defined to store the amount of likes and the comments	One variable is present that stores either the amount of likes or the comments	There are no variables which store the necessary information for the app to work correctly.
Code: Event Handlers Written	All necessary variables are updated inside of the onEvents.	Most necessary variables are updated inside of the onEvents.	Some of the necessary variables are updated inside of the onEvents.	None of the necessary variables are updated inside of the onEvents.
Code: Output Information	The screen correctly displays the amount of likes and the total comments. Sound plays when different buttons are clicked.	The screen correctly displays the amount of likes and the total comments.	The screen correctly displays either the amount of likes or the some amount of comments.	The screen does not display the amount of likes or the comments.
Code runs without errors.	No errors are present in the code.	At most one error is present in the code.	Some errors are present in the code.	Many errors are present in the code.
Coding Comments	Comments are used to correctly explain the purpose and function of all onEvents.	Comments are used to correctly explain the purpose and function of most onEvents.	Comments are used to explain the purpose and function of some onEvents.	Comments are not present

Submit

Wrap Up

Great job today!

Unit 4 - Lesson 5
Conditionals Explore

Warm Up

●○○

Unit 4 Lesson 5 - Warm Up

Prompt:

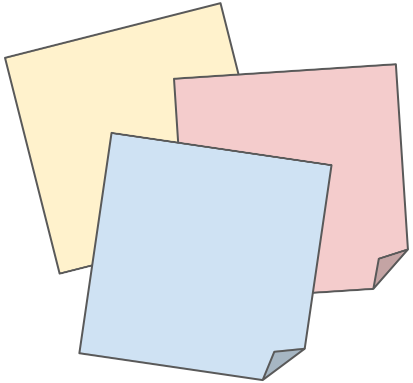
Imagine you want to make a decision about what to wear to an event. Name two pieces of information you'd want. How would you use them in your decision?

Activity

●●○

Conditionals Explore

You and your partner should have:
Small stacks of red, yellow, and blue stickies
At least three baggies
Pen/Pencil
Dry Erase Marker



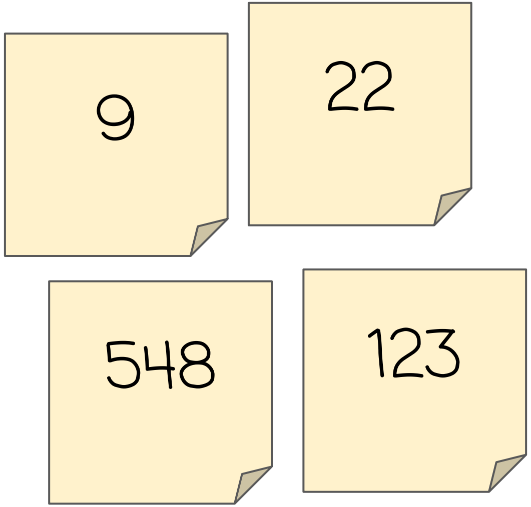
Information can be stored as...

Numbers

Made of the digits 0...9

No quotes

Yellow sticky

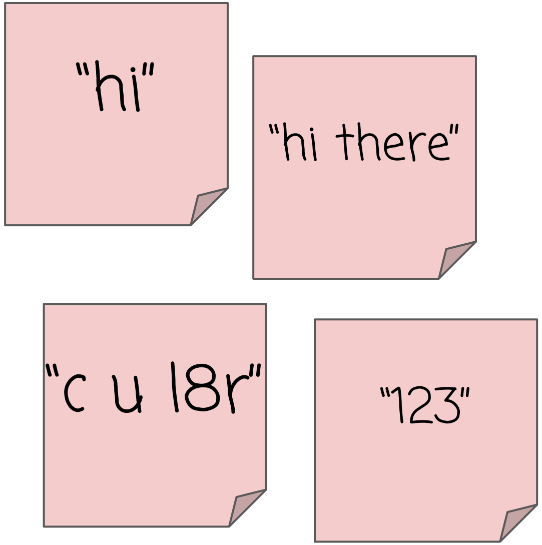


Strings

Made of any characters

Inside double quotes

Red sticky

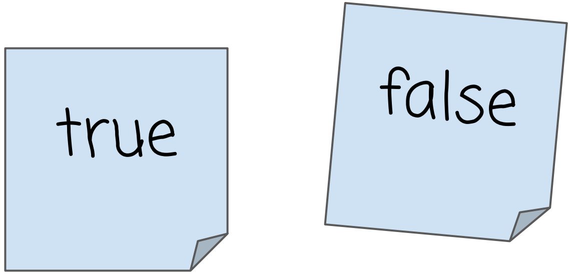


Information can be also be stored as a...

Boolean

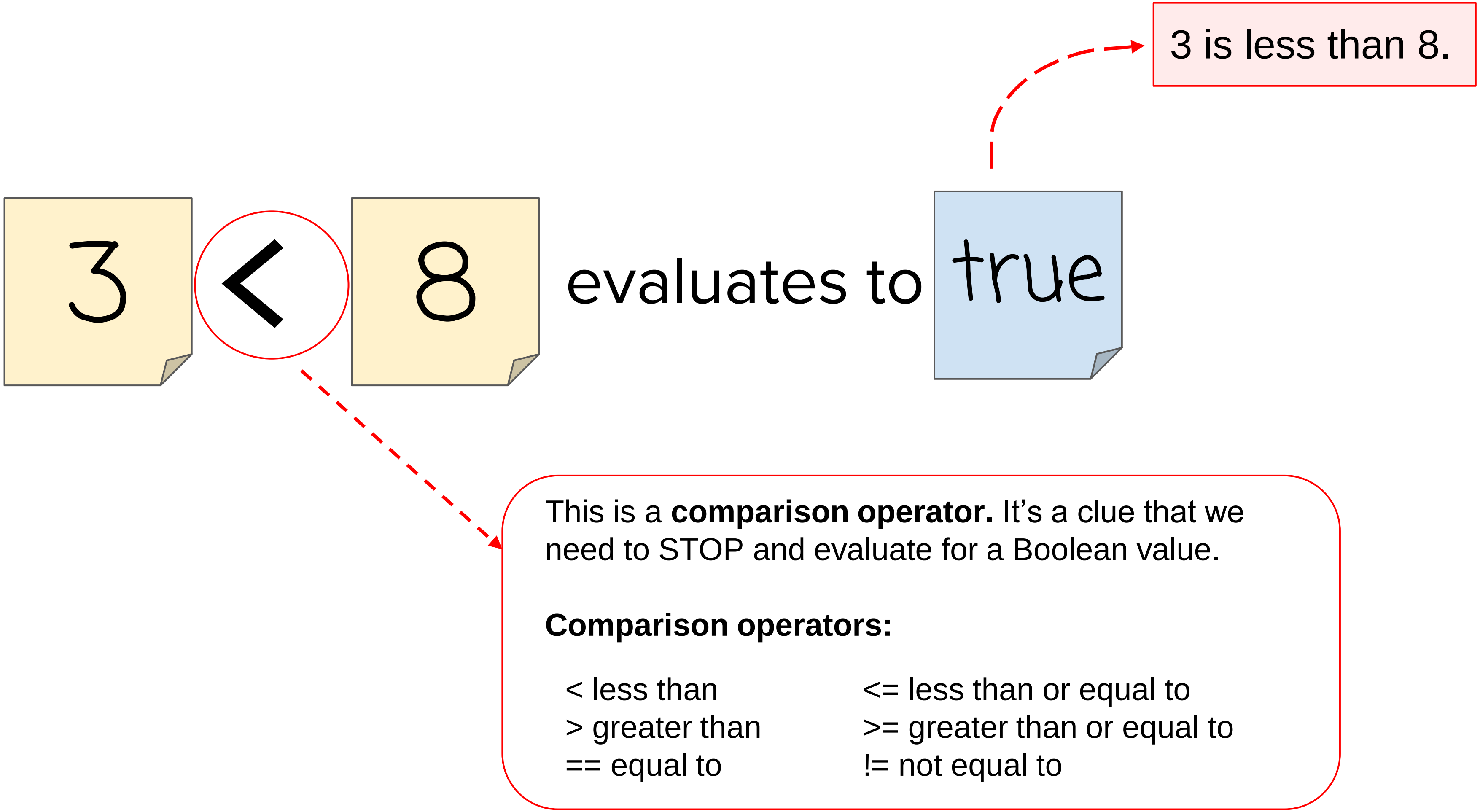
true or false

blue sticky



Do This:

Write true on one sticky note and false on another.



3 is less than 8.

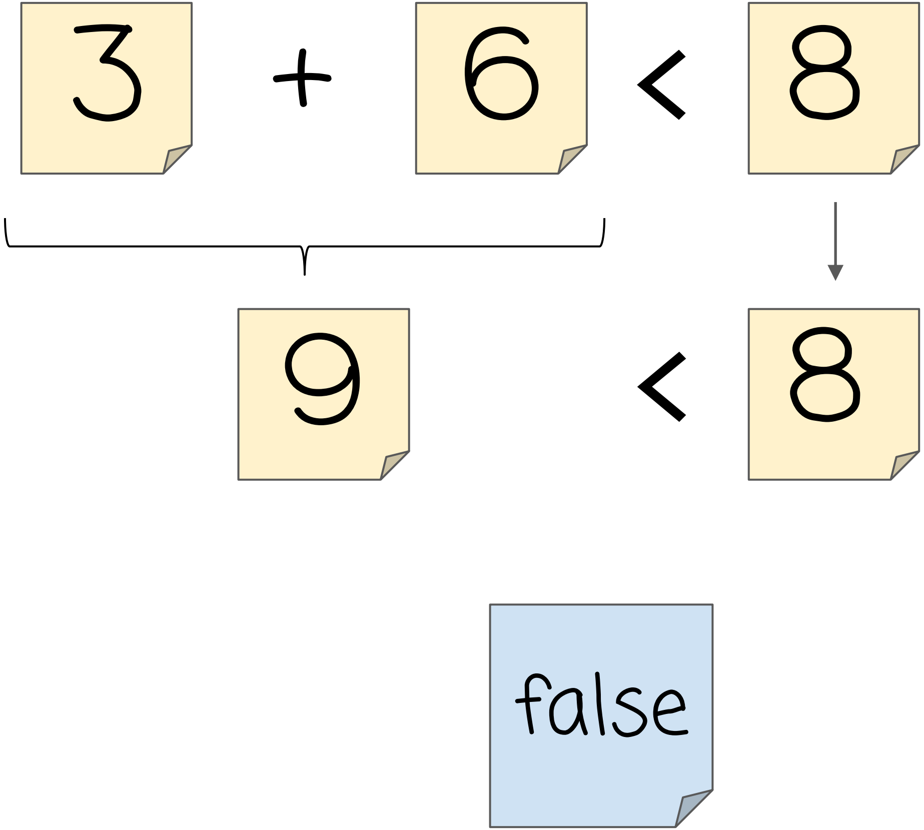
This is a **comparison operator.** It’s a clue that we need to STOP and evaluate for a Boolean value.

Comparison operators:

< less than	<= less than or equal to
> greater than	>= greater than or equal to
== equal to	!= not equal to

Unit 4 Lesson 5 - Activity

Each side of the comparison operator must be reduced to one sticky note before we can compare

