

در هنگام گفتگو با افراد مختلفی که در پروژه‌های توسعه نرم افزار، نقش‌های مختلفی را دارا می‌باشند، یکی از جالب‌ترین و اساسی‌ترین بحث‌ها تفاوت بین Desktop App و Web App می‌باشد، و این که پروژه بر اساس کدام مدل باید نوشته شود.

در اینترنت و در منابع معتبر، تفسیرهای متفاوتی از این دو وجود دارد، که گاه دقیقا با نظر من یکی بوده و گاه تا 180 درجه بر عکس هستند، آنچه که در ادامه می‌خوانید می‌تواند لزوما نظر شما نباشد.

گروهی از افراد بر این باور هستند که اجرای برنامه در محیط مرورگر (ظاهر مرورگر و نه Sandbox آن)، یکی از ملاک‌های ما بین Desktop App و Web App است، گروهی دیگر نیز اجرا شدن برنامه بر روی بستر اینترنت و یا شبکه‌ی محلی را جزو ملاک‌ها می‌دانند، و گروهی دیگر نیز زبان برنامه نویسی برنامه را ملاک می‌دانند، برای مثال اگر با HTML/JS باشد Web App است، اگر نه Desktop App است.

اما آنچه که در عمل می‌تواند تفاوت بین Desktop App را با یک Web App مشخص کند، رفتار و عملکرد خود آن برنامه است، نه بستر اجرای آن و این که آن رفتار منتج شده از چه کدی و چه زبان برنامه نویسی ای است.

اگر کمی دقیق به مطلب نگاه کنیم، می‌بینیم این که یک برنامه در چارچوب ظاهری یک مرورگر (نه Sandbox آن) اجرا شود، اصلا مقوله ای اهمیت دار نیست، کما این که برای مثال Silverlight اجازه می‌دهد، برنامه هم در داخل مرورگر و هم در بیرون از آن اجرا شود، و این کار با یک کلیک امکانپذیر است، آیا با همین یک کلیک برنامه از Web App به Desktop App تبدیل می‌شود یا بالعکس ؟

آیا یک برنامه مبتنی بر دلفی که تا همین یک ساعت پیش بر روی شبکه محلی در حال اجرا بوده، با انتقال پیدا کردن آن بر روی شبکه‌ی اینترنت، تبدیل به یک Web App می‌شود؟

آیا اگر ما با HTML/JS یک برنامه Native برای ویندوز فون بنویسیم که تک کاربره آفلاین باشد و اصلا سروری هم نداشته باشد، آیا Web App نوشته ایم ؟ **اصلی‌ترین** تفاوت مابین Desktop App و Web App که به تفاوت در عملکرد آنها و مزایا و معایب آنها منجر می‌شود، این است که انجام کارهایی که اپراتور با آنها در سمت کلاینت و سیستم مشتری سر و کار دارد، در کجا صورت می‌پذیرد؟

برای مثال در نظر بگیرید که یک دیتاگرید داریم که دارای Paging است، و ما از Page اول به Page بعدی می‌رویم، در یک Desktop App تنها اطلاعات از سرور گرفته می‌شود، و ترسیم خطوط و ستون‌ها و ردیف‌ها و ظاهر نمایشی دیتاگرید بر عهده کلاینت است، برای مثال اگر ستون قیمت داشته باشیم، و بخواهیم برای ردیف‌هایی که قیمت آنها زیر 10000 ریال است، قیمت به شکل سبز رنگ نمایش داده شود و برای بقیه ردیف‌ها به رنگ قرمز باشد، پردازش این مسئله و این if به عهده کلاینت است، اما در یک Web App، علاوه بر اطلاعات، تعداد زیادی tagهای مختلف، مانند table - tr - td و ... نیز به همراه اطلاعات آورده می‌شوند، که وظیفه نمایش ظاهری اطلاعات را بر عهده دارند، و آن if مثال ما یعنی رنگ سبز و قرمز در سمت سرور مدیریت شده است، و کلاینت در اینجا نمایش دهنده‌ی آن چیزی است که به صورت آماده از سرور آورده شده است.

