عنوان: **بررسی مشکلات AngularJS 1.x**
نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۴/۰۶/۲۶ ۱۹:۲۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, MVC, jQuery, AngularJS, Code Review

از اولین مقاله‌ای که در مورد AngularJS در این سایت منتشر کردم، بیش از دو سال می‌گذرد. در آن زمان فقط از این فریمورک تعریف و تمجید کردم؛ اما بد نیست بعد از چند تجربه‌ی کاری دلایل تنفری را که نسبت به آن پیدا کرده ام، نیز بیان کنم.
اگر عبارت why I hate angularjs را در گوگل جستجو کنید، می‌بینید که فقط من این عقیده را پیدا نکرده‌ام و افراد دیگری نیز هستند که مثل من فکر می‌کنند و حتی از لحاظ فنی AngularJS را به چالش کشیده‌اند. برای مثال سایت I hate angular بیشتر مقالاتی را که ضد AngularJS هستند، گردآوری کرده است و برای بررسی مشکلات Angular می‌تواند شروع خوبی باشد.
البته قصد ندارم که از نظر فنی Angular را نقد کنم؛ فقط قصد به اشتراک گذاری یک سری از مشکلات توسعه‌ی Single Page Applicationها را با استفاده از فریمورک Angular، دارم و این را در نظر داشته باشید که بعضی از این مشکلات در هنگام توسعه SPAها با فریمورک‌هایی از این دست، گریبان‌گیر شما می‌شوند و الزاما ربطی به AngularJS ندارند.

سازگار نبودن افزونه‌های jQuery با Angular

برنامه‌های واقعی فقط از تعدادی ng-repeat تشکیل نشده‌اند که ما از دیدن آن‌ها ذوق زده شویم. خواسته یا ناخواسته مجبوریم در برنامه‌های وب خودمان از افزونه‌های محبوب جی‌کوئری نیز استفاده کنیم. خوب، خیلی هم خوب! چندین راه حل پیش روی ماست:

روش اول - نادیده گرفتن angular

انگار نه انگار که از angular استفاده می‌کنیم و افزونه مورد نظر را بدون در نظر گرفتن وجود angular، کاملا عادی فراخوانی کنیم.
نتیجه: ممکن است بعضی وقت‌ها جواب بدهد، ولی اکثر مواقع، نتیجه عجیب غریب است و خطاها قابل فهم نیستند و توانایی اشکال زدایی آن‌ها را نخواهید داشت. دلیلش هم مشخص است؛ چون Angular فازی به نام کامپایل و اصطلاحا context مربوط به خودش را دارد و فراخوانی افزونه مورد نظر، خارج از context انگولار رخ می‌دهد و انگولار از وجود این افزونه بی خبر است. حال ممکن است به طور اتفاقی، فراخوانی افزونه قبل، مابین و یا حتی بعد از فاز کامپایل انگولار رخ دهد. باز هم فرض کنید که بر حسب اتفاق همه چیز خوب پیش رفت، اما اکنون سایر قابلیت‌های خوب انگولار مثل ng- model و model binding آن در دسترس نیستند و در آخر به این نتیجه می‌رسید که پس چرا دارم از انگولار استفاده می‌کنم.

روش دوم - استفاده از directiveهای محصور کننده

راه اصولی برای استفاده از افزونه‌های جی‌کوئری در AngularJS، استفاده از directiveهای تهیه شده برای آن افزونه است. اگر خوش شانس باشید، معمولا برای افرونه‌های معروف، directive انگولاری آن نیز تهیه شده است. اما این همه‌ی داستان نیست؛ فرض کنید که از کتابخانه jQuery file upload ، برای آپلود فایل می‌خواهید استفاده کنید. خوشبختانه directive انگولاری نیز برای آن تهیه شده است و مستندات استفاده از آن هم، تنها مثالی هست که برای آن فراهم شده است. اما فرض کنید که می‌خواهید مانند مثال استفاده از آن در jQuery، یک file input که کاربر تنها بتواند یک فایل را از طریق کشیدن و رها کردن آپلود کند، با استفاده از Directive انگولاری آن پیاده سازی کنید. اما کار با این directive، به آسانی مثال جی‌کوئری آن نیست. یک‌کم که جلوتر بروید می‌بینید که این directive گنگ طراحی شده است. البته بیشتر directive هایی که اصطلاحا wrapper برای افزونه‌های جی‌کوئری هستند این مشکل را دارند و کار با آن‌ها چندان لذت بخش نیست و باید ساعت‌ها با آنها کلنجار رفت تا به نتیجه‌ی دلخواه رسید و همه‌ی این‌ها را در نظر بگیرید که اگر با apiهای jQuery آن کار می‌کردید، دیگر این مشکلات را نداشتید.
قبلا نیز یک نمونه‌ی دیگر از مشکلات استفاده از این گونه directiveهای محصور کننده را تحت مقاله ای با عنوان استفاده از افزونه isotope در انگولار به اشتراک گذاشتم.

