عنوان: **اهمیت code review**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۱۶ ۱۴:۱۰:۰۰
آدرس: www.dotnettips.info
گروه‌ها: Refactoring

تا جایی که دقت کردم (در بلاگ‌هایی که منتشر می‌شوند) در آنسوی آب‌ها، « code review » یک شغل محسوب می‌شود. سازمان‌ها، شرکت‌ها و امثال آن از مشاورین یا برنامه نویس‌هایی با مطالعه بیشتر دعوت می‌کنند تا از کدهای آن‌ها اشکال‌گیری کنند و بابت اینکار هم هزینه می‌کنند.
اگر علاقمند باشید قسمتی از یک پروژه سورس باز دریافت شده از همین دور و اطراف را با هم مرور کنیم:

```
//It's only for code review purpose!
protected void Button1_Click1(object sender, EventArgs e)
{
        string  strcon;
        string strUserURL;
        string strSQL;
        string strSQL1;
        strSQL = "SELECT UserLevel FROM listuser " + "WHERE Username='" + TextBox2.Text + "' " + "And
Password='" + TextBox3.Text + "';";
        strSQL1 = "SELECT Pnumber FROM listuser " + "WHERE Username='" + TextBox2.Text + "' " + "And
Password='" + TextBox3.Text + "';";
        strcon = @"Data Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\bimaran.mdf;Integrated
Security=True;User Instance=True";
        SqlConnection myConnection = new SqlConnection(strcon);

        SqlCommand myCommand = new SqlCommand(strSQL, myConnection);
        SqlCommand myCommand1 = new SqlCommand(strSQL1, myConnection);
        myConnection.Open();

        strUserURL = (string)myCommand.ExecuteScalar();
        send = (string)myCommand1.ExecuteScalar();
        myCommand.Dispose();
        myCommand1.Dispose();
        myConnection.Close();


        if (strUserURL != null)
        {
            Label1.Text = "";

            url = "?Pn=" + code(send);
            FormsAuthentication.SetAuthCookie(TextBox2.Text, true);
            Response.Redirect("Page/" + strUserURL + url);
       }
        else
            Label3.Text = "چنین کاربری با این مشخصات ثبت نشده است.";
}
```

مروری بر این کد یا «مشکلات این کد»:
- کانکشن استرینگ داخل کدها تعریف شده. یعنی اگر نیاز به تغییری در آن بود باید کدهای برنامه تغییر کنند. آن هم نه فقط در این تابع بلکه در کل برنامه.
- از پارامتر استفاده نشده. کد 100 درصد به تزریق اس کیوال آسیب پذیر است.
- نحوه‌ی dispose شیء کانکشن غلط است. هیچ ضمانتی وجود ندارد که کدهای فوق سطر به سطر اجرا شود و خیلی زیبا به سطر بستن کانکشن استرینگ برسد. فقط کافی است این میان یک استثنایی صادر شود و تمام. به عبارتی این سایت فقط با کمتر از 30 کاربر همزمان از کار می‌افته. بعد نیاید بگید من یک سرور دارم با 16 گیگ رم ولی باز کم میاره! همش برنامه کند میشه. همش سایت بالا نمیاد!
- همین تعریف کردن متغیرها در ابتدای تابع یعنی این برنامه نویس هنوز حال و هوای ANSI C را دارد!

- مهم نیست لایه بندی کنید. ولی یک لایه در این نوع پروژه‌ها الزامی است و آن هم DAL نام دارد. DAL یعنی کثافت کاری نکنید. یعنی داخل هر تابع گپه گپه بر ندارید open و close بذارید. برید یک تابع یک گوشه‌ای درست کنید که این عملیات را محصور کند.
- همین وجود Button1 و Label1 یعنی تو خود شرح مفصل بخوان از این مجمل!

نظرات خوانندگان

نویسنده: amir hosein jelodari
تاریخ: ۱۳۹۰/۱۰/۱۶ ۱۹:۵۷:۴۸

سلام ...
خیلی خیلی جالب و مفیده این سبک آموزشی!

الان تو این تابع دو تا سلکت متفاوت از دیتابیس انجام نمیشه؟ نه؟ ... یعنی دو تا عملیات متفاوت داریم . خوب حالا بهتر نیست هر کدوم تو یه متود wrap بشه؟!

