INTRO TO JS

LEARNING OBJECTIVES

- Describe event-based web site behavior and list practical uses of JavaScript on a website's front end.
- Practice programmatic thinking by writing pseudocode.
- Define variables and identify best cases to use them.
- Describe strings, numbers, and boolean variable types.
- Use comparison operators to evaluate and compare statements.
- Apply conditionals to change the program's control flow

INTRO TO JS

DIRECTIONS

1. Find a new partner. Someone you haven't yet worked with!

2. Identify one person to DRAW and one person to TALK.

3. The person TALKING chooses a shape or object (circle, star, heart, slice of pizza, motorcycle etc).

4. Using explicit step-by-step directions, the talker must get their partner to draw the correct shape without ever stating what that shape is.

5. The person drawing must follow directions exactly, and **may not draw anything that their partner did not say exactly.**

INTRO TO PROGRAMMING

INTRO TO PROGRAMMING

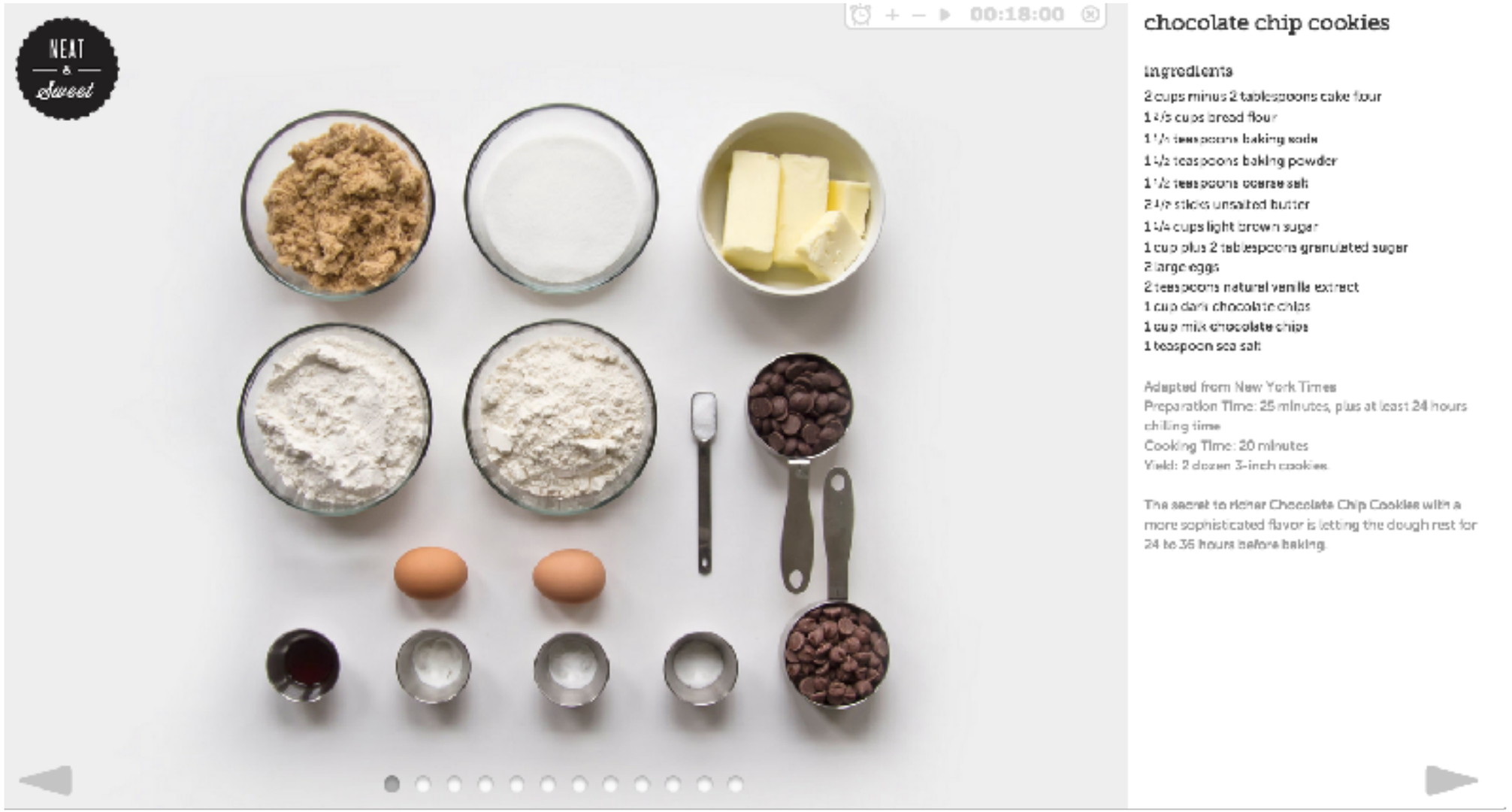
WHAT IS A PROGRAM?

- A program is a set of instructions that tell a computer what to do

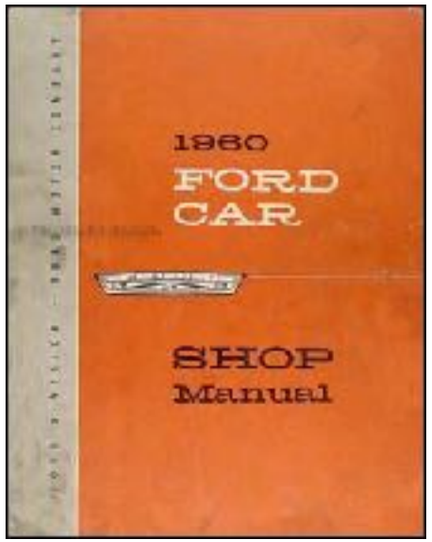
WHAT IS PROGRAMMING?

