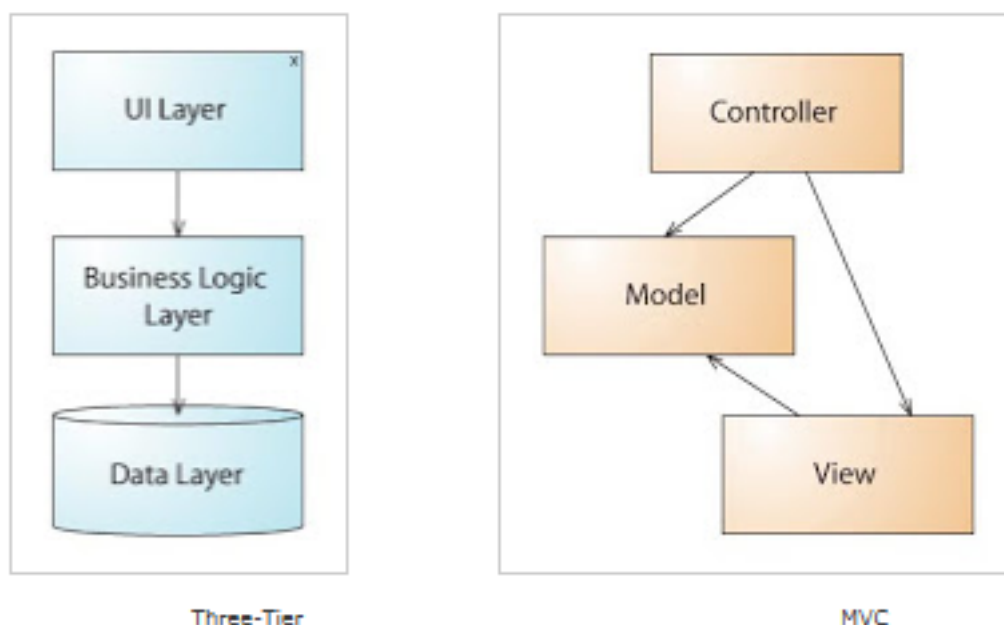


من تا به حال برنامه نویسی‌های زیادی را دیده‌ام که می‌پرسند «چه تفاوتی بین الگوهای معماری MVC و Three-Tier وجود دارد؟» قصد من روشن کردن این سردرگمی، بوسیله مقایسه هردو، با کنار هم قرار دادن آنها می‌باشد. حداقل در این بخش، من اعتقاد دارم، منبع بیشتر این سردرگمی‌ها در این است که هر دوی آنها، دارای سه لایه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



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

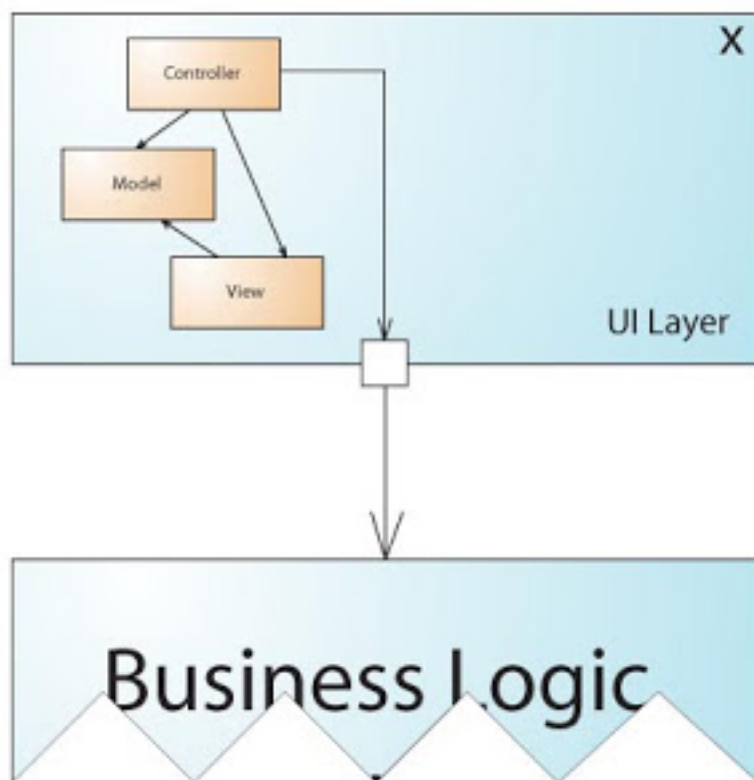
سیستم‌های سه لایه، واقعاً لایه‌ها را می‌سازند: لایه UI به لایه Business logic دسترسی دارد و لایه Business logic به لایه Data دسترسی دارد. اما لایه UI دسترسی مستقیمی به لایه Data ندارد و باید از طریق لایه Business logic و روابط آنها عمل کند. بنابراین می‌توانید فکر کنید که هر لایه، بعنوان یک جزء، آزاد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

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 تصمیم‌گیری منطقی است. چه نوع منطقی؟ نوعاً، کنترلر، ساخت و تغییر مدل را در اکشن‌های مربوطه، کنترل خواهد کرد. کنترلر سپس تصمیم‌گیری می‌کند که برای منطق داخلی، 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: ۲۲:۳۳ ۱۳۹۲/۰۳/۲۶

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 View و Controller در این لایه

نویسنده: یزدان
تاریخ: ۱:۳۳ ۱۳۹۲/۰۳/۲۷

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

نویسنده: fss
تاریخ: ۸:۴۱ ۱۳۹۲/۰۳/۲۷

منم به مدت دچار این ابهام بودم. ولی الان اینطور نتیجه می گیرم:

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: ۸:۴۹ ۱۳۹۲/۰۳/۲۷

همون لایه UI هم نیاز به جداسازی کدهای نمایشی از کدهای مدیریت کننده آن [برای بالابردن امکان آزمایش کردن و یا حتی استفاده مجدد قسمت های مختلف اون](#) داره. در این حالت شما راحت نمی تونید 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: ۹:۱۳ ۱۳۹۲/۰۳/۲۷

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

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

لایه business Logic در واقع لایه پیاده سازی Business پروژه شما می باشد با یک مثال عرض می کنم 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 می کنید.

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

این طور نیست دوست عزیز شما می تونید حتی برای Model هم لایه در نظر بگیرید که براحتی توسط لایه Business و کلا لایه های 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: ۱۷:۱۴ ۱۳۹۲/۰۳/۲۷

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