روش سوم - استفاده از directive هایی که به صورت native با انگولار نوشته شده‌اند

اما چرا به هنگام استفاده از directiveهای محصور کننده افرونه‌های جی‌کوئری، با مشکلات زیادی روبرو می‌شویم؟ دلیلش این است که انگولار می‌گوید بهتر است این افزونه‌ها با استفاده از خود angular بازنویسی شوند. برای مثال برای آپلود فایل می‌توان از کتابخانه‌ی با کیفیت ng-file-upload که هیچ وابستگی به jQuery ندارد استفاده کرد. اما آیا واقعا برای تمامی افزونه‌های جی‌کوئری معادلی برای AngularJs آن با همان کیفیت تهیه شده است؟ جواب مطمئنا خیر است. برای مثال در حالی که برای datagrid افزونه‌های بی شماری برای جی‌کوئری تهیه شده است، اما برای angular تنها یکی دو تا directive با کیفیت تهیه شده‌است که نه تنها قابلیت رقابت با معادل‌های jQuery شان را ندارند، آنچنان نیز stable نیستند و در مستندات خودشان هشدار

می‌دهند که فلان ویژگی در حال تست هست و هنوز پایدار نیست.

روش چهارم – نوشتن directive توسط خودتان

به عنوان آخرین راه حل باید خودتان دست به کار شده و برای افزونه مورد نظرتان directive بنویسید. اما نوشتن directive برای افزونه‌های پیچیده‌ی جی‌کوئری به سادگی مثال‌های آموزشی AngularJS همانند چگونگی نوشتن directive برای jQueryUI Datepicker نیست. اگر کدهای directiveهای نوشته شده برای افزونه‌های پیچیده را بررسی کنید، کدهایی را می‌بینید که برای شما منطقی نیست. برای مثال ممکن است با تعداد زیادی setTimeOut مواجه شوید که احتمالا با نحوه‌ی کامپایل HTML توسط انگولار مرتبط است. در کل باید بدانید که نوشتن directive برای تعداد زیادی از افزونه‌ها کار راحتی نیست و احتمالش هست که قید این را کار نیز بزنید.

پس اگر قصد توسعه SPA با هر فریمورکی مثل angular را داشته باشید، این را در نظر داشته باشید که دیر یا زود هنگام استفاده از افزونه‌های جی‌کوئری به مشکل برخواهید خورد.

بیشتر امکانات تو کار ASP.NET MVC را از دست خواهید داد

به هنگام توسعه‌ی برنامه با استفاده از فریم ورک‌های SPA، امکانات توکار ASP.NET MVC مثل اعتبارسنجی یکپارچه و strongly typed viewها را از دست خواهید داد. شاید یک سری پروژه در Github پیدا کنید که سعی کرده‌اند این‌ها را با یکدیگر سازگار کنند. اما به محض استفاده متوجه می‌شوید که اگر همه‌ی کارها را خودتان با Angular انجام بدهید راحت‌تر هستید تا استفاده از کتابخانه‌های آزمایشی و ناقص.

البته باز هم نمی‌گویم که این‌ها تقصیر AngularJS است. ذات توسعه‌ی SPAها، این گونه است و در توسعه‌ی SPA با هر فریمورکی به این مشکلات برخواهید خورد.

حال که یکسری مشکلات عمومی را بررسی کردیم، بدنیست نگاهی اختصاصی به خود AngularJS بیندازیم.

ضعف طراحی

اگر به تعدای از لینک‌های سایت ihateangular مراجعه کنید می‌بینید که هر کسی نظری دارد: یکی می‌گوید به هیچ وجه Directive ننویسید، یکی دیگر می‌گوید کنترلر ننویسید و تمامی کارها را در directiveهای سفارشی نوشته شده توسط خودتان انجام بدهید، کلا همه جا علیه performance این فریمورک صحبت می‌کنند و همگی به پیچیده بودن آن اذعان دارند.

اما نمی‌شود با چند مقاله‌ی موجود در اینترنت، یک فریمورک با این محبوبیت را زیر سوال برد. اما واقعا فکر می‌کنید که چرا نسخه‌ی 2 انگولار یک بازنویسی کامل است؟ دلیلش واضح است؛ این فریمورک از پایه اشکال دار بوده است و باید از اساس اصلاح شود. پس می‌توان نتیجه گرفت که اشکالات وارد شده به این فریمورک صحیح هستند.

