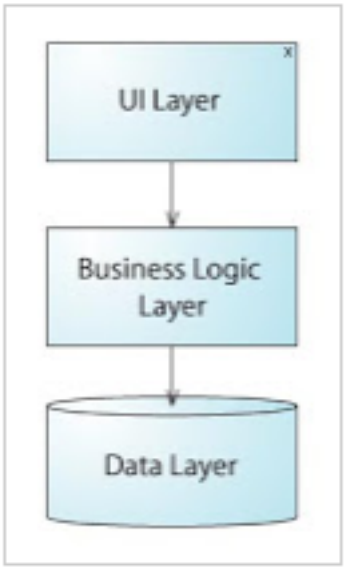
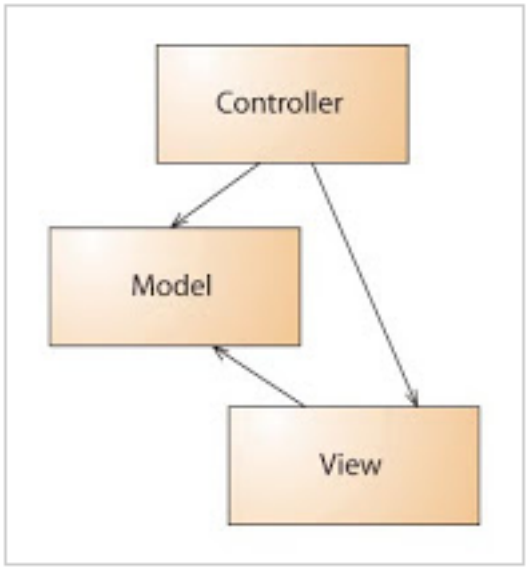
عنوان: MVC vs 3-Tier Pattern
نویسنده: مهرداد اشکانی
تاریخ: ۱۳۹۲/۰۳/۲۶ ۱۴:۰
آدرس: www.dotnettips.info
گروه‌ها: Design patterns, MVC, Architecture, معماری سه لایه, MVC معماری

من تا به حال برنامه نویس‌های زیادی را دیده‌ام که می‌پرسند «چه تفاوتی بین الگوهای معماری MVC و Three-Tier وجود دارد؟» قصد من روشن کردن این سردرگمی، بوسیله مقایسه هردو، با کنار هم قرار دادن آنها می‌باشد. حداقل در این بخش ، من اعتقاد دارم، منبع بیشتر این سردرگمی‌ها در این است که هر دوی آنها، دارای سه لایه متمایز و گره، در دیاگرام مربوطه‌اشان هستند.



Three-Tier



MVC

اگر شما به دقت به دیاگرام آنها نگاه کنید، پیوستگی را خواهید دید. بین گره‌ها و راه اندازی آنها، کمی تفاوت است.

معماری سه لایه

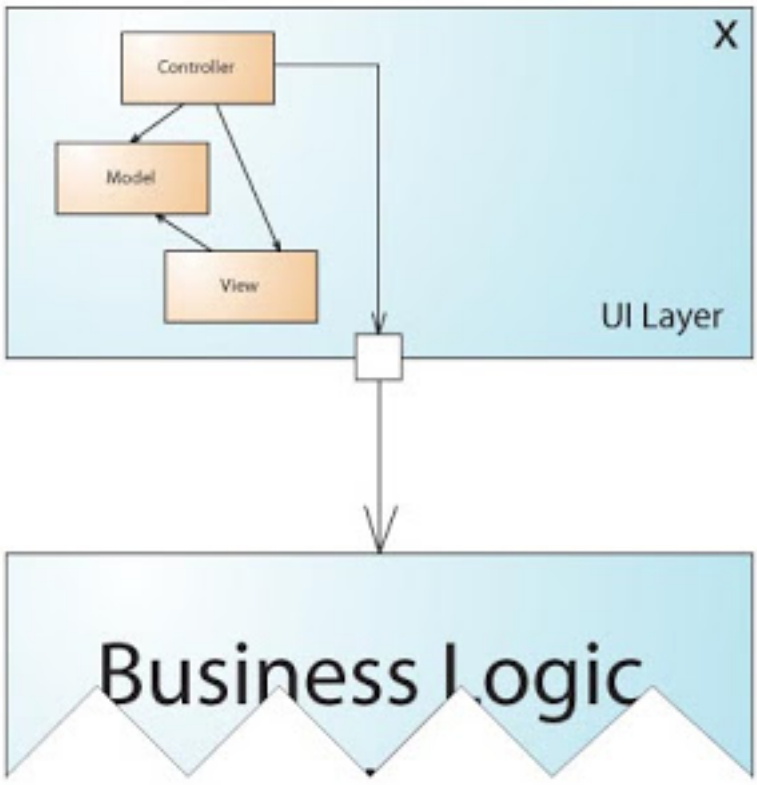
سیستم‌های سه لایه، واقعاً لایه‌ها را می‌سازند: لایه UI به لایه Business logic دسترسی دارد و لایه Business logic به لایه Data دسترسی دارد. اما لایه UI دسترسی مستقیمی به لایه Data ندارد و باید از طریق لایه Business logic و روابط آنها عمل کند. بنابراین می‌توانید فکر کنید که هر لایه، بعنوان یک جزء، آزاد است؛ همراه با قوانین محکم طراحی دسترسی بین لایه ها.

