JS - arrays gebruiken

* **Een array is een data store**
* **een array bevat geïndexeerd elementen**
* **indexen starten vanaf nul en verhogen met één voor elk element in de array**
* **voor toegang tot een element van een array, gebruik je de index tussen vierkante haken**
* **een array kan elk soort datatype bevatten, met inbegrip van andere arrays**

Declareren en initialiseren

Na de fundamentele primitieve gegevenstypen in JavaScript, is het tijd om naar een meer krachtige gegevensstructuur — de array over te stappen.

Wat is een array? Het is gewoon een lijst (een reeks) van waarden. In plaats van één enkele variabele om één waarde op te slaan, kan je een array variabele gebruiken voor het opslaan van een willekeurig aantal waarden. Die waarden vormen de elementen van de array.

Een array variabele declareer je als volgt:

// declareren

var first\_array = [];

Je kan een array declareren en initialiseren tegelijkertijd:

// declareren en initialiseren

var second\_array = [1, 2, 3];

Array() is een ingebouwde functie die als een constructor functie kan gebruiken om array's te maken:

var a = new Array();

Dat is het zelfde als:

var a = [];

Vermits array's gemaakt kunnen worden met een constructor functie zijn het eigenlijk objecten. Je kan dit nagaan met de typeof operator:

var a = new Array(5);

typeof a;

"object"

Vermits array's objecten zijn erven ze de eigenschappen en dus ook de methoden over van het moeder Object:

var a = new Array('Jan');

a.constructor;

function Array() { [native code] }

a.toString();

"Jan"

a.valueOf();

[

0: "Jan",

length: 1

]

Array's zijn wel objecten maar van een speciale soort:

* de namen van hun eigenschappen worden automatisch toegekend, het zijn getallen die index voorstellen te beginnen met 0;
* ze hebben een length eigenschappen die het aantal elementen in de array retourneert;
* ze hebben meer methoden dan een gewoon object;

Element ophalen

Hoe krijg je toegang tot de waarden die zijn opgeslagen in een array? De elementen in een array worden geïndexeerd met opeenvolgende nummers vanaf nul. Het eerste element is index (of positie) 0, de tweede heeft indexnummer 1, enzovoort. Hier is de drie-element array uit het vorige voorbeeld:

|  |  |
| --- | --- |
| Sleutel/ Index | Waarde |
| 0 | 1 |
| 1 | 2 |
| 2 | 3 |

Om een array-element op ta halen, geef je de index van dat element tussen vierkante haken mee. Dus, een [0] geeft je het eerste element van de array a, een [1] geeft het tweede, enzovoort:

a[0];

1

a[1];

2

Element wijzigen

Met de index kan je een element ook wijziggen. Om het tweede element (index 2) in de second\_aray te wijzigen:

second\_array[1] = 11;

Element toevoegen

Om een nieuw element toe te voegen gebruik je een nog niet gebruikte index:

second\_array[3] = 15;

Hiermee voeg je een vierde element aan de bestaande array toe.

Als je een gat laat tussen de opeenvolgende indexen zullen die tussenliggende posities undefined retourneren als je ze ophaalt.

Het doet er niet toe hoe de array werd gemaakt, de manier van elementen toevoegen is hetzelfde:

a[0] = 1;

a[1] = 2;

a[2] = 'David';

a;

[

0: 1,

1: 2,

2: "David",

length: 3

]

Er is een uitzondering op deze regel. Als slechts 1 argument meegeeft aan de constructor functie wordt beschouwd als de lengte van de array:

var a = new Array(5);

a;

[

0: undefined,

1: undefined,

2: undefined,

3: undefined,

4: undefined,

length: 5

]

Element deleten

Om een element de deleten gebruik je de delete operator. Na het verwijderen van een element uit de array blijft de lengte van de array onveranderd. Je krijgt een gat in je array. Als je het element opvraagt met de overeenkomstige index krijg je undefined.

typeof second\_array[2];

"number"

delete second\_array[2];

typeof second\_array[2];

"undefined"

De length eigenschap

De length eigenschap bevat het aantal elementen in de array. Je kan de length eigenschappen instellen. Er worden dan extra plaatsen in de array voorzien, maar de waarde ervan is undefined.

Alle array objecten hebben een length eigenschap.

var a = new Array('Jan');

a.length;

1

Je kan de lengte zelf instellen:

a.length = 2;

2

Een tweede element wordt toegevoegd, maar er is geen waarde aan toegekend.

a[1];

undefined

Je kan er een waarde aan toe kennen:

a[1] = 'Piet';

a[1];

"Piet"

Je kan een array ook inkorten door de lengte kleiner te maken:

a.length = 1;

a;

[

0: "Jan",

length: 1

]

Voorbeelden

fruit = [];

fruit[0] = 'appel';

fruit[3] = 'peer';

fruit[5] = 'ananas';

fruit['plopperdeplop'] = function() {return 'banaan'; };

console.log(`Aantal elementen in fruit array: ${fruit.length}`);

console.log(`Plopperdeplop: ${fruit['plopperdeplop']}`);

console.log(fruit[6]);

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

console.log(fruit[i]);

}