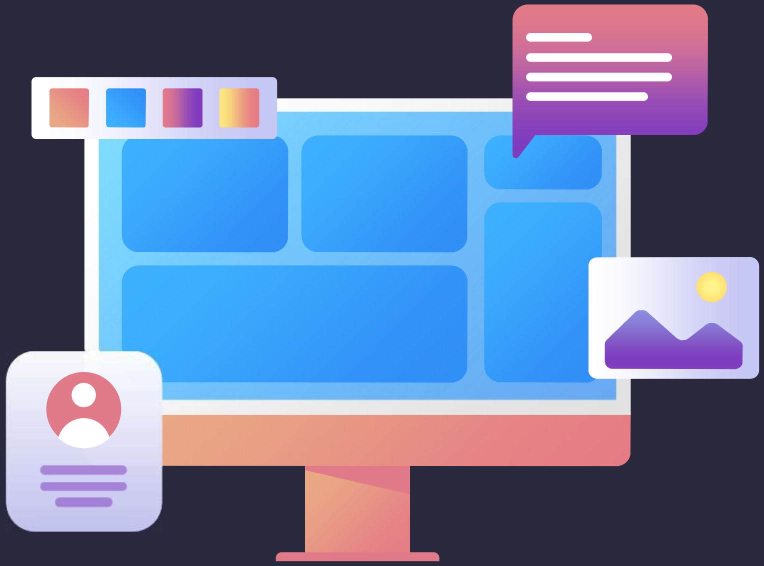
APPLESEEDS

JS CONDITIONS



APPLESEEDS

/TABLE OF CONTENTS

/01 What are conditions?

> = = vs = = =, booleans

/02 If statement

> Syntax, truthy and falsy, if else

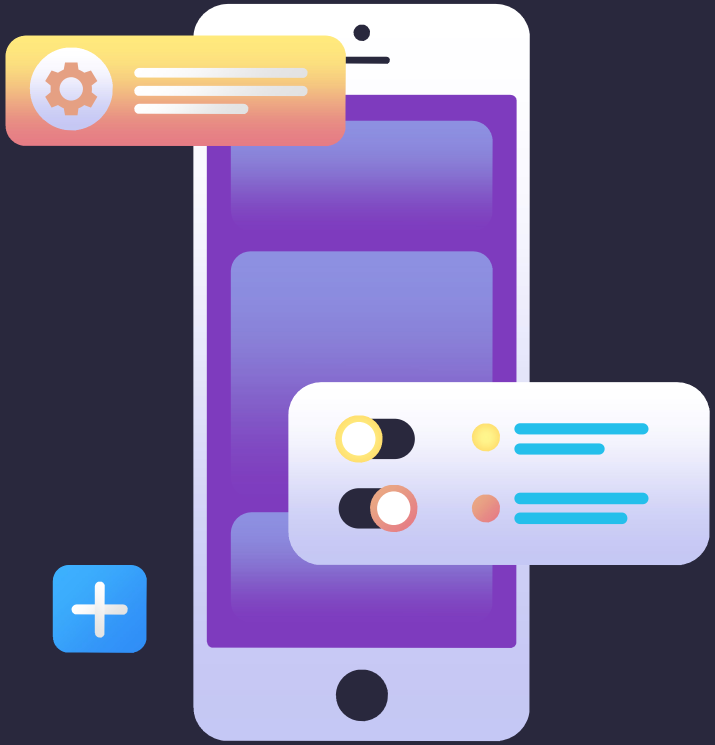
/03 Complex conditions

> Or, and, not

/04 Alternative if

> Tranary, switch

/01 What are conditions?



APPLESEEDS

What are conditions?

- We used js so far to run the code line by line.
- But what is we want some lines to execute only if some conditions are met?
- For example: if we have a string - if the string is longer than X: I want to print “ok” and if it’s not - print “not ok”