MVC

در مقابل، این Pattern ، لایه‌های سیستم را نگهداری نمی‌کند. کنترلر به مدل و View (برای انتخاب یا ارسال مقادیر) دسترسی دارد. View نیز دسترسی دارد به مدل . دقیقاً چطور کار می‌کند؟ کنترلر در نهایت نقطه تصمیم گیری منطقی است. چه نوع منطقی؟ نوعاً، کنترلر، ساخت و تغییر مدل را در اکشن‌های مربوطه، کنترل خواهد کرد . کنترلر سپس تصمیم گیری می‌کند که برای منطق داخلیش، کدام View مناسب است. در آن نقطه، کنترلر مدل را به View ارسال می‌کند. من در اینجا چون هدف بحث مورد دیگه‌ای می‌باشد، مختصر توضیح دادم.

چه موقع و چه طراحی را انتخاب کنم؟

اول از همه، هر دو طراحی قطعاً و متقابلاً منحصر بفرد نیستند. در واقع طبق تجربه‌ی من، هر دو آنها کاملاً هماهنگ هستند. اغلب ما از معماری چند لایه استفاده می‌کنیم مانند معماری سه لایه، برای یک ساختار معماری کلی. سپس من در داخل لایه UI ، از MVC استفاده می‌کنم، که در زیر دیاگرام آن را آورده ام.



نظرات خوانندگان

نویسنده: محسن اسماعیل پور
تاریخ: ۱۳۹۲/۰۳/۲۶ ۲۲:۳۳

3-Layer در واقع Architecture Style هست اما MVC یک Design Pattern هست پس مقایسه مستقیم نمیدونم کاری دست باشد یا نه اما میتونیم به این شکل نتیجه گیری کنیم:
Data Access: شامل کلاسهای ADO.NET یا EF برای کار با دیتابیس.
Business Logic: یا همان Domain logic که میتوان Model رو به عنوان Business entity در این لایه بکار برد.
UI Layer: بکارگیری Controller و View در این لایه

نویسنده: یزدان
تاریخ: ۱۳۹۲/۰۳/۲۷ ۱:۳۳

در برنامه نویسی 3لایه کار Business Logic به طور واضح و شفاف چی هست و چه کارهایی در این لایه لحاظ میشه ؟

نویسنده: fss
تاریخ: ۱۳۹۲/۰۳/۲۷ ۸:۴۱

منم یه مدت دچار این ابهام بودم. ولی الان اینطور نتیجه می‌گیرم:

mvc کلا در لایه UI قرار داره. یعنی اگر شما لایه BL و DAL رو داشته باشید، حالا میتونید لایه UI رو با یکی از روش ها، مثلا سیلورلایت، asp.net mvc یا asp.net web form پیاده کنید.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۳/۲۷ ۸:۴۹

همون لایه UI هم نیاز به جداسازی کدهای نمایشی از کدهای مدیریت کننده آن برای بالابردن امکان آزمایش کردن و یا حتی استفاده مجدد قسمت‌های مختلف اون داره. در این حالت شما راحت نمی‌تونید MVC و Web forms رو در یک سطح قرار بدی (که اگر اینطور بود اصلا نیازی به MVC نبود؛ نیازی به MVVM برای سیلورلایت یا WPF نبود و یا نیازی به MVP برای WinForms یا Web forms نبود).

