

Snelstartgids JavaScript

JAVASCRIPT INLEIDING	3
Wat Je Al Zou Moeten Weten	3
Wat is JavaScript?	3
Zijn Java en JavaScript Het Zelfde?	3
Wat kan een JavaScript doen?	3
JAVASCRIPT IN HTML	5
Hoe een JavaScript In een HTML Pagina in te voegen	5
Statements Eindigen met een Punt Komma?	6
Hoe om te gaan met oudere browsers?	6
WAAR PLAATS JE HET JAVASCRIPT	7
Externe JavaScript gebruiken	8
JAVASCRIPT VARIABELEN	9
Variabelen	9
Een Variabele Declareren	9
Een Waarde Toekennen Aan Een Variabele	9
Levensduur van Variabelen	9
CONDITIONELE STATEMENTS	10
Conditionele Statements	10
If Statement	10
If...else Statement	11
If...else if...else Statement	12
De JavaScript Switch Statement	13
JAVASCRIPT OPERATOREN	14
Wiskundige Operatoren	14
Opdracht Operatoren	14
Vergelijkings Operatoren	14
Logische Operatoren	15
String Operator	15
Conditionele Operator	15
JAVASCRIPT GEBEURTENISSEN	16
Onload en OnUnload	16
OnClick en OnDbIClick	16
OnMouseDown en OnMouseUp	16
OnMouseOver, OnMouseMove en OnMouseOut	16
OnKeyPress, OnKeyDown en OnKeyUp	16
OnSelect	16
OnFocus en OnBlur	16
OnChange	17
OnSubmit en OnReset	17
JavaScript heeft een groot nadeel.	18

JAVASCRIPT ALERTS	18
Waarschuwingsvenster maken	18
Bevestigingsvenster maken	18
JAVASCRIPT TIPS	19
JAVASCRIPT FUNCTIES	20
JavaScript Functies	20
Hoe een Functie te Definiëren	21
De return Opdracht (Statement)	21
JAVASCRIPT LOOPS	22
JavaScript Loops	22
De for Loop	22
De while loop	23
De do...while Loop	24

JavaScript Inleiding

JavaScript wordt in miljoenen webpagina's gebruikt om het design te verbeteren, formulieren te controleren, browsers te herkennen, cookies te maken, en veel meer.

JavaScript is de meest populaire scripting taal op het internet, en werkt in alle bekende browsers, zoals Internet Explorer, Mozilla, Firefox, Netscape, en Opera.

Wat Je Al Zou Moeten Weten

Voordat je verder gaat zou je al een basis kennis moeten hebben van:

- HTML / XHTML

Als je deze eerst wilt bestuderen kan je in het navigatie menu voor deze onderwerpen kiezen.

Wat is JavaScript?

- JavaScript is ontwikkeld om interactiviteit toe te voegen aan HTML pagina's
 - JavaScript is een scripting taal (een scripting taal is een lichtgewicht programmeer taal)
 - Een JavaScript bestaat uit uitvoerbare regels computercode
 - Een JavaScript wordt normaal direct in de HTML pagina ingevoegd.
 - JavaScript is een interpreter taal (dit betekent dat het script uitgevoerd wordt zonder compileren)
 - Iedereen kan JavaScript gebruiken zonder een licentie aan te schaffen
-

Zijn Java en JavaScript Het Zelfde?

NEE!

Java en JavaScript zijn twee compleet verschillende talen in concept en design!

Java (Sun Microsystems) is een krachtige en veel complexere programmeer taal - in de categorie van C and C++.

Wat kan een JavaScript doen?

