JAVASCRIPT TRAINING

Case 4 - Objecten en functies

Source

Download de source voor deze opdracht om te beginnen met de case. Pak de zip file uit zoals je in case 1 hebt geleerd. Als het goed is zie je nu in WebStorm onder de folder "opdrachten" de sources van case 4 staan.
Presentatie in PDF

Omschrijving

Zojuist is verteld hoe je objecten en functies creeert en hoe je hier vervolgens gebruik van kunt maken. Het doel van deze opdracht is het leren van omgaan met objecten, zowel in JSON als "normale" notatie. In deze opdracht leer je ook over de verschillende manieren van property/methoddeclaratie binnen javascript.

Voer voor het uitvoeren van de opdracht het de commando's "npm install" en "grunt watch" uit. Zorg ervoor dat er gedurende de opdracht geen fouten van grunt op de console verschijnen. Je mag ook, i.p.v. het commando "grunt watch" na het uitvoeren van de opdrachten het "grunt" commando draaien, ook hierbij geldt, zorg ervoor dat grunt geen fouten aangeeft.

Opdracht 1 - JSON

Open in je browser de URL "http://localhost:8000/opdrachten/case4/template.html" om de vierde case te laden. Je ziet het todoscherm van de applicatie voor je.

Open in WebStorm het bestand "opdrachten/case4/gebruikers.js". In dit bestand worden een aantal gebruikersobjecten gecreerd. Een meer gebruikelijke manier om dit te doen is middels het JSON formaat. Schrijf de huidige manier van objectdeclaratie om naar de JSON notatie. Doe ditzelfde voor todos.js. Refresh de browser om te controleren of pagina nog werkt, nadat je de bestanden hebt omgeschreven.

Opdracht 2 - Prototyping

Voor deze opdracht is er een View.js gedefinieerd. In deze file zie je op dit moment een constructor function staan die een viewNaam accepteerd. De bedoeling is dat we een View class maken waaraan we wat generieke view functionaliteiten aan gaan toevoegen. Doorloop onderstaande stappen om de opdracht te voltooien.

Instance variabele

1. Zorg ervoor dat de viewNaam parameter die als constructor parameter wordt meegegeven wordt opgeslagen als instance variabele in de class.
2. Refresh in je browser de pagina en gebruik FireBug om de class te instantieren met een viewNaam parameter. Controleer middels de console of de juiste instancenaam is gezet.
3. Welke constructor is er gedefinieerd op het prototype van het View object? Gebruik FireBug om het antwoord te bepalen.

Instance method

4. Maak een methode logVariable op het View object. Zorg ervoor dat deze methode geinherit kan worden door instanties van het View object. Zorg er dus voor dat dit een instance method wordt. De methode dient één variable te accepteren: variableToLog. De implementatie van deze methode dient de viewNaam uit de constructor te loggen en daarna de meegegeven variable op de console te loggen.
5. Refresh in je browser de pagina en gebruik FireBug om de class te instantieren met een viewNaam parameter. Roep vervolgens de net gemaakte methode aan op de objectinstantie en controleer of zowel de viewNaam als de meegegeven parameter worden gelogd.

Class variabele (static variabele)

6. Maak een constante op de class waarmee je aan kunt geven of je het loggen aan of uit wil zetten. Zorg ervoor dat je niet gebruik maakt van een instance

variabele, maar een class variabele. Zorg voor een duidelijke naam voor de constante.
7. Pas de methode die je in stap 4 hebt gemaakt aan, zodat deze a.d.v. de in stap 6 aangemaakte constante een logregel naar de console maakt. Dat betekend dat als de constante op false staat, er geen logregel op de console moet verschijnen.
8. Refresh in je browser de pagina en gebruik FireBug om de class te instantieren met een viewNaam parameter. Roep vervolgens de logVariable methode aan op de objectinstantie en controleer of zowel de viewNaam als de meegegeven parameter worden gelogd. Zet vervolgens de constante op false en roep nogmaals de logVariable methode aan. Controleer of er daadwerkelijk geen logregel verschijnt.

Class method (static method)

9. Maak een methode op de class waarmee je kunt controleren of er een console beschikbaar is om naar te loggen. Zorg ervoor dat je niet gebruik maakt van een instance method, maar een class method. Zorg voor een duidelijke naam voor de methode.
10. Pas de methode die je in stap 4 hebt gemaakt aan, zodat deze tevens a.d.v. het resultaat van de methode in stap 9 de log statements uitvoert. Als er geen console is, kan er immers niet gelogd worden.
11. Refresh in je browser de pagina en gebruik FireBug om de class te instantieren met een viewNaam parameter. Roep vervolgens de logVariable methode aan op de objectinstantie en controleer of zowel de viewNaam als de meegegeven parameter worden gelogd. Zet vervolgens de console op null of undefined en roep nogmaals de logVariable methode aan. Controleer of er geen logregel en geen foutmelding (error) verschijnt.

Prototype als JSON

12. Er zijn twee manieren om de logVariabele methode te declareren, 1 middels JSON en een andere decleratief. Probeer middels deze opdracht het prototype om te schrijven zodat deze in de andere declaratiestijl wordt gedeclareerd die je nu hebt. Dus als je het prototype met de logVariable methode nu middels JSON hebt gedeclareerd, schrijf hem dan om zodat er niet gebruik gemaakt wordt van JSON. En andersom, als je hem nu niet middels JSON hebt gedeclareerd, zorg er dan voor dat het prototype met de logVariable methode middels JSON wordt gedeclareerd.

Opdracht 3 - Extra opdracht

Mocht je tijd overhebben na het maken van bovenstaande opdrachten, dan kun je je middels deze extra opdracht verdiepen in het maken van objecten en wat standaard functies.

1. Pas het gebruikers.js bestand aan zodat er een Gebruiker object aan wordt toegevoegd met een 3-tal properties. Instantieer dit object vervolgens 3 keer zodat er een array ontstaat van 3 gebruikers.
2. Refresh de browser en controleer of nog steeds de gebruikers aan de linkerkant worden weergegeven. Als dit niet zo is, heb je net een fout gemaakt in stap 1. Als je het goed hebt gedaan dan valt je op dat er voor de gebruikende kant geen verschil is tussen de JSON gebruiker declaratie die er eerst stond, en de class instances die je er nu hebt staan, zoals je hebt gemaakt in stap 1.
3. Pas de View aan zodat de implementatie van de log methode middels de volgende regel een log statement uitvoert: "console.log(this.viewNaam + ' - ' + variableToLog);"
4. Refresh in je browser de pagina en gebruik FireBug om de class te instantieren met een viewNaam parameter. Roep vervolgens de logVariable methode aan op de objectinstantie met "gebruikers" (zonder aanhalingstekens) als parameter. Als het goed is zie je een dergelijke logregel verschijnen: "TodoView - [object Object],[object Object],[object Object]". Hoe denk je dat dit komt?
5. Voeg een instance method to aan het Gebruiker object genaamd "toString". Hiermee override je de toString methode van Object. Implementeer de toString methode zo dat het id en de gebruikersnaam van een Gebruiker worden teruggegeven.
6. Voer nogmaals stap 4 uit. Als het goed is zie je nu een dergelijke logregel verschijnen: "TodoView - [ID: 1 - Gebruikersnaam: jscript],[ID: 2 - Gebruikersnaam: cscherp],[ID: 3 - Gebruikersnaam: scala]".

PUBLISHED ON February 13th, 2015