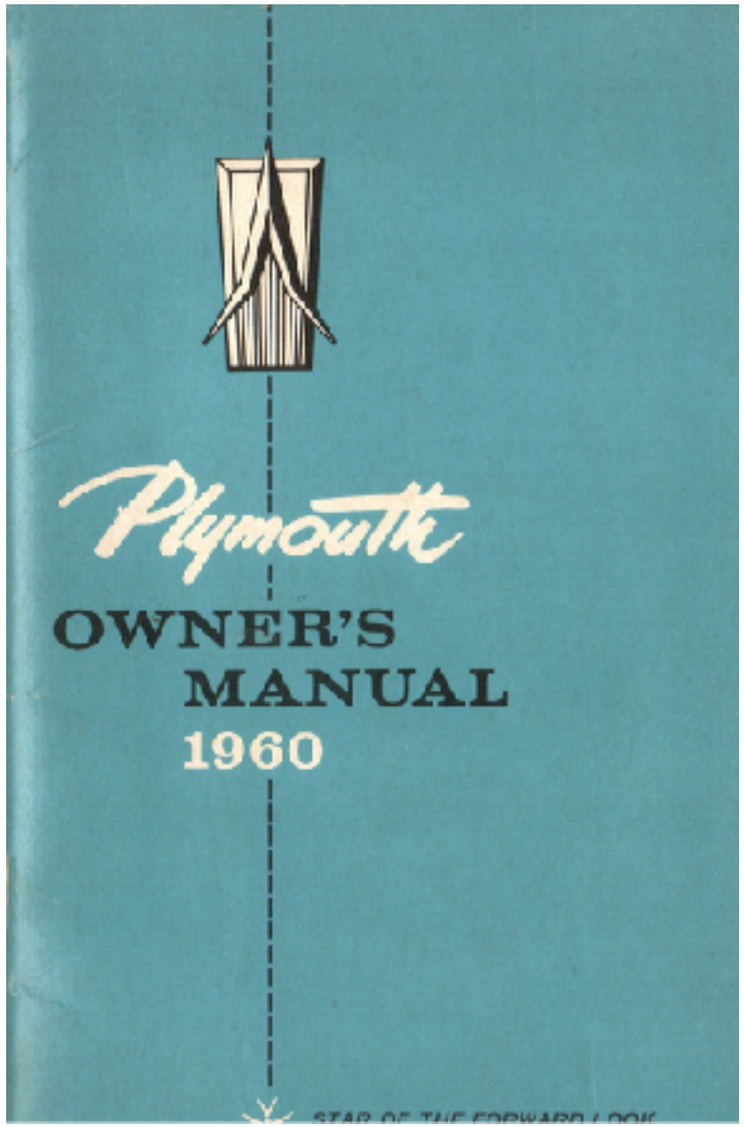
- Programming is the task of writing those instructions in a language that the computer can understand.

WHAT IS A PROGRAM?



WHAT IS A PROGRAM?





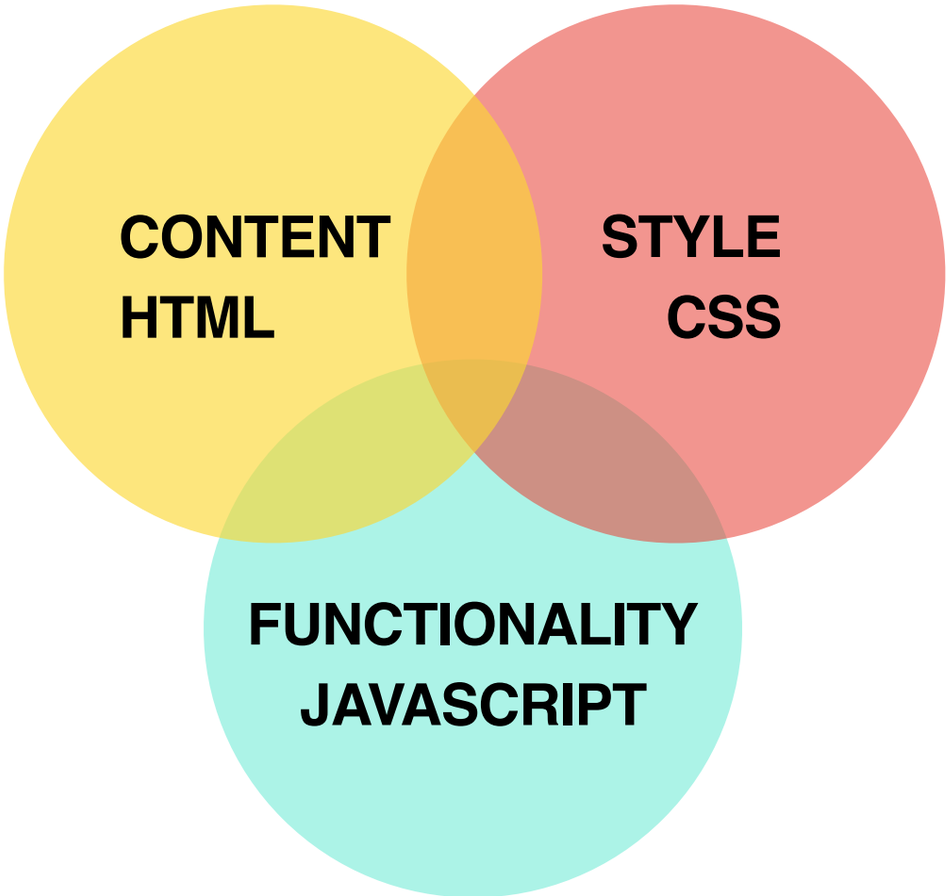
HOW COMPUTERS 'THINK'

SO HOW DO COMPUTERS THINK?