- **JavaScript geeft HTML designers een programmeer tool** - HTML schrijvers zijn normaal geen programmeurs, maar JavaScript is een scripting taal met een eenvoudige syntax! Bijna iedereen kan kleine stukjes code in zijn HTML pagina's invoegen.
- **JavaScript kan dynamische tekst in een HTML pagina maken** - Een JavaScript statement zoals: `document.write("<h1>" + name + "</h1>")` kan een variabele tekst in een HTML pagina zetten
- **JavaScript kan reageren op gebeurtenissen(events)** - Een JavaScript kan ingesteld worden om iets uit te voeren wanneer er iets gebeurt, bijvoorbeeld het klikken op een HTML element
- **JavaScript kan HTML elementen lezen en schrijven** - Een JavaScript kan de inhoud van een HTML element lezen en wijzigen
- **JavaScript kan gebruikt worden om data te valideren** - Een JavaScript kan gebruikt worden om formulier gegevens te valideren voordat deze naar de server worden gestuurd.

- **JavaScript kan gebruikt worden om de gebruikers' browser te detecteren** - Een JavaScript kan gebruikt worden om de browser van de bezoeker te detecteren, en afhankelijk van de browser een pagina laden welke speciaal voor die browser is gemaakt
- **JavaScript kan gebruikt worden om cookies te maken** - Een JavaScript kan gebruikt worden om informatie op de computer van de gebruiker op te slaan en weer te verkrijgen, hiermee zijn bijvoorbeeld terugkerende bezoekers te herkennen

JavaScript in HTML

De HTML `<script>` tag wordt gebruikt om JavaScript in een HTML pagina in te voegen.

Hoe een JavaScript In een HTML Pagina in te voegen

```
<html>
<body>
<script type="text/javascript">
document.write("Hallo Wereld!")
</script>
</body>
</html>
```

De bovenstaande code zal deze uitvoer geven:

```
Hallo Wereld!
```

Voorbeeld uitgelegd

Om een JavaScript in een HTML pagina te implementeren, gebruiken we de `<script>` tag (gebruikt ook het type attribuut om de scripttaal te).

Dus, de `<script type="text/javascript">` en `</script>` tags laten weten waar de JavaScript start en eindigt:

```
<html>
<body>
<script type="text/javascript">
...
</script>
</body>
</html>
```

Het woord **document.write** is een standaard JavaScript commando om uitvoer op een pagina te schrijven.

Door het commando `document.write` tussen de `<script type="text/javascript">` en `</script>` tags te gebruiken, zal de browser het als JavaScript herkennen en de regel code uitvoeren. In dit geval zal de browser `Hallo Wereld!` op de pagina schrijven:

```
<html>
<body>
<script type="text/javascript">
document.write("Hallo Wereld!")
</script>
</body>
</html>
```

N.b.: Als we de `<script>` tags niet hadden gebruikt, zou de browser het `document.write("Hallo Wereld!")` commando als pure tekst hebben gezien en dit volledig op de pagina hebben weergegeven.

Statements Eindigen met een Punt Komma?

Bij traditionele programmeer talen, zoals C++ en Java, moet elke regel code eindigen met een punt komma.

Veel programmeurs houden deze gewoonte aan wanneer ze JavaScript schrijven, maar over het algemeen zijn punt komma's **optioneel!** Als je meerdere statements op een regel wilt zetten moeten de statements wel gescheiden worden door een punt komma.

Hoe om te gaan met oudere browsers?

Browsers die geen JavaScript ondersteunen zullen het script als tekst weergeven. Om dit te voorkomen kunnen we de HTML commentaar tag gebruiken:

```
<script type="text/javascript">
<!--
document.write("Hallo Wereld!")
//-->
</script>
```

De twee slashes aan het einde van de commentaar regel(//) zijn het JavaScript commentaar symbool. Dit voorkomt dat de JavaScript compiler de regel vertaalt.

JavaScripts in de body sectie worden uitgevoerd terwijl de pagina laadt.

JavaScripts in dehead sectie worden uitgevoerd als erom gevraagd wordt.

Waar plaats je Het JavaScript

JavaScript in een pagina zal direct uitgevoerd worden terwijl de pagina ingeladen wordt door de browser. Dat is niet altijd wenselijk, soms willen we dat het JavaScript wordt uitgevoerd als de pagina geladen wordt, maar in andere gevallen willen we dat het JavaScript wordt uitgevoerd door een actie van de gebruiker.