در برنامه‌های Desktop آنچه که در سمت سرور وجود دارد، برای مثال یک WCF Service یا ASP.NET Web API است که فقط به رد و بدل کردن اطلاعات می‌پردازد، اما در Web App در سمت سرور ASP.NET MVC، ASP.NET Web Forms و PHP وجود دارند که علاوه بر اطلاعات برای کلاینت شما ظاهر صفحات را نیز آماده می‌کنند، و ظاهر اصلی صفحات از سمت سرور به سیستم مشتری ارسال می‌شوند، اگر چه که ممکن است در سمت کلاینت تغییراتی را داشته باشند.

به هر میزان رفتار برنامه ما شبیه به حالت اول باشد، برنامه ما Desktop App بوده و به هر میزان برنامه ما به حالت دوم نزدیک‌تر باشد، برنامه ما Web App است.

مزیت اصلی Web App‌ها در عدم انداختن بار زیاد بر روی دوش کلاینت‌های بعضا نحیف بوده، و عملا کلاینت به علت این که کار خاصی را انجام نمی‌دهد، پیش نیاز نرم افزاری و یا سخت افزاری خاصی ندارد، و این مورد Web App‌ها را به یک گزینه ایده آل برای وب سایت‌هایی تبدیل کرده است که با عموم مردم در ارتباطند، زیرا که امکان ارائه آسان برنامه وجود دارد و تقریبا همه می‌توانند از آن استفاده کنند.

با توجه به شناخت عموم از برنامه‌های Web App به توضیح بیشتر برنامه‌های Desktop App می‌پردازم.

مزیت اصلی Desktop App‌ها در سرعت عمل بالاتر (به علت این که فقط دیتا را رد و بدل می‌کند)، توانایی بیشتر در استفاده از منابع سیستمی مانند سرویس نوشتن، و امکانات محلی مانند ارائه Notification و ... است، و در کنار آن برای مثال یک Desktop App

می‌تواند به نحوی طراحی شود که به صورت Offline نیز کار کند. این مزیت‌ها باعث می‌شود که Desktop App ها گزینه ای مناسب برای برنامه‌های سازمانی باشند. ضعفی که از گذشته در Desktop App ها وجود داشته است، که البته به معماری Desktop App بر نمی‌گردد، بلکه متأثر از امکانات است، عدم Cross Platform بودن آنها بوده است، تا آنجا که Desktop App در نظر خیلی از افراد همان نوشتن برنامه برای سیستم عامل ویندوز است. با توجه به رویکرد جدی ای که در طول دو سال اخیر برای نوشتن برنامه Desktop App به شکل Cross Platform رخ داده است، خوشبختانه این مشکل حل شده است و اکنون لااقل دو راهکار جدی برای نوشتن یک برنامه Cross Platform با ویژگی‌های Desktop وجود دارد، که یکی از آنها راه حل‌های مبتنی بر HTML/JS است و دیگری راه حل‌های مبتنی بر C#/XAML در راه حل‌های مبتنی بر HTML/JS در صورتی که شما برنامه را به شکل Web App طراحی نکرده باشید، و برای مثال در آن از ASP.NET Web Forms و ASP.NET MVC، PHP و ... استفاده نکرده باشید، می‌توانید یک خروجی کاملاً Native با تمامی ویژگی‌های Desktop App برای انواع پلتفرم‌ها بگیرید. استفاده از فریم ورک هایی که با طراحی Desktop App سازگار هستند، مانند Angular JS، Kendo UI، Ext JS، و ... و استفاده از مدل طراحی Single Page Application می‌تواند سیستم کدنویسی ای ساده را فراهم آورد، که در آن شما با یک بار نوشتن برنامه می‌توانید خروجی اکثر پلتفرم‌های مطرح را داشته باشید، اعم از ویندوز فون، اندروید، iOS و ویندوز امروزه حتی مرورگرها با فراهم آوردن امکاناتی مانند Client side databases و Manifest based deployment اجازه نوشتن برنامه Desktop با HTML/JS را که حتی می‌تواند Offline کار کند را به شما ارائه می‌کنند. در کنار این راهکار، استفاده از C#/XAML برای نوشتن برنامه برای اکثر پلتفرم‌های مطرح بازار اعم از اندروید، iOS و Windows Phone و ویندوز، نیز به عنوان راهکاری دیگر قابلیت استفاده را دارا است. حرکت پر شتاب و پر انرژی جهانی برای توسعه Cross Platform Desktop Development، خوشبختانه توانسته است تا حد زیادی امتیاز نوشتن برنامه‌های Desktop را در سیستم‌های Enterprise بالا ببرد.