- Short answer — they don’t think!
- While computers don’t think, they act as if they do, by sequentially executing simple instructions.
- The only things a computer knows are the things we tell it.
- A computer doesn't learn to perform tasks like you and I — it needs to follow instructions every time it performs the task.

INTRO TO JS

THE THREE AMIGOS: STRUCTURE, STYLE, BEHAVIOR



JAVA VS. JAVASCRIPT

We're learning JavaScript in this class, not Java.

Java and JavaScript are completely different languages.

WHAT JAVASCRIPT CAN DO!

WHAT JAVASCRIPT CAN DO — ACCESS AND MODIFY CONTENT

We can use JavaScript to access and modify content on the page.

Note: we will usually do this in response to something the user does on the page, instead of when the page loads. But we'll get there!

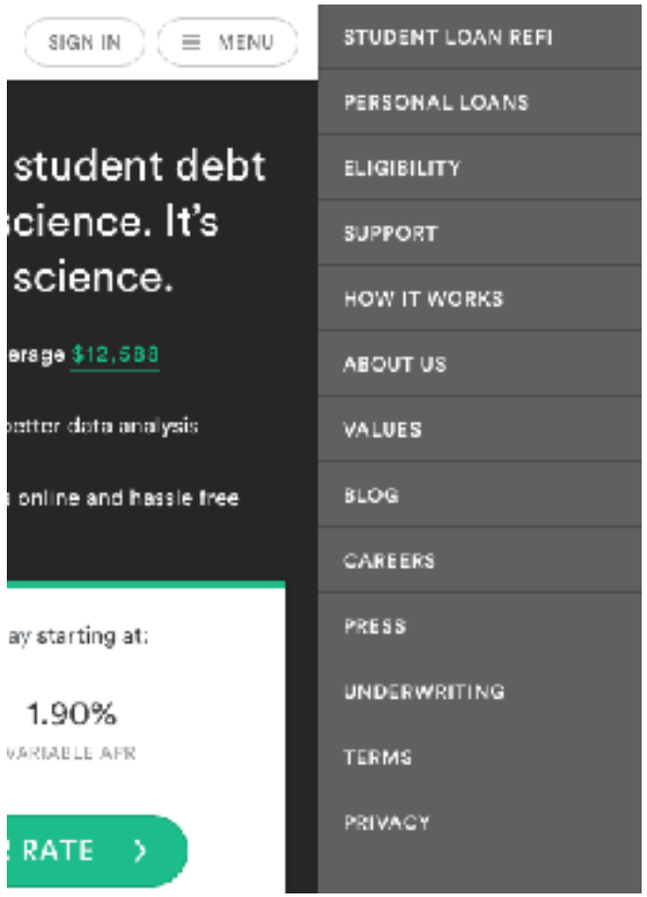
ACCESSING CONTENT:

- Find out what the user typed into the password field.
- Check to see whether the "remember me" checkbox is checked or not.

MODIFYING CONTENT:

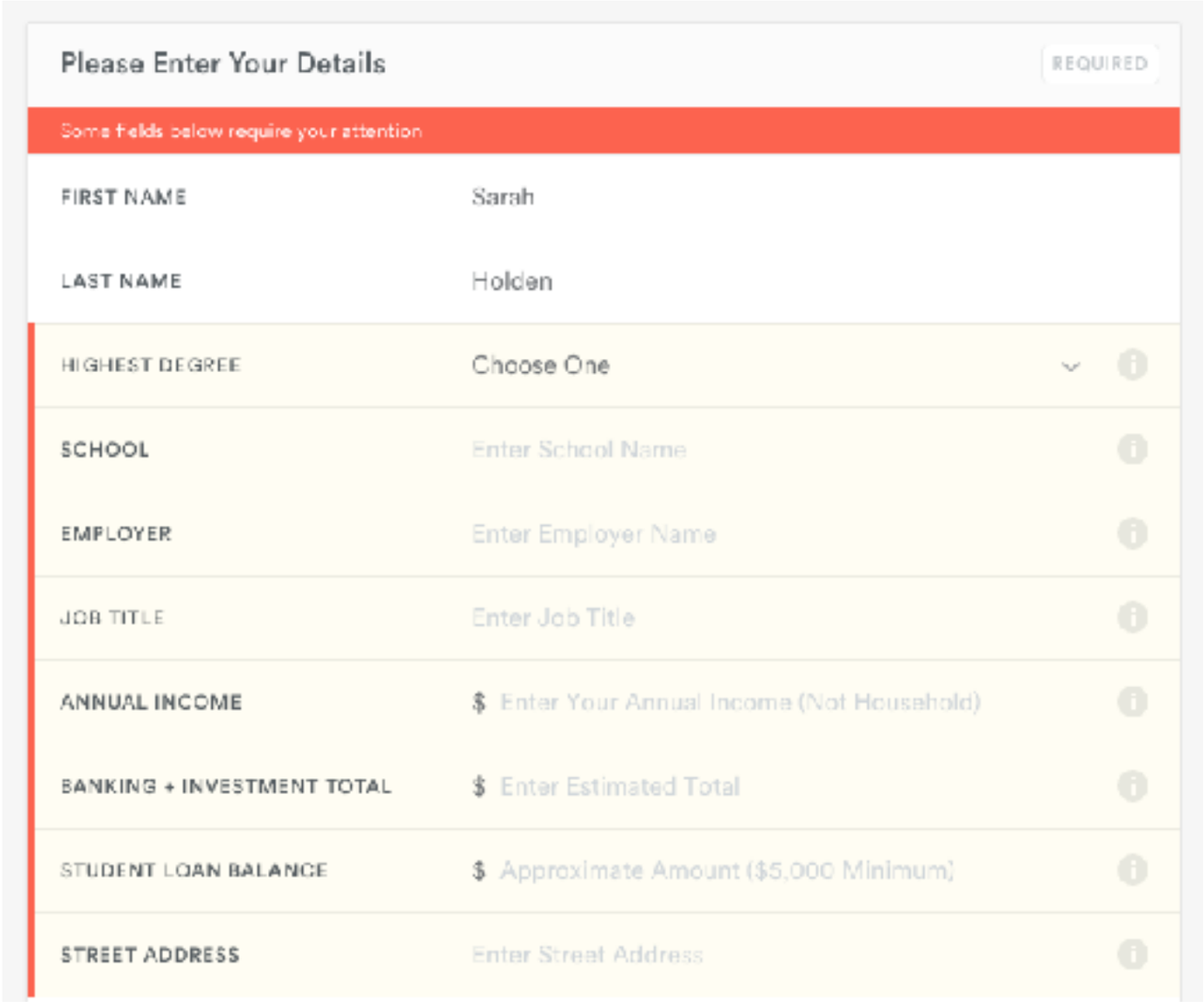
- Slide the nav out from the side of a page.
- Fade in a paragraph
- Fade in an image
- Add a red border to the password field

WHAT JAVASCRIPT CAN DO - MODIFYING CONTENT



Change the size, position, color, or other styles for an element

WHAT JAVASCRIPT CAN DO - MODIFYING CONTENT



Add an error message (and styles) to a form

WHAT JAVASCRIPT CAN DO — REACT TO EVENTS

JavaScript is referred to as an event-driven language.

We can wait, or listen, for the user to do something and then respond to that action.

We can write a series of instructions that will happen when an event occurs.

EVENT	ELEMENT	STEPS TO TAKE
click	button	***Slide in the*** nav ***from the side of the page***
mouseover	anchor	***Fade in a paragraph***
scroll	window	***If the user has scrolled 500px or more, fade in an image***
keyup	input	***Check to see if the input has enough characters and add a red border to the field if not***

WHAT JAVASCRIPT CAN DO — ADD LOGIC

We can use conditionals to make our pages seem smart and responsive.

Examples:

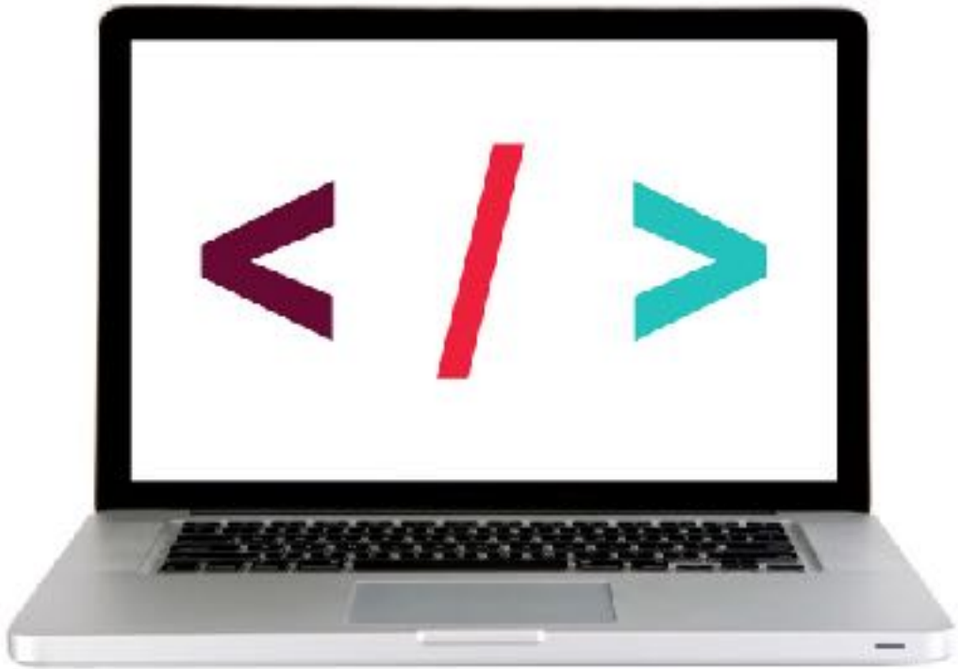
```
// when the user scrolls inside the browser window
   // if the user has scrolled 500px or more
      // fade in the image
   // else
      // fade out the image


// when the user tabs or clicks away from the password field
   // if the user has entered at least 8 characters
      // add a green border to the password field
   // else
      // add a red border to the password field
      // add an error message
```

WHAT JAVASCRIPT CAN DO — COMMUNICATE WITH THE SERVER

JavaScript can:

1. Tell the server that a user has done something.
2. Tell the server to save that interaction in a database.
3. Display the results of that interaction to all other users.

