thinkdev #4

Making decisions

Explore featured courses

The University of Glasgow

The Museum as a Site and Source for Learning

4.6 (75 reviews)

Find out more

Sentinel|9 & FutureLearn

NEW

Fundamentals of Business Strategy

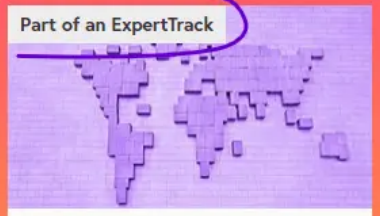
Find out more

University of Groningen, University of Cambridge & University Medical Center Groningen (UMCG)

Young People and Mental Health

4.7 (649 reviews)

Find out more



Coventry University

International Logistics: A Beginner's Guide to Logistics Management

Find out more

First, let's learn how to do some basic comparison

The relational operators:

```
3 > 2    // true
8 < 5    // false
9 >= 13  // false
6 <= 6   // true
```

Use the *strict equality operator* `===` to determine if two values are equal:

```
2 === 5 - 3           // true
89.0 === 89           // true
10 === '10'           // false
'hello' === 'Hello'   // false
true === false        // false
```

There’s a strict inequality counterpart:

```
2 !== 5 - 3           // false
89.0 !== 89           // false
10 !== '10'           // true
'hello' !== 'Hello'   // true
true !== false        // true
```

No two objects have the same value, even if they look alike:

```
const obj1 = { prop: "value" }
const obj2 = { prop: "value" }
console.log(obj1 === obj2) // false 😕
```

But, same object, same value:

```
const obj1 = { prop: "value" }
const obj2 = obj1
console.log(obj1 === obj2) // true
```

Let’s get to making decisions now.

The `if` statement

```
if (expression) {
  statement1
  statement2
  ...
}
```

```
if (expression) {
  statement1
  statement2
  ...
}
```

If `expression` is `true`, execute the statements in the curly brackets. Otherwise, ignore the statements.

Let's consider the FutureLearn example.

We represented a course like so in the previous lesson:

```
const course = {
  title: 'The Museum as a Site and ...',
  rating: 4.6,
  reviewsCount: 75,
  isNew: false,
  isPartOfAnExpertTrack: false,
}
```

Let’s determine if the course has a rating:

```
const course = {
  rating: 4.6,
  // ...
}

course.rating !== 0.0 // true
```

Now we can act accordingly:

```
const course = {
  rating: 4.6,
  // ...
}

if (course.rating !== 0) {
  console.log(`Rating: ${course.rating}`)
}
```

Let's add some logs around the `if` statement for clarity:

```
const course = {
  rating: 4.6,
  // ...
}

console.log('Before decision')
if (course.rating !== 0) {
  console.log(`Rating: ${course.rating}`)
}
console.log('After decision')
```

Here’s the output:

TODO: Improve the CSS of the <samp>s

```
Before decision
Rating: 4.6
After decision
```

What if the rating is 0?

```
const course = {
  rating: 0,
  // ...
}

console.log('Before decision')
if (course.rating !== 0) {
  console.log(`Rating: ${course.rating}`)
}
console.log('After decision')
```

Then the output is just this:

```
Before decision
After decision
```

How do we print a different message?

else

```
const course = {
  rating: 0,
  // ...
}

console.log('Before decision')
if (course.rating !== 0) {
  console.log(`Rating: ${course.rating}`)
} else {
  console.log('No rating')
}
```