Scripts in de head sectie: Script om uitgevoerd te worden als ze worden aangeroepen, of wanneer een gebeurtenis plaatsvindt, komen in de head sectie van de pagina.

```
<html>
<head>
<script type="text/javascript">
....
</script>
</head>
```

Scripts in de body sectie: Scripts die uitgevoerd moeten worden als de pagina geladen is komen in de body sectie. Als je een script in de body sectie plaatst genereert deze content voor de pagina.

```
<html>
<head>
</head>
<body>
<script type="text/javascript">
....
</script>
</body>
```

Scripts in en de body en de head sectie: Je kan een ontelbaar aantal scripts in je document zetten, dus je kan en scripts in de head sectie en scripts in de body sectie hebben.

```
<html>
<head>
<script type="text/javascript">
....
</script>
</head>
<body>
<script type="text/javascript">
....
</script>
</body>
```

Externe JavaScript gebruiken

Het kan gebeuren dat je een bepaald JavaScript in meerdere pagina's wilt uitvoeren, zonder dat je het script op elke pagina wilt schrijven.

Of eenvoudiger: Je kan JavaScript in een extern bestand opslaan. Sla het externe JavaScript bestand op met een .js bestands extensie.

N.b.: Het externe script heeft geen <script> tag!

Om het externe script te gebruiken, verwijfs je naar het .js bestand in het "src" attribuut van de <script> tag:

```
<html>
<head>
<script src="xxx.js"></script>
</head>
<body>
</body>
</html>
```

N.b.: Plaats de verwijzing waar je wilt dat het script wordt uitgevoerd!

JavaScript Variabelen

Een variabele is een container voor informatie die je wilt opslaan.

Variabelen

Een variabele is een "container" voor informatie die je wilt opslaan. De waarde van een variabele kan tijdens de uitvoer van het script veranderen. Je kan een variabele aanroepen met zijn naam om de waarde te zien of te wijzigen. Rules for variable names:

- Variabele namen zijn hoofdletter gevoelig
- Ze moeten beginnen met een letter of een underscore

BELANGRIJK! JavaScript is hoofdletter-gevoelig! Een variabele genaamd strname is niet dezelfde als de variabele genaamd STRNAME!

Een Variabele Declareren

Je kan een variabele maken met de 'var' statement.

```
var strname = een waarde
```

Je kan een variabele ook maken zonder de 'var' statement:

```
strname = een waarde
```

Een Waarde Toekennen Aan Een Variabele

Je kan zo een waarde toekennen aan een variabele:

```
var strname = "Erik"
```

Of zo:

```
strname = "Erik"
```

Nu heeft de variabele "strname" de waarde "Erik".

Levensduur van Variabelen

Als je een variabele declareerd binnen een functie, kan deze alleen binnen deze functie worden gebruikt. Als de functie stopt met uitvoeren wordt de variabele vernietigd. Deze variabelen heten locale variabelen. Je kan dus locale variabelen met dezelfde naam hebben in verschillende functies, omdat elke alleen wordt herkend binnen de functie waarin hij is gedeclareerd.

Als je een variabele declareerd buiten een functie, dan kunnen alle functies op de pagina deze gebruiken. Deze variabelen worden vernietigd als de pagina wordt gesloten.

Conditionele Statements

Conditionele statements worden in JavaScript gebruikt om verschillende acties uit te voeren gebaseerd op verschillende condities.

Conditionele Statements

Wanneer je code schrijft wil je vaak verschillende acties uitvoeren voor verschillende beslissingen. Je kan conditionele statements gebruiken om dit te doen.

In JavaScript hebben we de volgende conditionele statements:

- **if statement** - Gebruik deze als je code wilt laten uitvoeren als een gespecificeerde conditie waar is (true)
- **if...else statement** - Gebruik deze als je code wilt laten uitvoeren als een gespecificeerde conditie waar is (true) en andere code als de conditie niet waar is (false)
- **if...else if...else statement** - Gebruik deze als één van meerdere blokken code uitgevoerd moet worden.
- **switch statement** - Gebruik deze als je een blok code, van meerdere, wilt uitkiezen om uitgevoerd te worden.

