EXAMPLE 35: UNDERSTANDING == AND ===

EXAMPLE 35: UNDERSTANDING == AND ===

== IS KNOWN AS THE EQUALITY OPERATOR

=== IS KNOWN AS THE STRICT EQUALITY OPERATOR

=== IS PAIRED WITH !==

== IS KNOWN AS THE EQUALITY OPERATOR

== COMPARES VALUES - AND WILL PERFORM CONVERSIONS, SUCH AS FROM NUMBER TO STRING

=== IS KNOWN AS THE STRICT EQUALITY OPERATOR

=== COMPARES TYPES AND VALUES - AND WILL NOT PERFORM ANY CONVERSIONS

==

==

CONVERSIONS

FOR OBJECTS THERE IS NO DIFFERENCE BETWEEN THESE OPERATORS - ONLY FOR STRINGS AND NUMBERS :-)

===

===

NOT

== IS KNOWN AS THE EQUALITY OPERATOR

== COMPARES VALUES - AND WILL PERFORM CONVERSIONS, SUCH AS FROM NUMBER TO STRING

=== IS KNOWN AS THE STRICT EQUALITY OPERATOR

=== COMPARES TYPES AND VALUES - AND WILL NOT PERFORM ANY CONVERSIONS

== IS KNOWN AS THE EQUALITY OPERATOR

== COMPARES **VALUES** - AND WILL **PERFORM CONVERSIONS**, SUCH AS FROM NUMBER TO STRING

===

===

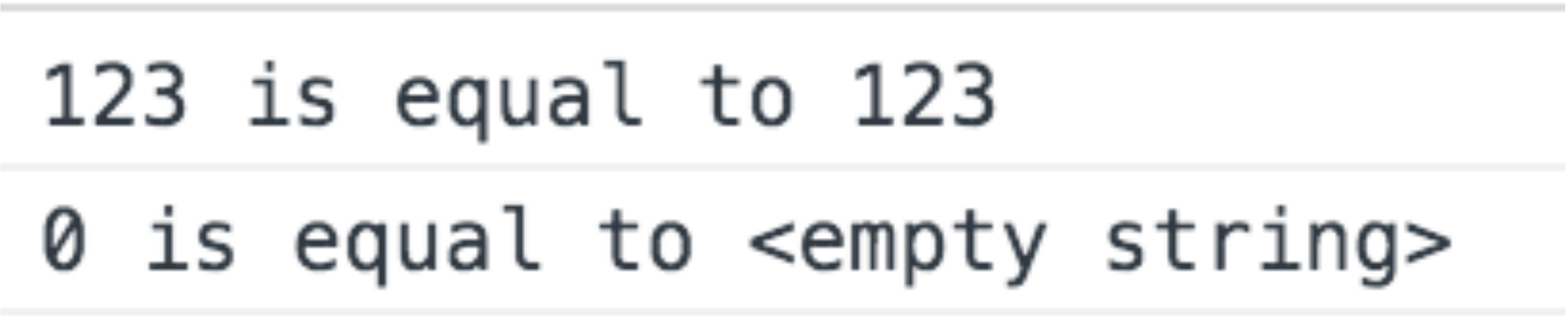
NOT

== COMPARES VALUES - AND WILL PERFORM CONVERSIONS, SUCH AS FROM NUMBER TO STRING

==COMPARE STRINGS AND NUMBERS - CONVERT STRING TO NUMBER (WE ALREADY SAW THIS)

```
  var someNumber = 123;
  var someStringThatLooksLikeANumber = "123";

  if(someNumber == someStringThatLooksLikeANumber) {
    console.log(someNumber + " is equal to " +
someStringThatLooksLikeANumber);
  }
```



```
123 is equal to 123
0 is equal to <empty string>
```

