Vue.js vs Angular

**Introductie:**

In deze paper ga ik 2 bekende en veel gebruikte javascript bibliotheken vergelijken. We beginnen met een introduciet, wat meer info over de frameworks. Wanneer is het ontstaan, wie is de founder en de rede van het ontstaan.

Angular:

Angular is geboren als GetAngular in 2009 opgericht door een hoopje ingenieurs waaronder Misko Hevery. Het werd redelijk snel populair nadat Misko Hevery een web applicatie her ontwikkelde met GetAngular en de oorspronkelijke hoeveelheid code deelde door 17.

Na deze daad begon google met sponseren en maakte er een open-source framework van: AngularJs. Met de belangrijke features: Two-way-data binding, Depindency injection en het gebruik van html-attributen als directives werd het al snel een enorm populair platform.

Vue.js:

Vue is een veel jongere en kleinere framework. Het is opgericht in 2014 door Evan You en het is gebaseerd op AngularJs en Backbone. Hoewel het nog in de kinderschoenen staat is het al razend polulair en veel kenners zeggen dat de populariteit doorheen de jaren enkel gaat stijgen. Veel websites prijzen het ook als zeer een beloftevolle framework zoals “Javascript weekly” en “HackerNews”. De nieuwe versie is Vue 2.0.

**Gelijkenissen**:

Vue en Angular lijken hard op elkaar. De rede waarom ze zo hard op elkaar lijken ligt ook aan het feit dat Evan You (de developer van Vue) zich heeft gebaseerd op Angular bij het ontwikkelen van Vue. Het zijn beide open-source frameworks dus iedereen kan het gebruiken en er zelf dingen bij aanpassen (voor eigen gebruik).

De 2 frameworks richten zich op dezelfde soort applicaties. De meeste dingen die je met vue kan, kan je ook met Angular. Ze worden allebei het meest gebruikt om een one-page-website te maken. Een website die zo weinig mogelijk data communiceert met de server.

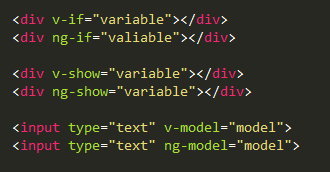
In plaats van per POST of GET een hele pagina op te vragen aan de server kan je met Angular of Vue een Json request doen en zo enkel de data opvragen die je nodig hebt. Zo zal je pagina niet helemaal herladen en zal alles sneller werken.

Ze proberen beide ook het mv\* principe te gebruiken. Model-View-Wildcart (mvc, mvvm,…). Je kan het dus gebruiken om je code te organiseren op een universele manier. Je bent in die structuur ook zeer vrij.

Ze weken beiden ook goed samen met laravel. Ik heb zelf 2 zeer kleine applicaties gemaakt met de 2 frameworks in hetzelfde laravel project en er is geen enkel probleem.

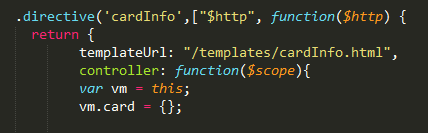
Met Vue kan je zelfs je initiële data rechtstreeks via laravel krijgen zonder eerst een Ajax call te doen.

Wat de 2 typeert zijn vooral de html attributen. Het zijn 2 frameworks die veel gebruik maken van attributen om zo een duidelijke communicatie te maken tussen de javascript en de html. Zo heb je “ng-model” voor Angular en “v-model” voor Vue, beide dezelfde functionaliteit. Je hebt



Het gebruik van al deze attributen zijn zeer kenmerkend aan deze frameworks. Ook het werken met de dubbele accolades is een typisch kenmerk.

Om je code te structureren gebruiken beide frameworken ook een manier om code te scheiden, op te delen in overzichtelijke delen. Zo gebruikt Angular Directives. Dit is een apart staand stuk javascript en Html die je onafhankelijk van elkaar kan gebruiken. Bij Angular kan je het stuk html dat er bij hoort ook in een andere file opslagen. Dit zorgt voor een veel overzichtelijkere code.

