عنوان: Best Practice هایی برای طراحی RESTful API - قسمت اول
نویسنده: محسن درپرستی
تاریخ: ۱۳۹۲/۱۰/۱۶ ۱۷:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.NET Web API, RESTful API, Best Practice

با آمدن Asp.Net Web API کار ساختن Web API ها برای برنامه نویس‌ها به خصوص دسته ای که با ساخت API و وب سرویس آشنا نبودند خیلی ساده‌تر شد . اگر با Asp.Net MVC آشنا باشید خیلی سریع می‌توانید اولین Web Service خودتان را بسازید .

در صفحه مربوط به Asp.Net Web API آمده است که این فریمورک بستر مناسبی برای ساخت و توسعه برنامه های RESTful است . اما تنها ساختن کنترلر و اکشن و برگشت دادن داده‌ها به سمت کلاینت ، به خودی خود برنامه شما رو تبدیل به یک RESTful API نمی‌کند .

مثل تمام مفاهیم و ابزارها ، طراحی و ساختن RESTful API هم دارای اصول و Best Practice هایی است که رعایت آنها به خصوص در این زمینه از اهمیت زیادی برخوردار است . همانطور که از تعریف API برمی آید شما در حال طراحی رابطی هستید تا به توسعه دهندگان دیگر امکان دهید از داده‌ها و یا خدمات شما در برنامه‌ها و سرویس هایشان استفاده کنند . مانند APIهای توئیتر و نقشه گوگل که برنامه‌های زیادی بر مبنای آنها ساخته شده اند . در واقع توسعه دهندگان مشتریان API شما هستند .

بهره وری توسعه دهنده مهمترین اصل

اینطور می‌توان نتیجه گرفت که اولین و مهمترین اصل در طراحی API باید رضایت و موفقیت توسعه دهنده در درک و یادگیری سریع API شما ،نه تنها با کمترین زحمت بلکه همراه با حس نشاط ، باشد. (تجربه کاربری در اینجا هم می‌تواند صدق کند). سعی کنید در زمان انتخاب از بین روش‌های طراحی موجود ، از دیدگاه توسعه دهنده به مسئله نگاه کنید . خود را به جای او قرار دهید و تصور کنید که می‌خواهید با استفاده از API موجود یک رابط کاربری طراحی کنید یا یک اپلیکشن برای موبایل بنویسید و اصل را این نکته قرار دهید که بهره وری برنامه نویس را حداکثر کنید. ممکن است گاهی بین طرحی که بر اساس این اصل برای API خود در نظر داریم و یکی از اصول یا استانداردها تعارض بوجود بیاید . در این موارد بعد از اینکه مطمئن شدیم این اختلاف ناشی از طراحی و درک اشتباه خودمان نیست (که اکثرا هست) ارجحیت را باید به طراحی بدهیم .

تهیه مستندات API

اگر برای پروژه وب سایتتان هیچ نوشته ای یا توضیحی ندارید ، جالب نیست اما خودتان ساختار برنامه خود را می‌شناسید و کار را پیش می‌برید. اما توسعه دهنده ای که از API شما می‌خواهد استفاده کند و به احتمال زیاد شما را نمی‌شناسد ، عضو تیم شما هم نیست ، هیچ ایده ای درباره ساختار آن ، روش نامگذاری توابع و منابع، ساختار Urlها ، چگونگی و گام‌های پروسه درخواست تا دریافت پاسخ ندارد ،و به مستندات شما وابسته است و تمام اینها باید در مستندات شما باشد. بیشتر توسعه دهنده‌ها قبل از تست کردن API شما سری به مستندات می‌زنند ، دنبال نمونه کد آموزشی می‌گردند و در اینترنت درباره آن جستجو می‌کنند .
ازینرو مستندات (کارامد) یک ضرورت است :
1- در مستندات باید هم درباره کلیت و هم در مورد تک تک توابع (پارامترهای معتبر ، ساختار پاسخ‌ها و ...) توضیحات وجود داشته باشد.
2- باید شامل مثالهایی از سیکل کامل درخواست‌ها / پاسخ‌ها باشند .
3- تغییرات اعمال شده نسبت به نسخه‌های قبلی باید در مستندات بیان شوند .
4- (در وب) یافتن و جستجو کردن در مستنداتی که به صورت فایل Pdf هستند یا برای دسترسی نیاز به Login داشته باشند سخت و آزاردهنده هستند.
5- کسی را داشته باشید تا با و بدون مستندات با API شما کار کند و از این روش برای تکمیل و اصلاح مستندات استفاده کنید.

رعایت نسخه بندی و حفظ نسخه‌های قبلی به صورت فعال برای مدت معین

یک API تقریبا هیچوقت کاملا پایدار نمی‌شود و اعمال تغییرات برای بهبود آن اجتناب ناپذیر هستند . مسئله مهم این است که چطور این تغییرات مدیریت شوند . مستند کردن تغییرات ، اعلام به موقع آنها و دادن یک بازه زمانی کافی برای ارتقا یافتن برنامه

هایی که از نسخه‌های قدیمی‌تر استفاده می‌کنند نکات مهمی هستند . همیشه در کنار نسخه بروز و اصلی یک یا دو نسخه (بسته به API و کلاینت‌های آن) قدیمی‌تر را برای زمان مشخصی در حالت سرویس دهی داشته باشید .