APPLESEEDS

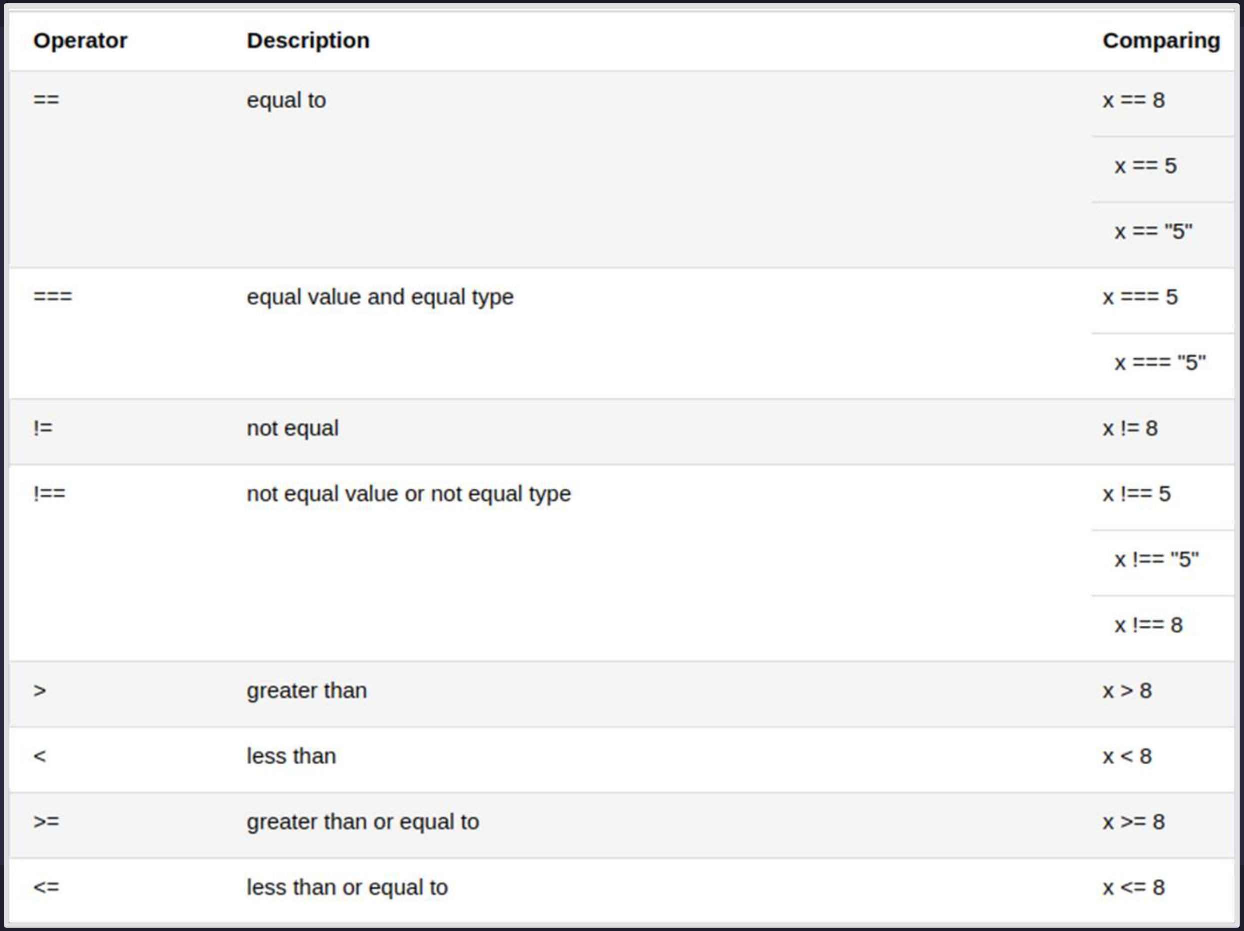
What are conditions?

- This is where conditions come in.
- Conditions allow us to check if some conditions is true or false - and based on that run specific code.
- **Every conditions must be resulted in true or false!**

What are conditions?

- What can we put inside a condition?
- Any logical operator: bigger than, smaller than, equal to, etc.
- How you write those expressions?

What are conditions?



Operator	Description	Comparing
==	equal to	x == 8
		x == 5
		x == "5"
===	equal value and equal type	x === 5
		x === "5"
!=	not equal	x != 8
!==	not equal value or not equal type	x !== 5
		x !== "5"
		x !== 8
>	greater than	x > 8
<	less than	x < 8
>=	greater than or equal to	x >= 8
<=	less than or equal to	x <= 8

APPLESEEDS

What are conditions?