AngularJs 2 یک بازنویسی کامل است

قبلا این موضوع در این نظرسنجی مطرح شده است. بازنویسی کامل یعنی این که خیلی چیزها به کل تغییر کرده‌اند و کدهای قبلی شما با نسخه‌ی جدید سازگار نیستند. بیشتر مطالبی که فراگرفته بودید دیگر کاربردی ندارد و دوباره مطالب جدیدی را باید یاد بگیرید. این را هم در نظر بگیرید که توسعه دهندگانی که در حال نوشتن directive هستند، احتمالا با آمدن نسخه 2 انگولار، مجبورند directive خود را بازنویسی کنند. آیا خودتان بودید، دیگر دل به کار می‌دادید؟!

نتیجه گیری

AngularJS فریمورک خیلی خوبی برای نوشتن برنامه‌های تست پذیر است و کسی منکر قابلیت‌های آن نیست. ولی این را نیز در نظر بگیرید که برای تست پذیر بودن، خیلی چیزها از جمله سادگی کار را از دست می‌دهید. معمولا می‌گویند که AngularJS کارهای مشکل را ساده می‌کند و کارهای ساده را مشکل.

پیشنهاد من این است که اگر هنوز AngularJS را فرا نگرفته‌اید، حداقل یادگیری آن را تا انتشار نسخه‌ی 2 آن به تعویق بیندازید. اگر AngularJS را بلد هستید، دیگر آن را در پروژه‌ای استفاده نکنید؛ چون دیگر کدهای شما در نسخه‌ی 2 کار نخواهد کرد و احتیاج

به انجام تغییرات گسترده‌ای در کدهای نوشته شده قبلی پیدا می‌کنید.

نظرات خوانندگان

نویسنده: مرتضی حاتمی
تاریخ: ۱۳۹۴/۰۶/۲۶ ۲۳:۲۷

سلام
حالا ما که هنوز سراغش نرفتیم به نظر شما سراغ کدوم فریمورک بریم
ایا ReactJS خوبه؟

نویسنده: محمود راستین
تاریخ: ۱۳۹۴/۰۶/۲۷ ۱۲:۳۵

ممنون بابت این مقاله. یکی از دلایل علاقه من به AngularJs استفاده از Ui Router هستش. میشه بپرسم به نظرتون چه فریمورک دیگه ای کاربردی شبیه به این درش وجود داره که بشه ازش استفاده کرد ؟

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۴/۰۶/۲۷ ۱۳:۱

شاید مطالعه ی سری مقالات EmberJs به شما کمک کنه.

نویسنده: علی پناهی
تاریخ: ۱۳۹۴/۰۶/۲۷ ۲۱:۴۲

نظرتون راجع به آنگولار 2 تا اینجای کار چیه؟
تا اینجای کار توانسته مشکلات را برطرف کند؟
توانسته با کتابخانه‌های دیگه ارتباط برقرار بکند؟

نویسنده: عباسی بهزاد
تاریخ: ۱۳۹۴/۰۶/۲۸ ۱۰:۱۱

سلام مهدی عزیز.
با اومدن Es6 انگولار باید عوض می‌شد.
در انگولار1 می‌توان انگولار 2 نوشت و تبدیلش کرد
توی انگولار 2 فقط استفاده از Decoratorها عوض شده که می‌تونید اونو خودتون باز نویسی کنید و ازش استفاده کنید.
اگه هم انگولار نمیخواید کار کنید اینگونه خرابش نکنید لطفا.
اونایی هم که نمی‌خوان انگولار کار کنن به نظر من Aurelia بهترین گزینه می‌تونه باشه.

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۴/۰۶/۲۸ ۱۱:۵۳

بنده قصد خراب کردن این فریمورک را نداشتم. اشکالات بیان شده همگی مربوط به مقالاتی هستند که توسعه دهندگان به اشتراک گذاشتند. من هم آن‌ها را تایید نکردم ولی وقتی می‌بینید که angular 2 همان اشکالات وارد شده به نسخه 1 را برطرف کرده است می‌توان نتیجه گرفت که اشکالات نسخه 1 صحیح بوده اند و باید اصلاح می‌شدند.
بنده هم نگفتم که انگولار را یاد نگیرید، گفتم "حداقل یادگیری آن را تا انتشار نسخه‌ی 2 آن به تعویق بیندازید."
نکته بحث "الان" هست نه گذشته. الان که قرار هست نسخه‌ی 2 منتشر شه و تغییرات زیادی داره چه دلیلی داره پروژه جدیدی را با آن شروع کنیم و بعد شروع به تغییر کدهایمان کنیم.
بله راهنمایی برای مهاجرت از نسخه‌ی 1 به نسخه 2 وجود دارد: AngularJS 1 to Angular 2 Upgrade Strategy
Angular 1 and Angular 2 integration: the path to seamless upgrade

اگر با این روش با آپگرید پروژتون که ممکن است چند هزار خط داشته باشد مشکلی ندارید، معطل نکنید همین امروز پروژه جدیدتون را با انگولار آغاز کنید.
باز هم نمی‌گویم انگولار بد است، همین الان میشود مقاله ای برای مزیت‌های نوشت. فقط قصدم این بود که به هنگام استفاده از انگولار و توسعه spa این موارد را هم در نظر داشته باشید.

