2. JS - Operatoren

**Operatoren bepalen welke acties we op variabelen en waarden willen uitvoeren.**

Wiskundige operatoren

|  |  |  |  |
| --- | --- | --- | --- |
| **operator** | **omschrijving** | **voorbeeld** | **uitkomst** |
| + | optellen | resultaat = getal + 5 | resultaat = 15 |
| - | aftrekken | resultaat = getal - 5 | resultaat = 5 |
| \* | vermenigvuldigen | resultaat = getal \* 5 | resultaat = 25 |
| / | delen | resultaat = getal / 5 | resultaat = 2 |
| % | modulo, de rest van een deling | resultaat = getal % 3 | resultaat = 1 |
| ++ | vermeerderen met 1 | getal ++ | getal = 11 |
| -- | verminderen met 1 | getal -- | getal = 9 |

Het is soms nodig om te testen of een getal even of oneven is. Met behulp van de modulo operator kan je dat gemakkelijk doen. Alle oneven getallen geven als resultaat 1 als je ze deelt door 2, terwijl alle even nummers 0 retourneren.

De string + operator

Om strings aan elkaar te plakken gebruik je de + operator (lees string + operator).

var voornaam = 'Barack';

var familienaam = 'Obama';

var naam = voornaam + ' ' + familienaam;

Toekenningsoperatoren

In het Engels heet dat een ***assignment operator*.**

Er bestaan twee soorten toekenningsoperators.

1. Enkelvoudige toekenningsoperator  
   Met de toekenningsoperator ken je een waarde toe aan een variabele. Je gebruikt die operator om een variabele te initialiseren.  
   var naam = 'Barack Obama';
2. Samengestelde toekenningsoperatoren  
   Samengestelde toekenningsoperatoren zijn een samentrekking van een wiskundige bewerking of een string bewerking met een toekenning.  
   Bijvoorbeeld getal = getal + 1 kan samengetrokken worden tot getal += 1. Met een string variabele kan je dat ook doen: naam = naam + ' de president van Amerika' wordt samengetrokken tot naam += ' de president van Amerika'.

|  |  |  |  |
| --- | --- | --- | --- |
| **operator** | **voorbeeld** | **samentrekking van** | **uitkomst** |
| += | getal += 5 | getal = getal + 5 | getal = 15 |
| -= | getal -= 5 | getal = getal - 5 | getal = 5 |
| \*= | getal \*= 5 | getal = getal \* 5 | getal = 50 |
| /= | getal /= 5 | getal = getal / 5 | getal = 2 |
| %= | getal %= 3 | getal = getal % 3 | getal = 1 |

Vergelijkingsoperatoren

|  |  |
| --- | --- |
| **operator** | **betekenis** |
| == | is gelijk aan |
| === | betekent is gelijk aan als ook het gegevenstype van beide operanden gelijk is; er treedt dus geen gegevenstypeconversie op bij de vergelijking |
| != | is verschillend van |
| !== | betekent is verschillend van; als het gegevenstype van beide operanden niet gelijk is levert de expressie altijd een false op; er treedt geen gegevenstypeconversie op bij de vergelijking |
| > | is groter dan |
| < | is kleiner dan |
| >= | is groter dan of gelijk aan |
| <= | is kleiner dan of gelijk aan |

Logische operatoren

Er zijn drie logische operatoren die werken met booleaanse waarden:

* ! – logical NOT (negation)
* && – logical AND
* || – logical OR

Operator precedence

Dezelfde regels gelden als in de wiskunde.

Bij logische operatoren heeft de **negatie** de hoogste prioriteit, dan de **en** en tenslotte de **of**.

Gebruik altijd haakjes zodat je de volgorde zelf controleert en niet afhankelijk bent van de JavaScript-engine.

Luie evaluatie

Als je verschillende logische operaties na elkaar plaatst, stop JavaScript met evalueren vanaf het moment dat de uitkomst duidelijk wordt. De rest van de expressie wordt niet meer geëvalueerd: true || false || true || false || true;

true

Aangezien dat allemaal OR operaties zijn en dus dezelfde prioriteit hebben, is het resultaat true als ten minste een van de operands waar is. Dus het volstaat de eerste operand te evalueren om de te weten wat het uiteindelijke resultaat zal zijn. De JavaScript-engine besluit lui te zijn en vermijdt onnodig werk door code die geen invloed meer kan hebben op het resultaat niet te evalueren.

Bronnen

Voor veel meer details over operatoren zie Mozilla Developer Network (MDN) "[Expressions and Operators](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Expressions_and_Operators)".