- Let’s talk about equal. In JS there are 2 types of equal. Double equal (= =) and triple equal (= = =). What is the difference?
- Double equal just compares the value.
- Triple compares the value and the type.
- For example:

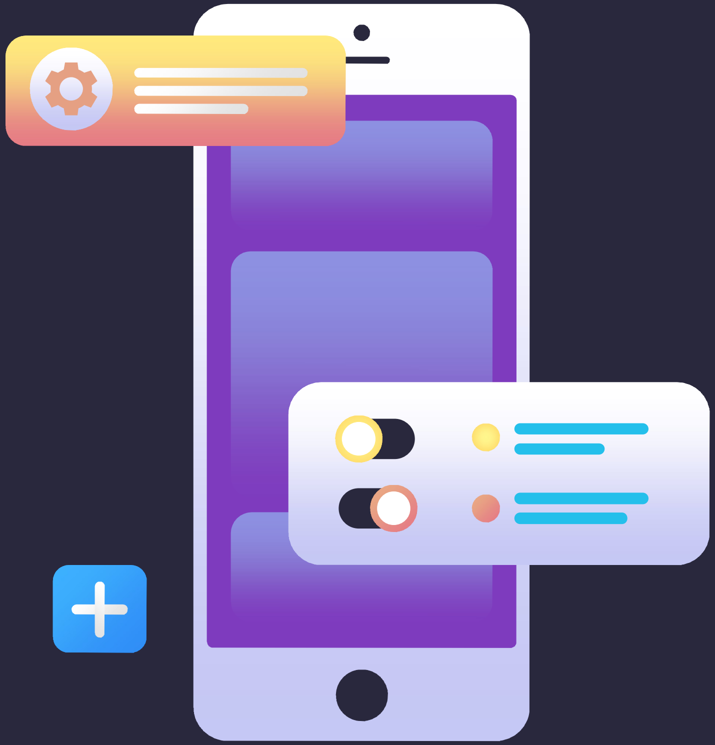
```
‘2’ = = 2    // true
‘2’ = = = 2  // false
```

What are conditions?

- Always use triple equal!
- Like const, we want to make our code with the minimum amount of bugs and errors, and using triple equal helps us to make sure that what we are comparing is really equal.

/02

If statement



APPLESEEDS

If statement

- The basic way to write a condition is with an “if statement”.
- Its syntax is like so:

```
if(condition){
   // code that runs if condition = true
}
```

APPLESEEDS

If statement

- For example:

```
const age = 19;

if(age > 18){
   console.log(‘you can vote’);
}
```

This will print, because age > 18 === true.

APPLESEEDS

If statement

- On the other hand:

```
const age = 17;
if(age > 18){
   console.log(‘you can vote’);
}
```

This will not print, because age > 18 === false

If statement

- As we said, the condition **always** results in true or false.
- But what happens if don’t use a condition with logical operators (like ===, or <=) but a variable?
- For example:
  Const name = ‘ori’
  if(name){ // do something if true }

APPLESEEDS

If statement

- What happens, basically, is that the JS engine **coerces** (like converts) the variable into a boolean value.
- The condition results will be **false** if the variable is equal to:
    - Empty string / ‘’/ “”/``
    - 0 / -0
    - false (boolean)
    - NaN/Null/undefined

APPLESEEDS

If statement

- For example:

```
const age = 0;
if(age){
    console.log('you are bigger than 0!');
}
```

  This will not print, because age is 0, and 0 is **falsy**, i.e - it turns into the boolean false by the JS engine.

APPLESEEDS

If statement

- On the other hand:

```
const name = ‘ori’
if(name){
   console.log(‘you have a name!’);
};
```

  This will print, because an none empty string is **truthy**, and it will be evaluated as true.

APPLESEEDS

If statement

- So we know how to run code if the condition is true.
- What if we want to run code if the condition is true, but run something else if the condition is false?
- For that we have - **if else**

APPLESEEDS

If statement

- The structure is like so:

```
if(condition){
   // code that runs if condition = true
} else {
   // code that runs if condition = false
}
```

APPLESEEDS

If statement

- For example:

```
const age = 17;
if(age > 18){
   console.log(‘can vote’)
}
else {
   console.log(‘cannot vote’)
}
```

This will print ‘cannot vote’