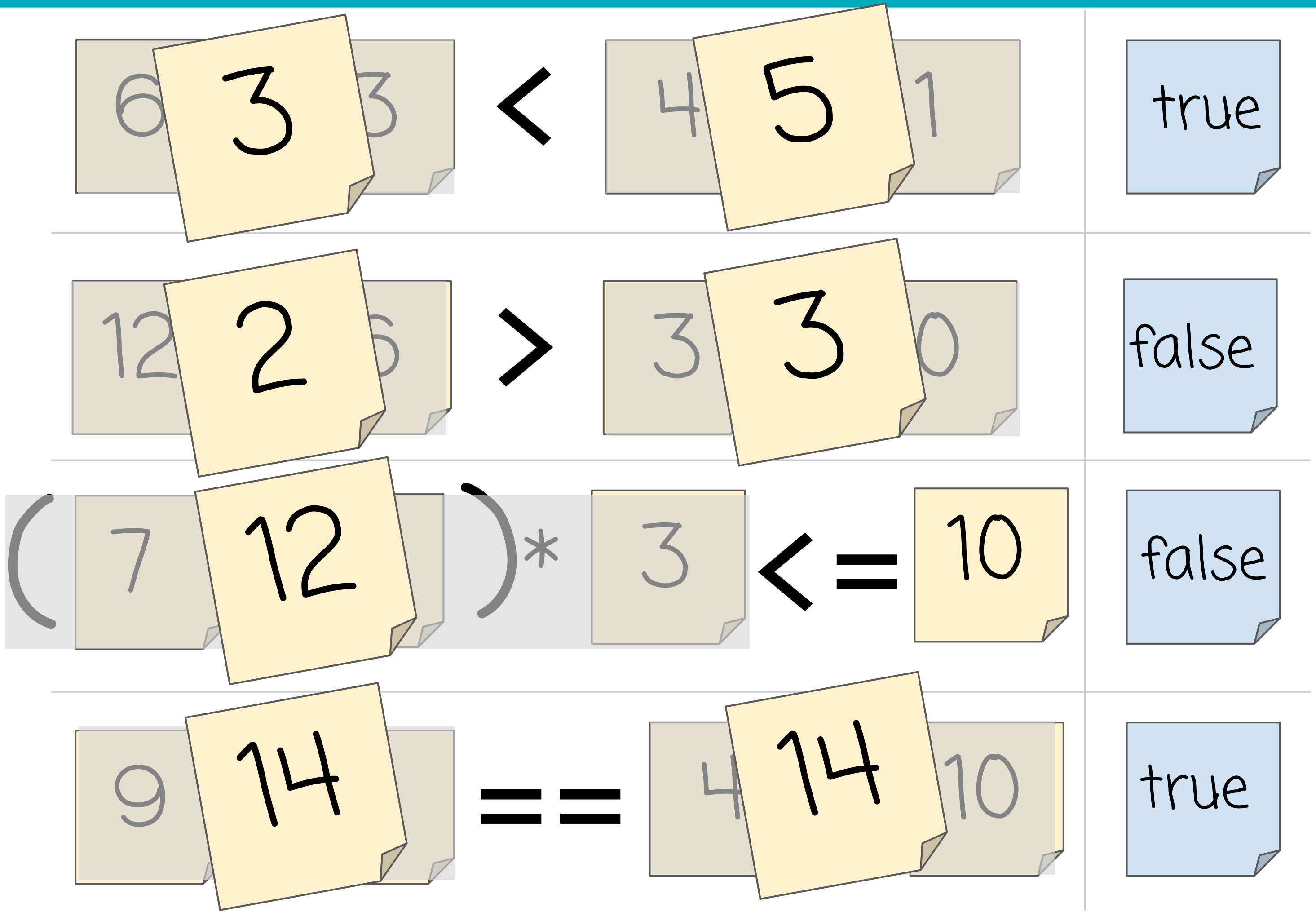


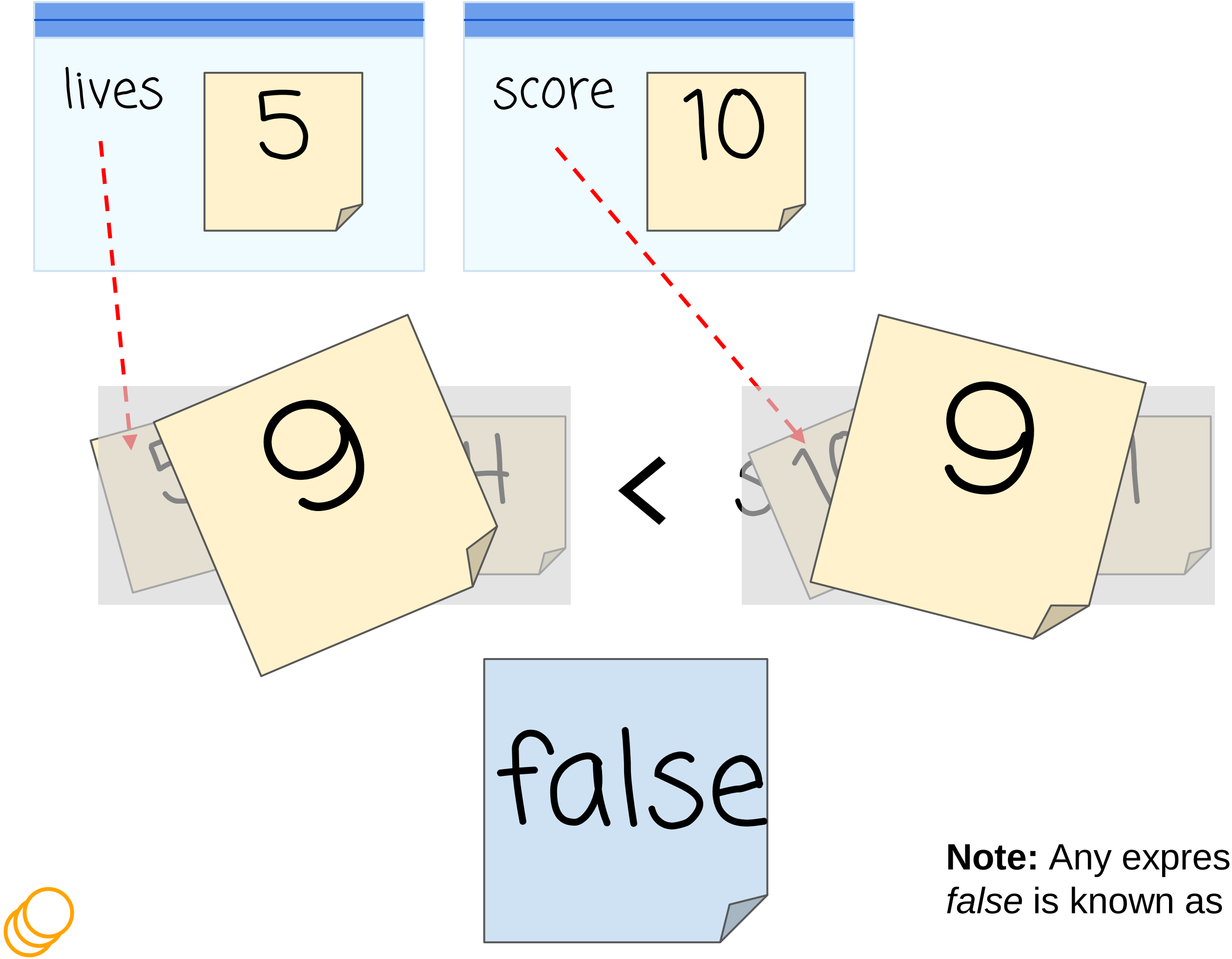
Unit 4 Lesson 5 - Activity

Do This:

Evaluate the expressions on each side of the comparison operator.

Then evaluate the expression for a Boolean value.





Any of the values in an expression can be a variable.

Do This:

Evaluate the expression for a Boolean value.

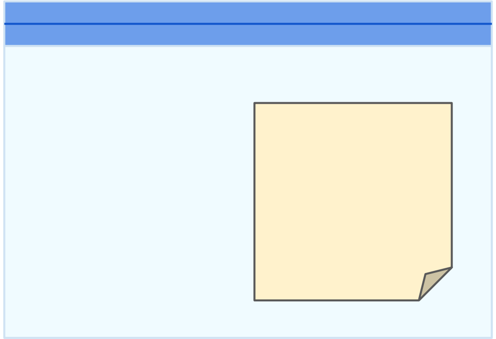
Note: Any expression that can be evaluate for *true* or *false* is known as a **Boolean Expression.**

Decision time!

Can I go to the movies? I'm allowed to if it's before 8 o'clock

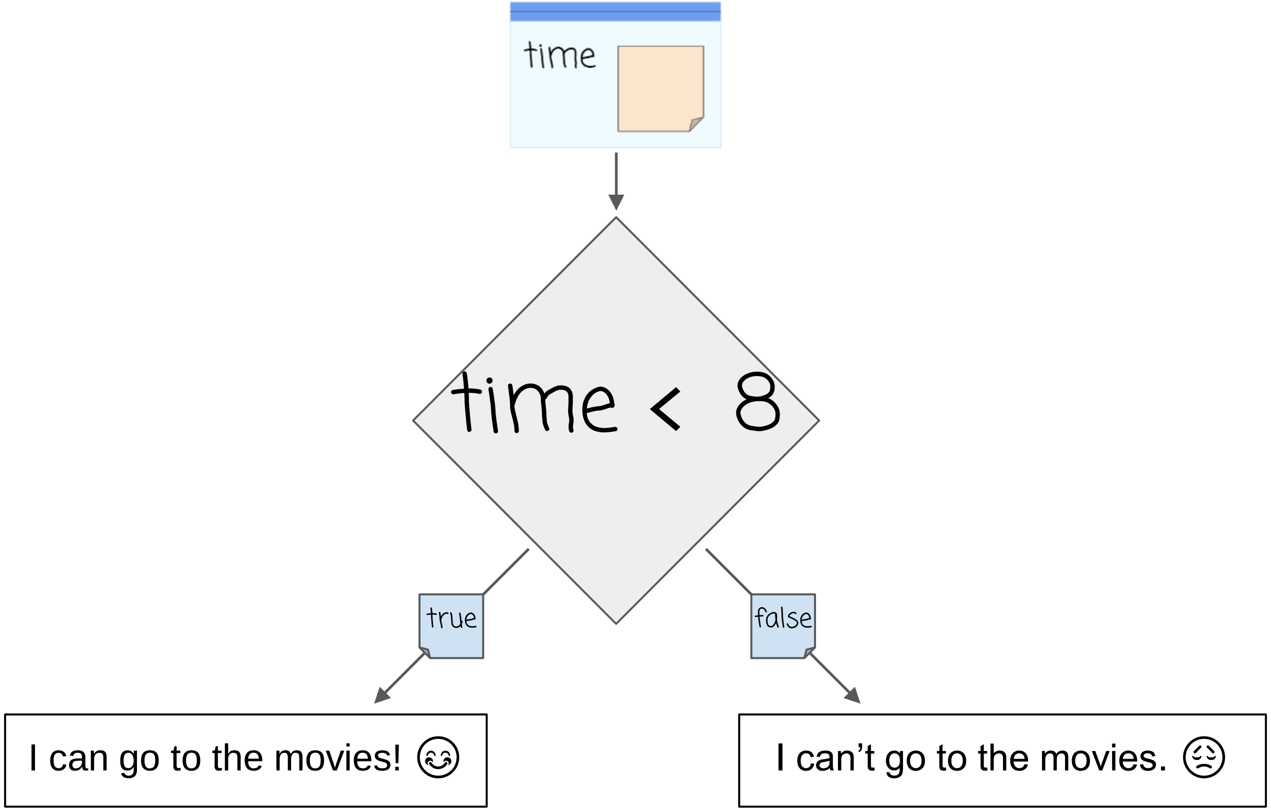
Do This:

- What information do I need to know?
- Create a baggie variable to store that information. Give it a name.



Unit 4 Lesson 5 - Activity

Can I go to the movies? I'm allowed to if it's before 8 o'clock



time

time < 8

true

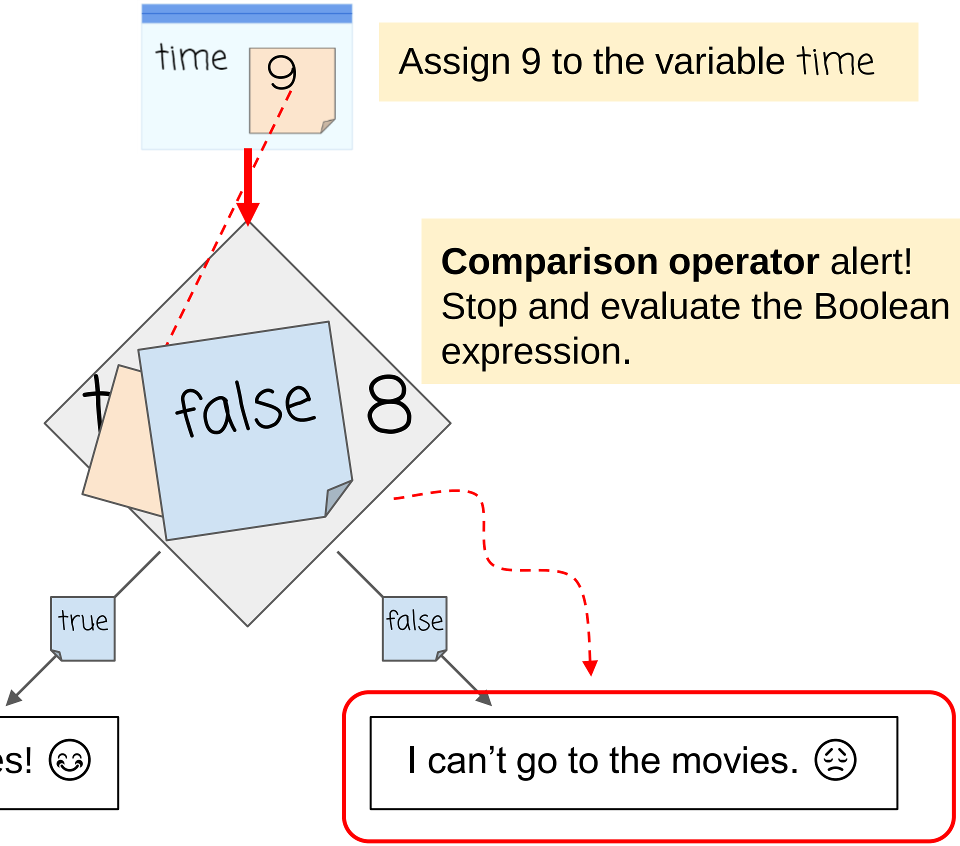
false

I can go to the movies! 😋

I can't go to the movies. 😟

Can I go to the movies? I'm allowed to if it's before 8 o'clock

It's 9 o'clock.
Can I go to
the movies?



Assign 9 to the variable time

Comparison operator alert!
Stop and evaluate the Boolean expression.

I can go to the movies! 😊

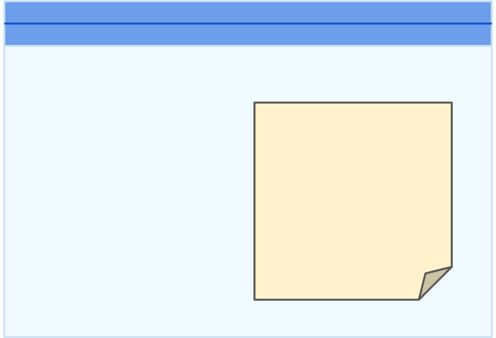
I can't go to the movies. 😔

Unit 4 Lesson 5 - Activity

Have I won the game? I have if my score * my lives is greater than 10.

