JavaScript Conditions

COEN 161

Comparison Operators

- Apart from ==, >, and <, JavaScript has additional operators for more complex comparisons

Operator	Description	Comparing	Returns
!=	not equal	x != 8	true
>=	greater than or equal to	x >= 8	false
<=	less than or equal to	x <= 8	true

*** x = 5**

Comparison Operators

- JavaScript has one unusual, quirky operator: ===
- “Triple Equals” not only compares two values, it compares types as well

Operator	Description	Comparing	Returns
==	equal to	x == 5	true
		x == "5"	true
===	equal value and equal type	x === 5	true
*** x = 5**		x === "5"	false

Comparison Operators

- The inverse operation of “triple equals” is !==

Operator	Description	Comparing	Returns
!=	not equal	x != 8	true
!==	not equal value or not equal type	x !== 5	false
		x !== "5"	true
*** x = 5**		x !== 8	true

Logical Operators

- You can combine comparisons using logical operators
- Here **x = 6** and **y = 3**

Operator	Description	Example
&&	and	(x < 10 && y > 1) is true
||	or	(x == 5 || y == 5) is false
!	not	!(x == y) is true

Comparing Different Types

- When comparing with a number, JavaScript will try to convert a string to a number to do the comparison
- An empty string is 0 and anything non-numeric is NaN, which is always false

Case	Value	Case	Value
2 < 12	true	2 == "John"	false
2 < "12"	true	"2" < "12"	false
2 < "John"	false	"2" > "12"	true
2 > "John"	false	"2" == "12"	false

Conditional Statements

- JavaScript supports the following conditional statements
  - if - executes the block of code if the condition is true
  - else - executes the block of code if the condition is false
  - else if - tests a new condition if the previous condition was false
  - switch - specifies many alternative blocks of code to run

The if Statement

- If the condition is true, the block of code is run
  - If there is no block { } then only the first line after the if is run

```
if (condition) {
    block of code to be executed if the condition is true
}
```

The else Statement

- Must follow an if statement, and only runs if the if condition is false
  - Like the if, if there is no block after an else, only the first line is run

```
if (condition) {
    block of code to be executed if the condition is true
} else {
    block of code to be executed if the condition is false
}
```

The else if Statement

- This specifies a new condition to check if the previous condition is false
  - Must be after an if statement, but can follow another else if statement
  - Like if and else, if there is no block, only the first line after the else if is run

```
if (condition1) {
    block of code to be executed if condition1 is true
} else if (condition2) {
    block of code to be executed if the condition1 is false
    and condition2 is true
} else {
    block of code to be executed if the condition1 is false and
    condition2 is false
}
```

Conditional (Ternary) Operator

- This operator assigns a value based on a condition
- Syntax: `variablename = (condition) ? value1:value2`
- Example:

```
var voteable = (age < 18) ? "Too young":"Old enough";
```

The switch Statement

- A switch evaluates an expression and has *cases* for different conditions

```
switch(expression) {
    case n:
        code block
        break;
    case n:
        code block
        break;
    default:
        code block
}
```

The break Keyword

- In a switch, the keyword break exits a block and stops executing the rest of the switch
- Break statements should be at the end of each case block otherwise the switch will continue to check each case condition

```
switch (new Date().getDay()) {
    case 6:
        text = "Today is Saturday";
        break;
    case 0:
        text = "Today is Sunday";
        break;
    default:
        text = "Looking forward to the Weekend";
}
```

The default Keyword

- The default case is executed if there is no other match for the expression
- It doesn’t have to be the last case in the switch statement

```
switch (new Date().getDay()) {
    default:
        text = "Looking forward to the Weekend";
        break;
    case 6:
        text = "Today is Saturday";
        break;
    case 0:
        text = "Today is Sunday";
}
```

Common Code Blocks

- Case blocks can/should only have one value
- If you have multiple cases for one block of code you can use the following

```
switch (new Date().getDay()) {
    case 4:
    case 5:
        text = "Soon it is Weekend";
        break;
    case 0:
    case 6:
        text = "It is Weekend";
        break;
    default:
        text = "Looking forward to the Weekend";
}
```

Resources

https://www.w3schools.com/js/js_comparisons.asp

https://www.w3schools.com/js/js_if_else.asp

https://www.w3schools.com/js/js_switch.asp