If Statement

Gebruik deze als je code wilt laten uitvoeren als een gespecificeerde conditie waar is (true).

Syntax

```
if (condition)
{
  code die wordt uitgevoerd als conditie true is}
```

N.b.: if is geschreven in kleine letters. Het gebruik van hoofdletters (IF) zou resulteren in een JavaScript error!

Voorbeeld 1

```
<script type="text/javascript">
//Schrijf "Goede morgen" groet als
//het vroeger is dan 10
var d=new Date()
var time=d.getHours()

if (time<10)
{
document.write("<b>Goede morgen</b>")
}
</script>
```

Voorbeeld 2

```
<script type="text/javascript">
//Write "Lunch-tijd!" als de tijd 11 is
var d=new Date()
var time=d.getHours()

if (time==11)
{
document.write("<b>Lunch-tijd!</b>")
}
</script>
```

N.b.: Als je variabelen vergelijkt gebruik je twee gelijk aan tekens(==)!

If...else Statement

Gebruik deze als je code wilt laten uitvoeren als een gespecificeerde conditie waar is (true) en andere code als de conditie niet waar is (false).

Syntax

```
if (condition)
{
code die wordt uitgevoerd als de conditie waar is(true)
}
else
{
code die wordt uitgevoerd als de conditie niet waar is(false)}

```

Voorbeeld

```
<script type="text/javascript">
//als de tijd kleiner dan 10 is,
//krijg je een "Goede morgen" groet.
//Anders krijg je een "Goedendag" groet.
var d = new Date()
var time = d.getHours()

if (time < 10)
{
document.write("Goede morgen!")
}
else
{
document.write("Goedendag!")
}
</script>
```

If...else if...else Statement

Gebruik deze als één van meerdere blokken code uitgevoerd moet worden op basis van de conditie.

Syntax

```
if (condition1)
{
code die wordt uitgevoerd als conditie1 waar(true) is
}
else if (condition2)
{
code die wordt uitgevoerd als conditie2 waar(true) is
}
else
{
code die wordt uitgevoerd als conditie1
en conditie2 niet waar zijn (false)
}
```

Voorbeeld

```
<script type="text/javascript">
var d = new Date()
var time = d.getHours()
if (time<10)
{
document.write("<b>Goede morgen</b>")
}
else if (time>10 && time<16)
{
document.write("<b>Goededag</b>")
}
else
{
document.write("<b>Hallo Wereld!</b>")
}
</script>
```

De JavaScript Switch Statement

Gebruik deze als je een blok code, van meerdere, wilt uitkiezen om uitgevoerd te worden.

Syntax

```
switch(n)
{
case 1:
    voer code blok 1 uit
    break
case 2:
    voer code blok 2 uit
    break
default:
    code die moet worden uitgevoerd als n
    verschilt van 1 en 2
}
```

Zo werkt het: Als eerste wordt een enkele expressie n (vaak een variabele), geëvalueerd. De waarde van de expressie wordt dan vergeleken met de waarde voor elke case. Als de waarde overeenkomt zal de code van die case worden uitgevoerd. Gebruikt **break** om te voorkomen dat de code van de volgende case ook wordt uitgevoerd.

Voorbeeld

```
<script type="text/javascript">
//Je krijgt een verschillende begroeting gebaseerd
//op welke dag het is. (Sunday=0),
//Monday=1, Tuesday=2, etc.
var d=new Date()
theDay=d.getDay()
switch (theDay)
{
case 5:
    document.write("Eindelijk Vrijdag")
    break
case 6:
    document.write("Super Zaterdag")
    break
case 0:
    document.write("Uitslaap Zondag")
    break
default:
    document.write("Ik kijk uit naar het weekeinde!")
}
</script>
```