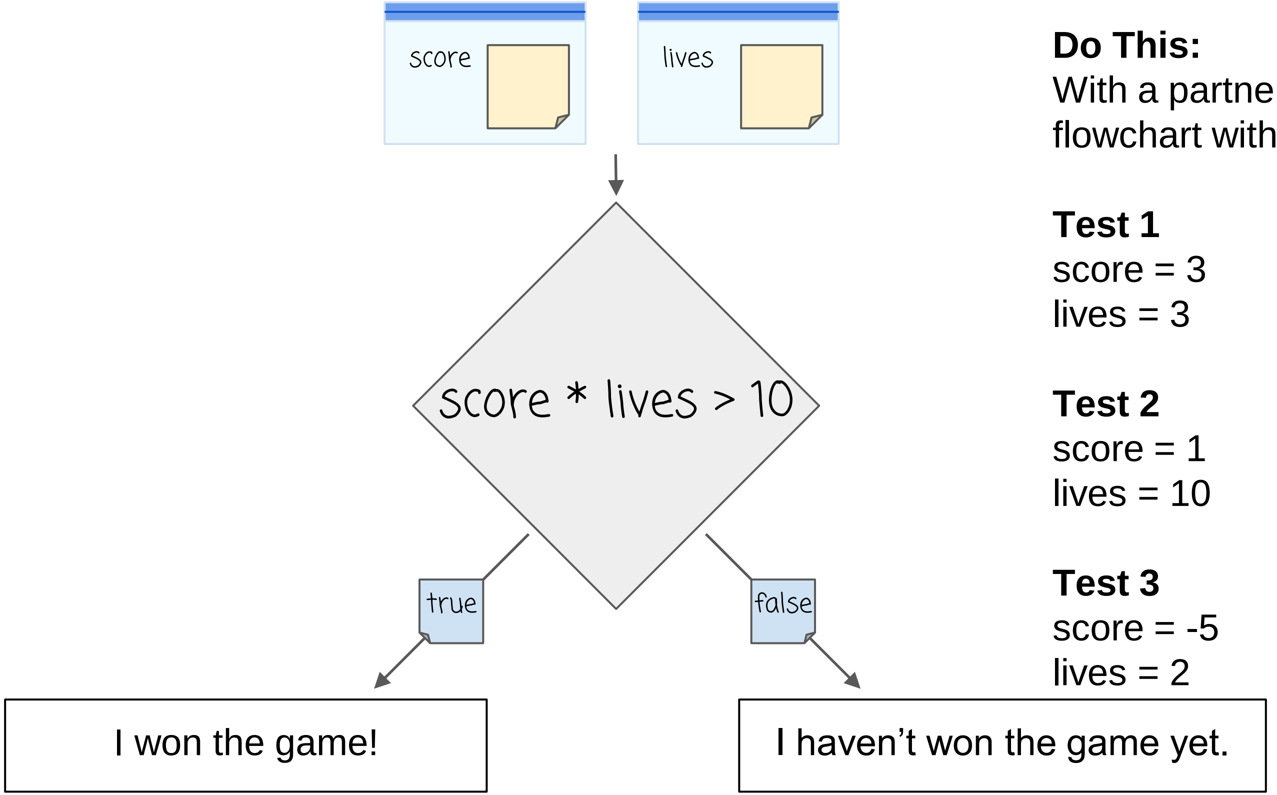
Do This:

- What information do I need to know?
- Create baggie variable(s) to store that information. Give them names.
- With a partner, discuss what the Boolean expression will look like. What will be compared?



Unit 4 Lesson 5 - Activity

Have I won the game? I have if my score * my lives is greater than 10.



Do This:
With a partner follow the flowchart with these inputs:

Test 1
score = 3
lives = 3

Test 2
score = 1
lives = 10

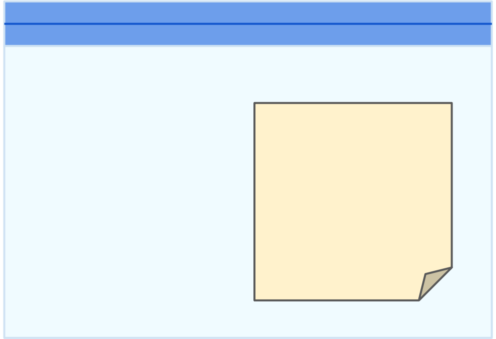
Test 3
score = -5
lives = 2

Challenge!!

Is my dog older than me if his age is converted to human years? Seven dog years equals one human year.

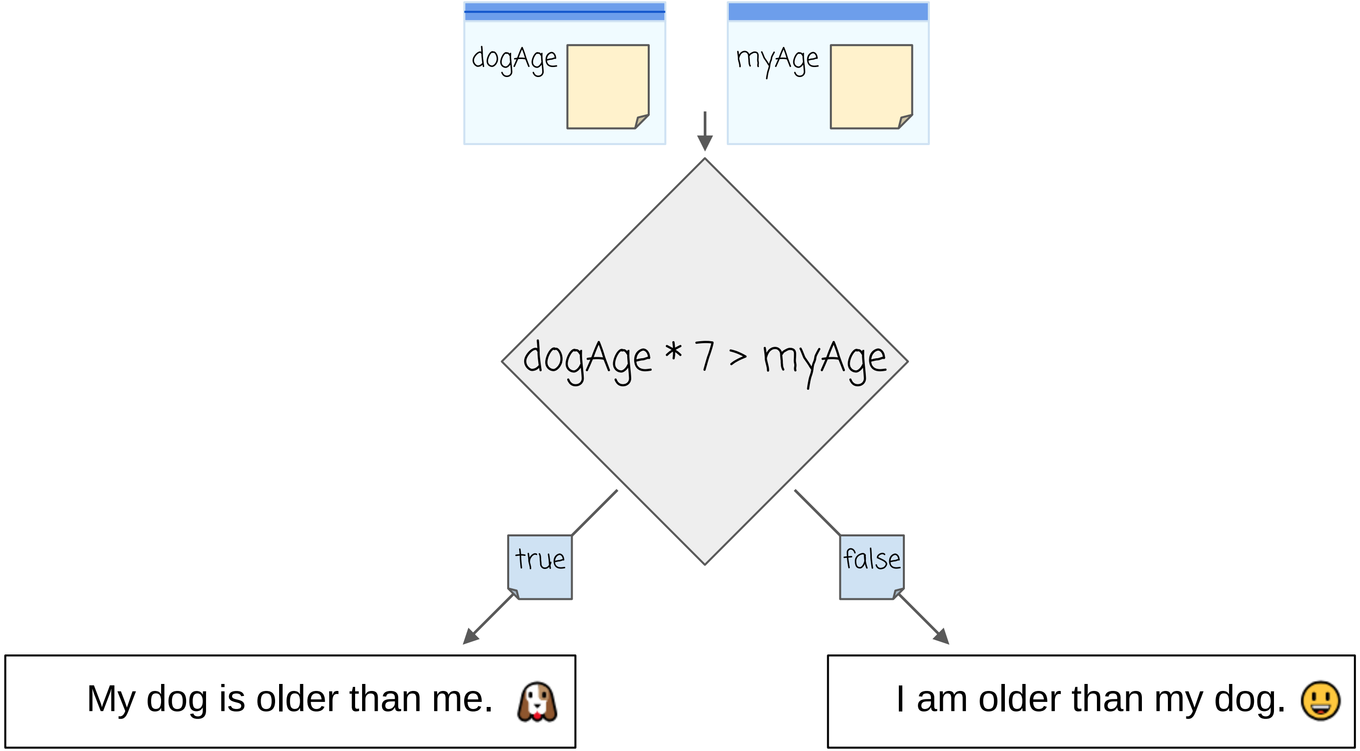
Do This:

- What information do I need to know?
- Create baggie variable(s) to store that information. Give them names.
- With a partner, discuss what the Boolean expression will look like. What will be compared?



Unit 4 Lesson 5 - Activity

Is my dog older than me if his age is converted to human years? Seven dog years equals one human year.



My dog is older than me. 🐶

I am older than my dog. 😃

What if my decision requires several steps?

Can I adopt a cat?

I can if I have 40 dollars

AND

I am over 14 years old

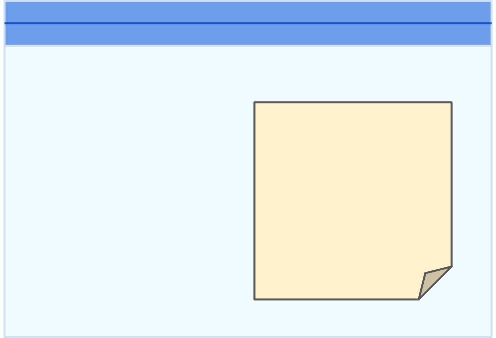
Unit 4 Lesson 5 - Activity

Can I adopt a cat?

I can if have 40 dollars
AND
I am over 14 years old

Do This:

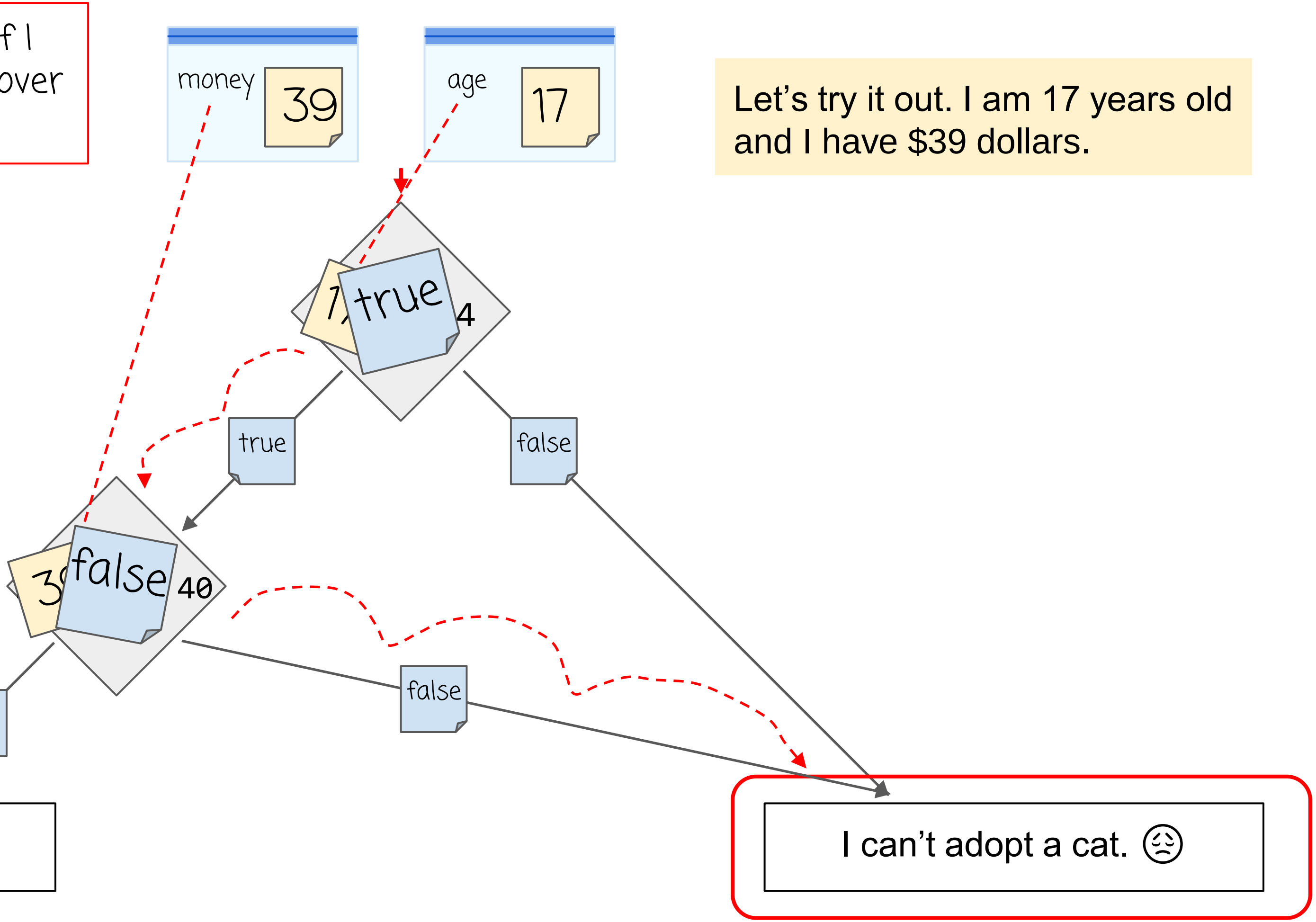
- What information do I need to know?
- Create baggie variable(s) to store that information. Give them names.
- With a partner, discuss how the flowchart might be set up. What will the Boolean expression(s) look like? What will be compared?



Unit 4 Lesson 5 - Activity

Can I adopt a cat? I can if I have 40 dollars AND I am over 14 years old.

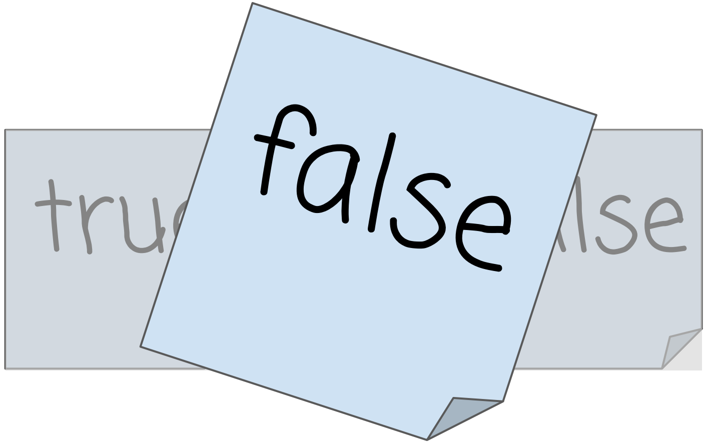
Let's try it out. I am 17 years old and I have $39 dollars.



Boolean values are a type of information, so they can also be evaluated in a Boolean expression using **logical operators.**

Logical Operators:

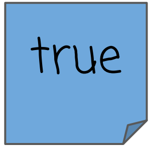
&&	AND
||	OR
!	NOT

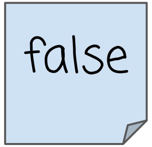


If something is true and (&&) false, it evaluates to false.

Let’s take a look at **Truth Tables**

In the next few slides, we are going to change the color of the sticky notes a little bit. Boolean values will still be on blue stickies, but true will be a dark blue, and false a light blue.





Unit 4 Lesson 5 - Activity

Truth Tables - used in evaluating Boolean expressions

&&		AND		
true	&&	true	→	true
true	&&	false	→	false
false	&&	true	→	false
false	&&	false	→	false

true

tr

fa

false and

&&

Both must be true for the Boolean expression to evaluate as true.

Unit 4 Lesson 5 - Activity

|| OR

true || true → true

true || false → true

false || true → true

false || false → false

Do This:

With a p esults for

each

tr

tr

fa

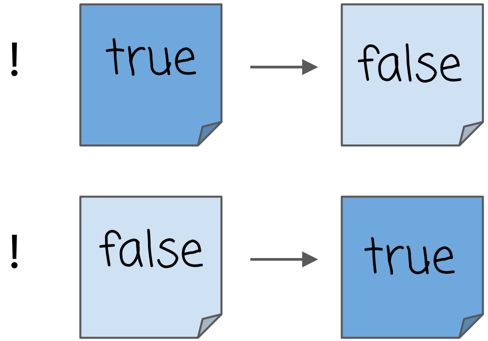
false or fal

||

Either may be true for the Boolean expression to evaluate as true.