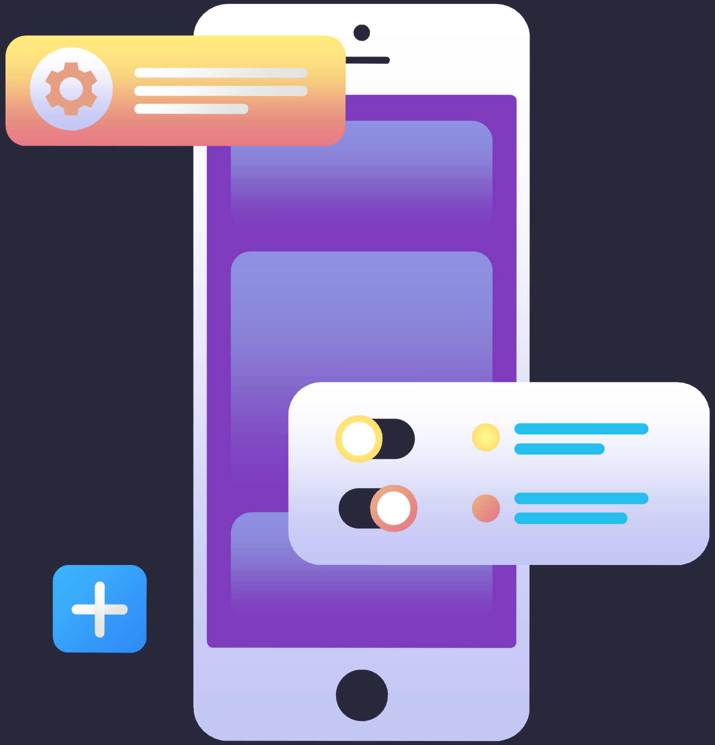
APPLESEEDS

If statement

- We can add more condition with every else.
  For example:

```
const age = 19;
if(age > 5){
    console.log(‘can go to school’)
}
else if(age > 14){
    console.log(‘can go to high school’)
} else {
    console.log(‘a baby’)
    // runs only if age < 14 and age <5
}
```

/03 Complex Conditions



Complex Conditions

- We can reverse the results of a logical comparison with !.
  For example, if I want to check that the variable name is not empty:
  - I can check (name ! = = ‘’)
  - I can check !(name)

APPLESEEDS

Complex Conditions

- For example:

```
const userName = ‘’;

if(!userName){
   console.log(‘no username!’)
}
```

APPLESEEDS

Complex Conditions

- Sometimes it’s easier to check if something is true, and else when its false
- And sometimes is nicer to check if something is false, and else if its true.
- Then if the 2nd is true, use !

APPLESEEDS

Complex Conditions

- We can check multiple things inside a condition, not only one thing.
- For example, if we want to check that your string can be a password, we can check if your string is longer than 9 characters **and** has at least 1 uppercase letter.
- We can also check if a your age is bigger than 18 **or** your IQ is higher than 120.

APPLESEEDS

Complex Conditions

- Let's introduce 2 new logical operators: **and**, **or**.
- **And** is written like so && (shift + 7)
- **Or** is written like so || (shift \)
- How do they work?

Complex Conditions

- Result = condition1 && condition2
  Results = true **only** if condition1 = true **and** condition2 = true
- For example:

```
1 < 2 && ‘ori’.length > 2 //true

1 < 2 && ‘ori’.length > 8 //false

2 < 1 && ‘ori’.length > 2 //false
```

APPLESEEDS

Complex Conditions

- Result = condition1 || condition2
  Results = true if condition1 is true **or** condition2 is true.
  For example:

```
‘bob’ === ‘bob’ || 19 > 5 // true

‘Bob’ === ‘Bob’ || 19 > 21 // true

‘Bob’ === ‘bob’ || 19 < 5 // false
```

