Hanzehogeschool Groningen
University of Applied Sciences

GRONINGEN PROGRAMMEERT_

Programmeren is makkelijk

handleiding
HTML les 3

Project van de Pedagogische Academie, Hanzehogeschool Groningen en Groningen Programmeert

Leerdoelen

- De kinderen snappen het verschil tussen JavaScript en HTML
- De kinderen snappen hoe JavaScript met HTML code aangeroepen kan worden
- De kinderen kunnen uitleggen welke code (zowel HTML als JavaScript) verantwoordelijk is voor bepaalde eigenschappen van site

Materialen

- Computer met internetverbinding per tweetal
- Inlogcodes voor de programmeeromgeving

Lesverloop

In deze les leren de leerlingen kennis maken met JavaScript. Met deze programmeertaal kunnen de leerlingen de concepten die ze hiervoor met de Unplugged lessen geleerd hebben, zoals variabelen, voorwaarden en herhalingen, toepassen in een 'echte' programmeertaal.
De les begint met een korte uitleg en demonstratie van hoe JavaScript gebruikt wordt en hoe HTML en JavaScript code 'samenwerken'. Vanuit HTML wordt JavaScript code aangeroepen, die vervolgens invloed heeft op wat er op de site gebeurt.
Vervolgens gaan de leerlingen zelfstandig aan het werk met de opdrachtkaarten. Tussentijds kunnen verschillende dia's uit de presentatie getoond worden, wanneer een significant deel van de klas tegen een probleem aanloopt welke met behulp van de presentatie uitgelegd kan worden.

Deel 1: uitleg van JavaScript

In de afgelopen lessen stond HTML centraal. Met HTML kun je de opmaak, het uiterlijk van de website, beïnvloeden. Echt programmeren, zoals met variabelen, voorwaarden en herhalingen, is met HTML niet mogelijk. Daarom wordt nu JavaScript geïntroduceerd. De leerlingen hebben nu genoeg ervaring opgedaan met HTML en zijn klaar om door te gaan met JavaScript.
Als voorbeeld om JavaScript uit te leggen wordt de <button>-tag gebruikt. In de presentatie staat een slide waarin met HTML een knop op de site geprogrammeerd staat. Deze knop heeft een 'onclick'-attribuut. Dit attribuut geeft aan wat er gebeurt wanneer er op de knop gedrukt wordt door de bezoeker van de site. De waarde van het 'onclick'-attribuut is "zeg_hoi()". Aan de (- en)-haak is te herkennen dat het hier om een JavaScript *functie* gaat. Wanneer er op de knop geklikt wordt, zal de browser op zoek gaan naar de functie met de naam "zeg_hoi()". Hij zoekt hiervoor in de JavaScript code, een apart stuk code wat los staat van de HTML code. Binnen de programmeeromgeving is deze code te vinden *onder* het vak van de HTML code.
In de presentatie staat de JavaScript code op de volgende dia. Hier staat de functie "zeg_hoi()" gedefinieerd. Dat is te herkennen aan het *function* woord. Daarna komt de naam van de functie, in dit geval zeg_hoi(). Daarna volgt de { haak, en even later de } haak. Tussen de { en } haken staat de inhoud van de functie, datgene dat moet gebeuren wanneer de functie aangeroepen wordt. In dit geval verschijnt er een pop-up met de tekst "Hallo vreemdeling!". De alert()-functie zorgt voor de pop-up, de tekst binnen de (- en)-haken geeft aan wat er in de pop-up moet komen te staan.

In het tweede voorbeeld van de presentatie staat een nieuwe functie beschreven, de functie “zeg()”. Tussen de haakjes staat het woord “hoi”. Dat betekent dat de functie “zeg()” het woord “hoi” meekrijgt, als het ware als een cadeautje (de technische term hiervoor is een ‘parameter’). In de JavaScript code van “zeg()” staat vervolgens beschreven wat de functie met deze parameter doet. In de JavaScript code staat “zeg()” geschreven als “zeg(woord)”. Dat betekent dat het cadeautje, het argument dat met de HTML code is meegegeven, het label “woord” krijgt. Vervolgens wordt dit label gebruikt om het “woord” meteen te gebruiken in de functie alert(). Zodoende laat de pop-up de waarde van het “woord” zien.
In het derde voorbeeld wordt voorwaardelijk programmeren uitgelegd. De voorwaarde wordt getoetst met behulp van een variabele. Wanneer de variabele met de naam ‘meubel’ de waarde ‘lamp’ heeft, wordt de functie “doe_dit()” aangeroepen. In alle andere gevallen, wanneer de variabele een andere waarde heeft, zal deze functie niet aangeroepen worden. Let erop dat het dubbele =-teken betekent dat de waarde van een variabele gecontroleerd wordt, en niet dat de variabele de waarde ‘waarde’ krijgt, wat wel zou gebeuren wanneer er een enkel =-teken gebruikt zou worden.
In het laatste voorbeeld wordt een herhalingslus uitgelegd. Een herhalingslus, de *for*-loop, heeft drie eigenschappen of argumenten. Het eerste argument geeft de beginwaarde van de teller i aan, in dit geval 1. Het tweede argument geeft aan tot wanneer de functie herhaal moet worden. In dit geval gaat de lus *zolang* i kleiner is dan 10. Het laatste argument, in dit geval en een veelvoorkomende notatie, geeft aan dat de teller i na afloop van *een herhalingsronde* vergroot wordt. De functie “doe_dit()” wordt in dit voorbeeld 9 keer uitgevoerd. Wanneer i na de 9^{e} iteratie de waarde 10 heeft gekregen, zal de lus niet nog een keer herhaald worden, omdat er niet aan de voorwaarde “i < 10” wordt voldaan.

Deel 2: aan de slag!

De kinderen kunnen nu zelfstandig met de programmeeromgeving aan de slag. Zoals eerder genoemd kan de zelfstandige verwerking kort onderbroken worden om klassikaal een nieuw aspect van JavaScript te introduceren, zoals de functie met parameters of een lus.

Afsluiting

Vraag de leerlingen of alles duidelijk is. Bij gevallen van onduidelijkheid kunnen de desbetreffende tags nogmaals uitgelegd worden aan de hand van voorbeelden binnen de programmeeromgeving, of kunnen de benodigde slides opnieuw vertoond worden.