LET'S TAKE A LOOK



https://kinhr.com/

INTRO TO PSEUDO CODE

PSEUDO CODE

When we write a program, we need to figure out a way to translate the ideas that are in our heads into code.

We can use pseudo code to help us do this.

Pseudo code is:

- A detailed yet readable description of what a computer program must do
- Written in plain english rather than in a programming language
- A way to "plan out" your program before coding it

THE IMPORTANCE OF PLANNING

PSEUDO CODE — CONDITIONS

```
// if _____
  // Steps to take if the condition is true (indented)
// else
  // Steps to take if the condition is false
```

PSEUDO CODE — PASSING SCORE

PSEUDO CODE — PASSING SCORE

```
// player score = 130

// if the player's score is greater than 120
   // player has passed level
// else
   // player should retry level
```

PSEUDO CODE — EVENTS

```
// WHEN the user _____ the _______
    // steps to take when that event occurs
    // all steps should be indented
```

```
// WHEN the user clicks the button
   // fade in the image
   // display a message that says "You win!"
```

PSEUDO CODE — ALL TOGETHER NOW!

```
// WHEN the user _____ the _______
    // if _____
        // Steps to take if condition is true
    // else
        // Steps to take if condition is false
```

```
// WHEN the user scrolls in the browser window
   // if the user has scrolled more than 500px
        // Fade in the image
   // else
        // Fade out the image
```

LAB — SURVEY

KEY OBJECTIVE

- Practice programmatic thinking by writing pseudo code to solve a basic problem

TYPE OF EXERCISE

- Group of 2

TIMING

5 min

1. Image we have a simple webpage with a button, a ring, and Frodo.
2. Every time users click on the button, the Ring of Power gets put on Frodo's finger, which makes Frodo disappear. If users click on the button again, the Ring of Power gets taken off of Frodo's finger, and Frodo reappears.

VARIABLES

EXERCISE — READING AND GUESSING

KEY OBJECTIVE

- Read a sample JavaScript file and see if you can guess what will happen.

TYPE OF EXERCISE

- Reading exercise (Groups of 3 - 4)

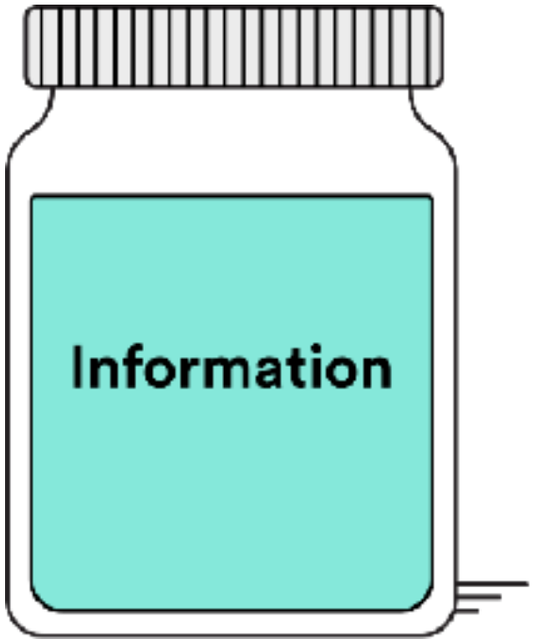
EXECUTION

1 min

1. Follow the instructions in this JS Bin
2. After reading the instructions, hit the "Run" button in the console tab to confirm your guesses.

WHAT ARE VARIABLES?

- We can tell our program to remember (store) values for us to use later on.
- The "container" we use to store the value is called a variable



CREATING VARIABLES

DIRECTIONS

1. We'll be using the console to practice creating variables. It's where JavaScript is interpreted and run. You can use it to practice writing JavaScript!
2. Open a new JS Bin session and select the "Console" tab. You can close any other tabs.

DECLARING A VARIABLE

```
var age = 29;
```

VARIABLE CONVENTIONS

RULES:

1. Should be "camel case" — First word starts with a lowercase letter and any following words start with an uppercase letter.
2. Names can only contain: letters, numbers, $ and _
3. No dashes, no periods.
4. Cannot start with a number
5. Case sensitive - numberofstudents is not the same as numberOfStudents


```
var numberOfStudents = 10;
```

Guideline: Names should be descriptive:


```
var lastName = "Sanchez";
```

```
var x = "Sanchez";
```

JAVASCRIPT — UPDATING THE VALUE OF A VARIABLE

Declaring a variable:

```
var champion = "Max";
```

Update the value of the variable:

```
champion = "Madeline";
```

ASSIGNMENT OPERATORS

ASSIGNMENT OPERATORS

	Initial Value	Operator	Example	Result
Assign value to variable	var num = 8	=	num = 6	6
Add value to variable	var num = 8	+=	num += 6	14
Subtract value from variable	var num = 8	-=	num -= 6	2

ASSIGNMENT OPERATORS

Question: What will the variable totalAmount hold after these lines run?

```
var totalAmount = 6;
totalAmount += 4;
totalAmount -= 2;
```

ASSIGNMENT OPERATORS

Question: What value will the variable score hold after these lines run?

```
var score = 6;
score + 2;
```

EXERCISE — VARIABLES

KEY OBJECTIVE

- Practice declaring and assigning variables

TYPE OF EXERCISE

- Individual/paired

LOCATION

- Score Keeper (Codepen)

EXECUTION

5 min

1. Hook up the +10, -1 and -5 buttons

DATA TYPES

WHAT CAN BE STORED IN VARIABLES?

