CS7025 Programming for Digital Media

Lesson 3 – Conditions & Loops

Comparison Operators in JavaScript

==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to

Boolean operators

- While comparison operators return boolean values
- We can also compute with boolean values using boolean operators

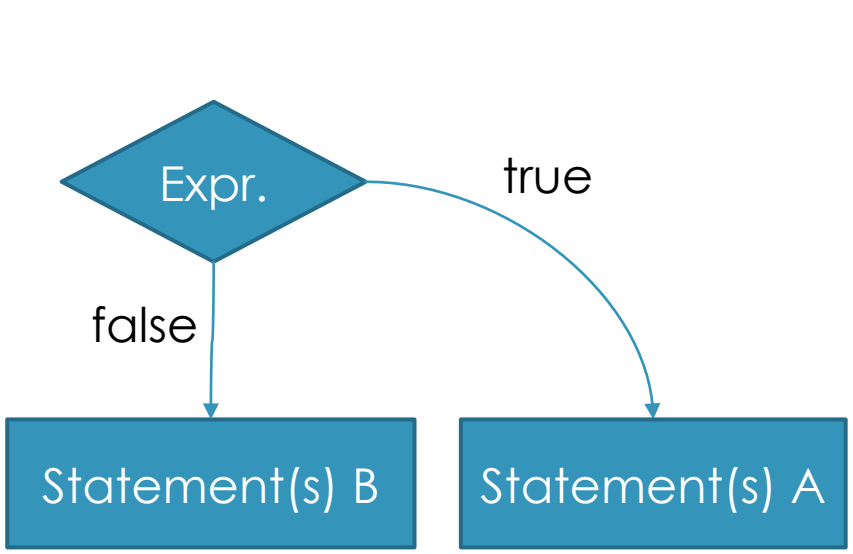
  AND – both A and B are true (&& in JavaScript)

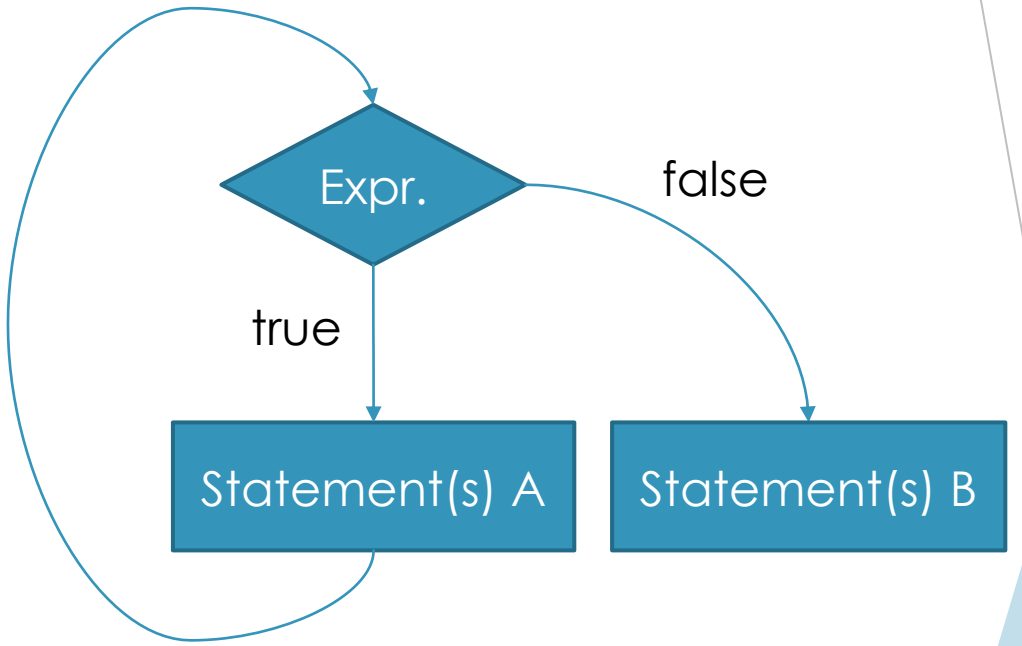
  OR – either A or B is true (| | in JavaScript)

  XOR – one and only one of A or B is true (doesn't exist in JavaScript)

