Arrays

Wat zijn arrays

- Een variabele kan telkens maar één waarde bevatten, maar soms is het nuttig om bepaalde variabelen te koppelen in een reeks van gerelateerde data. In Javascript kunnen we hiervoor een array of reeksvariabele voor gebruiken.

- Een array of reeksvariabele is een geordende verzameling van elementen. Elk element heeft een genummerde positie. Het eerste element staat op positie nul en alle genummerde posities samen vormen de index.

Arrays declareren

- Het declareren van een array kan op verschillende manieren.

```
// Een lege reeksvariabele.
let leeg = new Array();


// Een lege reeksvariabele met plaats voor 5 elementen.
let werkdag = new Array(5);
```

```
// Een lege array met plaats voor 5 elementen die erna gevuld wordt.
let werkdag = new Array(5);
werkdag[0] = “Maandag”;
werkdag[1] = “Dinsdag”;
werkdag[2] = “Woensdag”;
werkdag[3] = “Donderdag”;
werkdag[4] = “Vrijdag”;


// Een array met de vijf werkdagen als elementen.
let werkdag = new Array(“Maandag”, “Dinsdag”, “Woensdag”, “Donderdag”,
“Vrijdag”);


// Een array met de vijf werkdagen als elementen.
let werkdag = [“Maandag”, “Dinsdag”, “Woensdag”, “Donderdag”, “Vrijdag”];
```

Arrays uitlezen

- We kunnen de elementen in een array gemakkelijk opvragen en uitlezen.

```
// Geef de volledige array terug met waarde en lengte.
console.log(werkdag);

// Geef één element terug op basis van de positie in de index.
console.log(werkdag[0]);

// Slaat een element uit de array op als aparte variabele.
let maandag = werkdag[0];
```

Lengte van arrays opvragen

- Elke array heeft een bepaalde hoeveelheid elementen, maar soms weten we niet op voorhand hoeveel elementen er in een array staan.
- Via de eigenschap ‘length’ kunnen we het aantal elementen van een array eenvoudig opvragen.

```
// Aantal elementen van een array opvragen.
console.log(werkdag.length);
```

Arrays eenvoudig aanpassen

- We kunnen elementen toevoegen, vervangen of verwijderen in een array.

```
// Voegt op positie 5 een element toe aan de array.
werkdag[5] = “zaterdag”;

// Vervangt op positie 0 een element in de array.
werkdag[0] = “zondag”;

// verwijder de elementen in de array door de lengte aan te passen.
werkdag.length = 0;
```

Methoden en functies

- Er bestaan verschillende methoden en functies om de inhoud van een array aan te passen.

- In deze cursus bekijken we enkel *de meest gebruikte*…

Join en split

- De join-methode zet alle elementen in de array om naar een string en voegt ze samen tot één lange string.

- Via de spit-methode kunnen we één lange string omzetten naar een array met meerdere elementen. Bij de split-methode moeten we altijd het scheidingsteken meegeven.

Sort

- Via de sort-methode kunnen we alle elementen in de array sorteren in alfabetische volgorde van **A** naar **z**.

Reverse

- Via de reverse-methode kunnen we de volgorde van alle elementen in de array omdraaien.

Push en unshift

- Via de push-methode kunnen we elementen aan het einde van een array toevoegen.

- Met de unshift-methode kunnen we elementen aan het begin van een array toevoegen.

Pop en shift

- Via de pop-methode kunnen we één element aan het einde van een array verwijderen.

- Met de shift-methode kunnen we één element aan het begin van een array verwijderen.

Concat

- Met de concat-methode kunnen we twee arrays samenvoegen tot één nieuwe array.

Functies

Wat zijn functies

- Een functie is een stukje code met een naam.
- Functies zorgen ervoor dat we bepaalde stukjes code makkelijk meermaals kunnen gebruiken zonder deze opnieuw te moeten schrijven.
- Dit is onderhoudsvriendelijker, want we moeten slechts op één plaats onze stukjes code beheren.