نویسنده: fss
تاریخ: ۱۳۹۲/۰۳/۲۷ ۹:۱۳

دوست عزیز من متوجه منظور شما نشدم. حرف من اینه که MVC، MVVM، MVP و .. در سطح UI پیاده میشن.

نویسنده: مهرداد اشکانی
تاریخ: ۱۳۹۲/۰۳/۲۷ ۹:۱۸

لایه business Logic در واقع لایه پیاده سازی Business پروژه شما می‌باشد با یک مثال عرض می‌کنم فرض کنید در لایه UI شما لازم دارید یک گزارش از لیست مشتریانی که بالاترین خرید را در 6 ماه گذشته داشته اند و لیست تراکنش مالی آنها را بدست آورید.برای این مورد شما توسط کلاسهای و متدهای لازم ، در لایه Business Logic این عملیات را پیاده سازی می‌کنید.

نویسنده: مهرداد اشکانی
تاریخ: ۱۳۹۲/۰۳/۲۷ ۹:۳۸

این طور نیست دوست عزیز شما می‌تونید حتی برای Model هم لایه در نظر بگیرید که براحتی توسط لایه Business و کلا لایه‌های دیگر در دسترس باشد.که این مورد الان در MVC خیلی کاربرد دارد.مواردی که من عرض کردم برای رفع ابهام بین معماری چند لایه و Pattern MVC بود.

نویسنده: داود
تاریخ: ۱۳۹۲/۰۳/۲۷ ۱۷:۱۴

به بنظر بنده هم معماری رو نمی‌شود با الگو مقایسه کرد به هر حال خود الگوی mvcها یک سری لایه داره و تا اونجایی هم که می‌دونم فرق tier با layer اینه که tierها رو از لحاظ فیزیکی هم جدا می‌کنند

نویسنده: وحید فرهمندیان
تاریخ: ۱۳۹۳/۱۲/۰۴ ۱۷:۱۶

لایه کسب و کار مغز برنامه شما میباشد. یک زمانی میخواهید معادله ریاضی حل کنید در این لایه و زمانی نیز نیاز است مقداری داده از انباره داده خود بخوانید. لذا UI درخواست محاسبه معادله یا استخراج گزارش را به کسب و کار میدهد، کسب و کار بررسی میکند تا درخواست را پاسخ دهد. اگر برای پاسخ نیاز به انباره داده بود به لایه داده میفرستد تا مطابق با آن درخواست داده‌های مناسب استخراج شده و برگشت داده شوند.
نکته ای که وجود دارد این است که لایه داده حتما نباید با یک پایگاه داده ارتباط برقرار کند، و لایه UI نیز نباید شخصا کار پردازشی یا منطقی انجام دهد و این کارها باید به لایه کسب و کار ارجاع داده شوند.

نویسنده: ریوف مدرسی
تاریخ: ۱۳۹۴/۰۴/۰۵ ۱۸:۳۲

به نظر من این درست نیست که بگیم mvc کلا در لایه UI می‌باشد، ببینید شما وقتی می‌خواهید یک application را توسعه بدید علاوه بر اهداف بیزینسی application که قصد توسعه ان را دارید یکسری اهداف تکنولوژیکی هم مد نظر دارید، حالا بعضی از این اهداف می‌تواند بسیار پایه ای باشد مانند اینکه شما این application را بصورت web app , win app یا mobile app یا ترکیبی پیاده سازی کنید، وبعضی تصمیمات هم بعد از این تصمیم گیری انجام میشوند، برای مثال شما در نظر می‌گیرید که می‌خواهید این application را به صورت یک web app پیاده سازی کنید، حال شما ممکن است به عنوان طراح نرم افزار یا یک معمار یکسری concern ریز و درشت دیگر برایتان ایجاد شود، باز هم برای مثال: این application باید SOC را در تمامی سطوح درنظر بگیرد و decoupling به یک سطح مناسب برساند، ویا اینکه هدف این که لایه business را طوری طراحی کنیم که به یک repository خاص یا یک ui framework وابسته نباشد، اما هدف من از این همه اسمان ریسمان‌ها این بود که بگم که مثلا نمی‌توان گفت که فلان تکنولوژی باید در فلان موقعیت و به فلان روش استفاده شود بلکه این شما هستید که بر حسب concernهای خود چگونه تصمیم بگیرید.