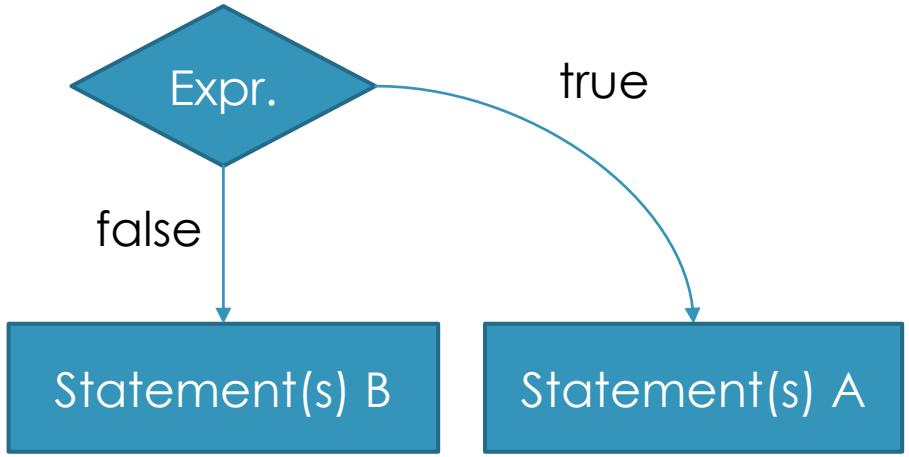
  NOT – X is not true, i.e. it is false (! In JavaScript)
- In fact, most modern programming languages allow us to extend this idea beyond boolean values, so that we can say that a variable with a value is true and one with no value is false.

Remember booleans and the flow of a program?





Conditional Statements



When we want to perform different actions for different decisions

Syntax:

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

Conditional Statements: if

```
if ( expression ) {
   // statement(s) to be executed only if expression is true;
}
// statement(s) to be executed whether or not expression is true;
```

```
if (hour < 18) {
     greeting = "Good day";
}
```

Conditional Statements: if … else

```
if ( expression ) {
   // statement(s) to be executed only if expression is true;
} else {
   // statement(s) to be executed only if expression is false;
}
// statement(s) to be executed whether or not expression is true;
```

```
if (hour < 18) {
        greeting = "Good day";
} else {
      greeting = "Good evening";
  }
```

Conditional Statements: chained conditionals

```
if ( expression1 ) {
    // statement(s) to be executed only if
    // expression1 is true;
} else if ( expression2 ) {
    // statement(s) to be executed only if
    // expression2 is true;
} else {
    // statement(s) to be executed only if
    // neither expression is true;
}
    // statement(s) to be executed
    // whether or not the expressions are true;
```

```
function greeting(theTime){
    let greeting;
     if (theTime < 10) {
          greeting = "Good morning";
     } else if (theTime < 20) {
          greeting = "Good day";
     } else {
          greeting = "Good evening";
     }
    return greeting;
}
```

Conditional Statements: nested conditionals

```
if ( expression1 ) {
   // statement(s) to be executed only if expression1 is true;
} else {
   if ( expression2 ) {
      // statement(s) to be executed if expression 1 is false and expression2 is true;
   } else {
      // statement(s) to be executed only if expression1 and expression2 are false;
   }
}
// statement(s) to be executed whether or not expressions are true
```

Conditional Statements: multiple expressions 'and' &&

```
if ( expression1 && expression 2 ) {

     // statement(s) to be executed only if
     // both expression1 and expression2
     // are true;

} else {

     // statements to be executed if either
     //expression1 or expression2 are false;
}

// statement(s) to be executed whether or
// not expressions are true;
```

```
if (theTime > 10 && theTime < 20 ) {
        greeting = "Good day";
} else {
        greeting = "Good evening";
}
```

Conditional Statements: multiple expressions ‘or’ | |

```
if ( expression1 || expression2 ) {
    // statement(s) to be executed if either expression1 OR expression2 are true;
} else {
    // statements to be executed only if both expression1 and
    // expression2 are false;
}
// statement(s) to be executed whether or not expressions are true;
```

Conditional Statements: ‘not’ !

```
if ( !expression ) {
   // statement(s) to be executed if the expression is not true;
}
```

