

Web Developer

Programmazione - Javascript e Typescript

Docente: Shadi Lahham

Flow

Conditions and loops

Shadi Lahham - Web development

if/else

The if statement

Use `if` to tell JS which statements to execute, based on a condition

```
if (condition) {  
  // statements to execute  
}  
let x = 5;
```

```
if (x > 0) {  
  console.log('x is a positive number!');  
}
```

Comparison Operators

```
let myFavoriteNumber = 28;
```

== Equality

```
myFavoriteNumber == 28
```

```
myFavoriteNumber == '28'
```

```
28 == '28'
```

=== Strict equality

```
myFavoriteNumber === 28
```

!= Inequality

```
myFavoriteNumber != 29
```

!== Strict inequality

```
myFavoriteNumber !== '28'
```

```
28 !== '28'
```

Common mistake

Do not confuse the assignment operator = with ==

Comparison Operators

```
let myFavoriteNumber = 28;
```

```
> Greater than  
myFavoriteNumber > 25  
'28' > 25
```

```
>= Greater than or equal  
myFavoriteNumber >= 28  
'28' >= 25
```

```
< Less than  
myFavoriteNumber < 30  
'28' < 30
```

```
<= Less than or equal  
myFavoriteNumber <= 28  
'28' <= 28
```

Logical Operators

Operators

<code>&&</code>	<code>and</code>
<code> </code>	<code>or</code>
<code>!</code>	<code>not</code>

When combining together multiple conditions, use parentheses to group

```
let myAge = 28;
if ((myAge >= 0 && myAge < 3) || myAge > 90) {
  console.log('You\'re not quite in your peak.');
}
```

Truthy vs Falsey

If you don't use a comparison or logical operator, JS tries to figure out if the value is "truth-y"

```
let catsRule = true;  
if (catsRule) {  
  console.log('Yay cats!');  
}
```


Truthy vs Falsey

Values that are "false-y":

false, the empty string (""), the number 0, the number -0, undefined, null, NaN

```
let firstName = '';  
if (firstName) {  
  console.log('Hello, ' + firstName);  
}
```

```
let points = 0;  
if (points) {  
  console.log('You have ' + points + ' points');  
}
```

```
let firstName;  
if (firstName) {  
  console.log('Your name is ' + firstName);  
}
```

Short-Circuit Evaluation

JS evaluates logical operators from left to right and stops evaluating as soon as it knows the answer

`(falsey && anything) => falsey`

`(truthy || anything) => truthy`

Examples:

```
let nominator = 5;
let denominator = 0;
if (denominator !== 0 && (nominator/denominator > 0)) {
  console.log('Thats a valid, positive fraction');
}
```

The if/else statement

```
let age = 28;
if (age > 16) {
  console.log('Yay, you can drive!');
} else {
  console.log('Sorry, but you have ' + (16 - age) + ' years till you can drive.');
```

The if/else if/else statement

```
let age = 20;
if (age >= 35) {
  console.log('You can vote AND hold any place in government!');
} else if (age >= 25) {
  console.log('You can vote AND run for the Senate!');
} else if (age >= 18) {
  console.log('You can vote!');
} else {
  console.log('You have no voice in government!');
}
```

while & for

The while loop

The while loop tells JS to repeat statements until a condition is true

```
while (expression) {  
  // statements to repeat  
}  
let x = 0;  
while (x < 5) {  
  console.log(x);  
  x = x + 1;  
}
```

The for loop

```
for (initialize; condition; update) {  
  // statements to repeat  
}
```

```
for (let i = 0; i < 5; i = i + 1) {  
  console.log(i);  
}
```

The for loop

Some developers prefer to declare variables in advance

```
let people = ['James', 'Richard', 'Robert'];  
let i = 0;  
let len = people.length;  
let text = '';  
for (; i < len; i++) {  
    text += people[i] + ',';  
}  
console.log(text);
```

However this is not required.

