Front End
Software Development

Introduction to JavaScript (weeks 1 - 6)

Week 02

Agenda

- Questions
- Boolean Operators
- Conditionals
  - If/Else
  - Switch
- Loops
- User Input

Questions


THE WORLD
BEFORE SOCIAL MEDIA
LinkedIn
Pinterest
YouTube
Instagram
foursquare
Imgur
Reddit
WordPress
Address
Skype
Twitter
Tumblr
Facebook
@successfulmotivational

Boolean Operators

- Boolean Values
  - true / false
  - Digital: 1 (true) / 0 (false)
- Equality Operators
  - ==, >, >=, <, <=, !=
  - <left value> <*operator*> <right value>

```
9 == 4 // false
4 == 4 // true
1 >= 6 // false
3 != 2 // true
```

- Logical Operators
  - &&, ||, ! *(AND, OR, NOT)*

```
(5 == 4) && (4 == 4) // false
(5 == 4) || (4 == 4) // true
!(5 == 4) // true
```

AND (&&)

X	Y	XY
0	0	0
0	1	0
1	0	0
1	1	1

OR (||)

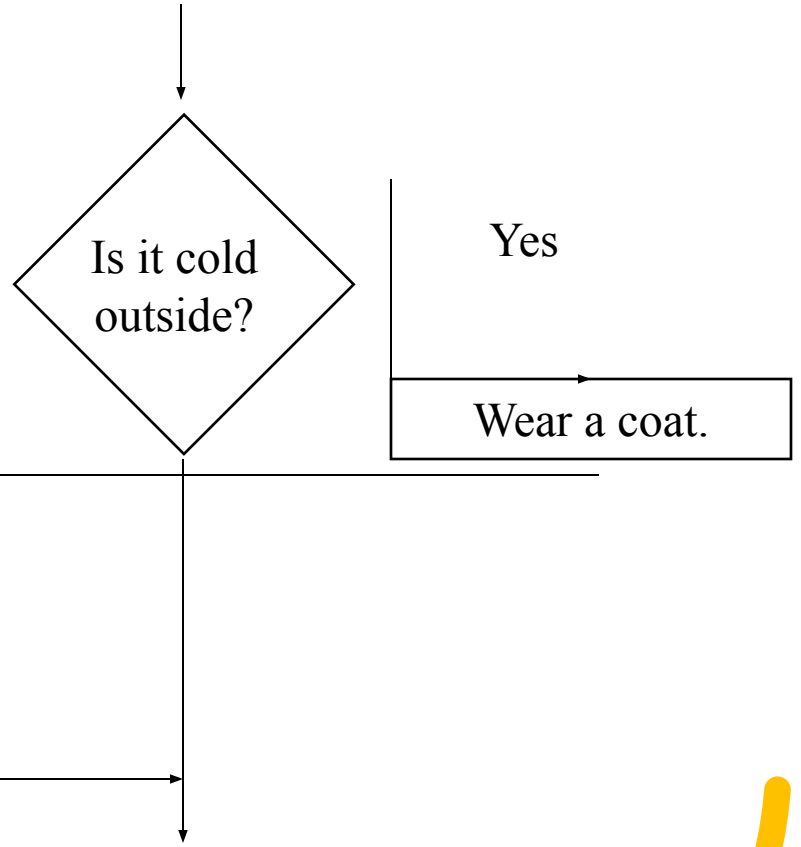
X	Y	XY
0	0	0
0	1	1
1	0	1
1	1	1

NOT (!)

X	X’
0	1
1	0

Conditionals
(if / else)