! NOT

The ! NOT table is easy! The results are the opposite of the Boolean value.



not true evaluates to: **false**

not false evaluates to: **true**

Use logical operators to combine several Boolean expressions into one expression.

Can I adopt a cat?

I can if I have 40 dollars

AND

I am over 14 years old

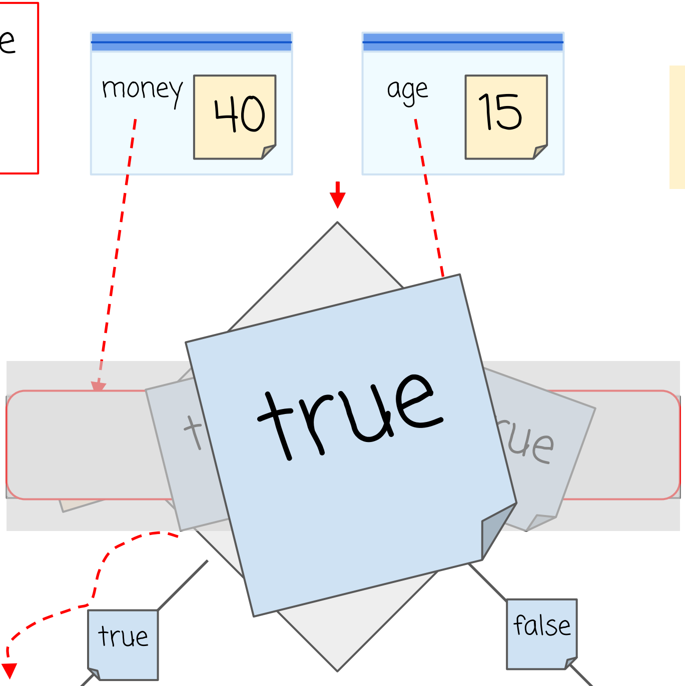
Unit 4 Lesson 5 - Activity

Can I adopt a cat? I can if I have 40 dollars AND I am over 14 years old.

Let's try it out. I am 15 years old and I have $40 dollars.

Truth Table

true && true → true



I can adopt a cat! 😋

I can't adopt a cat. 😟

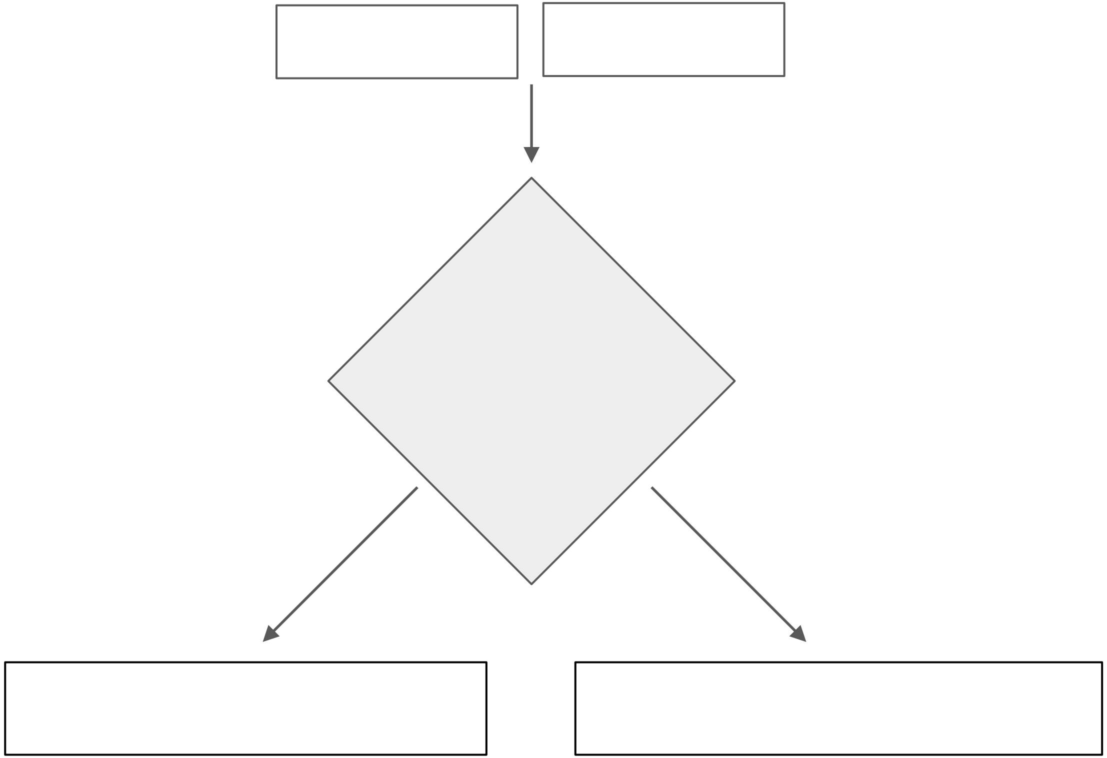
Unit 4 Lesson 5 - Activity

Do This:

Now it’s your turn to make a flowchart. On a scrap sheet of paper:

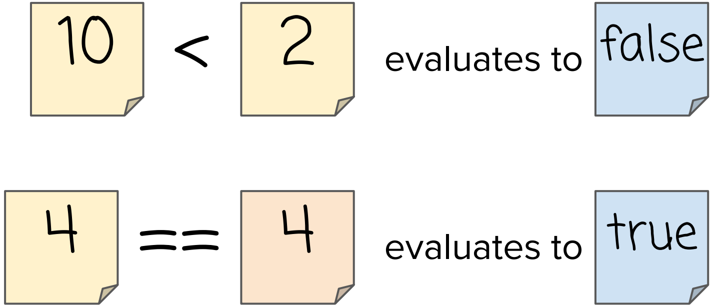
- Think back to the warm-up where we discussed making a decision on what to wear to an event.
- Write down the variables in boxes at the top.
- Inside the diamond shape, create the Boolean expression that will be used to make the decision.
- Draw True/False lines to the possible decisions.
- Challenge: Use logical operators (&&, ||, or !) in your Boolean expression. Add extra branches with multiple decisions.
- Test your flowchart with a friend!

Decision:



Key Takeaways

- A **Boolean Value** is a data type that is either true or false.
- **Comparison Operators** <, >, <=, >=, ==, != indicate a Boolean expression
- Each side of the Boolean expression is reduced to a single value
- Single values are compared and result in a Boolean value (true or false)

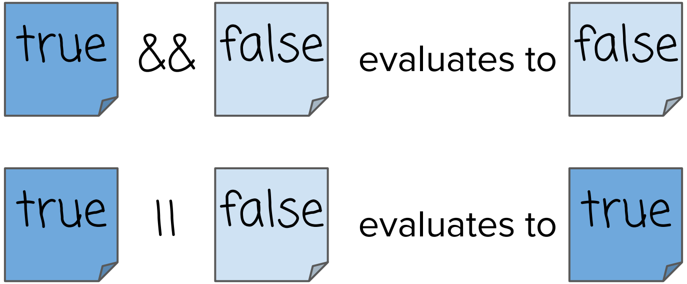


evaluates to

evaluates to

Key Takeaways

- Boolean expressions can also include **Logical Operators** &&, ||, != (AND, OR, NOT). Both sides of the logical operator are reduced to a single Boolean value
- A truth table is used to evaluate the reduced Boolean expression to a single Boolean value
- A decision is made with the single Boolean value
- A flowchart illustrates the steps of making a decision with a Boolean expression



true && false evaluates to false

true || false evaluates to true

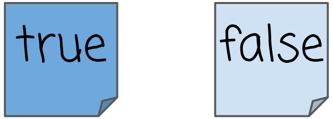
Wrap Up

●●●

Unit 4 Lesson 5 - Wrap Up

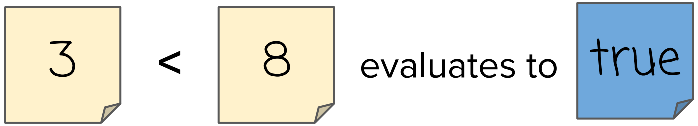
Unit 4 Lesson 5 - Wrap Up

Boolean Value: true or false



true false

Boolean Expression: evaluates to either true or false



3 < 8 evaluates to true

Unit 4 - Lesson 6
Conditionals Investigate

Warm Up

●○○

Unit 4 Lesson 6 - Warm Up

Prompt:

A water park will let a visitor on a ride if they are 48 or more inches tall OR they are 14 years old or older.

Make a flowchart for this decision. Make sure to use comparison operators (<, >, ==, etc.) and logical operators (&&, ||, !) when you write your Boolean expression.

Unit 4 Lesson 6 - Warm Up

"when": Means there is an onEvent to respond to user input. The app does something "when" the user clicks.

"if": Means there is a conditional statement that decides what pieces of code to run. The app does something "if" a boolean expression evaluates to true.

Activity

●●○

Unit 4 Lesson 6 - Activity

Lemon Squeeze Pt 1

Do This

Play the game at least once. Then with a partner, choose one of the three code sections below

- Section 1: lines 1 - 13
- Section 2: lines 16 - 30
- Section 3: lines 33 - 53

Read the code in your section carefully, making sure you understand how each line works.

Discuss

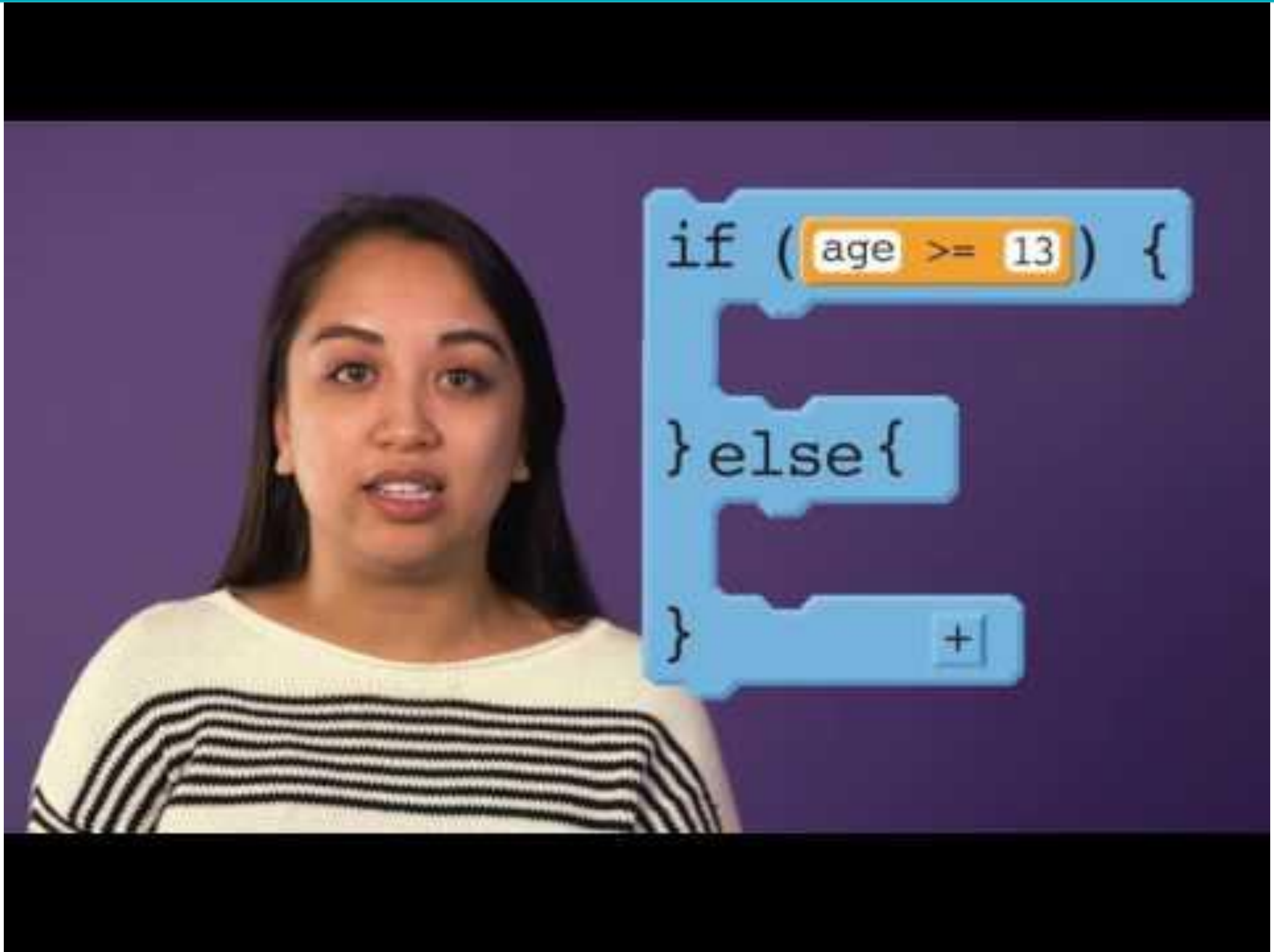
Find partners from the two other groups and:

- Explain what your section does
- Call out any lines of code you thought were interesting or confusing
- Ask good questions about how their section works, for example: "I don't understand this line" or "What does this command do?" or "Could your section have been written another way?"

Modify

- Right now the game keeps going when the player has 0 lives. Fix this problem.

Unit 4 Lesson 6 - Activity



```
if ( age >= 13 ) {

}else{

}                  +
```

Unit 4 Lesson 6 - Activity

```
var age = [illegible];
if (age >= 17) {
  write("You can see an R rated movie alone.");
} else if (age >= 13) {
  write("You can see a PG 13 movie alone.");
} else {
  write("You can see a G rated movie.");
}
```

Lemon Squeeze Pt 2

Do This

- Play the game at least once.
- Discuss with a partner what's changed since last time in how the game works.
- Find the if-else-if statement that was added to the program.
- Draw a flowchart for the if-else-if statement
- Modify the program so that the lemon becomes even smaller when the user has more than 15 points.