Functies definiëren

- Om een zelfgeschreven functie in Javascript te kunnen gebruiken, moeten we deze definiëren.

- Er bestaan twee manieren om functies te definiëren:
  - De functie expressie
  - De functie declaratie

Functie expressie

- Een functie expressie is een gewone variabele declaratie waarbij de waarde van de variabele een functie is.

- De functie is enkel beschikbaar na de definitie.

Functie expressie

```
let vierkant = function(lengte) {
       return lengte * lengte;
};
console.log(vierkant(5));
```

Functie declaratie

- Een functie declaratie is ‘hoisted’ (omhooggebracht) en geeft ons meer vrijheid over de locatie in onze code.

- De functie is hierdoor ook beschikbaar in de code die voorafgaat aan de functie declaratie.

Functie declaratie

```
console.log(kubus(5));
function kubus(lengte) {
	return lengte * lengte * lengte;
}
```

Opbouw van functies

Een zelfgeschreven functie bevat ‘meestal’ volgende onderdelen:

- Een unieke naam
- Een lijst met nodige parameters
- Een body met statements

```
function functieNaam(parameter1, parameter2) {
    // statements waarin we de parameters gebruiken
}
```

Unieke naam

Voor de naamgeving van een functie moeten we dezelfde regels hanteren als voor de naamgeving van een variabele:

- Het eerste karakter moet een letter of underscore (_) zijn.
- De naam mag zowel hoofdletters als kleine letters bevatten.
- De naam mag letters, getallen, underscore en dollartekens bevatten.
- De naam mag geen spaties of andere lees- of speciale tekens bevatten.
- De naam mag geen gereserveerd woord zijn.

Unieke naam

- Een functie kan ook geen naam hebben. We spreken dan van een anonieme functie of IIFE (Immediately-Invoked Function Expression).

- Deze functie wordt meteen uitgevoerd en kunnen we niet meer hergebruiken.

```
( function(parameter) {
    // statements waarin we de parameter gebruiken;
}(waardeParameter));
```

```
// Bereken de prijs inclusief 21% btw voor een bedrag exclusief btw.
// Uitkomst in console is 121.
( function(bedrag) {
      console.log(bedrag * 1.21);
}(100));
```

Lijst met parameter(s)

- Een parameter is een waarde die door de functie intern wordt gebruikt.
- De parameters staat na de unieke functienaam tussen haakjes. Indien er meerdere parameters nodig zijn in een functie, worden deze gescheiden met behulp van een komma.
- Parameters zijn niet verplicht in een functie. Sommige functies kunnen perfect zonder parameters werken. De haakjes na de functienaam blijven dan leeg.

```
// Schrijf 'Test' in een HTML-element.
function schrijfTest() {
	document.getElementById("uitkomst").innerHTML = "Test";
}
schrijfTest();



// Tel twee getallen op.
function optellen(getal1, getal2) {
	return getal1 + getal2;
}
console.log(optellen(12, 15));
console.log(optellen(1, 2));
```

```
// Geef een naam op als variabele en gebruik deze via een functie.
let volledigeNaam = “Kimberly Willems”;

function weergevenNaam(naam) {
       console.log(“Uw naam is “ + naam + “.”);
}
weergevenNaam(volledigeNaam);
```

Waarden retourneren

- Functies kunnen één enkelvoudige waarde (getal, string, boolean, …) als resultaat teruggeven via het statement ‘return’.

- Na het statement ‘return’ wordt de functie automatisch beëindigd.

Opgelet! Wanneer we meerdere waarden willen retourneren uit een functie, moeten we deze wrappen in een object. Objecten zien we in het volgende hoofdstuk.

```
// Bereken het kwadraat van het getal 5.
function kwadraat(getal) {
	return getal * getal;
}
console.log(kwadraat(5));
```

```
// Bereken het kwadraat van het getal 5.
function kwadraat(getal) {
	return getal * getal;
}
let uitkomst = kwadraat(5);
console.log(uitkomst);
```