داشتن یک روش مناسب برای اعلام تغییرات و ارائه مستندات و البته دریافت بازخورد از استفاده کنندگان

تعامل با کاربران برنامه باید از کانال‌های مختلف وجود داشته باشد .از وبلاگ ، Google Groups ، Mailing List و دیگر ابزارهایی که در اینترنت وجود دارند برای انتشار مستندات ، اعلام بروزرسانی‌ها ، قرار دادن مقالات و نمونه کدهای آموزشی ، پرسش و پاسخ با کاربران استفاده کنید .

مدیریت خطاها به شکل صحیح که به توسعه دهنده در آزمودن برنامه اش کمک کند.

از منظر برنامه نویسی که از API شما استفاده می‌کند هرآنچه در آنسوی API اتفاق می‌افتد یک جعبه سیاه است . به همین جهت خطاهای API شما ابزار کلیدی برای او هستند که خطایابی و اصلاح برنامه در حال توسعه اش را ممکن می‌کنند . علاوه بر این ، زمانی که برنامه نوشته شده با API شما مورد استفاده کاربر نهایی قرار گرفت ، خطاهای به دقت طراحی شده API شما کمک بزرگی برای توسعه دهنده در عیب یابی هستند .

-1 از Status Code های HTTP استفاده کنید و سعی کنید تا حد ممکن آنها را نزدیک به مفهوم استانداردشان بکار ببرید .

-2 خطا و علت آن را به زبان روشن توضیح دهید و در توضیح خساست به خرج ندهید .

-3 در صورت امکان لینکی به یک صفحه وب که حاوی توضیحات بیشتری است را در خطا بگنجانید .

رعایت ثبات و یکدستی در تمام بخش‌های طراحی که توانایی پیش بینی توسعه دهنده را در استفاده از API افزایش می‌دهد .

داشتن مستندات لازم است اما این بدین معنی نیست که خود API نباید خوانا و قابل پیش بینی باشد . از هر روش و تکنیکی که استفاده می‌کنید آن را در تمام پروژه حفظ کنید . نامگذاری توابع/منابع ، ساختار پاسخ‌ها ، Urlها ، نقش و عملیاتی که HTTP methodها در API شما انجام می‌دهند باید ثبات داشته باشند . از این طریق توسعه دهنده لازم نیست برای هر بخشی از API شما به سراغ فایل‌ها راهنما برود . و به سرعت کار خود را به پیش می‌برد .

انعطاف پذیر بودن API

API توسط کلاینت‌های مختلفی و برای افراد مختلفی مورد استفاده قرار می‌گیرد که لزوما همه‌ی آنها ساختار یکسانی ندارند و API شما باید تا جای ممکن بتواند همه آنها را پوشش دهد . محدود بودن فرمت پاسخ ، ثابت بودن فیلدهای ارسالی به کلاینت ، ندادن امکان صفحه بندی ، مرتب سازی و جستجو در داده‌ها به کلاینت ، داشتن تنها یک نوع احراز هویت ، وابسته بودن به کوکی و ... از مشخصات یک API منجمد و انعطاف ناپذیر هستند .

اینها اصولی کلی بودند که بسیاری از آنها مختص طراحی API نیستند و در تمام حوزه‌ها قابل استفاده بوده ، جز الزامات هستند . در قسمت‌های بعدی نکات اختصاصی‌تری را بررسی خواهیم کرد .

نظرات خوانندگان

نویسنده: وحید
تاریخ: ۱۳۹۲/۱۰/۱۷ ۱:۱۴

با سلام یعنی شما میگویید برای web api میبابیست یک لایه جدا تعریف نمود و cors را در آن لحاظ نمود؟

نویسنده: محسن درپرستی
تاریخ: ۱۳۹۲/۱۰/۱۷ ۹:۱۲

لایه جدا از چه چیزی ؟ اگر منظورتان در Asp.Net باشد ، پروژه هایی که با استفاده از Asp.Net Web API ساخته می‌شوند خود یک سیستم مستقل هستند .

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۹ ۲۲:۳۹

سایت silverreader که از asp.net web api استفاده می‌کنه، نگارش اول کارش با این آدرس شروع میشه

https://silverreader.com/api/v1

نویسنده: شهروز جعفری
تاریخ: ۱۳۹۲/۱۲/۰۶ ۷:۳۶

بحث cors یک چیز است و بحث Best Practice هایی برای طراحی RESTful API یک چیز .در کل زمانی که ما می‌خواهیم یک سرویس ارائه دهیم و باقی از آن استفاده کنند داستانش با یک متد که فقط خودمان از آن استفاده می‌کنیم فرق می‌کند. باید به خیلی چیزها حواسمان باشد.

نویسنده: شهروز جعفری
تاریخ: ۱۳۹۲/۱۲/۰۶ ۷:۳۹

من یک سری مطلب درباره نسخه بندی در API پیدا کردم باهاتون به اشتراک میذارم شاید مفید باشه:

http://www.lexicalscope.com/blog/2012/03/12/how-are-rest-apis-versioned/

http://stackoverflow.com/questions/10742594/versioning-rest-api