JavaScript Operatoren

Wiskundige Operatoren

Operator	Omschrijving	Voorbeeld	Resultaat
+	Optellen	x=2 y=2 x+y	4
-	Aftrekken	x=5 y=2 x-y	3
*	Vermenigvuldigen	x=5 y=4 x*y	20
/	Delen	15/5 5/2	3 2.5
%	Modulus (restwaarde van delen)	5%2 10%8 10%2	1 2 0
++	Vergroten	x=5 x++	x=6
--	Verkleinen	x=5 x--	x=4

Opdracht Operatoren

Operator	Voorbeeld	Is Hetzelfde Als
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
=	x=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

Vergelijkings Operatoren

Operator	Omschrijving	Voorbeeld
==	is gelijk aan	5==8 geeft false terug
===	is gelijk aan (kijk naar waarde en type)	x=5 y="5" x==y geeft true terug x===y geeft false terug
!=	is niet gelijk	5!=8 geeft true terug
>	is groter dan	5>8 geeft false terug
<	is kleiner dan	5<8 geeft true terug
>=	is groter dan of gelijk aan	5>=8 geeft false terug
<=	is kleiner dan of gelijk aan	5<=8 geeft true terug

Logische Operatoren

Operator	Omschrijving	Voorbeeld
&&	en	x=6 y=3 (x < 10 && y > 1) geeft true terug
	of	x=6 y=3 (x==5 y==5) geeft false terug
!	niet	x=6 y=3 !(x==y) geeft true terug

String Operator

Een string is meestal tekst, bijvoorbeeld "Hallo Wereld!". Om twee of meer string variabelen aan elkaar te koppelen gebruiken we de + operator.

```
txt1="wat een hele"  
txt2="fijne dag!"  
txt3=txt1+txt2
```

De variabele txt3 bevat nu "Wat een helefijne dag!".

Om een spatie tussen de twee strings te maken kan je een spatie toevoegen in de expressie of in een van de strings.

```
txt1="Wat een hele"  
txt2="fijne dag!"  
txt3=txt1+" "+txt2  
of  
txt1="Wat een hele "  
txt2="fijne dag!"  
txt3=txt1+txt2
```

De variabele txt3 bevat nu "Wat een hele fijne dag!".

Conditionele Operator

JavaScript kent ook een conditionele operator welke een waarde geeft aan een variabele gebaseerd op een conditie.

Syntax

```
variablename=(condition)?value1:value2
```

Voorbeeld

```
greeting=(visitor=="PRES")?"Geachte President ":"Geachte"
```

Als de variabele visitor gelijk is aan PRES, dan de string "Geachte President " in de variabele greeting zetten. Zoniet dan "Geachte" in de variabele greeting zetten.

JavaScript Gebeurtenissen

Een belangrijke categorie eigenschappen die je in JavaScript kan gebruiken, zijn de gebeurtenissen (events), die voor een object kunnen optreden.

Onload en OnUnload

De gebeurtenis 'onload' treedt op als de browser klaar is met het laden van het document. OnUnload treedt op bij het verwijderen van het document uit het browservenster of frame.

Je kan de attributen Onload en OnUnload gebruiken voor de elementen 'body' en 'frameset'.

OnClick en OnDbClick

De gebeurtenis OnClick treedt op als de gebruiker met de muis (of ander aanwijsapparaat) op een object klikt.

De gebeurtenis OnDbClick treedt op als de gebruiker met de muis dubbelklikt op een object. De attributen OnClick en On DbClick kan je gebruiken voor de meeste elementen.

OnMouseDown en OnMouseUp

De gebeurtenis OnMouseDown treedt op als de primaire knop van de muis (of ander aanwijsapparaat) wordt ingedrukt boven een object.

OnMouseUp treedt op als de muisknop weer wordt losgelaten boven het object.

(OnClick is dus een combinatie van OnMouseDown en OnMouseUp.)

Deze attributen kan je ook voor de meeste elementen gebruiken.