- Flow chart

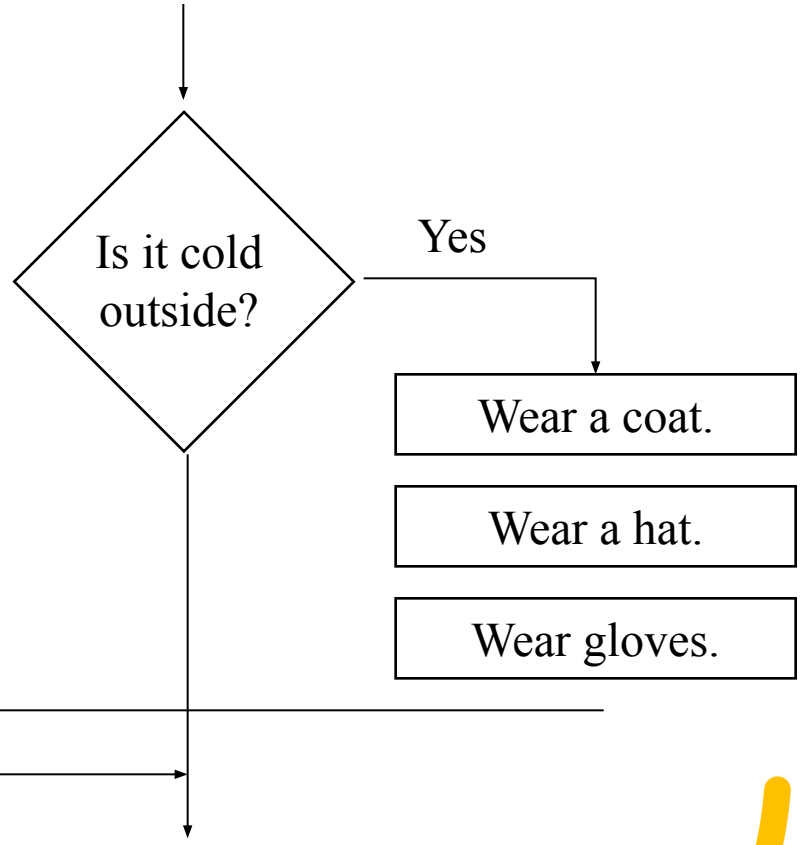



```
if (isColdOutside){
    wearCoat();
}
```

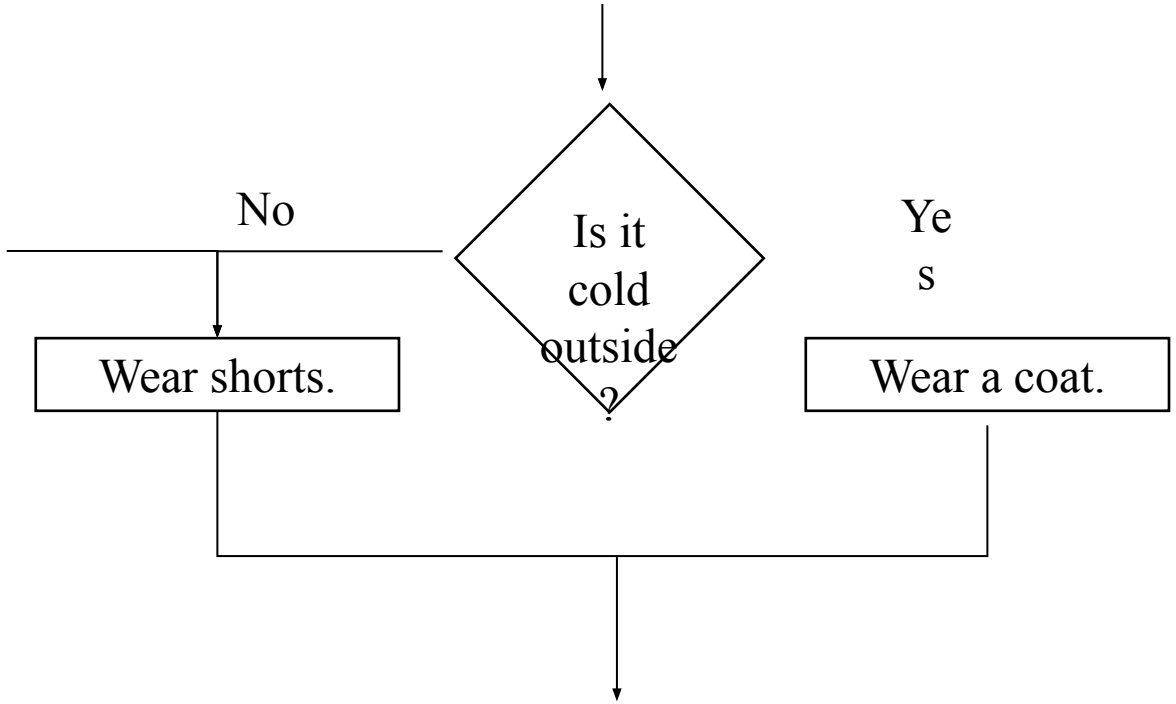
Conditionals
(if / else)

- Code Block

```
if (isColdOutside){
   wearCoat();
   wearHat();
   wearGloves();
}
```