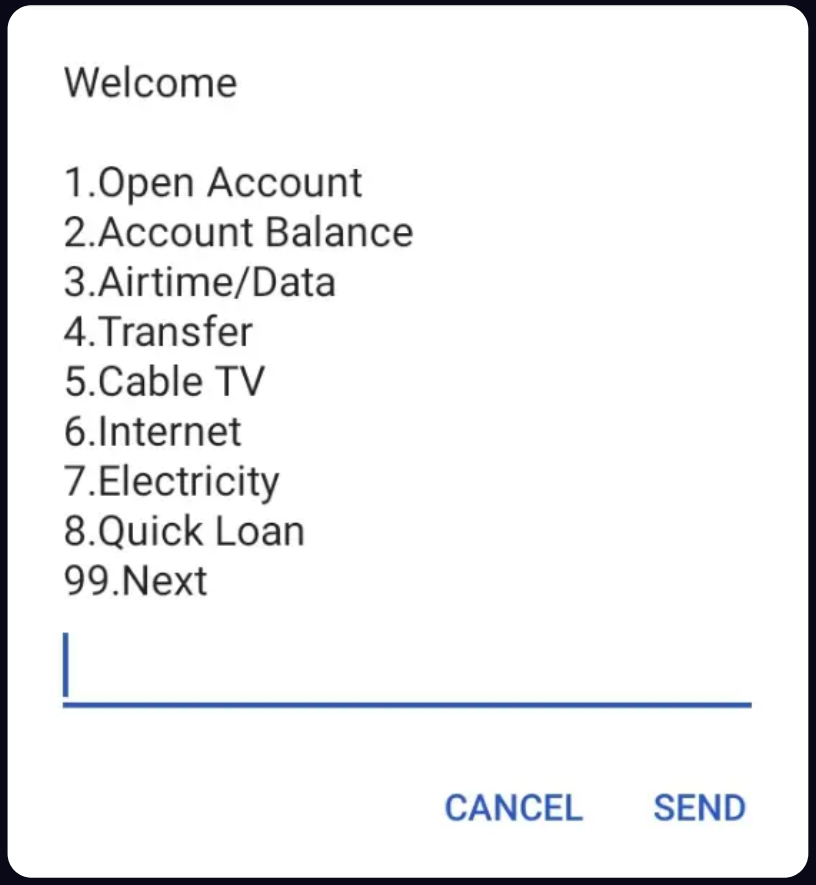
```
  rating: 0,
  // ...
}

console.log('Before decision')
if (course.rating !== 0) {
  console.log(`Rating: ${course.rating}`)
} else {
  console.log('No rating')
}
console.log('After decision')
```

The result:

```
Before decision
No rating
After decision
```

Let's consider another example.



A USSD menu with different options to choose from.

How do we express the several alternatives?

else if

```
const choice = 1 // Could be any other number

if (choice === 1) {
  console.log('Open Account')
} else if (choice === 2) {
  console.log('Account Balance')
} // ...
```

You can use a final `else` to handle any other choice.

```
const choice = 1 // Could be any other number

if (choice === 1) {
  console.log('Open Account')
} else if (choice === 2) {
  console.log('Account Balance')
} // ...
else {
  console.log('Invalid choice')
}
```

TODO: Reduce this and the next slide to one.

We can now make decisions based on single conditions

But what if we have multiple conditions?

TODO: Screenshot here

Logical operators

Logical operators

OR

Logical operators

OR, AND

Logical operators

OR, AND, and NOT.

TODO: Screenshot demonstrating OR

OR

`expression1 || expression2`

One expression must be `true` for the result to be `true`.

```
if (filename.endsWith(".docx") || filename.endsWith(".do
  console.log(filename, "is a Word document")
}
```

TODO: Screenshot demonstrating AND

AND

`expression1 && expression2`

Both expressions must be `true` for the result to be `true`.

```
if (user && user.role === "ADMIN") {
  // Show something only admins should see
}
```

TODO: Screenshot demonstrating NOT

NOT

`!expression`

- The result is `false` if `expression` is `true`.
- The result is `true` if `expression` is `false`.

```
const arr = []

console.log(arr.length) // 0, which is falsy

if (!arr.length) {
  console.log("The array is empty")
}
```

```
const arr = []

console.log(arr.length) // 0, which is falsy

if (!arr.length) {
  console.log("The array is empty")
}
```

```
const arr = []

console.log(arr.length) // 0, which is falsy

if (!arr.length) {
  console.log("The array is empty")
}
```

One final thing ...

The expressions used in making decisions don’t have to be boolean; JavaScript automatically converts them.

TODO: Code sample demoing non-boolean conditions.

TODO: Code sample demoing non-boolean conditions with logical ops.

The following values convert to `false`. We call them *falsy* values:

TODO: Code sample showing all falsy values.

All other values convert to `true`, so they are *truthy*:

TODO: Code sample showing some truthy values.