نویسنده: aliaghdam
تاریخ: ۱۳۹۰/۱۰/۱۶ ۲۲:۴۳:۲۷

اندر همین باب!!!
http://www.aliaghdam.ir/2011/06/blog-post_1694.html

نویسنده: shahin kiassat
تاریخ: ۱۳۹۰/۱۰/۱۶ ۲۲:۵۶:۵۹

سلام.
به نظر من هر دو Select باید در یک تابع باشه.چون شرط هر دو select یکسان هست و برنامه نویس این کد از نتیجه ی هر دو query در محدوده ی همین تابع استفاده کرده.

نویسنده: Meysam Hooshmand
تاریخ: ۱۳۹۰/۱۰/۱۶ ۲۳:۳۲:۵۴

ان الله مع الصابرین
یک نکته، اگر دست به آچار نباشیم و بزار نشناسیم، گاهی اوقات این تر و تمیز کد نویسی ،خودش میشه مایه دردسر، خصوصا برای همونایی که تو حال و هوای قدیما هستند. مثلا، داشتن لایه ای برای wrap کردن، sp‌ها، بدون جنریتور،!!!!
کثافت کاری، عجب کلمه بدیه
!

نویسنده: Javad Darvish Amiry
تاریخ: ۱۳۹۰/۱۰/۱۷ ۰۰:۰۴:۲۶

- همین وجود Button1 و Label1 یعنی تو خود شرح مفصل بخوان از این مجمل!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۱۷ ۰۰:۰۴:۳۳

همون، مگر اینکه این «جنریتور»ها با چهارتا علامت تعجب به داد برسه و گرنه کمی صحبت کردن در مورد زیرساخت این‌ها کمی باعث سرگیجه میشه؛ خصوصا تفکر در مورد آن‌ها.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۱۷ ۰۰:۲۸:۴۱

هر دو کوئری رو میشه تبدیل به یک کوئری کرد : SELECT Pnumber, UserLevel FROM listuser where blabla و اون وقت بهتر خواهد بود که از DataReader استفاده شود. البته نه در این متد.
و کسانی هم که نمی‌خواهند از ORM استفاده کنند، Wrappers خیلی خوبی بعد از دات نت 4 برای ADO.NET اومده. مطلب در موردش

در سایت هست تحت عنوان Micro ORMs

نویسنده: A. Karimi
تاریخ: ۱۳۹۰/۱۰/۱۷ ۰۰:۳۰:۲۴

این بنده خدا سعی کرده نام گذاری مجارستانی رو رعایت کنه ولی ...! دوتا امتیاز منفی دیگه، یکی برای اینکه نامگذاری به این سبک توی سی شارپ منسوخه و دیگری اینکه همون نام گذاری رو هم برای متغییر اول رعایت نکرده.

نویسنده: Parham.D
تاریخ: ۱۳۹۰/۱۰/۱۷ ۰۷:۴۸:۳۴

سلام. چطور میشه روش کد نویسی صحیح را یاد گرفت؟ در اثر تجربه به دست میاد، یا برای این کار مرجع و آموزشی وجود داره؟ کدنویسی بلدم، خوب نویسی نه؟! چطور این مشکل را حل کنم؟

نویسنده: zahra abdolahi
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۰:۳۸:۱۷

من چند وقتیه که مشترک بلاگ شما شدم. پست هاتون بسیار مفید و کاربردیه. خواستم ازتون تشکر کرده باشم.
در مورد این پست هم موافقم باهاتون. ولی من تو شرکت هایی که کار کردم معمولا یه قوانینی در این زمینه وجود داره که موقع شروع به کار برنامه نویس بهش می دن. هرچند شرکت تا شرکت فرق می کنه و گاهی به طور کامل رعایت نمیشه ولی حداقل موقعی که از اواسط یک پروژه واردش میشیم و قراره در ادامه کدهای دیگران کد بنویسیم مفیده.
البته من خودم رو برنامه نویس با تجربه ای نمی دونم اما به شخصه با وجود لایه DAL بسیار موافقم. نظم خاصی به کد میده و باعث میشه موقع تست یا بازبینی کد دیگه درگیر دیتابیس نباشیم.
یه خواهشی هم دارم میشه لطف کنید یه کم در مورد لزوم تحلیل درست پیش از شروع به کدنویسی هم بنویسید.
پایدار باشید.