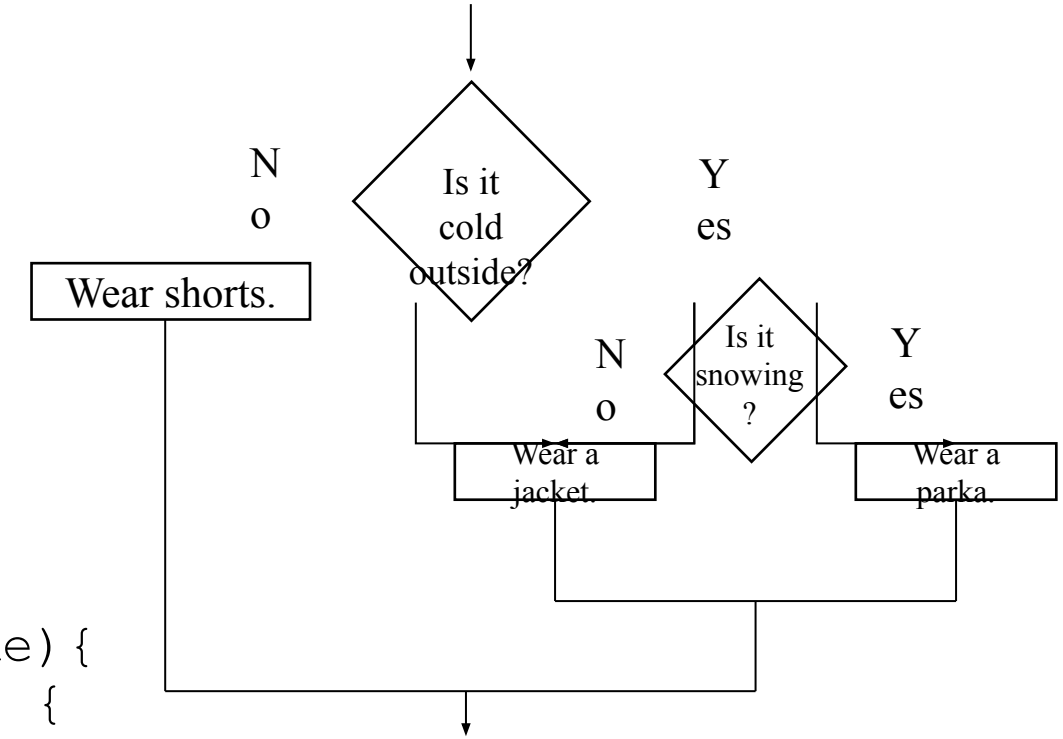
Conditionals
(if / else)



- if / else

```
if (isColdOutside){
  wearCoat();
}
else {
  wearShorts();
}
```

Conditionals
(if / else)



- Nested

```
if (isColdOutside){
  if (isSnowing) {
    wearParka();
  }
  else {
    wearJacket();
  }
}
else {
  wearShorts();
}
```