Unit 4 Lesson 6 - Activity

```
( day == "Tuesday" || day == "Thursday" ) || ( age > 12 && age < 21 )
```

Lemon Squeeze Pt 3

Do This

- Play the game at least once.
- Discuss with a partner what's changed since last time in how the game works.
- Find the if-else-if statement that was added
- Draw a flowchart for the if-else-if statement that was added
- Modify the app to add one more username and password. You'll need to edit both the code and user interface. Warning: Use a fake password. It's never a good idea to use real passwords directly in your code like this.

Wrap Up

●●●

Unit 4 Lesson 6 - Wrap Up

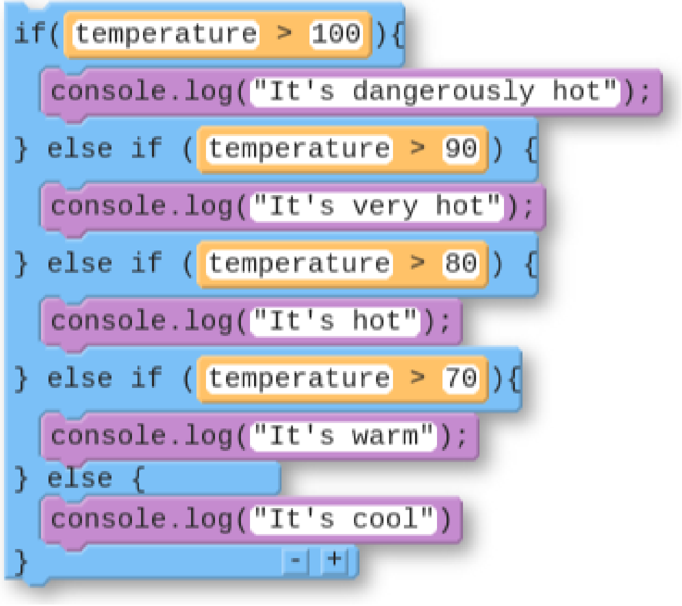
Prompt:

What is the difference between an if-statement, an if-else statement, and an if-else-if statement?

How are they similar?

Checking Multiple Conditions with If-Else-If

Name	Code (Block)	Code (Text)

Checking Multiple Conditions with If-Else-If	
	if(temperature > 100){
	  console.log("It's dangerously hot");
	} else if (temperature > 90) {
	  console.log("It's very hot");
	} else if (temperature > 80) {
	  console.log("It's hot");
	} else if (temperature > 70){
	  console.log("It's warm");
	} else {
	  console.log("It's cool")
	}
	
		if(temperature > 100){
		  console.log("It's dangerously hot");
		} else if (temperature > 90) {
		  console.log("It's very hot");
		} else if (temperature > 80) {
		  console.log("It's hot");
		} else if (temperature > 70){
		  console.log("It's warm");
		} else {
		  console.log("It's cool")
		}

Unit 4 - Lesson 7
Conditionals Practice

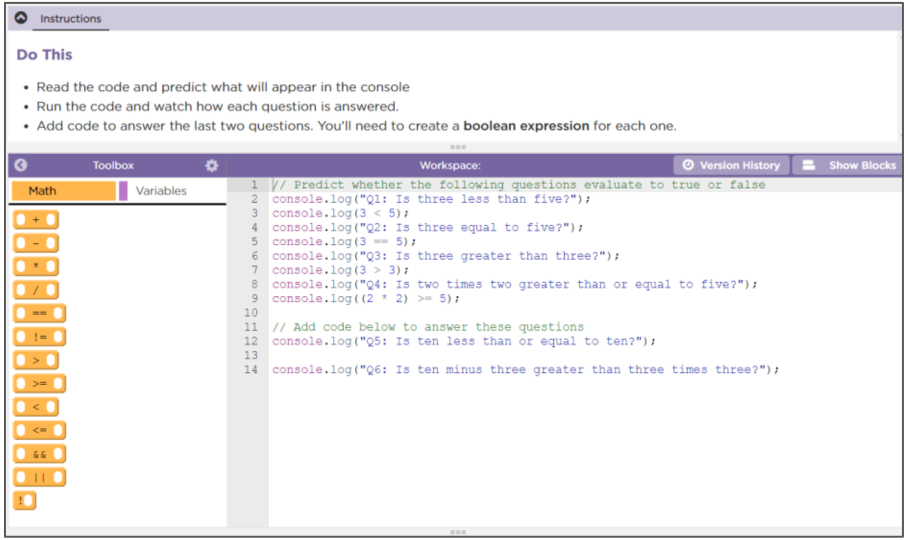
Warm Up

● ○ ○

Activity

●●○

Conditionals Practice:



Do This

- Read the code and predict what will appear in the console
- Run the code and watch how each question is answered.
- Add code to answer the last two questions. You'll need to create a **boolean expression** for each one.

```
// Predict whether the following questions evaluate to true or false
console.log("Q1: Is three less than five?");
console.log(3 < 5);
console.log("Q2: Is three equal to five?");
console.log(3 == 5);
console.log("Q3: Is three greater than three?");
console.log(3 > 3);
console.log("Q4: Is two times two greater than or equal to five?");
console.log((2 * 2) >= 5);

// Add code below to answer these questions
console.log("Q5: Is ten less than or equal to ten?");

console.log("Q6: Is ten minus three greater than three times three?");
```

Lesson 7: Conditionals Practice

Saved 13 days ago

Do This:

- Navigate to Lesson 7, Level 2 on Code Studio

Unit 4 Lesson 7 - Activity

Debugging: the process of finding and fixing problems in code

Describe

The Problem

What do you expect it to do?

What does it actually do?

Does it always happen?

Hunt

For Bugs

Are there warnings or errors?

What did you change most recently?

Explain your code to someone else

Look for code related to the problem

Try

Solutions

Make a small change

Document

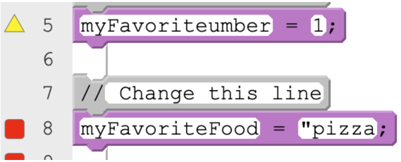
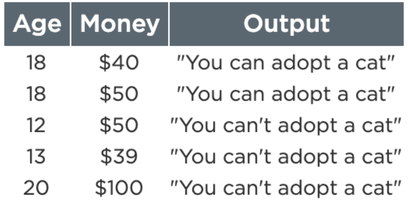
As You Go

What have you learned?

What strategies did you use?

What questions do you have?

Specific Debugging Skill for the Day

<table>
<tr><th>Error Type</th><th>Example</th><th>How to Debug</th></tr>
<tr><td><b>Syntax Error:</b> Your code doesn’t follow the rules of the programming language</td><td>Writing a variable name in quotes
Using a variable that doesn’t exist</td><td>Check warnings and errors

myFavoriteumber = 1;
// Change this line
myFavoriteFood = "pizza;</td></tr>
<tr><td><b>Logic Error:</b> Your code follows the rules of the programming language but doesn’t do what you intend</td><td>Writing if-else-if statements in the wrong order
Updating the property of the wrong element</td><td>Test your code
<table><tr><th>Age</th><th>Money</th><th>Output</th></tr><tr><td>18</td><td>$40</td><td>"You can adopt a cat"</td></tr><tr><td>18</td><td>$50</td><td>"You can adopt a cat"</td></tr><tr><td>12</td><td>$50</td><td>"You can't adopt a cat"</td></tr><tr><td>13</td><td>$39</td><td>"You can't adopt a cat"</td></tr><tr><td>20</td><td>$100</td><td>"You can't adopt a cat"</td></tr></table></td></tr>
</table>

Practice checking for both syntax errors and logic errors in your programs.

Wrap Up

Unit 4 Lesson 7 - Wrap Up

Prompt:

What aspects of working with conditionals do you feel like clicked today?

What do you still feel like you have trouble with?

Unit 4 - Lesson 8
Conditionals Make

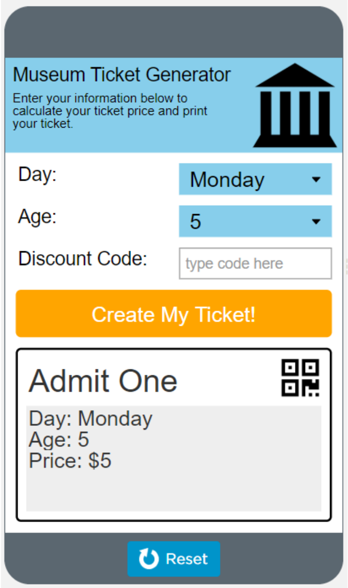
Warm Up

●○○

Activity

●●○

Conditionals Make: Museum Ticket Generator



Museum Ticket Generator

Enter your information below to calculate your ticket price and print your ticket.

Day: Monday

Age: 5

Discount Code: type code here

Admit One

Day: Monday
Age: 5
Price: $5

Lesson 8: Conditional Make

Saved a few seconds ago

Do This:

- Navigate to Lesson 8, Level 2 on Code Studio

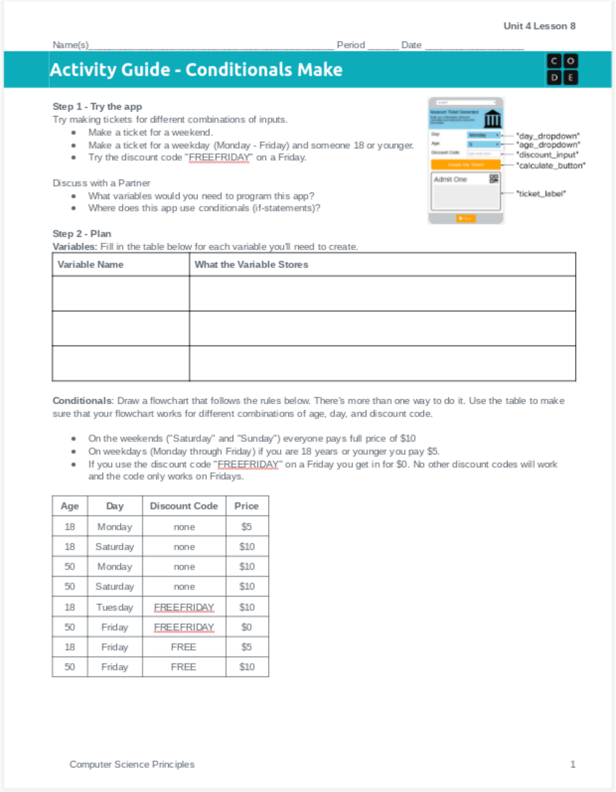
Step 1 - Try the app

- Try making tickets for different combinations of inputs.
- Make a ticket for a weekend.
- Make a ticket for a weekday (Monday - Friday) and someone 18 or younger.
- Try the discount code "FREEFRIDAY" on a Friday.

Discuss with a Partner

- What variables would you need to program this app?
- Where does this app use conditionals (if-statements)?

Do This: Make the Museum Ticket Generator!



Unit 4 Lesson 8

Name(s)_______________________________________ Period ______ Date ____________________

Activity Guide - Conditionals Make

Step 1 - Try the app

Try making tickets for different combinations of inputs.

- Make a ticket for a weekend.
- Make a ticket for a weekday (Monday - Friday) and someone 18 or younger.
- Try the discount code "FREEFRIDAY" on a Friday.

Discuss with a Partner

- What variables would you need to program this app?
- Where does this app use conditionals (if-statements)?

Step 2 - Plan

Variables: Fill in the table below for each variable you'll need to create.

Variable Name	What the Variable Stores

Conditionals: Draw a flowchart that follows the rules below. There's more than one way to do it. Use the table to make sure that your flowchart works for different combinations of age, day, and discount code.

- On the weekends ("Saturday" and "Sunday") everyone pays full price of $10
- On weekdays (Monday through Friday) if you are 18 years or younger you pay $5.
- If you use the discount code "FREEFRIDAY" on a Friday you get in for $0. No other discount codes will work and the code only works on Fridays.

Age	Day	Discount Code	Price
18	Monday	none	$5
18	Saturday	none	$10
50	Monday	none	$10
50	Saturday	none	$10
18	Tuesday	FREEFRIDAY	$10
50	Friday	FREEFRIDAY	$0
18	Friday	FREE	$5
50	Friday	FREE	$10

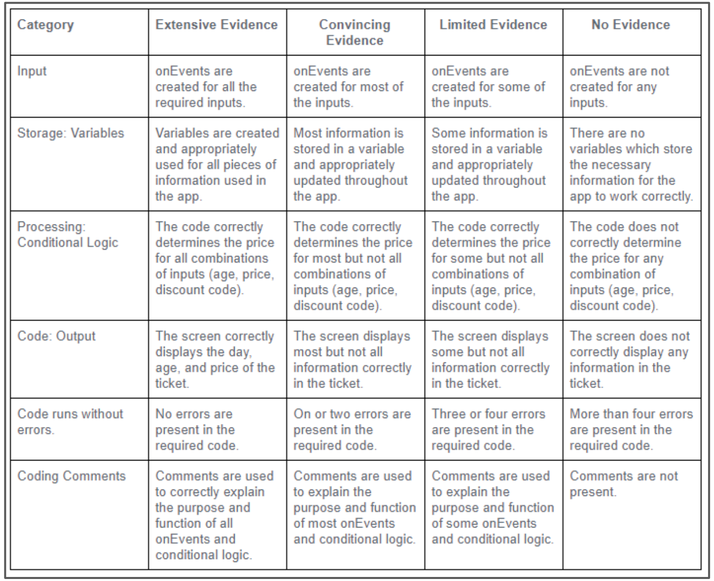
Computer Science Principles 1

Use the activity guide to plan out your code, including the variables you'll create and a flowchart of the conditional statement you'll need to write.

Step 3 includes steps you can follow to build the app, or you can use your own process.

Unit 4 Lesson 8 - Activity

Don't forget to check the rubric before hitting submit!