نویسنده: m.jamshidi
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۱:۵۴:۴۳

سلام، خسته نباشید، ما وقتی با c# برنامه می نویسیم اصولا متغیرهایی را که نیاز داریم را در ابتدای تابع تعریف می کنیم پس منظورتان از جمله "تعریف کردن متغیرها در ابتدای تابع یعنی این برنامه نویس هنوز حال و هوای ANSI C را دارد!" چیست؟ یعنی هرجا نیاز داشتیم باز هم متغیر تعریف کنیم. موفق باشید

نویسنده: m.jamshidi
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۲:۲۶:۵۶

عجب!!!
آخه مگه دو تا select متفاوت از یه جدول توی یک فرمان انجام میشه!
اونم توی دستوری مثل login جالبه .....

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۲:۵۶:۱۶

در ANSI C ، متغیرها فقط ابتدای تابع (scope block) مجاز به تعریف هستند؛ و هر زمان که به دنبال تعریف متغیرها در ابتدای متدی گشتید یعنی هنوز حال و هوای ANSI C برقرار است. اما از زمان سی++ به بعد خیر؛ هر چند کامپایلر با هر دو حالت مشکلی ندارد.
بحث تکمیلی در اینجا : (^)
به صورت خلاصه: از سی++ به بعد متغیرها را در نزدیکترین مکانی که اولین استفاده از آنها صورت می‌گیرد تعریف کنید تا بهتر بدانید که به چه علتی تعریف شده. همچنین خیالتان هم راحت می‌شود که به اشتباه جای دیگری استفاده نشده یا نمی‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۳:۰۲:۰۴

هیچ ایرادی نداره که حتی 10 کوئری متفاوت هم در یک متد صادر شوند. این business logic که می‌گن همینه. بسته به منطق تجاری که قرار است پیاده سازی شود، شما هر تعداد کوئری که نیاز داشتید را فراخوانی کنید.
فقط بحث اینجا است که فایل code behind ، محل پیاده سازی Data access layer و همچنین محل تعریف هیچ نوع منطق تجاری نیست. این‌ها باید از فایل code behind دور شوند. اینجا فقط محل استفاده نهایی از منطق تهیه شده است.

نویسنده: ناصر طاهری
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۴:۰۵:۱۶

درود بر شما
به نظر من برای این کد همون کلمه کثافت کاری که استفاده کردید مناسبه. من خواستم پروژه ای رو توسعه بدم که با همچین کدی برخوردم. واقعا خیلی بد بود. خیلی

نویسنده: ناصر طاهری
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۴:۰۸:۰۶

:)) فکر کردم عکس برای قسمت عکس مخاطب انتخاب میشه. مثل اینکه اشتباه شد.پاکم نمیشه.شرمنده

نویسنده: rahmat rezaei
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۶:۱۴:۳۳

شک ندارم خیلی از ماها زمانی کدهای بدتر از این هم نوشتیم. بنابراین اینقدر کد بیچاره را مسخره نکنید.
طرف اگر بیاد اینجا و نظرات شما رو بخونه که سرشو میندازه پایین و برنامه نویسی رو میذاره کنار و احتمالا میره سمت تحلیل و طراحی!
دستگیری کنید از ضعیفان نه سرکوفت بزنید!

نویسنده: Mehdi
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۶:۴۰:۳۵

با سلام و ممنون از مطلب خوب شما اقای نصیری عزیز
یک سوال در مورد نحوه dispose کردن شی کانکشن دارم
- شما گفتید که اگر استثنایی اون وسط اتفاق بیفته دیگه به متد dispose نمیرسه و کانکشن باقی میمونه خب این درسته حالا اگه این کدها رو داخل بلاک using قرار بدیم و اتفاقا استثنایی اتفاق بیفته اون وقت شی کانکشن dispose میشه یا به همون صورت قبلی رفتار میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۶:۵۱:۰۹

using توسط کامپایلر به try/finally ترجمه میشه و قسمت dispose رو در قسمت finally قرار میده. بنابراین اگر این وسط استثنایی رخ بده، حتما قسمت finally اجرا خواهد شد.

نویسنده: Mehdi
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۷:۱۷:۳۹

بسیار ممنون
پس به نظر شما نوشتن using هم بهتر و هم راحت تر از نوشتن try catch finally هستش درسته؟

نویسنده: وحید نصیری

تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۷:۳۸:۴۱

بله. من خودم همیشه از using استفاده می‌کنم. جمع و جورتر هست و تمیزتر. به علاوه مثلا زمان dispose خودش بررسی می‌کنه که آیا شیء الان نال هست یا نه. در کل یک سرویس رایگان هست از طرف کامپایلر!