Count backwards:

```
for (let c=20; c > 0; c--) {  
    // do something with c  
}
```


The break statement

To prematurely exit a loop, use the break statement

```
for (let current = 100; current < 200; current++) {  
  console.log('Testing ' + current);  
  if (current % 7 == 0) {  
    console.log('Found it! ' + current);  
    break;  
  }  
}
```

Diving deeper

// how many times is each of the following executed:

// initialize? condition? update? statements?

```
for (initialize; condition; update) {  
    // statements to repeat  
}
```

// what does this code do?

```
for (; 8; ) {  
    console.log('hi');  
}
```

// what does this code do?

```
while (') {  
    console.log('print me please');  
}
```

switch

The switch statement

// using if-else

```
let dayOfWeek = 'Monday';
```

```
let message;
```

```
if (dayOfWeek === 'Monday') {
```

```
  message = "It's Monday!";
```

```
} else if (dayOfWeek === 'Wednesday') {
```

```
  message = "It's Wednesday!";
```

```
} else if (dayOfWeek === 'Friday') {
```

```
  message = "It's Friday!";
```

```
} else {
```

```
  message = 'Invalid day of the week';
```

```
}
```

```
console.log(message);
```

The switch statement

```
// using switch
let dayOfWeek = 'Monday';
let message;

switch (dayOfWeek) {
  case 'Monday':
    message = "It's Monday!";
    break;
  case 'Wednesday':
    message = "It's Wednesday!";
    break;
  case 'Friday':
    message = "It's Friday!";
    break;
  default:
    message = 'Invalid day of the week';
}

console.log(message);
```

Switch with fall-through

```
// forgot to use break
let dayOfWeek = 'Monday';
let message;

switch (dayOfWeek) {
  case 'Monday':
    message = "It's Monday!";
  case 'Wednesday':
    message = "It's Wednesday!";
    break;
  case 'Friday':
    message = "It's Friday!";
    break;
  default:
    message = 'Invalid day of the week';
}

console.log(message); // what's the output?
```

Switch with intentional fall-through

```
let dayOfWeek = 'Monday';
let message;

switch (dayOfWeek) {
  case 'Monday':
    // fall-through
  case 'Tuesday':
    // fall-through
  case 'Friday':
    message = "It's a weekday";
    break;
  case 'Saturday':
    // fall-through
  case 'Sunday':
    message = "It's the weekend";
    break;
  default:
    message = 'Invalid day of the week';
}
console.log(message);
```

Your turn

1. Big numbers

- Write a function named `greaterNum` that:
 - takes 2 arguments, both numbers.
 - returns whichever number is the greater (higher) number.
- Call that function 2 times with different number pairs, and log the output to make sure it works (e.g. "The greater number of 5 and 10 is 10.").

2.Universal Translator

- Write a function named `helloWorld` that:
 - takes 1 argument, a language code (e.g. "es", "de", "en")
 - returns "Hello, World" for the given language, for atleast 3 languages. It should default to returning English.
- Call that function for each of the supported languages and log the result to make sure it works.

3. Grade master

- Write a function named `assignGrade` that:
 - takes 1 argument, a number score.
 - returns a grade for the score, either "A", "B", "C", "D", or "F".
- Call that function for a few different scores and log the result to make sure it works.

Bonus

4. One to many

- Write a function named `oneToMany()` that:
 - takes 2 arguments, a noun and a number.
 - returns the number and pluralized form, like "5 cats" or "1 dog".
- Call that function for a few different scores and log the result to make sure it works.
- Bonus: Make it handle a few collective nouns like "sheep" and "geese".

5.Odd or even

- Write a for loop that will iterate from 0 to 20.
- For each iteration, it will check if the current number is odd or even, and report that to the screen (e.g. "2 is even").

6.Easy multiplication

- Write a for loop that will iterate from 0 to 10.
- For each iteration of the for loop, it will multiply the number by 9 and log the result (e.g. "2 * 9 = 18").
- Bonus: Use a nested for loop to show the tables for every multiplier from 1 to 10 (100 results total).

7. Grade checker

- Write a loop that tests the function that you wrote earlier “assignGrade”.
- Check every value from 60 to 100:
 - your log should show
 - “For 88, you got a B.”
 - “For 89, you got a B.”
 - “For 90, you got an A.”
 - etc.

References

[JavaScript for Loop](#)

[JavaScript while Loop](#)

[JavaScript Switch](#)

Old resource

[Eloquent JavaScript - values, variables, and control flow](#)