OnMouseOver, OnMouseMove en OnMouseOut

De volgende gebeurtenissen hebben te maken met beweging van de muisaanwijzer:

OnMouseOver treedt op als de muisaanwijzer op een object wordt geplaatst, als de muisaanwijzer boven het object wordt verplaatst treedt OnMouseMove op en als de muisaanwijzer van het object wordt afgehaald treedt OnMouseOut op.

Ook deze attributen kunnen voor de meeste elementen worden gebruikt.

OnKeyPress, OnKeyDown en OnKeyUp

De gebeurtenis OnKeyPress treedt op als een toets wordt ingedrukt en losgelaten boven een object.

OnKeyDown treedt op als een toets wordt ingedrukt boven een object en OnKeyUp treedt op als de toets wordt losgelaten boven het object.

(OnKeyPress is dus een combinatie van OnKeyDown en OnKeyUp.)

Deze attributen kan je voor de meeste elementen gebruiken.

OnSelect

De gebeurtenis OnSelect treedt op wanneer de gebruiker tekst in een invoervak selecteerd. Dit attribuut kan je alleen gebruiken bij de HTML elementen 'input' en 'textarea'.

OnFocus en OnBlur

De gebeurtenis OnFocus treedt op wanneer een object de focus krijgt en OnBlur treedt op als het object de focus verliest.

(Een object heeft de focus als het klaar is voor het ontvangen van invoer.)

Deze attributen kan je gebruiken bij de elementen 'a' en 'area', en bij de formulier elementen 'button', 'input', 'label', 'select' en textarea.

OnChange

De gebeurtenis OnChange treedt op wanneer een besturingselement de focus verliest en de waarde van het element is gewijzigd nadat het de focus kreeg.

Dit attribuut kan je alleen gebruiken voor de elementen 'input', 'select' en 'textarea'.

OnSubmit en OnReset

De gebeurtenis OnSubmit treedt op wanneer een formulier wordt verzonden. De gebeurtenis OnReset treedt op wanneer een formulier wordt gewist.

Deze attributen kunnen uitsluitend worden gebruikt bij het element 'form'.

JavaScript heeft een groot nadeel.

JavaScript wordt uitgevoerd in de clientbrowser op de clientcomputer, als de clientbrowser JavaScript ondersteunt en als JavaScript niet is uitgeschakeld.

Om er voor te zorgen dat oudere browsers die geen JavaScript ondersteunen en browsers waarin JavaScript is uitgeschakeld het script niet als tekst weergeven zetten we het script tussen zogenaamde 'comment'-tags.

Voorbeeld van printknop:

```
<script type="text/javascript">
<!--
var strPrint = "<img alt='Afdrukken' onclick='window.print();'
src='printer.gif' style='cursor: hand;'";
document.write(strPrint);
-->
</script>
```

JavaScript Alerts

Waarschuwingvenster maken

Waarschuwingvenster met tekst en een OK knop.

```
window.alert("Dit is een alert!")
```

Bevestigingsvenster maken

Bij een interactieve webpagina is het bevestigingsvenster (confirm) onmisbaar. Via een confirm kan de bezoeker op een vraag reageren met OK of Annuleren.

Een bevestigingsvenster gebruik je altijd samen met de stuuropdrachten 'if' en 'else'. Als de bezoeker op OK klikt dan worden de opdrachten uitgevoerd die bij 'if' horen, wordt er op annuleren gedrukt dan worden de opdrachten uitgevoerd die bij 'else' horen.

```
window.confirm("Dit is een confirm!")
```

In de bovenstaande voorbeelden mag de objectnaam 'window' weggelaten worden. Zonder objectnaam geldt de opdracht voor het actieve browservenster.

JavaScript Tips

JavaScript is hoofdletter gevoelig

Een functie met de naam "mijnfunctie" is niet dezelfde als "mijnFunctie". Let daarom goed op bij het gebruik van hoofdletters en kleine letters wanneer je een variabele, een object of functie creëert of aanroept.

Symbolen

Open symbolen zoals ({ [" ' moeten ook een bijpassend sluit symbool hebben, zoals ' "] }).