Conditional Statements: using &&, || and ! together

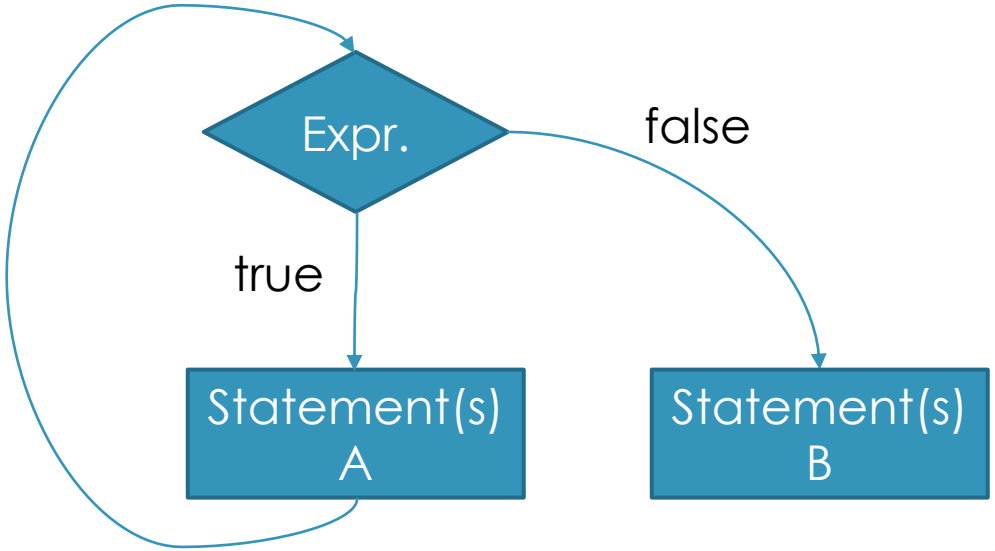
```
if (  ( expression1 && expression2 ) || !expression3 ) {
   // statement(s) to be executed if the expression is not true;
}
```

Loops

When we want to run the same code over and over again

JavaScript supports different kinds of loops:

- **for** - loops through a block of code a number of times
- **for/in** - loops through the properties of an object
- **while** - loops through a block of code while a specified condition is true
- **do/while** - also loops through a block of code while a specified condition is true



Loops: for

Syntax:

```
for (start; expression; step) {
    // code block to be executed
    // statement(s) to be executed until expression is true;
}
// statement(s) to be executed after the for loop has been completed;
// I.e. when the expression is no longer true
```

- **start** is executed (one time) before the execution of the code block.
- **expression** defines the condition for executing the code block.
- **step** is executed (every time) after the code block has been executed.

Loops: for

```
for (let i = 0; i < 5; i++) {
   // statement(s) to be until i >= 5;
}
// statement(s) to be executed after the for loop has been completed;
// I.e. when i is no longer < 5
```

// ++ is a shorthand way to say **`i = i + 1;`**

```
for (let i = 0; i < 5; i++) {
   console.log("The number is " + i );
}
```

Loops: for/In

```
let sequence = [1,2,3,4,5];
let position;
for ( position in sequence ) {
   // statement(s) to be executed once for each element in the sequence;
   // note that you need to use the syntax sequence[position] to access
   // the value at that position in the sequence
}
// statement(s) to be executed after each element has been processed;
```

```
let numbers = [34,2,56,8,10];
let x;
for (x in  numbers) {
 console.log("num " + x + " is "+numbers[x]);
}
```

```
let student = {name: 'John Doe', age: 25};
for (i in students) {
  console.log("student: " + student[i]);
}
```

Loops: while

```
while (expression) {
   // statement(s) to be executed while the expression is true;
}
// statement(s) to be executed if the expression is false or after it becomes false;
```

(The statements inside the loop may or may not be executed)

```
let i = 0;
while (i < 10) {
 console.log("The number is " + i );
  i++;
}
```

Loops: Do/While

```
do {
   //statement(s) to be executed while the expression is true;
}
while (expression);
//statement(s) to be executed after the expression becomes false;
```

(The statements inside the loop will be executed **at least once**, even if the condition is false as the statements appear before the condition is tested).

```
let i = 0;
do {
 console.log("The number is " + i );
  i++;
}
while (i < 10);
```

Loops: for vs. while

- **for** is great when the program knows in advance the number of times it will need to go through the loop
- **for/in** is used when you want to repeat some code for each element of a string or array (list of values) or an object
- **while** is useful for situations where the program won't know until later on how many times it needs to loop
- **do/while** is similar to while, but better if you always want to execute the statements in the loop at least once

Loops: break

Jump out of a loop when the condition is no longer met (END of LOOP)

```
for (start; expression1; step) {
    // statement(s) to be executed until expression1 is true;
     if (expression2) {
         break;
     }
}
// statement(s) to be executed after the
for loop has been completed or after break
```

The break command will cause the program to immediately exit the loop, even if expression1 is still true

```
for (let i = 0; i < 10; i++) {
  if (i === 3) { break; }
  console.log("The number is " + i );
}
```

Loops: continue

"jumps over" one iteration in the loop (START of LOOP)

```
for (start; expression1; step) {
     // statement(s) to be executed until expression1 is true;
    if (expression2) {
        continue;
    }
}
// statement(s) to be executed when
// expression1 is/becomes false
```

The continue command will cause the program to immediately exit the current iteration of the loop

```
for (let i = 0; i < 10; i++) {
  if (i === 3) { continue; }

  console.log("The number is " + i );
}
```

Try it yourself

Thank You

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin