Functies zijn data

**Functies in JavaScript zijn eigenlijk gegevens. Dit is een belangrijk concept om later begrijpen welke 'rare' dingen we met functies kunnen doen.**

Beschrijving

Als functies data zijn, moet je net als bij variabelen, een functie kunnen toewijzen aan een variabele:

var eenFunctie = function () {

return 1;

};

Deze manier van het definiëren van een functie wordt soms aangeduid als een letterlijke functienotatie (function literal notation).

Het deel function () {return 1;} is een *functieexpressie*. Een functieexpressie kan optioneel een naam hebben. In dat geval heet het een benoemde functieexpressie (NFE: named function expression). Dus, dit is ook toegestaan, maar in de praktijk zelden gebruikt.

var eenFunctie = function mijnFunctie() {

return 1;

};

Er lijkt geen verschil tussen een benoemde functie-expressie en een functiedeclaratie. Maar ze zijn in feite verschillend. De enige manier om een onderscheid te maken tussen de twee is te kijken naar de context waarin ze worden gebruikt. Functiedeclaraties mogen alleen voorkomen in de programmacode (in een body van een andere functie of in het hoofdprogramma).

De typeof operator, op een variabele die een functie waarde bevat, retourneert de tekenreeks function:

function eenFunctie() {

return 1;

}

var nogEenFunctie = function () {

return 1;

};

typeof eenFunctie;

"function"

typeof nogEenFunctie;

"function"

Omdat functies gegevens zijn die toegewezen worden aan variabelen, gelden dezelfde regels voor de naamgeving van functies als voor de naamgeving variabelen: de naam van een functie mag niet beginnen met een getal. De naam kan een willekeurige combinatie van letters, cijfers, het onderstrepingsteken en het dollar teken bevatten.

**Een voorbeeld**

var optellen = function() {

var i;

var result = 0;

for (i = 0; i < arguments.length; i++) {

result += arguments[i];

}

return result;

}

optellen(1, 2, 3, 4);