Extra spaties

JavaScript negeert extra spaties. Je kunt extra spaties toevoegen om je script meer leesbaar te maken.

Deze regels zijn hetzelfde:

```
naam="Hege"  
naam = "Hege"
```

Afbreken van een code regel

Je kunt een regel code binnen een tekst string afbreken met een backslash. Het voorbeeld hieronder wordt wel goed getoond:

```
document.write("Hallo \  
Wereld!")
```

N.B: Je mag een regel code niet afbreken op de volgende manier:

```
document.write \  
("Hallo wereld!")
```

Bovenstaand voorbeeld genereert een foutcode.

Invoegen van speciale leestekens

Je kunt speciale leestekens invoegen (zoals " ' ; &) met een backslash:

```
document.write ("You \& I sing \"Happy Birthday\".")
```

Het voorbeeld hierboven geeft de volgende output:

You & I sing "Happy Birthday".

Commentaar

Je kunt commentaar toevoegen aan JavaScript code door het commentaar vooraf te laten gaan door twee slashes "///
sum=a + b //Bereken het totaal

Je kunt ook commentaar toevoegen aan de JavaScript code door te beginnen met "/*" en te eindigen met "*/".

```
sum=a + b /*de som w*/
```

het gebruik van "/*" en "*/" is de enige manier om meerdere regels commentaar toe te voegen:

```
/* Dit is een blok commentaar.
```

```
Het bevat meerdere regels*/
```

JavaScript Functies

Een functie is een hergebruikbaar stukje code, welke gestart wordt door een gebeurtenis of als de functie wordt aangeroepen.

JavaScript Functies

Om te voorkomen dat de browser een script wil uitvoeren als de pagina wordt geladen, kan je het script in een functie zetten.

Een functie bevat code die wordt uitgevoerd door een gebeurtenis of door een aanroep van deze functie.

Je kan een functie van waar dan ook in de pagina aanroepen (of zelfs vanuit andere pagina's als de functie in een extern .js bestand staat.)

Functies kunnen in de <head> en in de <body> sectie van een document gedefiniëerd worden. Maar, om er zeker van te zijn dat de functie is gelezen/geladen door de browser, voordat de functie is aangeroepen, is het verstandig om de functie in de <head> sectie te zetten.

Voorbeeld

```
<html>
<head>
<script type="text/javascript">
function displaymessage()
{
alert("Hello World!")
}
</script>
</head>
<body>
<form>
<input type="button" value="Click me!"
onclick="displaymessage()" >
</form>
</body>
</html>
```

Als de regel: alert("Hello world!!") in het bovenstaande voorbeeld, niet in een functie zou zijn geplaatst, zou het uitgevoerd zijn zodra de pagina werd geladen. Nu, zal het script niet worden uitgevoerd voordat een gebruiker op de knop klikt. We hebben een onClick gebeurtenis(event) aan de knop toegevoegd, welke de functie displaymessage() zal uitvoeren zodra er op de knop wordt geklikt.

Hoe een Functie te Definiëren

De syntax voor het maken van een functie is:

```
function functienaam(var1, var2, ..., varX)
{
  enige code
}
```

var1, var2, etc zijn variabelen of waarden die in de functie nodig zijn. De { en de } definiëren het begin en het eind van de functie.

N.b.: Een functie zonder parameters moet wel de haakjes bevatten () achter de functie naam:

```
function functienaam()
{
  enige code
}
```

N.b.: Vergeet niet dat JavaScript hoofdletter gevoelig is! Het woord function moet in kleine letters worden geschreven, anders volgt er een JavaScript error! Let ook op dat je een functie exact met de zelfde naam (en hoofdletters) aanroept als in de functie naam is gedeclareerd.

De return Opdracht (Statement)

De return opdracht wordt gebruikt om een waarde te specificeren welke door de functie wordt terug gegeven.

Dus, functies die een waarde terug moeten geven, moeten de return opdracht gebruiken.