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۲۰ ۱۳:۵

مطلبی هم من چند سال قبل در این مورد نوشته بودم
« چرا در سازمان‌ها برنامه‌های وب جایگزین برنامه‌های دسکتاپ شده‌اند (یا می‌شوند)؟ »

نویسنده: علیرضا
تاریخ: ۱۳۹۳/۰۴/۲۱ ۱۰:۱۱

به نظر من این بحث به همین سادگی نیست و انتخاب پلتفرم اجرای پروژه به پارامترها و ویژگی‌های زیادی مرتبط هست. بطور مثال سرعت توسعه برنامه‌های ویندوز حداقل در قسمت طراحی رابط کاربری سریعتر و ساده‌تر از وب هست. و یا در مثال دیگر رفتار غیر یکسان مرورگرها مشکلاتی را در طراحی نرم افزارهای بزرگ ایجاد می‌کند و مشکل ساز میشه من بعد از سال‌ها طراحی سیستم‌های سازمانی روش استفاده ترکیبی از پلتفرم‌های مختلف را انتخاب کردم بطور مثال قسمت مدیریت یک سیستم را بصورت ویندوزی و قسمت رابط کاربری را با وب ... طراحی کردم. متأسفانه طراحی اولیه زبان HTML با هدف نمایش اطلاعات بوده و بهبودهای اخیر از جمله وب 2 پاسخی منطقی به نیاز به توسعه نرم افزارهای Cross Platform بوده ولی هنوز هم با پیچیدگی‌های زیادی روبروست. به نظر قابلیت‌های نرم افزار تحت وب بیش از واقعیت بزرگ نمایی شده و هنوز هم در برخی راه کارها استفاده از نرم افزارهای تحت ویندوز گزینه مناسب‌تری خواهد بود اما این به معنی چشم پوشی بر مزایای منحصر به فرد وب نخواهد بود و هنوز انتخاب پلتفرم بستگی زیادی به نیازمندی‌ها و امکانات پروژه خواهد داشت.

نویسنده: رحیم
تاریخ: ۱۳۹۳/۰۴/۲۱ ۲۲:۴۴

سلام لطفا در مورد این جمله بیشتر توضیح دهید
در کنار این راهکار، استفاده از C#/XAML برای نوشتن برنامه برای اکثر پلتفرم‌های مطرح بازار اعم از اندروید، iOS و Windows Phone و ویندوز، نیز به عنوان راهکاری دیگر قابلیت استفاده را دارا است.
آیا منظور شما استفاده از نرم افزارهای شرکت ثالث نظیر xamarin و می‌باشد یا تکنولوژی جدیدی از سمت مایکروسافت ارائه شده که این امکان رو میسر می‌کند

نویسنده: یاسر مرادی
تاریخ: ۱۳۹۳/۰۴/۲۲ ۸:۴۳

بله، منظور روش‌های ارائه شده مبتنی بر پلتفرم Xamarin است، البته در نظر بگیرید این کار بدون کمک‌های فنی ارائه شده توسط مایکروسافت و همچنین رفع مشکل لایسنس Portable Class Library ها و ... از سوی مایکروسافت امکانپذیر نبود.
مایکروسافت بنظر قصد پشتیبانی فنی و مالی و در نهایت خرید Xamarin رو داره، و بنظر نمی‌آد که بخواد این مسیر رو از نو پیش بره، چون واقعا کار زیادی می‌بره