Category	Extensive Evidence	Convincing Evidence	Limited Evidence	No Evidence
Input	onEvents are created for all the required inputs.	onEvents are created for most of the inputs.	onEvents are created for some of the inputs.	onEvents are not created for any inputs.
Storage: Variables	Variables are created and appropriately used for all pieces of information used in the app.	Most information is stored in a variable and appropriately updated throughout the app.	Some information is stored in a variable and appropriately updated throughout the app.	There are no variables which store the necessary information for the app to work correctly.
Processing: Conditional Logic	The code correctly determines the price for all combinations of inputs (age, price, discount code).	The code correctly determines the price for most but not all combinations of inputs (age, price, discount code).	The code correctly determines the price for some but not all combinations of inputs (age, price, discount code).	The code does not correctly determine the price for any combination of inputs (age, price, discount code).
Code: Output	The screen correctly displays the day, age, and price of the ticket.	The screen displays most but not all information correctly in the ticket.	The screen displays some but not all information correctly in the ticket.	The screen does not correctly display any information in the ticket.
Code runs without errors.	No errors are present in the required code.	On or two errors are present in the required code.	Three or four errors are present in the required code.	More than four errors are present in the required code.
Coding Comments	Comments are used to correctly explain the purpose and function of all onEvents and conditional logic.	Comments are used to explain the purpose and function of most onEvents and conditional logic.	Comments are used to explain the purpose and function of some onEvents and conditional logic.	Comments are not present.

Submit

Wrap Up

Unit 4 Lesson 8 - Wrap Up

Great job today!

Unit 4 - Lesson 9
Functions Explore/Investigate

Warm Up

●○○

Prompt

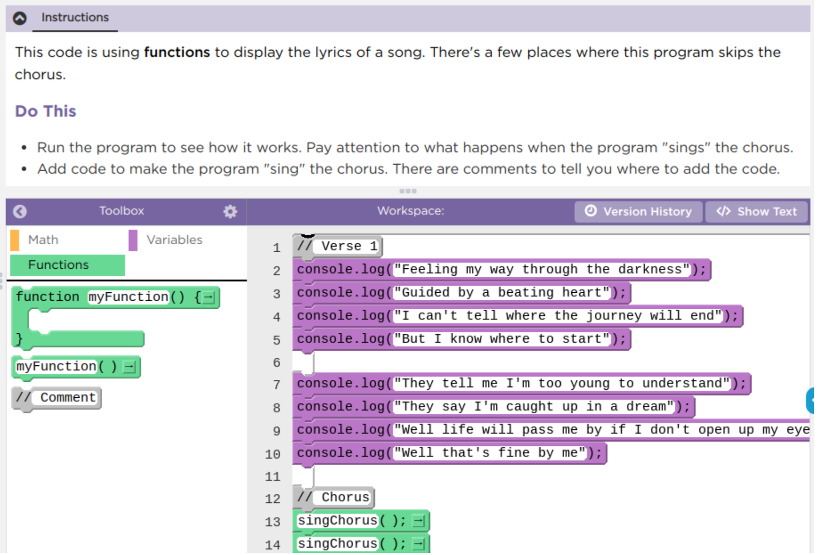
- In Style 1, what line of the song do you sing after line 09? What about in Style 2?
- Style 2 uses fewer lines to write. Are there fewer lyrics to sing?
- What are the benefits of writing a song in Style 2?

Activity

●●○

Unit 4 Lesson 9 - Activity

Functions Explore / Investigate



Instructions

This code is using **functions** to display the lyrics of a song. There's a few places where this program skips the chorus.

Do This

- Run the program to see how it works. Pay attention to what happens when the program "sings" the chorus.
- Add code to make the program "sing" the chorus. There are comments to tell you where to add the code.

```
// Verse 1
console.log("Feeling my way through the darkness");
console.log("Guided by a beating heart");
console.log("I can't tell where the journey will end");
console.log("But I know where to start");

console.log("They tell me I'm too young to understand");
console.log("They say I'm caught up in a dream");
console.log("Well life will pass me by if I don't open up my eye
console.log("Well that's fine by me");

// Chorus
singChorus( );
singChorus( );
```

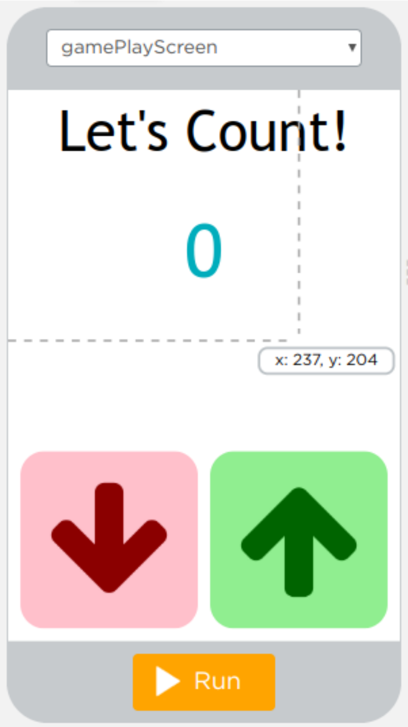
Lesson 9: Functions Explore / Investigate

Saved 3 days ago

Do This:

- Navigate to Lesson 9, Level 2 on Code Studio

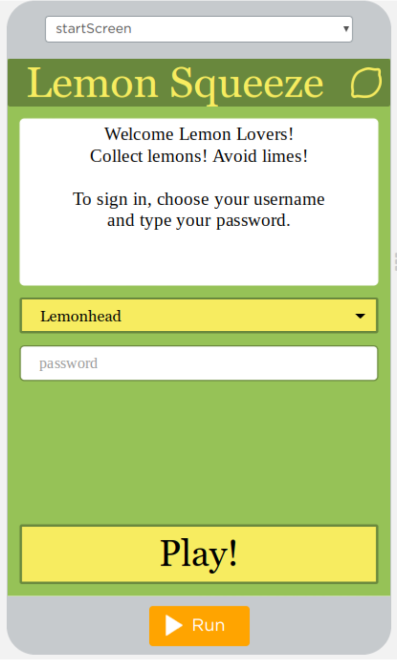
Unit 4 Lesson 9 - Activity



Level 4

- On what line(s) is the updateScreen functions being declared?
- On what lines is the updateScreen function being called?
- How does the updateScreen function make the program easier to write and understand?

Level 5



Welcome Lemon Lovers!
Collect lemons! Avoid limes!

To sign in, choose your username
and type your password.

Lemonhead

password

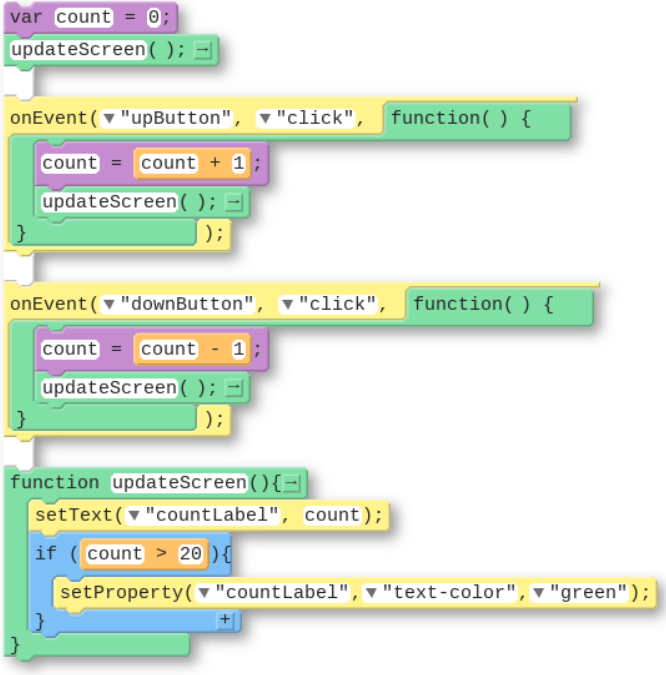
- How does the newly added updateScreen function help keep your code better organized or reduce the amount of repeated code?

Wrap Up

●●●

The updateScreen() Pattern

Code (Block)



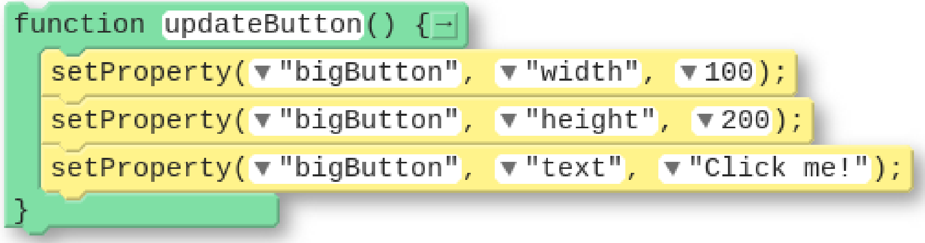
```
var count = 0;
updateScreen( );

onEvent( "upButton", "click", function( ) {
  count = count + 1;
  updateScreen( );
} );

onEvent( "downButton", "click", function( ) {
  count = count - 1;
  updateScreen( );
} );

function updateScreen(){
  setText( "countLabel", count);
  if ( count > 20 ){
    setProperty( "countLabel", "text-color", "green");
  }
}
```

Code (Text)

```
var count = 0;
updateScreen();

onEvent(“upButton”, “click”, function(){
  count = count + 1;
  updateScreen();
});

onEvent(“downButton”, “click”, function(){
  count = count - 1;
  updateScreen();
});

function updateScreen(){
  setText(“countLabel”, count);
  if(count > 20){
    setProperty(“countLabel”, “text-color”, “green”);
  }
}
```

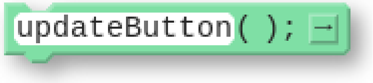
Unit 4 Lesson 9 - Wrap Up

Function: a named group of programming instructions. Also referred to as a “procedure”.



```
function updateButton() {
  setProperty("bigButton", "width", 100);
  setProperty("bigButton", "height", 200);
  setProperty("bigButton", "text", "Click me!");
}
```

Function Call: a command that executes the code within a function



```
updateButton( );
```

Prompt:

Reflecting on today's lesson about functions:

What did you learn?
What are you uncertain about?

Unit 4 - Lesson 10
Functions Practice

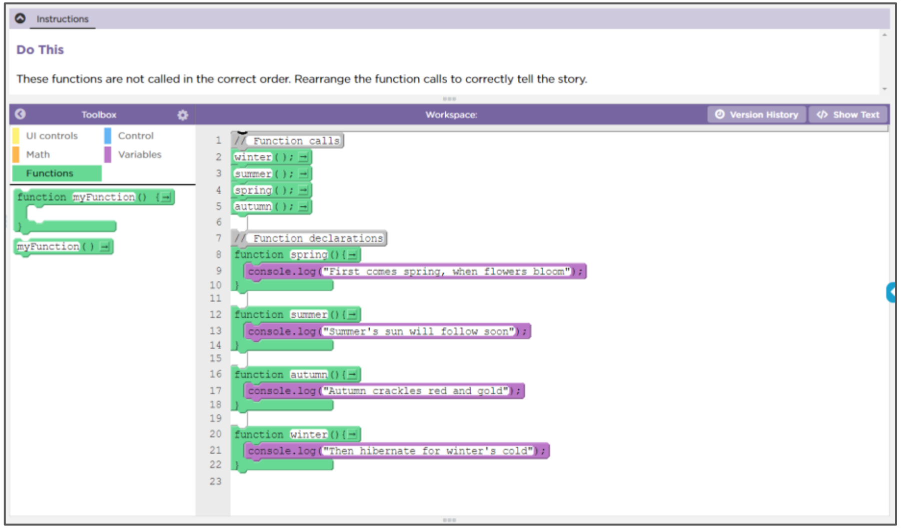
Warm Up

● ○ ○

Activity

●●○

Functions Practice



Do This

These functions are not called in the correct order. Rearrange the function calls to correctly tell the story.

```
// Function calls
winter();
summer();
spring();
autumn();

// Function declarations
function spring(){
  console.log("First comes spring, when flowers bloom");
}

function summer(){
  console.log("Summer's sun will follow soon");
}

function autumn(){
  console.log("Autumn crackles red and gold");
}

function winter(){
  console.log("Then hibernate for winter's cold");
}
```

Do This:

- Navigate to Lesson 10, Level 2 on Code Studio

Debugging Variable Scope: Functions

Video
Download Link

Create Variables Once, At the Top, Outside Functions or onEvent()

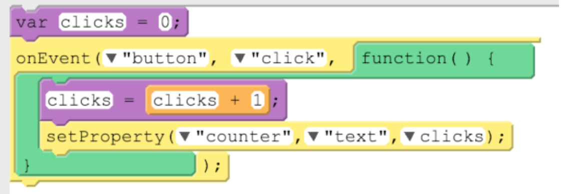
When you create variables you should:

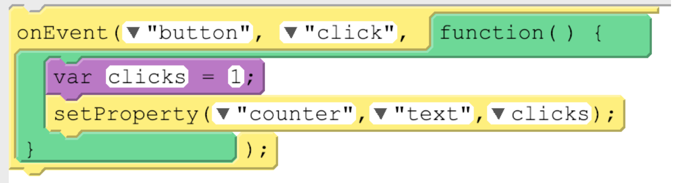
- **Use `var` only once.** You don't need to create variables twice and this can cause errors.
- **Create your variables at the top of your program.** This keeps your code organized and easier to read for you and others.
- **Create your variables outside any `function` or `onEvent()` bocks.**

Global vs. Local Variables

There's two types of variables, global and local, and so far we've only used global variables. Here's the main difference between global and local variables.

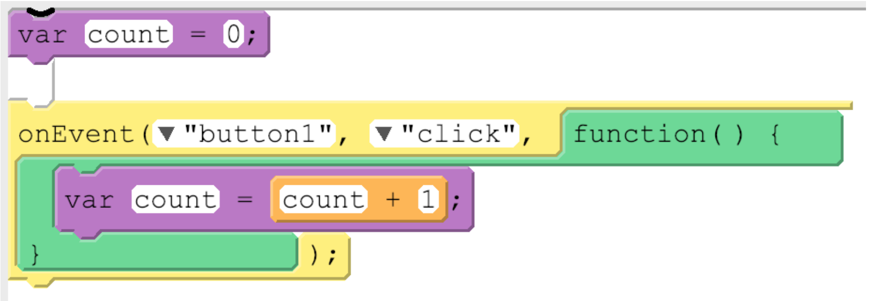
Type of Variable	How It Works	How Created	Picture

Global	Permanent. Can be used anywhere in your code.	var used outside an onEvent()	
			var clicks = 0;
			onEvent("button", "click", function() {
			  clicks = clicks + 1;
			  setProperty("counter", "text", clicks);
			});

Local	Temporary. Can be used only in the part of the code where it was created, like inside an onEvent(). Deleted once the onEvent() is done running.	var used inside an onEvent()	
			onEvent("button", "click", function() {
			  var clicks = 1;
			  setProperty("counter", "text", clicks);
			});

Avoiding Local Variables and Debugging

Local variables will eventually be useful but for now they're most likely to just be confusing. The biggest issue you'll fun into right now with local variables is accidentally using `var` inside of an `onEvent()` or `function`. Here's what the code usually looks like:



```
var count = 0;

onEvent(▼"button1", ▼"click", function( ) {
  var count = count + 1;
});
```

This code is pretty confusing. While it looks like there's only one variable being used, it actually has two variables, one local and one global, and they're both named count! Changing the value on one will have no impact on the other. This can cause unexpected behavior in your code and it can get tricky to debug.

The best way to avoid these issues is to **make sure for now that you're not using** `var` **inside of an** `onEvent()` **or** `function`. If you run into a tricky debugging problem, check if you're accidentally creating a local variable.

Wrap Up

Prompt:

What aspects of working with functions clicked today?

What do you still feel like you have trouble with?

Unit 4 - Lesson 11
Functions Make

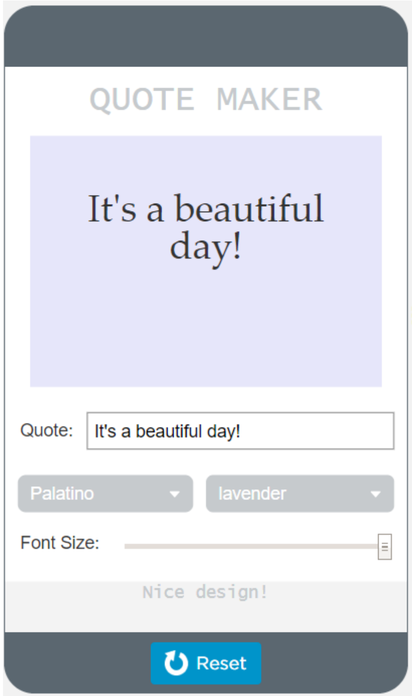
Warm Up

●○○

Activity

●●○

Functions Make:
Quote Maker App



QUOTE MAKER

It's a beautiful day!

Quote: It's a beautiful day!

Palatino

lavender

Font Size:

Nice design!

Lesson 11: Functions Make

Saved a day ago

Do This:

- Navigate to Lesson 11, Level 2 on Code Studio

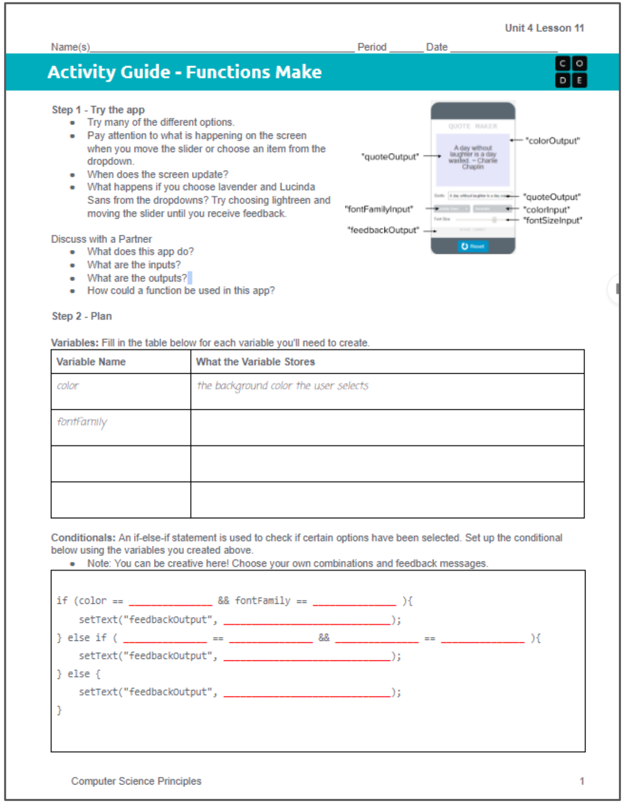
Do This:

- Try many of the different options
- Pay attention to what is happening on the screen when you move the slider or choose an item from the dropdown
- When does the screen update?
- What happens if you choose **lavender** and **Lucinda Sans** from the dropdowns? Try choosing **lightgreen** and moving the slider until you receive different feedback.

Prompts:

- What does this app do?
- What are the inputs?
- What are the outputs?
- How could a function be used in this app?

Do This: Make the Quote Maker!



Unit 4 Lesson 11

Name(s)_______________________________________ Period ______ Date __________________

Activity Guide - Functions Make

Step 1 - Try the app

- Try many of the different options.
- Pay attention to what is happening on the screen when you move the slider or choose an item from the dropdown.
- When does the screen update?
- What happens if you choose lavender and Lucinda Sans from the dropdowns? Try choosing lightreen and moving the slider until you receive feedback.

Discuss with a Partner

- What does this app do?
- What are the inputs?
- What are the outputs?
- How could a function be used in this app?

Step 2 - Plan

Variables: Fill in the table below for each variable you'll need to create.

Variable Name	What the Variable Stores
color	*the background color the user selects*
fontFamily	

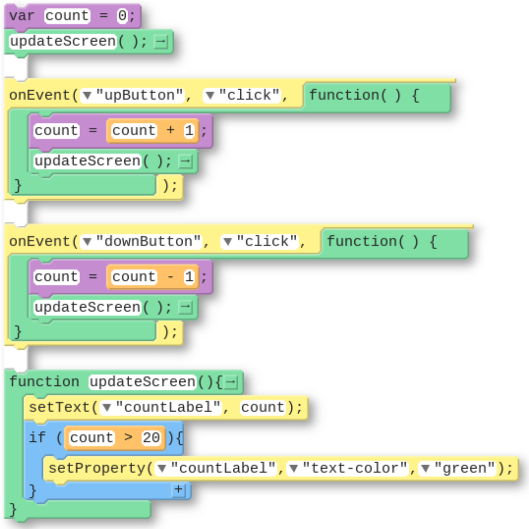
Conditionals: An if-else-if statement is used to check if certain options have been selected. Set up the conditional below using the variables you created above.

- Note: You can be creative here! Choose your own combinations and feedback messages.

```
if (color == ____________ && fontFamily == ____________ ){
    setText("feedbackOutput", _________________________);
} else if ( ____________ == ____________ && ____________ == ____________ ){
    setText("feedbackOutput", _________________________);
} else {
    setText("feedbackOutput", _________________________);
}
```

Computer Science Principles 1

The updateScreen() Pattern

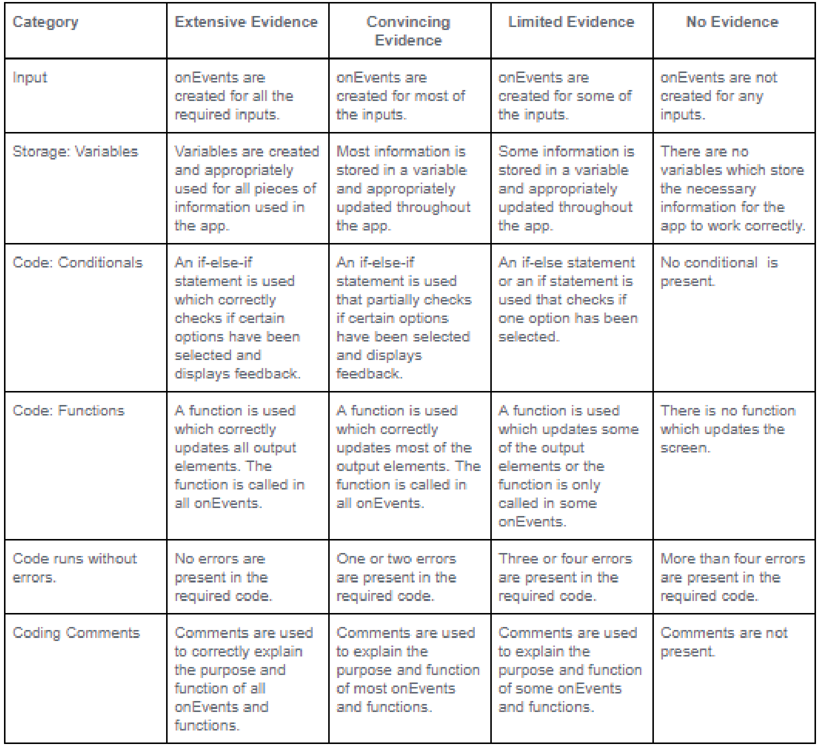


```
var count = 0;
updateScreen( );

onEvent( "upButton", "click", function( ) {
  count = count + 1;
  updateScreen( );
} );

onEvent( "downButton", "click", function( ) {
  count = count - 1;
  updateScreen( );
} );

function updateScreen(){
  setText( "countLabel", count);
  if ( count > 20 ){
    setProperty( "countLabel", "text-color", "green");
  }
}
```

```
var count = 0;
updateScreen();

onEvent("upButton", "click", function() {
  count = count + 1;
  updateScreen();
});

onEvent("downButton", "click", function() {
  count = count - 1;
  updateScreen();
});

function updateScreen(){
  setText("countLabel", count);
  if (count > 20){
    setProperty("countLabel","text-color","green");
  }
}
```

Unit 4 Lesson 11 - Activity

Don't forget to check the rubric before hitting submit!



Category	Extensive Evidence	Convincing Evidence	Limited Evidence	No Evidence
Input	onEvents are created for all the required inputs.	onEvents are created for most of the inputs.	onEvents are created for some of the inputs.	onEvents are not created for any inputs.
Storage: Variables	Variables are created and appropriately used for all pieces of information used in the app.	Most information is stored in a variable and appropriately updated throughout the app.	Some information is stored in a variable and appropriately updated throughout the app.	There are no variables which store the necessary information for the app to work correctly.
Code: Conditionals	An if-else-if statement is used which correctly checks if certain options have been selected and displays feedback.	An if-else-if statement is used that partially checks if certain options have been selected and displays feedback.	An if-else statement or an if statement is used that checks if one option has been selected.	No conditional is present.
Code: Functions	A function is used which correctly updates all output elements. The function is called in all onEvents.	A function is used which correctly updates most of the output elements. The function is called in all onEvents.	A function is used which updates some of the output elements or the function is only called in some onEvents.	There is no function which updates the screen.
Code runs without errors.	No errors are present in the required code.	One or two errors are present in the required code.	Three or four errors are present in the required code.	More than four errors are present in the required code.
Coding Comments	Comments are used to correctly explain the purpose and function of all onEvents and functions.	Comments are used to explain the purpose and function of most onEvents and functions.	Comments are used to explain the purpose and function of some onEvents and functions.	Comments are not present.

Submit

Wrap Up

Unit 4 Lesson 11 - Wrap Up

Great job today!

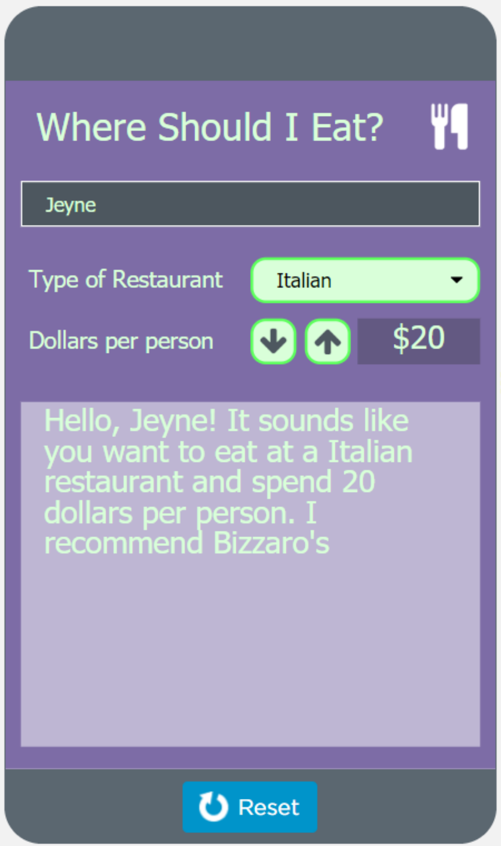
Unit 4 - Lesson 12
Project - Decision Maker App Part 1

Warm Up

●○○

Activity

●●○



Jeyne

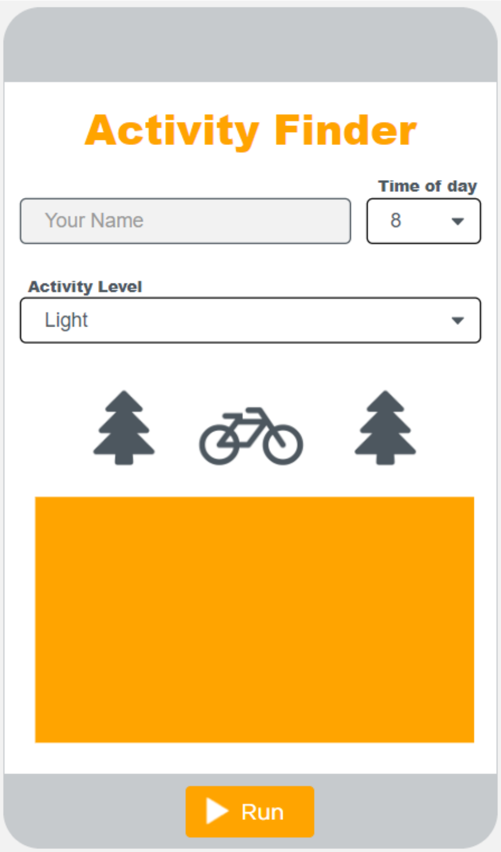
Type of Restaurant: Italian

Dollars per person: $20

Hello, Jeyne! It sounds like you want to eat at a Italian restaurant and spend 20 dollars per person. I recommend Bizzaro's

Level 2: Sample App #1

- What does this app do?
- What are the inputs?
- What are the outputs?
- What variables do you think would be necessary for this app to work?
- What kinds of conditional logic do you think are necessary to make it work?
- How could a function be used in this app?



Activity Finder

Time of day

Your Name

8

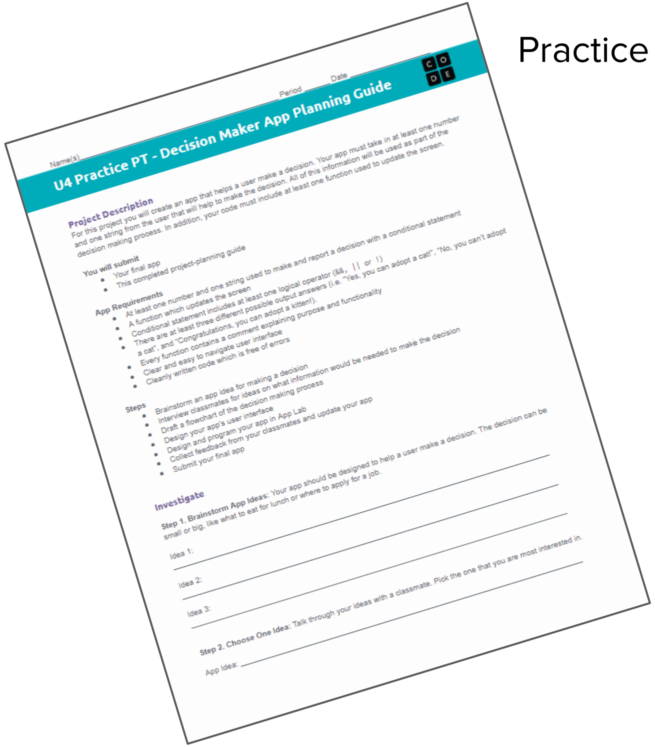
Activity Level

Light

Level 3: Sample App #2

- What does this app do?
- What are the inputs?
- What are the outputs?
- What variables do you think would be necessary for this app to work?
- What kinds of conditional logic do you think are necessary to make it work?
- How could a function be used in this app?

Practice PT: Plan the Decision Maker App



Name(s)__________ Period ______ Date ______

U4 Practice PT - Decision Maker App Planning Guide

Project Description

For this project you will create an app that helps a user make a decision. Your app must take in at least one number and one string from the user that will help to make the decision. All of this information will be used as part of the decision making process. In addition, your code must include at least one function used to update the screen.

You will submit

- Your final app
- This completed project-planning guide

App Requirements

- At least one number and one string used to make and report a decision with a conditional statement
- A function which updates the screen
- Conditional statement includes at least one logical operator (&&, || or !)
- There are at least three different possible output answers (i.e. "Yes, you can adopt a cat!", "No, you can't adopt a cat", and "Congratulations, you can adopt a kitten!).
- Every function contains a comment explaining purpose and functionality
- Clear and easy to navigate user interface
- Cleanly written code which is free of errors

Steps

- Brainstorm an app idea for making a decision
- Interview classmates for ideas on what information would be needed to make the decision
- Draft a flowchart of the decision making process
- Design your app's user interface
- Design and program your app in App Lab
- Collect feedback from your classmates and update your app
- Submit your final app

Investigate

Step 1. Brainstorm App Ideas: Your app should be designed to help a user make a decision. The decision can be small or big, like what to eat for lunch or where to apply for a job.

Idea 1: __________

Idea 2: __________

Idea 3: __________

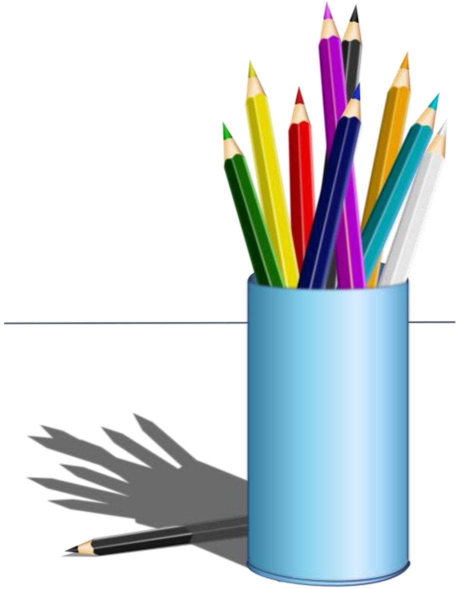
Step 2. Choose One Idea: Talk through your ideas with a classmate. Pick the one that you are most interested in.

App Idea: __________

You should have:

Practice PT Decision Maker App Planning Guide

Pen/Pencil



App Requirements:

- At least one number and one string used to make and report a decision with a conditional statement
- A function which updates the screen and is called at least twice in the program.
- Conditional statement includes at least one logical operator (&&, ||, or !)
- There are at least three different possible output answers (i.e. "Yes, you can adopt a cat!", "No, you can't adopt a cat", and "Congratulations, you can adopt a kitten!).
- Every function contains a comment explaining purpose (what it dos) and functionality (how it works).
- Clear and easy to navigate user interface.
- Cleanly written code which is free of errors.

Unit 4 Lesson 12 - Activity

Steps for today:

Step 1: Brainstorm App Ideas
Step 2: Choose One Idea
Step 3: Survey Your Classmates
Step 4: Storing Information
Step 5: Flowchart
Step 6: Design User Interface

Wrap Up

Unit 4 - Lesson 13
Project - Decision Maker App Part 2

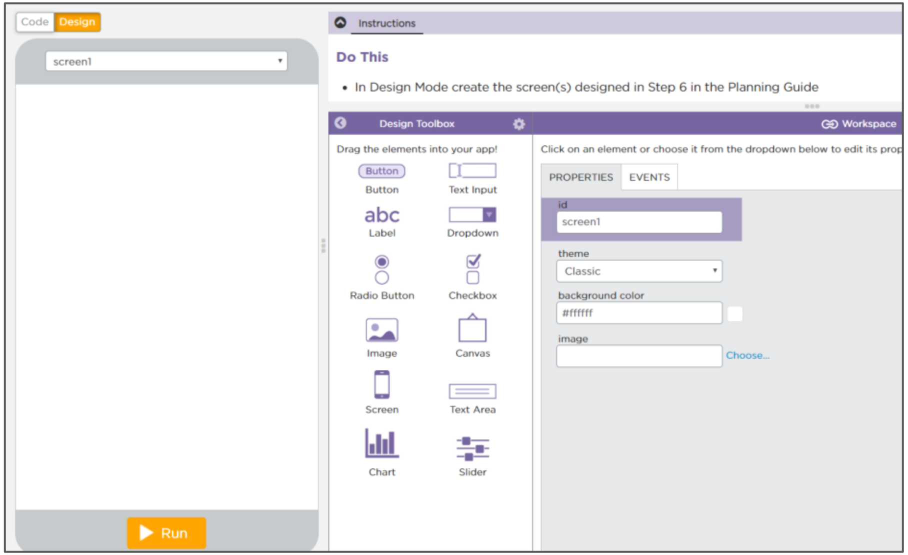
Warm Up

● ○ ○

Activity

●●○

Build the Decision Maker App



Do This

- In Design Mode create the screen(s) designed in Step 6 in the Planning Guide

Drag the elements into your app!

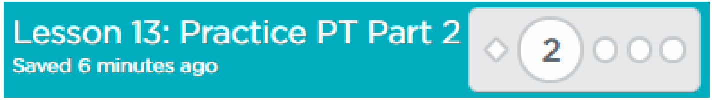
Click on an element or choose it from the dropdown below to edit its prop

id: screen1

theme: Classic

background color: #ffffff

image:



Lesson 13: Practice PT Part 2

Saved 6 minutes ago

Do This:

- Navigate to Lesson 13, Level 2 on Code Studio

Unit 4 Lesson 12 - Activity

Steps for today:

Level 2: Design Mode

Level 3: Create the Variables

Level 4: Create the function

Level 5: Add onEvents

Debugging: the process of finding and fixing problems in code

Describe
The Problem

What do you expect it to do?
What does it actually do?
Does it always happen?

Hunt
For Bugs

Are there warnings or errors?
What did you change most recently?
Explain your code to someone else
Look for code related to the problem

Try
Solutions

Make a small change

Document
As You Go

What have you learned?
What strategies did you use?
What questions do you have?

Wrap Up

Unit 4 - Lesson 14
Project - Decision Maker App Part 3

Warm Up

●○○

Activity

●●○

Do This: Divide into groups of 3-4

Test the Decision Maker App

Lesson 14: Practice PT Part 3

Saved a few seconds ago

Do This: Navigate to Lesson 13, Level 2 on Code Studio

Testing Instructions

Do This:

- Each app is tested by at least **two** other students
- The creator of the app watches others use the app and records feedback from the testers and things the creator noticed while observing someone else using the app.
    - **For example:** the creator of the app may notice that the user has difficulty figuring out which button to click on the app to make it run. The creator notes this down in the Planning Guide.

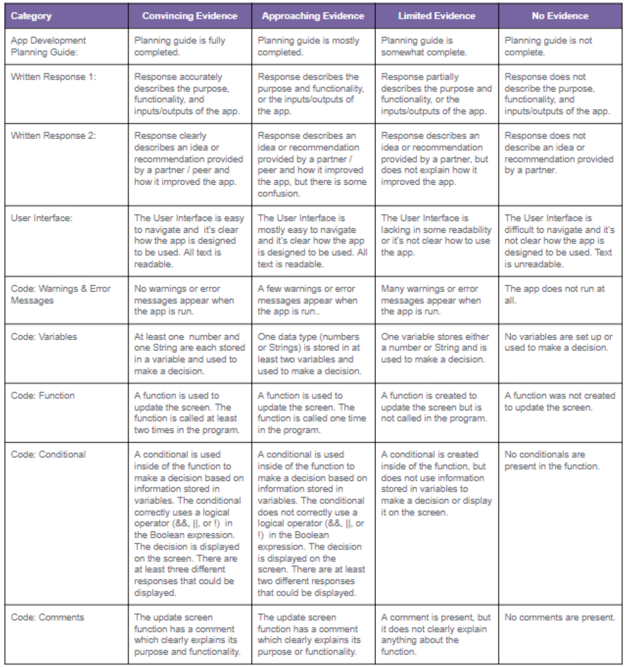
Finish Your App

Do This:

- Complete Steps 9 & 10:
  - Pick Improvements (Planning Guide)
  - Complete Your App (Level 2)

- Check the rubric on Level 3 before submitting!

Submit



Category	Convincing Evidence	Approaching Evidence	Limited Evidence	No Evidence
App Development Planning Guide:	Planning guide is fully completed.	Planning guide is mostly completed.	Planning guide is somewhat complete.	Planning guide is not complete.
Written Response 1:	Response accurately describes the purpose, functionality, and inputs/outputs of the app.	Response describes the purpose and functionality, or the inputs/outputs of the app.	Response partially describes the purpose and functionality, or the inputs/outputs of the app.	Response does not describe the purpose, functionality, and inputs/outputs of the app.
Written Response 2:	Response clearly describes an idea or recommendation provided by a partner / peer and how it improved the app.	Response describes an idea or recommendation provided by a partner / peer and how it improved the app, but there is some confusion.	Response describes an idea or recommendation provided by a partner, but does not explain how it improved the app.	Response does not describe an idea or recommendation provided by a partner.
User Interface:	The User Interface is easy to navigate and it's clear how the app is designed to be used. All text is readable.	The User Interface is mostly easy to navigate and it's clear how the app is designed to be used. All text is readable.	The User Interface is lacking in some readability or it's not clear how to use the app.	The User Interface is difficult to navigate and it's not clear how the app is designed to be used. Text is unreadable.
Code: Warnings & Error Messages	No warnings or error messages appear when the app is run.	A few warnings or error messages appear when the app is run..	Many warnings or error messages appear when the app is run.	The app does not run at all.
Code: Variables	At least one number and one String are each stored in a variable and used to make a decision.	One data type (numbers or Strings) is stored in at least two variables and used to make a decision.	One variable stores either a number or String and is used to make a decision.	No variables are set up or used to make a decision.
Code: Function	A function is used to update the screen. The function is called at least two times in the program.	A function is used to update the screen. The function is called one time in the program.	A function is created to update the screen but is not called in the program.	A function was not created to update the screen.
Code: Conditional	A conditional is used inside of the function to make a decision based on information stored in variables. The conditional correctly uses a logical operator (&&, ||, or !) in the Boolean expression. The decision is displayed on the screen. There are at least three different responses that could be displayed.	A conditional is used inside of the function to make a decision based on information stored in variables. The conditional does not correctly use a logical operator (&&, ||, or !) in the Boolean expression. The decision is displayed on the screen. There are at least two different responses that could be displayed.	A conditional is created inside of the function, but does not use information stored in variables to make a decision or display it on the screen.	No conditionals are present in the function.
Code: Comments	The update screen function has a comment which clearly explains its purpose and functionality.	The update screen function has a comment which clearly explains its purpose or functionality.	A comment is present, but it does not clearly explain anything about the function.	No comments are present.

Wrap Up

Complete the Reflection Questions in the Planning Guide:

- **Question 1:** Provide a written response that:
  - describes the overall purpose of the program
  - describes the functionality of your app
  - describes the input and outputs of your app (Approx 150 words)

- **Question 2:** This project was created using a development process that required you to incorporate the ideas of your partner and feedback from your classmates. Provide a written response that describes one part of your app that was improved through input from EITHER your partner or feedback you received from classmates. Include:
  - Who specifically provided the idea or recommendation
  - What their idea or recommendation was
  - The specific change you made to your app's user interface or functionality in response to the recommendation
  - How you believe this change improved your app (Approx 150 words)

Unit 4 - Lesson 15
Assessment Day

Activity

●●○

Unit Assessment

Unit Assessment