DATA TYPES:

1. Numeric	2. String	3. Boolean
Handles numbers	Consists of letters and/or other characters	Handles true or false values
Ex: 200.54 Ex: 893	Ex: 'GA@ga.co' Ex: "How are you user?"	Ex: true Ex: false
Used for tasks that involve counting or calculating	Used when working with any kind of text Written with single or double quotes	Used when there are two options for a value (i.e. yes/no, on/off, true/false)

NUMBERS

MORE ABOUT NUMBERS

```
10
```

Whole numbers

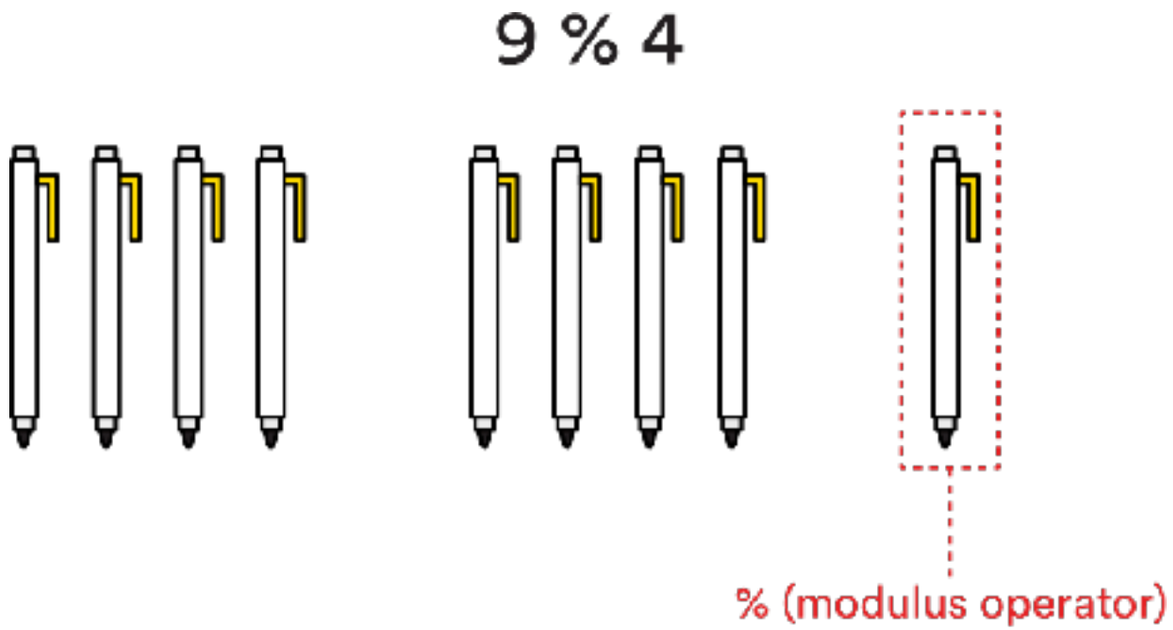
```
22.75
```

Number that uses a decimal to represent a fraction

ARITHMETIC OPERATORS

	Operator	Example	Result
Addition	+	2 + 4	6
Subtraction	-	8 - 1	7
Multiplication	*	2 * 3	6
Division	/	4 / 2	2
Modulus	%	4 % 2	0

ARITHMETIC OPERATORS



STRINGS

MORE ABOUT STRINGS

A STRING:

- Stores textual information
- Is surrounded by quotes

```
"How is the weather today?"
```

```
'Cold'
```

STRING CONCATENATION

- To take two strings (or a combination of strings and variables) and stick them together, use the **+** operator.
- This is called string concatenation.

```
var name = "Suzie Smith";
var greeting = "Hello " + name;
// greeting will be: "Hello Suzie Smith"
```

QUIZ

COMMON MISTAKES

"Bill" = var name;

COMMON MISTAKES

```
var name = "Bill";
```

COMMON MISTAKES

```
var total score = 20;
```

COMMON MISTAKES

```
var totalScore = 20;
```

COMMON MISTAKES

```
var fullName = Suzie Smith;
```

COMMON MISTAKES

```
var fullName = "Suzie Smith";
```

COMMON MISTAKES

```
Var fullName = "Bill Smith";
```

COMMON MISTAKES

```
var fullName = "Bill Smith";
```

COMMON MISTAKES

```
var score = "5";
score += "6";
```

COMMON MISTAKES

```
var score = 5;
score += 6;
```

EXERCISE — VARIABLES

KEY OBJECTIVE

- Practice writing conditionals

TYPE OF EXERCISE

- Individual/paired

EXECUTION

6 min

1. Follow the instructions under "Part 1" in this JS Bin.
2. Keep this tab open.

EXERCISE — VARIABLES

KEY OBJECTIVE

- What are variables? Why would we want to use variables?

