Basic concepts

Empty page : about:blank

PARAMETERS

Function parameters are the names listed in the function definition.

Function arguments are the values received by the function.

STRING

There r multiple functions/manipulations with string but for now lets talk bout .substring.

- If you write down 2 parameters, it will slice down the rest of the string
- If the parameter is negative, it will count backwards
- Its like the string I learned when using python, almost the same.

Data types:

- String
- Booleans
- symbol
- undefined
- object
- function
- number

MATH OPERATORS:

++ = increment, var a = 10, a++: now a = 11

-- = decrement, var a=10, a--: now a = 9

ASSIGNMENT OPERATORS:

It is like += is the same as x = x + y, so var number = 20. Number += 5; outcome is: 25.

COMPARISON OPERATORS:

Operator	Description	Example
==	Equal to	5 == 10 false
===	Identical(equal and of the same type)	5 === 10 false
!=	Not equal to	5!=10 true
!==	Not identical	10!=10 false
>	Greater than	10 > 5 true
>=	Greater than or equal to	10>=5 true
<	Less than	10<5 false
< =	Less than or equal to	10>==5 false

They only work if compared with the same data type; numbers with numbers, strings with string etc.

BOOLEAN OPERATORS:

&& = returns true if both operands are true

|| = returns true, if one of the operands is true

! = returns true if the operand is false and false if the operand is true

You can check all type of data; comparison operator` s always return true or false

CONDITIONAL (TERNARY) OPERATORS:

This is how it will look like:

Variable = (condition) ? value!: value2

Example:

Var isAdult = (age<18) ? 'too young': 'old enough';

If the var age value is below 18, the value of isAdult will be 'too youn' and vice versa.

Logical NOT returns true if the operand is false.

String operators with concatenation

Basically the syntax will look like this:

Var name = statement ? if statement: else statement;

Conditionals and loops

THE IF STATEMENT

We use **if** to specify a block of code that we want to be executed if a specified condition is true.

```
If (condition) {
    Statements
}
```

The statements will only be executed if the specify condion is true. the function will act like the function doesn't exist if its false.

Alert() is an automatic function in javascript,

which generate a popup alert box.

We can use the **else** statement to specify a block of code that will be execute if the condition is false.

```
If (expression) {
    //executed if condition is true
}
Else {
    //executed if condition is false
}
```

There is another way to do this check (var1 > var2 for example) using the ? operator:

a > b ? alert(a) : alert(b). **(kijk terug naar conditional operators)**

ELSE IF STATEMENT

The else if statement is useful because it lets us specify a new condition if the first condition is false.

Example:

```
Var course = 1;
If (course == 1) {
        Document.write("html tutorial");
} Else if (course == 2){
        Document.write("css tutorial");
} Else {
        Document.write("javascript tutorial");
}
```

This is whats happening:

- If course is equal to 1: output html tutorial
- Else if course is equal to 1: output css tutorial;
- If none of the above condition is true then outpur: javascript tutorial.

Course is equal to 1 so we get the first statement.

The final **else** statement 'ends' the else if statement.

It should always be written after the is and else if statements.

SWITCH

We can use the switch statement to perform different actions based on different conditions.

```
Switch (expression){
        Case n1:
                Statements
                Break;
        Case n2:
                Statements
                Break;
        Default:
                Statements
}
```

This expression is evaluated once. The value of the expression is compared with the values pf each **case** and if theres a match, that block of code is executed. You can write as many case statements as you need.

The break keyword essentially switches the switch statement off. Breaking out of the switch block stops the execution of more code and case testing inside the block.

Usually a break should be put in each case statement.

Often there will be no match, but we still need the program to output something.. for this we use the default keyword which specifies the code to run if theres no case match.

FOR LOOPS

Javascript has three types of loops: for, while and do.

The classic for loop:

```
For (statement1; statement2; statement3) {
    Code  block to be executed
}
```

Or (same code but different wording)

```
For (initialization ; condition ; iteration){
    Code block to be executed
}
```

- Statement 1 is executed before the loop (the code block) starts
- Statement 2 defines the condition for running the loop
- Statement 3 is executed each time after the loop has been executed.

As you can see, the classic for loop has three components, or statements.

You can also initiate more than one value in statement 1, using commas to separate them.

If statement 2 returns true, the loop will start over again, if it returns false, the loop will end. Statement 2 is also optional but only if you put a break inside the loop or else it will never end!

Statement 3 is used to change the initial variable. It can do anything and it is also optional only if you increment your values inside the loop, like this:

```
Var 1 = 0;
For (; I < 10; ){
    Document.write(i);
    i++
}
```

WHILE LOOP

The while loop repeats through a block of code, but only as long as specified condition is true.

```
While (condition){
    Code block
```

```
}
```

The condition can be any conditional statement that returns true or false(a Boolean value).

Example:

```
Var i=0;
While (i <= 10) {
        Document.write( i + " <br/>");
        i ++; // of je can i = i + 1;
}
```

The loop in this code will continue to run as long as i is less than or equal to 10. And each time the loop runs, it will increase by 1.

Make sure that the condition in a while loop eventually becomes false.

THE DO… WHILE LOOP

This loop will execute the code block once, before checking if the condition is true, and then it will repeat the loop as long as the condition is true.

```
Do {
        Code block
}
While (condition);
```

Note the semicolon at the end of the do..while loop.

BREAK KEYBOARD

There are 3 different ways to stop a loop:

- Using the break keyword
- Return
- Continue

Unlike the break statement, the continue statement breaks only one iteration in the loop and continues with the next iteration.

Example:

```
For (i=0; i<=10; i++){
        If (i==5) {
                Continue;
        }
        Document.write(i+"<br/>");
}
```

The value 5 will not be printed because continue skips that iteration of the loop.