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۴/۰۶/۲۸ ۱۲:۲

سلام؛
تجربه‌ی کاری با فریم ورک‌های دیگر را ندارم. ولی در مورد ReactJ.NET جمله " REACT ♥ C# AND ASP.NET MVC" می‌تواند برای توسعه دهندگان ASP.NET MVC جالب باشد. باید این‌ها را در یک پروژه واقعی استفاده کرد تا متوجه مزایا و معایبشان شد.

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۴/۰۶/۲۸ ۱۲:۱۶

اگر Angular 1 and Angular 2 integration قسمت why upgrade را مطالعه کنید می‌بینید که بیشتر مشکلات برطرف شده اند.
حتی Future work مثل Server-side rendering خیلی جذاب و مفید می‌تونه باشه.

نویسنده: محسن کریمی
تاریخ: ۱۳۹۴/۰۶/۲۸ ۲۱:۴۵

سلام
بنده هم بیش از 2 سال است که با Angular کار می‌کنم مشکلاتی که از نظر من بیشتر مهمه و باید اول پروژه قبل از انتخاب، در نظر گرفته شود شامل موارد زیر هستش
وضعیت Angular (و فکر می‌کنم بقیه SPA ها) در کامپوننت‌ها خیلی بد و شما در خیلی موارد انتخاب ندارید و از کامپوننت‌های pure javascript و جی کوئری هم بیشتر مواقع نمیشه استفاده کرد و تنها یک راه اینه که خودتون کامپوننت‌های مورد نیاز خودتونو بنویسید که اینم با توجه به Resource و زمان پروژه خیلی جاها امکان پذیر نیست. کامپوننت هایی که با Angular پیاده شده اند اکثرا یا از لحاظ Performance واقعا افتضاح هستند مثل Kendo یا خیلی ساده هستند که جوابگو نیستن و بازم مجبورید که به سمت پیاده سازی کامپوننت برید.

بحث Performance هم در صورتی که با یک پروژه سنگین روبرو باشید مطمئنا اذیت خواهد کرد و خیلی جاها مجبورید به سمت Pure javascript حرکت کنید و همیشه باید حواستون به watch ها، Bindها و ... باشه(dirty watch) تو بحث Performance هم اگر بخوایید از Patternهای خاصی استفاده کنید (Flux) تا حدودی نسبت به pure javascript دستتون بسته است و باید خیلی چیزها را خودتون طراحی و پیاده سازی کنید.

مشکل سوم هم لود اولیه صفحتونه که می‌تونه مشکل ساز بشه

البته من فکر می‌کنم این موارد در همه SPAها تقریبا موجود هستش و باید سیر تکاملی خودشونو طی کنن

نویسنده: عباسی بهزاد
تاریخ: ۱۳۹۴/۰۶/۲۹ ۱۶:۴

سلام. به jsblocks.com یک نگاه بندازید تا لااقل خیالتان نسبت به performance آسوده باشه :)

نویسنده: علی پناهی

تاریخ: ۱۳۹۴/۰۷/۱۱ ۱۷:۲۳

مسلما واکشی اطلاعات از صفحات وب انکارناپذیر است ولی وقتی که کل وب سایت با WebApi پیاده سازی می‌شود و از فریم ورک‌های تحت کلاینت استفاده شود، افراد دیگر به اطلاعات با ساختار دست پیدا می‌کنند و خیلی راحت‌تر از وب سایت‌های دیگر امکان واکشی اطلاعات وجود دارد.
راه حلی برای این موضوع وجود دارد؟

نویسنده: محسن خان
تاریخ: ۱۳۹۴/۰۷/۱۱ ۱۹:۴۰

«خیلی راحت‌تر از وب سایت‌های دیگر امکان واکشی اطلاعات وجود دارد» : خیر. مگر آنکه CORS - Cross Origin Resource Sharing را فعال کرده باشید.

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۴/۰۷/۱۱ ۱۹:۵۴

اگر منظورتون نمایش اطلاعاتی که به فرمت JSON دریافت می‌شوند، در قالب‌های HTML هست می‌توانید از کتاب خانه هایی به مانند Handlebars استفاده کنید.