APPLESEEDS

Complex Conditions

`AND truth table`

condition1	condition2	result
T	T	T
T	F	F
F	T	F
F	F	F

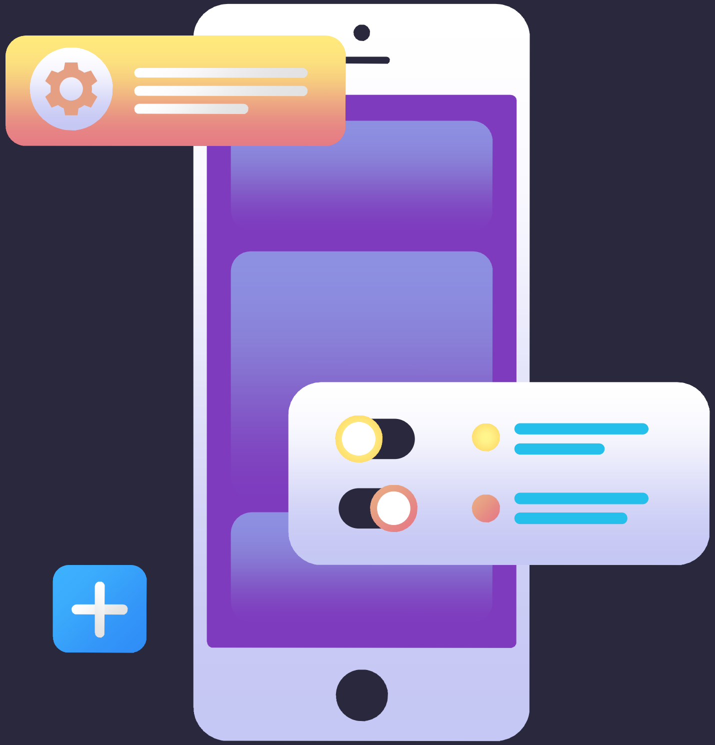
Complex Conditions

OR truth table

condition1	condition2	result
T	T	T
T	F	T
F	T	T
F	F	F

/04

Alternative to if



APPLESEEDS

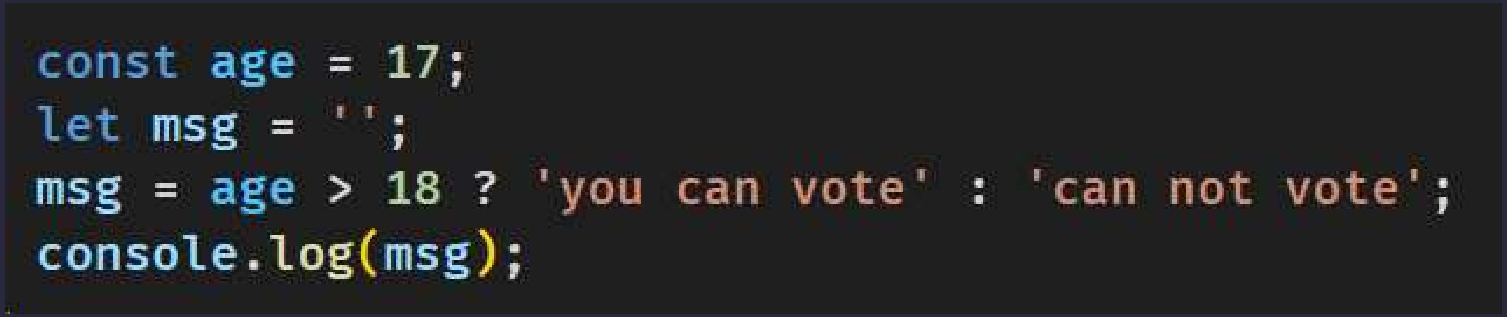
Alternative if - ternary

- There is a shorthand way to write an if statement, if there is one if-else.
- It's called “ternary”, because it uses the ternary operator (the combination of ? and :)
- The structure is like so:

```
Condition ? code if true : code if false
```

Alternative if - ternary

- For example:

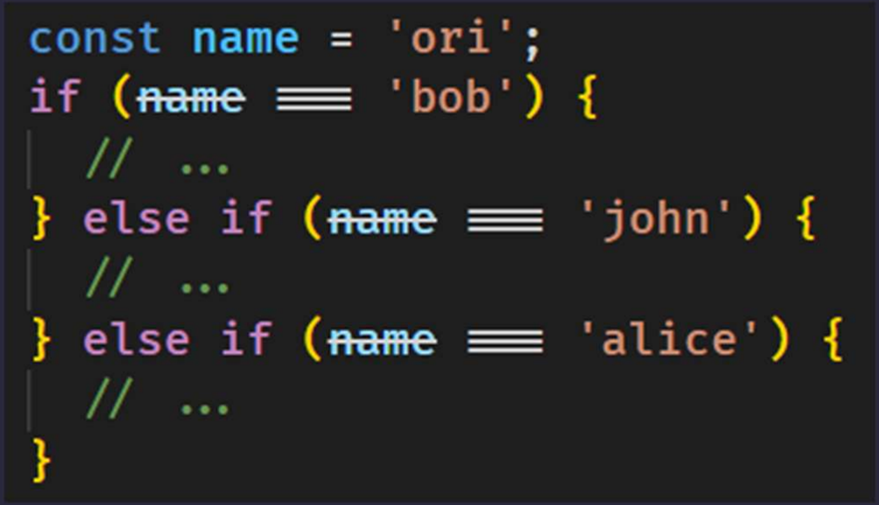


```
const age = 17;
let msg = '';
msg = age > 18 ? 'you can vote' : 'can not vote';
console.log(msg);
```

This will print ‘can not vote’, because age > 18 is false, so the part after the : is executed.