نویسنده: A. Karimi
تاریخ: ۱۳۹۰/۱۰/۱۷ ۱۸:۴۲:۵۷

حرف شما درسته فقط در مورد تحلیل و طراحی، کسی که نمیتونه یک قطعه کد با نظم رو ایجاد کنه خیلی خیلی خطرناک تر هست که بره تحلیل و طراحی کنه.

نویسنده: Mohsen
تاریخ: ۱۳۹۰/۱۰/۱۸ ۱۳:۴۸:۵۴

پس هیچ یک از شما برنامه های فاکس رو که طرف یک جدول تعریف کرده با 70 تا فیلد و اسمشون رو از 1 تا 60 گذاشته(در مورد کدهای نوشته شده چیزی نمیگم!)رو تا به حال پشتیبانی نکرده تا درد منو کشیده باشه!):

نویسنده: Mohsen
تاریخ: ۱۳۹۰/۱۰/۱۸ ۱۳:۵۶:۵۰

اما گذشته از بحث مثالی که زدم(اشتباه و بی اهمیت بودن برنامه نویس به چیزی که خلق می کند!) واقعا وجود امکانات هم در نوشتن کد تمیز و اصولی واقعا تاثیر گذاره.یعنی با وجود هزارتا داستان مثل Intellisense و ابزارهای Refactor که با IDE ای با قدرت VS موجود هستند،دیگه بی انصافیه که طوری کد نوشته شود که دیگران از آن چیزی متوجه نشوند...

نویسنده: Mohsen Najafzadeh
تاریخ: ۱۳۹۰/۱۰/۱۸ ۱۴:۰۵:۱۰

سلام مهندس
مواردی که سایت فقط با کمتر از 30 کاربر همزمان از کار می‌افته رو اگه لیست کنید ممنون می شم

نویسنده: ناصر طاهری
تاریخ: ۱۳۹۰/۱۰/۱۸ ۱۵:۳۲:۲۴

درسته اما بیشتر مواقع آدم باید بهش برخوره که بتونه بره دنبال کاری که تو اون زمینه مورد تمسخر(که مسخره کردن نیست بلکه یک انتقاده) قرار گرفته. :)

نویسنده: Mohammad Safdel
تاریخ: ۱۳۹۰/۱۰/۱۸ ۲۳:۵۰:۲۵

ممنون. متاسفانه تو اغلب شرکتهای ایرانی شعارشون اینه که "کدی که کار می کنه نباید تغییر بدن"! بنابراین Code Review یعنی کشک.

نویسنده: مهدی موسوی
تاریخ: ۱۳۹۰/۱۰/۲۰ ۱۷:۳۳:۱۱

سلام.
اگر کدی آزمایش شده، مرور شده و "کار میکنه"، دیگه نیازی به تغییر اون وجود نداره. برنامه نویس ها عموما در دوران حرفه ای خودشون، حداقل یک بار با "وسوسه بازنویسی همه چیز از نو" روبرو میشن، وسوسه ای که در ابتدا، افق های روشنی رو برامون ترسیم میکنه، اما در انتها، منجر به داشتن کدی به مراتب بدتر از اون چیزی که در ابتدا داشتیم، میشه.

وقتی کدی قدیمی (که بدون مشکل کار میکنه) رو دور میندازیم، در حقیقت داریم زمانی رو که صرف رفع ایرادهای موجود در اون

کرده بودیم (که میتونه روزها، هفته ها یا ماه ها باشه) رو هدر میدیم. گذشته از این، چون احتمالا به تمام بخش های کد و عملکرد اون اشراف نداریم، چیزهایی ممکنه در کد ببینیم که به نظرمون احمقانه بیاد و حذف اونها، باز موجب از کار افتادن بخش هایی از سیستم بشه که Debug کردن اون، مستلزم صرف زمانی هستش که تیم قبلی اون زمان رو یکبار صرف این کار کرده بوده. بنابراین، نمی تونیم به عنوان یه اصل کلی عبارت "کدی که کار میکنه رو نباید تغییر داد" رو رد کنیم! این مساله، باید بازای Case های مختلف، بدقت بررسی بشه و بعد در مورد اون Case خاص، نظر داده بشه.

طبیعتا، با دیدن کد آورده شده در این پست میشه به این مساله پی برد که نویسنده اون کد، در وهله اول، با اصول و مفاهیم اولیه نوشتن یک کد تمیز، بیگانه بوده. چنین افرادی، ابتدا باید آموزش ببینن و مرور یا عدم مرور کد اونها، در طولانی مدت، هیچ سودی در پی نخواهد داشت.