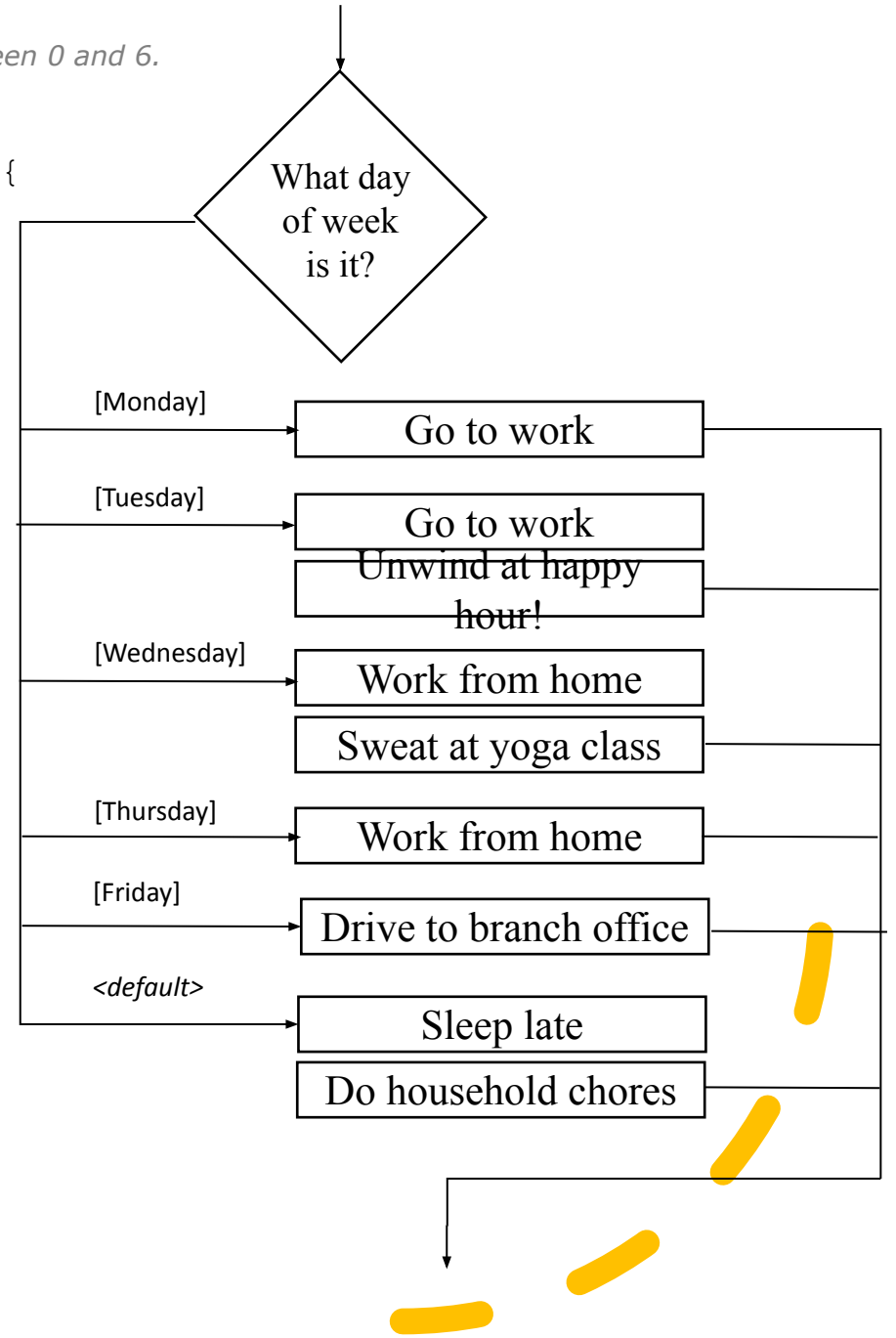
OR

```
if ((isColdOutside) && (isSnowing)) {
  wearParka();
}
else if (isColdOutside) {
  wearJacket();
}
else {
  wearShorts();
}
```

Conditionals
(switch)

The getDay() method returns the weekday as a number between 0 and 6. (Sunday=0, Monday=1, Tuesday=2 ..)

```
switch (new Date().getDay()) {
  case 1: // Monday
    goToWork();
    break;
  case 2: // Tuesday
    goToWork();
    unwindAtHappyHour();
    break;
  case 3: // Wednesday
    workFromHome();
    sweatAtYoga();
    break;
  case 4: // Thursday
    workFromHome();
    break;
  case 5: // Friday
    driveToBranch();
    break;
  default: // Sat(0),Sun(6)
    sleepLate();
    doHouseholdChores();
}
```

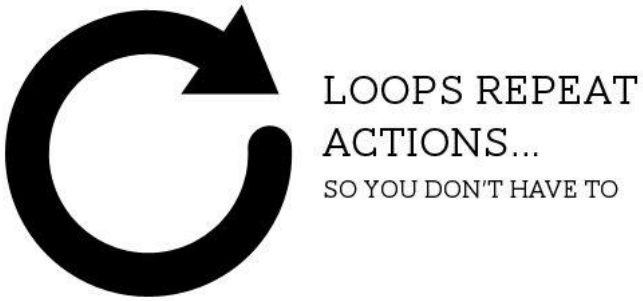


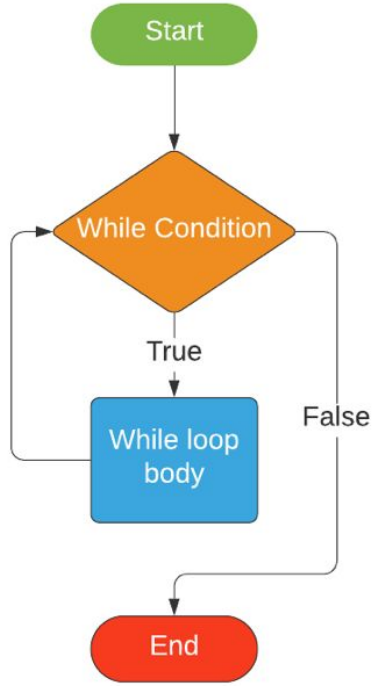
DEMO

Conditional / Branching Logic

Loops
(standard)