Alternative if - switch case

- If we have a lot of if else it can be annoying to write. For example:



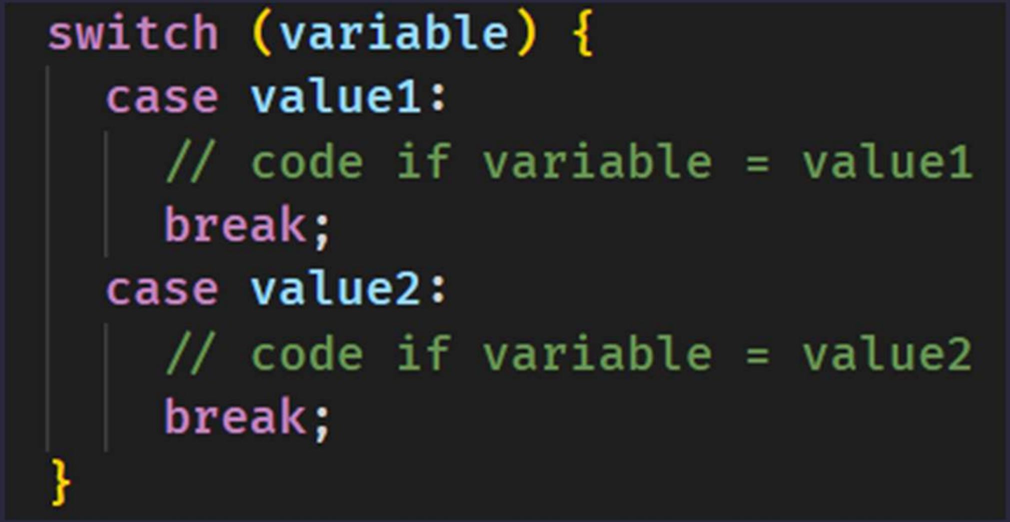
```
const name = 'ori';
if (name === 'bob') {
  // ...
} else if (name === 'john') {
  // ...
} else if (name === 'alice') {
  // ...
}
```

We can write it in a more readable way.

APPLESEEDS

Alternative if - switch case

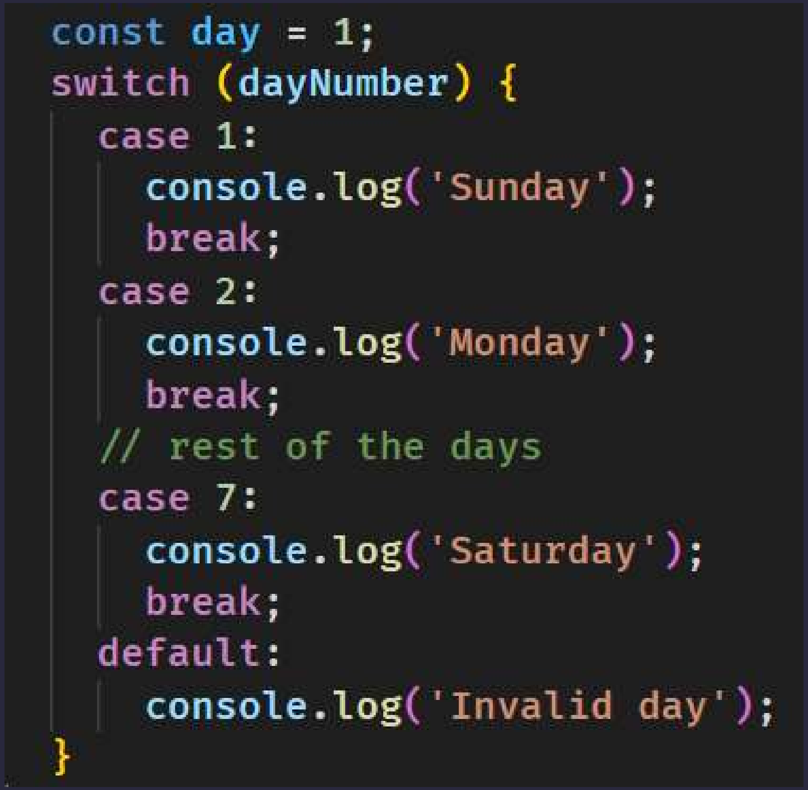
- It’s called switch case.
- The structure is:



```
switch (variable) {
  case value1:
    // code if variable = value1
    break;
  case value2:
    // code if variable = value2
    break;
}
```

APPLESEEDS

Alternative if - switch case



```
const day = 1;
switch (dayNumber) {
  case 1:
    console.log('Sunday');
    break;
  case 2:
    console.log('Monday');
    break;
  // rest of the days
  case 7:
    console.log('Saturday');
    break;
  default:
    console.log('Invalid day');
}
```