موفق باشید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۲۰ ۲۰:۴۶:۰۹

این کد بالا هم «کار می‌کنه». این فرد یا این شرکت اگر جرات داره اون برنامه رو در طی یک سایت بذاره روی اینترنت! خیلی‌ها علاقمند هستند تا کمی اهمیت مرور کدها رو به اون‌ها به نحو مقتضی یادآوری کنند!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۲۰ ۲۰:۵۲:۱۴

«نمی تونیم به عنوان یه اصل کلی عبارت "کدی که کار میکنه رو نباید تغییر داد" رو رد کنیم»
این مساله فقط زمانی رخ می‌ده که هیچ تستی وجود نداشته باشه. هیچ باگی در وهله اول به عنوان یک آزمون واحد جدید تعریف و سپس بررسی نشده باشه. در این صورت چون باگ‌ها به نحو شایسته‌ای مستند نشدن، سیستم دربرابر تغییرات شکننده خواهد بود، همچنین دلایل وجودی قسمت‌های «احمقانه» کد هم مشخص نخواهد بود.

نویسنده: Shima2012
تاریخ: ۱۳۹۰/۱۰/۲۰ ۲۱:۳۱:۰۱

با سلام خدمت شما استاد عزیز
در لینکی که در ادامه قرار دادم سمپلی از روشی که مدتی در ASP.Net WebForm استفاده میکردم رو قرار دادم.
میخواستم خواهش کنم اگه میشه لطف کنید و مشابه این پست، در پست دیگری روش من رو هم در سایت قرار بدید تا شما و دوستان دیگر نظرات و انتقادات خود را ارائه کنند.
باور کنید انتقادات شما اساتید بزرگوار بسیار در پیشرفت من موثر است.
پیشاپیش ممنون از لطف شما.

http://s2.picofile.com/file/7243730642/TestApp.rar.html

نویسنده: مهدی موسوی
تاریخ: ۱۳۹۰/۱۰/۲۰ ۲۲:۳۲:۳۱

سلام.
من به برخی از مشکلات کد شما اشاره می کنم:

1. ترکیب کدهای DAL و کد UI.
2. عدم جداسازی Business Object ها.
3. ایجاد کلاسی به اسم SqlHelper و قرار دادن N تا متود static در این کلاس، نشون میده که برنامه از هیچ یک از معیارهای موجود پیروی نمیکنه. مطلقا دلیلی نداره که متودها static تعریف بشن و ... حتی اگر قرار باشه چنین کلاسی تعریف کنیم، (که من

کاملا باهاش مخالفم)، باید متودهای اونو بر اساس Property ی Singleton ای از کلاس به بیرون Expose کنیم. نه اینکه همه متودهاشو static بذاریم و کلاس رو هم sealed کنیم و ... چنین کلاسی هرگز قابل توسعه نیست.
4. وقتی همه متودهای کلاس static هستش، چرا Constructor ای private برای اون کلاس تعریف کرده اید؟ تازه بهتر بود جای اینکه کلاس رو sealed تعریف می کردید، اونو static تعریف می کردید (اگر مجاب میشدیم که تعریف چنین کلاسی صحیح هستش).
5. catch کردن کلیه SqlException ها در متود CloseCnt.
6. عدم استفاده از Parametric Command ها (و در نتیجه فراهم اومدن امکان SQL Injection).
7. عدم اجرای StyleCop روی کد (و در نتیجه، نوشتن Comment های مربوط به هر تابع بر اساس Style ای من در آوردی، عدم رعایت Spacing، نوشتن if ها در یک خط و ...)
8. و ...

موفق باشید.

نویسنده: Shima2012
تاریخ: ۱۳۹۰/۱۰/۲۰ ۲۲:۵۳:۵۳

واقعاً ممنون
لطف کردید.
شما مثالی دارید که شبیه مثالی که من زدم باشه و استاندارد هایی که گفتید رو رعایت کرده باشه تا من بتونم یاد بگیرم؟

نویسنده: پریسا زاهدی
تاریخ: ۱۳۹۳/۱۰/۱۹ ۱۱:۵۷

سلام
اگر از Try Catch استفاده نکنیم, با استفاده از Using پس چطوری به اطلاعات استثنای رخ داده شده دسترسی پیدا کنیم که بخواهیم لاگش کنیم ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۹ ۱۲:۱۷

تهیه لاگ خودکار را واگذار کنید به ابزارهایی مانند ELMAH