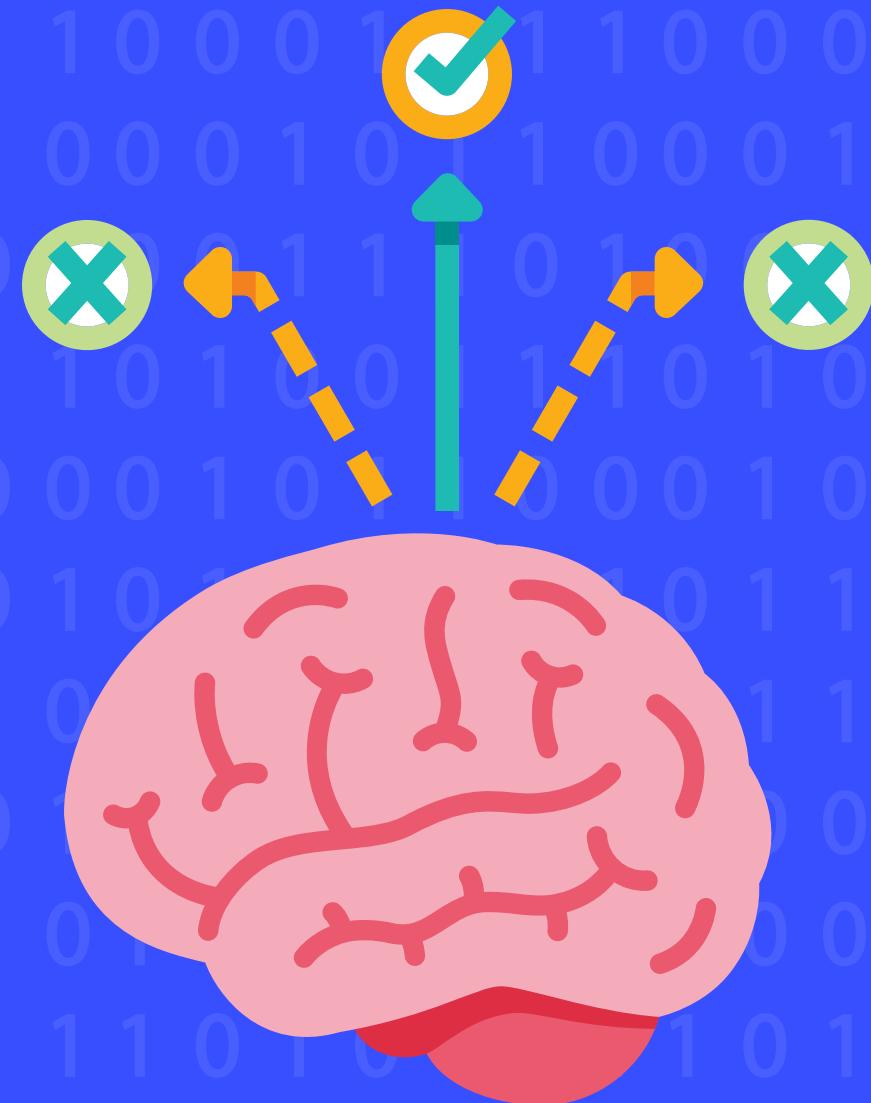


BOOLEAN LOGIC



GOALS

- Understand Comparison Operators
- Write Conditionals
- Work with Boolean Operators

COMPARISONS



```
> // greater than  
< // less than  
>= // greater than or equal to  
<= // less than or equal to  
== // equality  
!= // not equal  
=== // strict equality  
!== // strict non-equality
```

COMPARISONS



> // greater than

< // less than

>= // greater than or equal to

<= // less than or equal to

= // equality

!= // not equal

== // strict equality

IGNORE THESE FOR NOW

!== // strict non-equality

SOME EXAMPLES



```
10 > 1;      //true
0.2 > 0.3;   //false
-10 < 0;     //true
50.5 < 5;    //false
0.5 <= 0.5;  //true
99 >= 4;     //true
99 >= 99;    //true
'a' < 'b';   //true
'A' > 'a';   //false
```

Notice these all return a Boolean!

Though it's uncommon, you can compare strings. Just be careful, things get dicey when dealing with case, special characters, and accents!

COMPARISONS



> // greater than

< // less than

>= // greater than or equal to

== // equality

!= // not equal

=== // strict equality

!== // strict non-equality

**IT'S TIME TO TALK
ABOUT EQUALITY!!**



= =
= =

VS



= =
= =
= =

`==` (double equals)

- Checks for equality of value, but not equality of type.
- It coerces both values to the same type and then compares them.
- This can lead to some unexpected results!

== (double equals)



```
5 == 5;          //true
'b' == 'c';     //false
7 == '7';        //true
0 == '';         //true
true == false;  //false
0 == false;      //true
null == undefined; //true
```

==== (triple equals)

Checks for equality of value AND type



```
5 === 5; //true  
1 === 2; //false  
2 === '2'; //false  
false === 0; //false
```

//Same applies for != and !==

```
10 != '10'; //false  
10 !== '10'; //true
```



GO WITH
TRIPLE
EQUALS!

console.log()

prints arguments to the console

(we need this if we're going to start working with files!)



