JAVASCRIPT TRAINING

Case 8 - Uitwerkingen

Download de sources van de uitwerking

1. Blader door de verschillende javascript sources en de js/app folders. Probeer te begrijpen wat de functionaliteiten van de verschillende files zijn. Wat valt je op over de opzet van de applicatie? Welk pattern wordt er toegepast?

De applicatie is opgezet volgens het MV*(VM/C/P/W) pattern. De view is verantwoordelijk voor het uiteindelijke renderen van Todo en Gebruiker instanties. In de view ligt dan ook de logica om met de DOM om te gaan. De controllers zijn verantwoordelijk voor het aansturen van de view. Zo vind bijvoorbeeld het afhandelen van websocket events op de controllers plaats. Als laatste zijn er nog de Storage instanties, die zorgen voor communicatie met de "backend", in dit geval de REST service die voor de daadwerkelijke opslag zorgt.

Het javascript pattern wat werd toegepast toen je begon met de applicatie is het module pattern, wat je in case 6 geleerd hebt.

2. Open vervolgens het bestand "template.html". Wat valt je op/welk nadeel kun je herkennen aan de huidige opzet?

Het bestand template.html bevat nu alle modules die binnen de applicatie bekend zijn. Deze zijn te vinden in de <script> tags onderaan de pagina. Als er nieuwe modules bijkomen, of de modulenaam zou veranderen, dan zal dit template.html bestand moeten worden aangepast. Tevens zal er altijd gezorgd moeten worden dat de <script> tags in de juiste volgorde staan.

5. Refresh je browser en controleer of alles nog werkt als voorheen. Bekijk tevens de FireBug console en controleer dat er geen errors meer optreden. Bekijk tevens het "Net" tabblad in FireBug en controleer dat alle javascript bestanden worden ingeladen. Wat valt je op?

Alle modules worden nog steeds ingeladen, ondanks dat alle <script> tags onderaan de pagina verwijderd zijn. Tevens worden deze in de juiste volgorde ingeladen. De gebruikte AMD implementatie, RequireJS, zorgt hiervoor. RequireJS wordt geinitialiseerd door de enige overgebleven <script> die in de <head> van de pagina staat. Het bestand dat als eerste door RequireJS wordt ingeladen is de module die in het attribuut "data-main" in de <script> tag staat. In dit geval is dat *js/init* wat resulteert in dat het bestand *js/init.js* zal worden ingeladen.

6. Welke taak denk je dat de minification met AMD uitvoert? Naar welke file denk je dat het resultaat zal worden weggeschreven?

De "requirejs" taak zorgt voor de minification. Het resultaat wordt naar "target/build-AMD.js" weggeschreven. Als de buildoption "optimize" van waarde "none" naar "uglify" wordt gezet, wordt er trouwens pas echt minification toegepast. Met "none" als waarde zal de "requirejs" taak er alleen voor zorgen dat alle gevonden modules in de outputfile in de juiste volgorde worden geconcateneerd.

7. Voer de taak die de minification uitvoert uit en bekijk het resultaat. Wat valt je op aan de gegenereerde outputfile?

In de output file, build-AMD.js, staan alle modules in de juiste volgorde achter elkaar.

8. Zonder de outputfile worden modules met AMD gevonden in een file die als naam gelijk is aan de modulenaam. Hoe denk je dat RequireJS de modules nu vind?

Als je goed kijkt naar de "define" en "require" statements, zie je dat deze nu uit 3 parameters bestaat. De tweede en derde parameters zijn net als voorheen de dependencies die gebruikt worden en de function die de body van de module representeerd. De eerste parameter die in de build-AMD.js bij de "define" en "require" statements staat is echter door het minification proces toegevoegd. Deze parameter geeft de modulenaam weer, die gelijk is aan de bestandslocatie + bestandsnaam minus de extensie ".js" van het bestand waar de module oorspronkelijk voor de minification zich bevond.

Als deze modulenaam parameter wordt toegevoegd, en er dus van 3 parameters bij "require" en "define" statements gebruik wordt gemaakt, dan weet RequireJS tijdens het laden van de applicatie a.d.v. deze modulenaam, dat er geen daadwerkelijke javascript file hoeft te worden opgehaald. RequireJS zal dan eerst kijken of er reeds een module geladen is met de modulenaam, wat het geval is als alle modules in 1 keer worden ingeladen zoals met het bestand build-AMD.js het geval is.

PUBLISHED ON March 6th, 2014