Ook vue hanteert een zelfde soort principe maar noemt dit Components. Componenten zijn ook kleine afgescheide stukken html en javascript die je meerdere keren onafhankelijk van elkaar kan gebruiken.

Hier zie je een directive in angular. Achter de variabele templatUrl zie je de path waar het stuk Html zit dat hoort bij deze directive. Je ziet ook wat variabelen zoals vm.card. Deze variabele hebben een scope enkel binnen deze directive.



Hier zie je een component in Vue. Achter de variabbele template zie je een id waar het stuk html van deze component zit. De id wordt gebruikt in de html pagina. Hier worden de stukken html niet in aparte kleine files gestopt. Dit kan wel gedaan worden met een installeren van een package.

In de data functie zie je ook een paar variabelen die een scope hebben enkel binnen dat component.

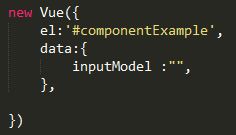
Zoals je ziet zijn er wel wat gelijkenissen tussen componenten bij vue en Directives bij Angular.

Bij zowel Vue als Angular kan je ook gebruik maken van filters. Dit is een heel makkelijke en snelle manier om data vervormen naar een bepaald type. Veel filters doen hetzelfde maar ze krijgen meestal wel een andere naam. Ze worden bij beide frameworks onderscheden van de rest met een “|” teken.

Links ziet u een filter in werking bij vue, recht een filter bij Angular. Ze hebben beide dezelfde functie.

Two-way databinding is ook 1 van de belangrijkste kenmerken van zowel Vue als Angular. Dit is een zeer praktisch en veel gebruikt principe omdat het een hele hoop code achter de schermen doet waarvan je je niets moet aantrekken. Bij beide frameworks werkt dit met het attribuut “model”.

Two-way databinding houdt in dat data wordt upgedate in 2 richtingen. Als je een bepaalde variabele vast hangt aan een input met “ng-model” bij Angular of “v-model” bij Vue. Zal een wijziging van het de input variabele of een wijziging van de variabele in de javascript zorgen voor een algemene verandering van de variabele. Zo kan je data doorgeven naar de javascript en hierop reageren zonder een event te triggeren (bijvoorbeeld een key-down). Met computed properties of $watch events wordt dit nog interessanter maar hier kom ik later op terug.

Met two-way binding is het mogelijk om iets in een input te typen en het te printen met maar een paar lijnen code.

Met deze paar lijnen code wordt het h1 element per key-down aangepast. Dit allemaal door een variabele vast te hangen aan een input field.

Al deze gebruiksvriendelijke en intuïtieve zaken zorgen voor 2 zeer praktische frameworks die beide aan te raden zijn bij het maken van een single web application.

Nu we over de gelijkenissen tussen Angular en Vue zijn gegaan gaan we nu beginnen met de verschillen.

**Verschillen:**

Ik heb 2 zeer simpele single web applicaties gemaakt met behulp van laravel als backand framework. De bedoeling van de pagina is dat je een hoop artikels hebt waarvan enkel de titel wordt getoond. Bij het klikken op een artikel wordt het hele artikel getoond + een form waar je het artikel kan veranderen.

Ik heb deze eenvoudige applicatie 2 maal gemaakt. In Vue en in Angular om zo de verschillen tussen de 2 frameworks goed te begrijpen.

Ik ben er zelf nog niet uit welk framework ik verkies aangezien ze allebei voordelen en nadelen hebben.

Eerst en vooral heb ik gemerkt dat ik met Vue een pak minder code heb geschreven.

Om het verschil duidelijk te maken heb ik een foto genomen van het uitgezoemde beeld rechts vanboven bij sublime tekst editor.