Running Code From a File

app.js

```
● ● ●  
//Put your code in the JS File  
alert('Hello from JS!');  
  
//Won't show up!!  
"hi".toUpperCase();  
  
//Will show up!  
console.log("hi".toUpperCase());
```

demo.html

```
● ● ●  
<!DOCTYPE html>  
<html>  
<head>  
  <title>JS Demo</title>  
  <script src="app.js"></script>  
</head>  
<body>  
  
</body>  
</html>
```

Write your code
in a .js file

Include your script
in a .html file

CONDITIONALS

Making decisions with code



IF

Run code *if* a given condition is true



```
let rating = 3;  
  
if (rating === 3) {  
  console.log("YOU ARE A SUPERSTAR!");  
}
```

ELSE IF

If not the first thing, maybe this other thing??



```
let rating = 2;

if (rating === 3) {
  console.log("YOU ARE A SUPERSTAR!");
}
else if (rating === 2) {
  console.log("MEETS EXPECTATIONS");
}
```

ELSE IF

You can have multiple *else ifs*



```
let rating = 1;

if (rating === 3) {
  console.log("YOU ARE A SUPERSTAR!");
}
else if (rating === 2) {
  console.log("MEETS EXPECTATIONS");
}
else if (rating === 1) {
  console.log("NEEDS IMPROVEMENT");
}
```

ELSE

"If nothing else was true, do this..."

```
let rating = -99;

if (rating === 3) {
  console.log("YOU ARE A SUPERSTAR!");
}
else if (rating === 2) {
  console.log("MEETS EXPECTATIONS");
}
else if (rating === 1) {
  console.log("NEEDS IMPROVEMENT");
}
else {
  console.log("INVALID RATING!");
}
```

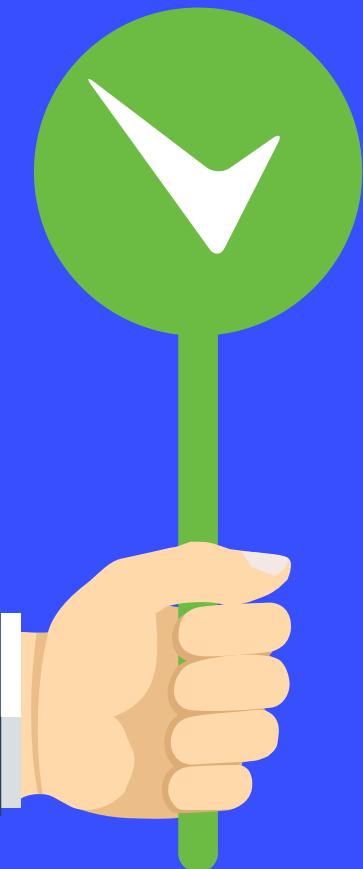
NESTING



We can nest conditionals inside conditionals

```
● ● ●  
  
let password = "cat dog";  
if (password.length >= 6) {  
  if (password.indexOf(' ') !== -1) {  
    console.log("Password cannot include spaces");  
  }  
  else {  
    console.log("Valid password!!")  
  }  
}  
else {  
  console.log("Password too short!");  
}
```

TRUTHY & FALSY VALUES



- All values have an inherent truthy or falsy boolean value
- Falsy values:
 - `false`
 - `0`
 - `""` (empty string)
 - `null`
 - `undefined`
 - `NaN`
- Everything else is truthy!



LOGICAL OPERATORS

& &

AND

| |

OR

!

NOT

AND (&&)

Both sides must be true in order
for the whole thing to be true



```
1 <= 4 && 'a' === 'a'; //true
```

```
9 > 10 && 9 >= 9; //false
```

```
'abc'.length === 3 && 1+1 === 4; //false
```

AND (&&)

Both sides must be true in order
for the whole thing to be true



```
let password = 'taco tuesday';

if(password.length >= 6 && password.indexOf(' ') === -1){
    console.log("Valid Password!");
}
else {
    console.log("INVALID PASSWORD!");
}
```

OR (||)

(pipe character above enter key)

If one side is true, the whole thing is true



//only one side needs to be true!

```
1 !== 1 || 10 === 10 //true
```

```
10/2 === 5 || null //true
```

```
0 || undefined //false
```

OR (||)



```
let age = 76;

if(age < 6 || age >= 65){
    console.log('You get in for free!');
}
else {
    console.log('That will be $10 please');
}
```

NOT (!)

! expression returns true if the expression is false



```
!null //true
```

```
! ( 0 === 0 ) //false
```

```
!( 3 <= 4 ) //false
```

OPERATOR PRECEDENCE

NOT (!) has higher precedence than && and ||
&& has higher precedence than ||

!

&&

||

You can alter this using parens ()

TERNARY OPERATOR

```
condition ? expIfTrue: expIfFalse
```