TYPE OF EXERCISE

- Turn and Talk

EXECUTION

30 sec

1. Describe variables. Explain why we would want to use variables in our programs.
2. What are the three data types in JS? Can you think of an example of each?

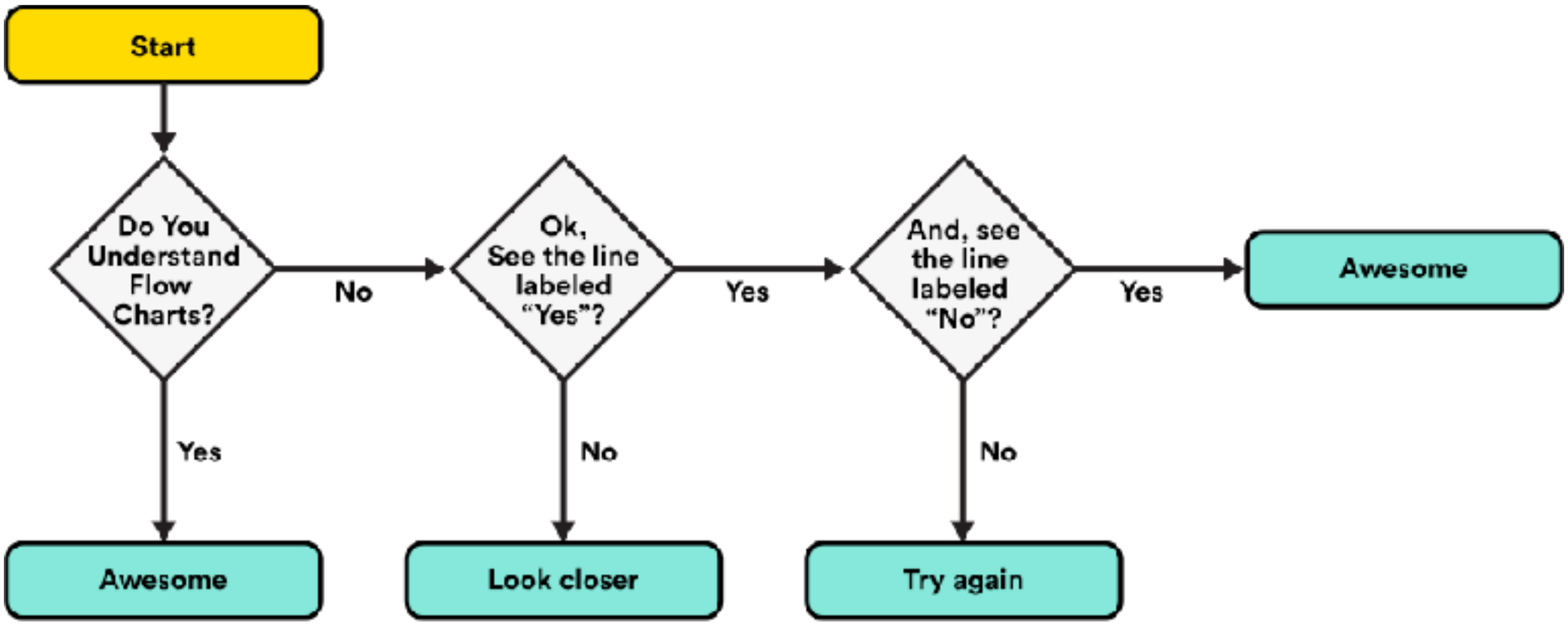
LEARNING OBJECTIVES

- Describe event-based web site behavior and list practical uses of JavaScript on a website's front end.
- Practice programmatic thinking by writing pseudocode.
- Define variables and identify best cases to use them.
- Describe strings, numbers, and boolean variable types.
- Use comparison operators to evaluate and compare statements.
- Apply conditionals to change the program's control flow

CONTROL FLOW

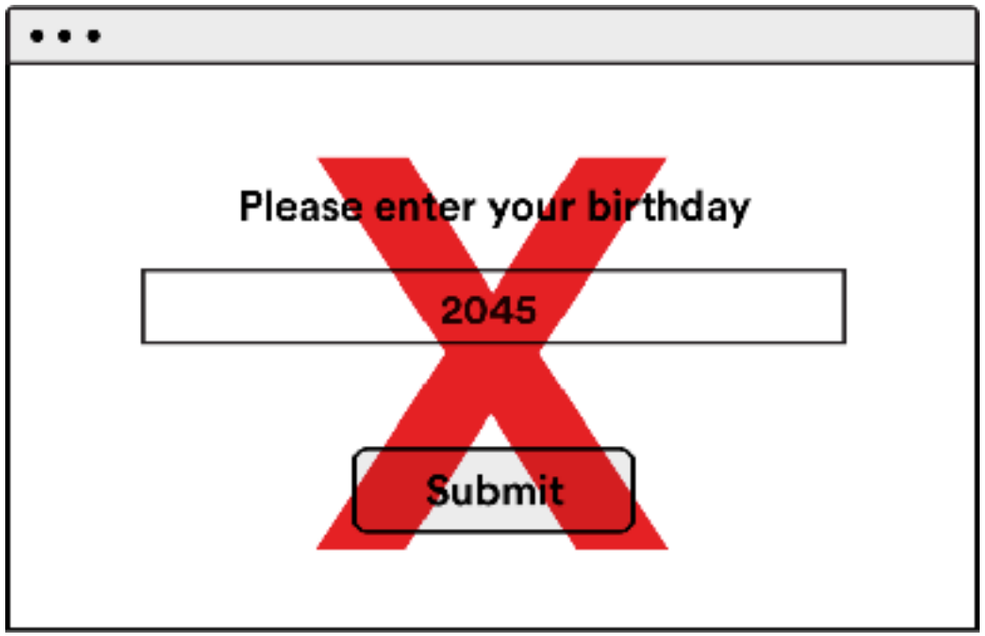
IF STATEMENTS

CONTROL FLOW



CONDITIONAL LOGIC

In JavaScript (and coding in general) you'll need to make comparisons all the time:



Is the year less than or equal to 2017 and more than or equal to 1900? If the answer to this question is "true," then we know the user has entered a valid year, and that he or she was born somewhere between 1900 and 2017.

COMPARISON OPERATORS

COMPARISON OPERATORS

Comparison Operators	
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

COMPARISON OPERATORS

Equality Operators	
`===`	Strict equal to
`==`	Equal to
`!==`	Strict not equal to
`!=`	Not equal to

COMPARISON OPERATORS

```
7 === 7  //true
7 === "7"  //false
0 === false  //false
false === "false"  //false
```

[This is because a string and number are not the same]

[0 is always false in boolean statements]

['false' is a string and JS will not evaluate strings that has the words false as false]

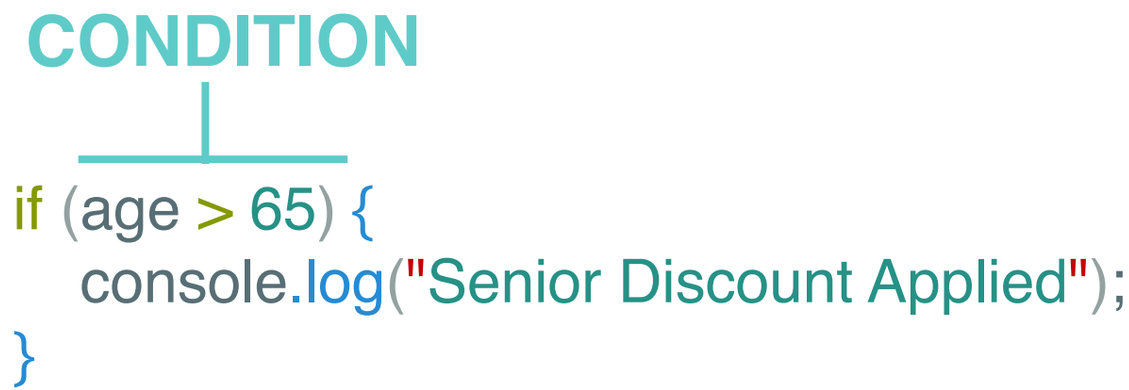
ASSIGNMENT VS. COMPARISON — DON'T GET THEM CONFUSED!

Assignment	Comparison
=	==

var number = 7;	if (number === 8) {
	  // Do something
	}

CONDITIONAL STATEMENTS

JAVASCRIPT — IF STATEMENT



JAVASCRIPT — IF STATEMENT

CONDITION

```
if (answer === 38) {
    console.log("Your answer is correct!");
}
```

JAVASCRIPT — IF STATEMENT

COMPARISON OR EQUALITY OPERATOR

```
if (age > 65) {
    console.log("Senior Discount Applied");
}
```

Comparison Operators	
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

Equality Operators	
===	Strict equal to
==	Equal to
!==	Strict not equal to
!=	Not equal to

ELSE IF STATEMENTS

```
if (age > 65) {
    console.log("Senior Discount Applied");

} else if (age < 18) {
    console.log("Student Discount Applied");

}
```

ELSE IF STATEMENT

IF STATEMENTS

```
if (age > 65) {
    console.log("Senior Discount Applied");

} else if (age < 18) {
    console.log("Student Discount Applied");

} else {
      console.log("Sorry, you don't qualify for a discount");
}
```

ELSE STATEMENT

LOGICAL OPERATORS

MULTIPLE CONDITIONS

&&	and
||	or
!	not

MULTIPLE CONDITIONS

```
if (name === "GA"  && password === "YellowPencil"){
    //Allow access to dashboard
}
```

MULTIPLE CONDITIONS

```
if (name === "GA"  && password === "yellowpencil" || name === "GA-GUEST"  && password === "workshop"){
    //Allow access to dashboard
}
```

EXERCISE — CONDITIONALS

KEY OBJECTIVE

- Practice writing conditionals

TYPE OF EXERCISE

- Individual/paired

EXECUTION

6 min

1. Follow the instructions under "Part 2" in this JS Bin.

LAB

LAB — ICE CREAM

KEY OBJECTIVE

- Practice writing JS to decide whether or not to buy an ice cream cone.

TYPE OF EXERCISE

- Individual/Partner

EXERCISE

1. Follow the steps in this JS Bin.

LEARNING OBJECTIVES

- Describe event-based web site behavior and list practical uses of JavaScript on a website's front end.
- Practice programmatic thinking by writing pseudocode.
- Define variables and identify best cases to use them.
- Describe strings, numbers, and boolean variable types.
- Use comparison operators to evaluate and compare statements.
- Apply conditionals to change the program's control flow

EXIT TICKETS