Dit verschil is niet helemaal in Angulars nadeel omdat ik vind dat Angular beter gestructureerd is. Bij angular heb ik 3 directives aangemaakt. Dit was niet echt nodig voor deze kleine applicatie maar dit zorgt er wel voor dat een uibreiding later makkelijk implementeerbaar zal zijn.

Eerste directive zorgt voor de lijst met titles, de 2de directive zorgt voor het tonen van het geselecteerde artikel. De 3de directive zorgt voor het kleine formulier om een artikel aan te passen.

Op goede aan de applicatie met angular is ook dat de 3 html templates die horen bij deze directives in aparte html files konden worden gezet. Dit zorgt voor een enorme boost in de overzichtelijkheid van het project en de hergebruikbaarheid van deze directives. Dus de extra lijnen code geef ik graag aan een betere en beter uitbreidbare structuur.

Bij Vue is de code een pakje minder. Dit komt ook omdat ik maar met 1 component heb gewerkt. Dit omdat de communicatie tussen verschillende directives niet heel handig is als je het een light weight project wil houden. Bij Vue zou ik een aparte Vues instantie moeten aanmaken en deze gebruiken als “brug” om te communiceren tussen verschillende componenten. Dit vind ik een bizarre manier. Daarom heb ik het in 1 component gezet. Deze code zal minder uitbreidbaar zijn maar het behoud de overzichtelijkheid wel.

Bij Vue heb ik de aparte html templates ook niet in aparte files kunnen zetten. Dit is niet bij vue ingebouwd in plaats daarvan refereer ik naar een div in de html pagina waar ik de template tussen heb gezet. Ik moet er wel bijzeggen dat het mogelijk is om de templates in aparte files op te slagen, maar dit met behulp van een package.

Vue heeft dit intentioneel zo gedaan en zegt dat het makkelijker is voor de developers als de templates in dezelfde file als de componenten staan. Dit omdat ze zo nauw samen werken en lastig is om tijdens het developen heel de tijd te switchen van file.

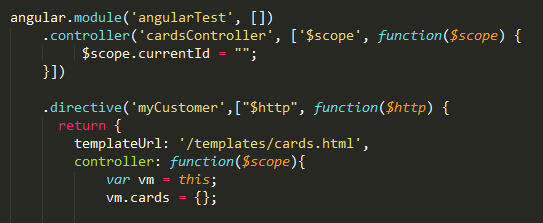
Ze zeggen ook dat het sneller zal werken eenmaal het gedployt is. Angular laad de files in met behulp van ajax en dit zal zorgen voor langere laadtijden.

Ik weet niet of ik het hier volledig met eens ben. Ik heb nog geen ervaring met large scale applicaties in vue maar ik denk dat het na een tijd chaotisch zal worden zonder het opsplitsen van de templates.

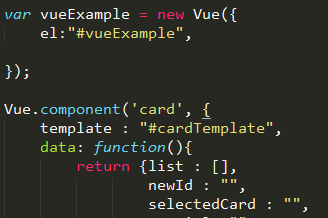
Complexiteit

Tijdens het maken van deze eenvoudige applicatie heb ik gemerkt dat er wel een verschil is in complexiteit. Angular is veel lastiger en frustrerender om met te werken qua code.

Je moet per module ook een controller aanmaken die je dan moet gebruiken met “ng-controller”. Dit maakt alles wat ingewikkelder en minder goed begrijpbaar, zeker voor kleine projecten.



Deze foto toont aan hoe je een nieuwe directive aan kunt maken in angular. Eerst moet er een module zijn waaraan je een directive koppelt.



Deze code toont hoe je een component aan kunt maken in Vue. Er moet ook een Vue intnantie zijn waarin deze gebruikt kan worden. Op deze foto heet deze vueExample.

Hieraan zie je duidelijk een verschil in complexiteit. Angular heeft een hoop meer tools en zorgt ook dat je een complexere applicatie kan bouwen maar voor klein schalige projecten is het minder gemakkelijk en zorgt deze grote hoeveelheid tools voor een grotere leer curve en een ingewikkeldere aanpak.