== COMPARES VALUES - AND WILL PERFORM CONVERSIONS, SUCH AS FROM NUMBER TO STRING

✓

```
null == undefined
```

== COMPARES VALUES - AND WILL PERFORM CONVERSIONS, SUCH AS FROM NUMBER TO STRING

==COMPARE BOOLEAN (TRUE/FALSE) AND STRINGS OR NUMBERS - MORE ON THIS WHEN WE GET TO TRUTHY AND FALSEY

✓ “” == 0

== IS KNOWN AS THE EQUALITY OPERATOR

== COMPARES VALUES - AND WILL PERFORM CONVERSIONS, SUCH AS FROM NUMBER TO STRING

===

===

NOT

== IS KNOWN AS THE EQUALITY OPERATOR

== COMPARES VALUES - AND WILL PERFORM CONVERSIONS, SUCH AS FROM NUMBER TO STRING

=== IS KNOWN AS THE STRICT EQUALITY OPERATOR

=== COMPARES TYPES AND VALUES - AND WILL NOT PERFORM ANY CONVERSIONS

==

==

CONVERSIONS

=== IS KNOWN AS THE STRICT EQUALITY OPERATOR

=== COMPARES TYPES AND VALUES - AND WILL NOT PERFORM ANY CONVERSIONS

=== IS KNOWN AS THE STRICT EQUALITY OPERATOR

=== COMPARES TYPES AND VALUES - AND WILL NOT PERFORM ANY

ACTUALLY A LOT SIMPLER THAN ==, DESPITE ITS WEIRD SYNTAX.

WITH ===, WHAT YOU SEE IS WHAT YOU GET. NO TYPE CONVERSIONS, NO WEIRD STUFF GOING ON:-)

WITH ===, WHAT YOU SEE IS WHAT YOU GET. NO TYPE CONVERSIONS, NO WEIRD STUFF GOING ON:-)

===COMPARE STRINGS AND NUMBERS - ALWAYS FALSE!

WITH ===, WHAT YOU SEE IS WHAT YOU GET. NO TYPE CONVERSIONS, NO WEIRD STUFF GOING ON:-)

✓

```
null !== undefined
```

WITH ===, WHAT YOU SEE IS WHAT YOU GET. NO TYPE CONVERSIONS, NO WEIRD STUFF GOING ON:-)

===COMPARE BOOLEAN (TRUE/FALSE) AND STRINGS OR NUMBERS - MORE ON THIS WHEN WE GET TO TRUTHY AND FALSEY

✓

“” !== 0

==

==

CONVERSIONS

=== IS KNOWN AS THE STRICT EQUALITY OPERATOR

=== COMPARES TYPES AND VALUES - AND WILL NOT PERFORM ANY CONVERSIONS

== IS KNOWN AS THE EQUALITY OPERATOR

== COMPARES VALUES - AND WILL PERFORM CONVERSIONS, SUCH AS FROM NUMBER TO STRING

=== IS KNOWN AS THE STRICT EQUALITY OPERATOR

=== COMPARES TYPES AND VALUES - AND WILL NOT PERFORM ANY CONVERSIONS

==

==

CONVERSIONS

FOR OBJECTS THERE IS NO DIFFERENCE BETWEEN THESE OPERATORS - ONLY FOR STRINGS AND NUMBERS :-)

===

===

NOT

FOR OBJECTS THERE IS NO DIFFERENCE BETWEEN THESE OPERATORS - ONLY FOR STRINGS AND NUMBERS :-)

TYPE CONVERSIONS ARE NEVER DONE WHILE COMPARING OBJECTS.

JAVASCRIPT QUIRKS

ARE ANNOYING WHEN YOU FIRST ENCOUNTER THEM, BUT OVER TIME THEY BECOME AMUSING, EVEN

✓STRANGE SPECIAL VALUES

✓STRINGS AND NUMBERS

STRANGE COMPARISON OPERATORS

TRUTHY AND FALSEY

JAVASCRIPT QUIRKS

ARE ANNOYING WHEN YOU FIRST ENCOUNTER THEM,
BUT OVER TIME THEY BECOME AMUSING, EVEN

✓STRANGE SPECIAL VALUES

✓STRINGS AND NUMBERS

✓STRANGE COMPARISON OPERATORS

TRUTHY AND FALSEY