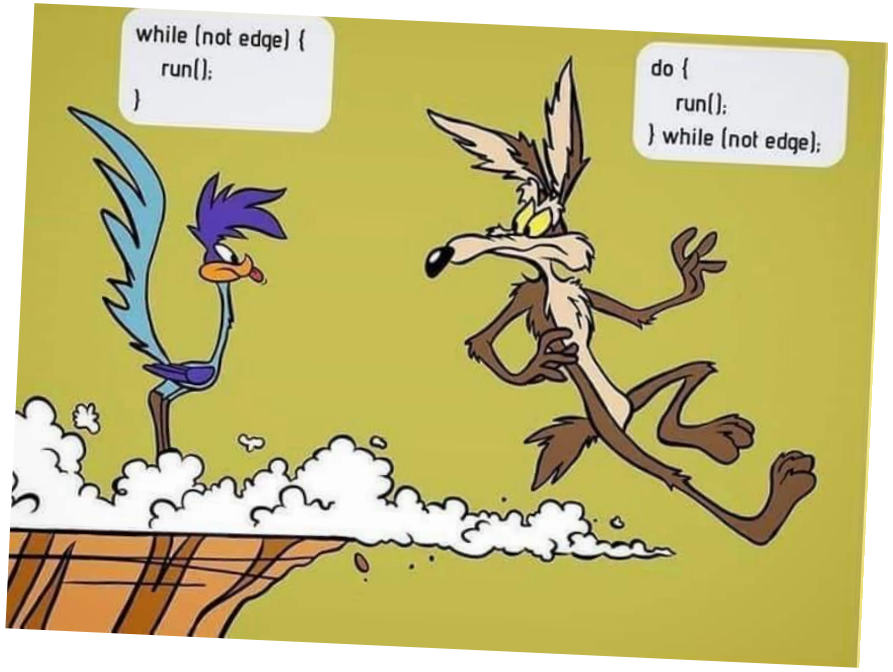
- Non-Deterministic
  Not predictable. Some type of "sentinel" or control value is needed to determine when to stop.
  -
```
while (true){
    // code to repeat
}
```
  -
```
do {
    // code to repeat
} while (true);
```
- Deterministic
  Number of loops are known in advance (i.e. 1 to 100)
  -
```
for (...; ...; ...) {
    // code to repeat
}
```





Loops
(enhanced)

- Enhanced Loops (foreach)
  - for/**in**
    Loops through the index
  - for/**of**
    Loops through the values

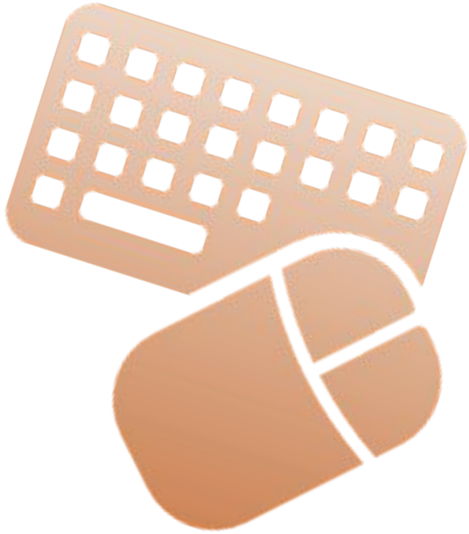



```
let names = [ "George", "Ava" ];

for(let index in names) {
  console.log(index); // To get value, use names[index]
} // 0, 1

for(let name of names) {
  console.log(name); // No way to get index or position
} // George, Ava
```

DEMO

Loops *(for / while / foreach)*

User Input



- Web Browser
  - Interaction with user

```
let count = prompt('How many?');
let index = 0;
while (index < count) {
  alert(++index); // This is annoying...
}
```

  - TEMPORARY!
    Better solutions exist, but we're not ready for them yet…

DEMO

User Input / Reading Input