Voorbeeld

Onderstaande functie zou de vermenigvuldiging van twee getallen terug moeten geven (a en b):

```
function prod(a,b)
{
  x=a*b
  return x
}
```

Als je de bovenstaande functie wil aanroepen, moet je twee parameters meegeven:

```
product=prod(2,3)
```

De waarde die wordt teruggegeven uit de prod() functie is 6, en zal in een variabele genaamd product worden opgeslagen.

JavaScript Loops

In JavaScript worden loops gebruikt om een zelfde blok code een aantal keer te doorlopen of zolang een bepaalde conditie waar (true) is.

JavaScript Loops

Wanneer je code schrijft wil je vaak dat een zelfde blok code zich herhaald. In plaats van deze code meerdere keren in het script te plaatsen kan je een loop gebruiken om zo een taak uit te voeren.

In JavaScript kennen we twee soorten loops:

- **for** - loopt een gespecificeerd aantal keer door de code
 - **while** - loopt door de code zolang een bepaalde conditie waar is.
-

De for Loop

De for loop wordt gebruikt als je van te voren weet hoe vaak de code uitgevoerd moet worden.

Syntax

```
for (var=startwaarde;var<=eindwaarde;var=var+operator)
{
    code om uit te voeren
}
```

Voorbeeld:

Uitleg: Het onderstaande voorbeeld definieert een loop welke start met i=0. De loop gaat door zolang i kleiner of gelijk is aan 10. Er zal steeds 1 bij i opgeteld worden als de loop een keer doorlopen wordt.

N.b.: De operator kan ook negatief zijn en kan elke vergelijkings opdracht hebben. (<=)

```
<html>
<body>
<script type="text/javascript">
var i=0
for (i=0;i<=10;i++)
{
document.write("Het getal is " + i)
document.write("<br />")
}
</script>
</body>
</html>
```

Resultaat

```
Het getal is 0  
Het getal is 1  
Het getal is 2  
Het getal is 3  
Het getal is 4  
Het getal is 5  
Het getal is 6  
Het getal is 7  
Het getal is 8  
Het getal is 9  
Het getal is 10
```

De while loop

De while loop wordt gebruikt als je de loop wilt uitvoeren zolang een gespecificeerde conditie true is.

Syntax

```
while (var<=eindwaarde)  
{  
    code die uitgevoerd moet worden  
}
```

N.b.: De <= kan elke vergelijkings opdracht zijn.

Voorbeeld

Uitleg: Het onderstaande voorbeeld definieert een loop welke start met **i=0**. De loop zal doorgaan zolang **i** kleiner of gelijk is aan 10. Er zal steeds 1 bij **i** opgeteld worden als de loop een keer doorlopen wordt.

```
<html>  
<body>  
<script type="text/javascript">  
var i=0  
while (i<=10)  
{  
document.write("Het getal is " + i)  
document.write("<br />")  
i=i+1  
}  
</script>  
</body>  
</html>
```

Resultaat

```
Het getal is 0  
Het getal is 1  
Het getal is 2  
Het getal is 3  
Het getal is 4  
Het getal is 5  
Het getal is 6  
Het getal is 7  
Het getal is 8  
Het getal is 9  
Het getal is 10
```

De do...while Loop

De do...while loop is een variant op de while loop. Deze loop zal altijd de code ÉÉN keer uitvoeren en de loop herhalen zolang de gespecificeerde conditie true is. De code wordt dus altijd op zijn minst een keer uitgevoerd, zelfs als de conditie false is, omdat de code wordt uitgevoerd voordat de conditie getest wordt.

Syntax

```
do  
{  
    code die uitgevoerd moet worden  
}  
while (var<=eindwaarde)
```

Voorbeeld

```
<html>  
<body>  
<script type="text/javascript">  
var i=0  
do  
{  
document.write("Het getal is" + i)  
document.write("<br />")  
i=i+1  
}  
while (i<0)  
</script>  
</body>  
</html>
```

Resultaat

```
Het getal is 0
```