عنوان: **اهمیت ارائه‌ی برنامه‌های دات نت به صورت release**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۸/۱۴ ۲۰:۱۸:۳۱
آدرس: www.dotnettips.info
برچسب‌ها: Tips

یکی از مواردی که بعضی از همکاران هنگام ارائه برنامه‌های خود رعایت نمی‌کنند، تفاوت قائل نشدن بین حالت release و debug در زمان کامپایل پروژه، برای ارائه نهایی است.
هنگام استفاده از حالت release ، گزینه‌های بهینه سازی کامپایلر فعال شده و همچنین debug symbols از اسمبلی نهایی تولید شده حذف می‌گردد (بنابراین حجم اسمبلی نهایی نیز کمتر خواهد شد). لازم به ذکر است در حالت release ، میزان مصرف حافظه برنامه تولید شده نیز کمتر از حالت debug خواهد بود. گاهی از اوقات سرعت اجرای این دو حالت تا چندین برابر در بعضی از الگوریتم‌ها می‌توانند متفاوت باشند.
مطابق مستندات موجود، وجود debug symbols هیچگونه تاثیری بر روی کارآیی یک برنامه دات نت ندارند.
لازم به ذکر است که عمده بهینه‌سازی‌ها در دات نت توسط JIT compiler صورت می‌گیرد (تا 99 درصد) و نه توسط کامپایلر زبان مورد استفاده (به همین جهت است که عده‌ای اعتقاد دارند در نهایت و هنگام اجرا تفاوتی مابین زبان‌های مختلف دات نت وجود نخواهد داشت). بر روی JIT compiler نیز می‌توان تاثیر گذاشت و نحوه عملکرد آنرا تغییر داد (حتی بر روی یک اسمبلی کامپایل شده). برای مثال یک فایل ini کنار اسمبلی پروژه خود ایجاد کنید (xyz.ini که در اینجا xyz.exe نام فایل اجرایی برنامه است).
محتویات این فایل می‌تواند به صورت زیر باشد:

```
[.NET Framework Debugging Control]
GenerateTrackingInfo=1
AllowOptimize=0
```

در اینجا می‌توان بهینه سازی را فعال و غیر فعال کرد و یا مطابق تنظیمات فوق برنامه را جهت دیباگ آماده نمود. (این روش با اسمبلی‌های ASP.Net کار نمی‌کند)
در دات نت فریم ورک 2 ، TrackingInfo مربوط به JIT compiler همواره تولید خواهد شد اما می‌توان بر روی بهینه سازی نهایی به صورت فوق نیز تاثیرگذار بود.

نکته:
اگر می‌خواهید هنگام مشاهده گزارش خطا، شماره سطر مورد نظر نیز در کدهای شما گوشزد شود فایلهای pdb - program database تولید شده را هم ارائه دهید. حال شاید بخواهید هم برنامه را در حالت release ارائه دهید و هم pdb آن تولید شود، در این حالت باید خط فرمان کامپایل برنامه، با سوئیچ /debug:pdbonly اجرا شود.
این مورد را در قسمت خواص پروژه، گزینه build و با کلیک بر روی دکمه advanced نیز می‌توان تنظیم نمود. (حالت پیش فرض release در VS.Net است)

خلاصه‌ی کلام: لطفا هنگام ارائه نهایی، گزینه release را از بالای صفحه در VS.Net انتخاب کنید. با تشکر!

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۱۳۸۷/۱۰/۱۴ ۱۸:۵۶:۰۰

با سلام
در vs 2008
گزینه release برای web site موجود نیست! چرا؟
چه باید کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۱۴ ۱۹:۱۸:۰۰

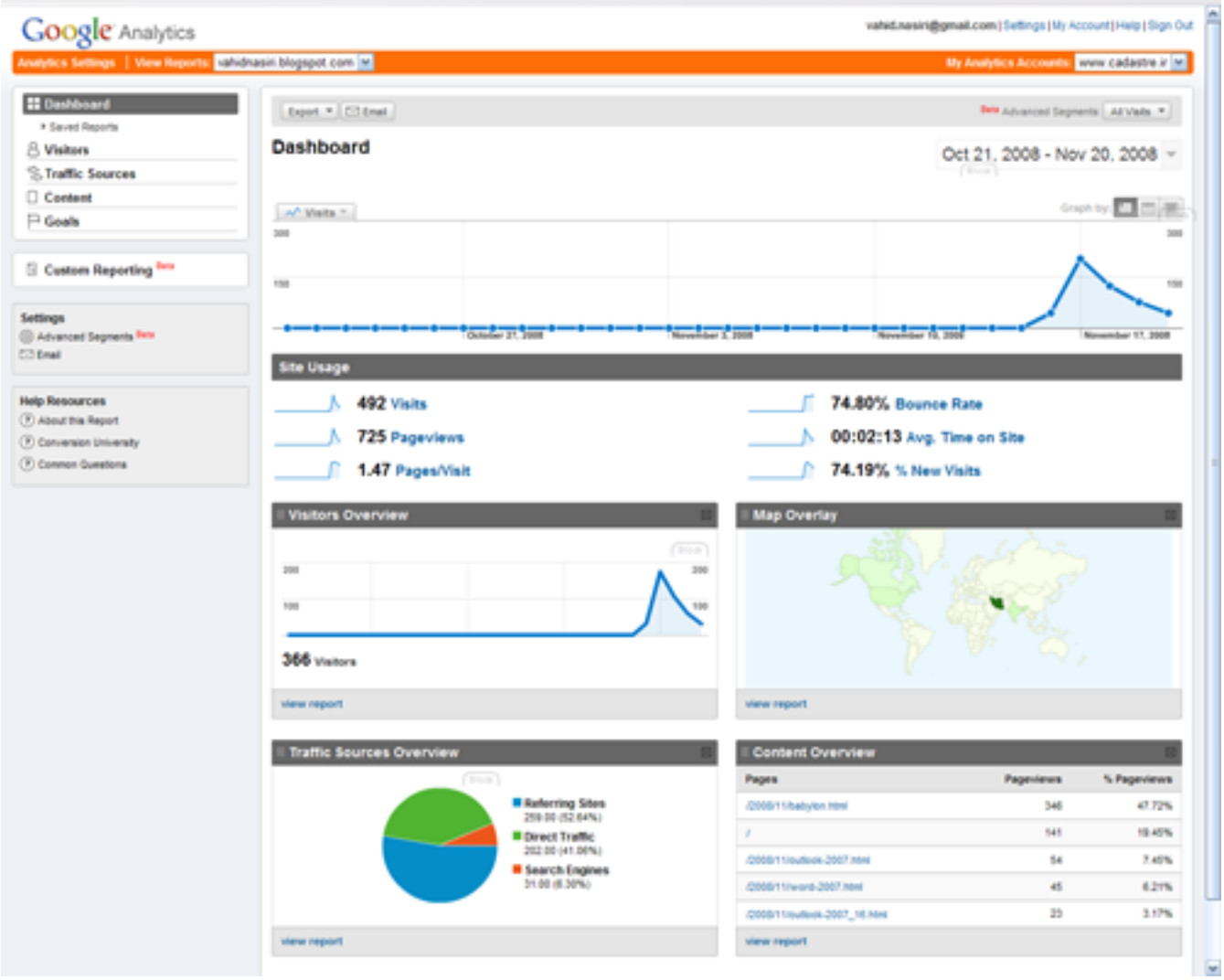
من همیشه از حالت web application برای ASP.Net استفاده می‌کنم.
+ شما در ASP.Net همیشه می‌تونید نحوه کامپایل شدن را در وب کانفیگ هم تعیین کنید. به این صورت:
"compilation defaultLanguage="c#" debug="false
برای مطالعه بیشتر
http://aspnetresources.com/articles/debug_code_in_production.aspx
و یا استفاده از روش زیر که در آن retail را باید true کنید:
http://msdn.microsoft.com/en-us/library/ms228298(VS.80).aspx

عنوان: **استفاده از Google Analytics در ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۰۲ ۰۹:۴۱:۴۸
آدرس: www.dotnettips.info
برچسب‌ها: Tips

قبل از استفاده از بلاگر، در سایت wordpress وبلاگ داشتم، که به‌دلایلی کنسل شد. تفاوت محسوسی را که اینجا مشاهده می‌کنم، نبود قسمت آمار سایت است. در سایت wordpress آمار مبسوطی را از بازدید کنندگان سایت می‌توانید در کنترل پنل مدیریتی وبلاگ مشاهده کنید، اما در اینجا خیر.

به همین جهت اولین کاری را که انجام دادم استفاده از سرویس رایگان persianstat بود که انصافا هم با کیفیت است و قابل مقایسه با آماری که wordpress ارائه می‌دهد، می‌باشد.

جالب اینجا است که هر چند هاست اینجا، گوگل است اما استفاده‌ی خودکار از ابزار Google analytics در آن مهیا نیست. احتمالا علت آن آماده نبودن API آن است که قرار است به زودی ارائه شود، بنابراین ارزش وقت گذاشتن را دارد.



برای استفاده از Google analytics ، پس از ثبت نام و ورود به آن، سایت مورد نظر را معرفی کرده (در قسمت Add Website Profile) و نهایتا یک کد جاوا اسکریپتی به شما خواهد داد که می‌توانید آنرا به صفحات مورد نظر خود در سایت اضافه نمائید تا

تحت کنترل آماری قرار گیرد. محدودیتی هم در مورد تعداد سایت وجود ندارد و با یک اکانت می‌توانید چندین سایت را معرفی کرده و تحت کنترل قرار دهید.

اگر از ASP.Net استفاده می‌کنید، تنها کافی است به master page سایت مراجعه کنید و پیش از بسته شدن تگ body ، اسکریپت مربوط به Google analytics را اضافه کنید تا تمام سایت را تحت کنترل قرار دهید.

یا اگر علاقمند بودید که اینکار را به صورت "شیک‌تری" انجام دهید، می‌توان از این http module استفاده کرد. به این صورت ابتدا تگ بسته شدن body به صورت خودکار پیدا شده و سپس اسکریپت به پیش از آن اضافه می‌شود.

این روش بار بزرگ تهیه آمار سایت را حذف خواهد کرد. عموما دیتابیس جمع آوری آمار سایت خیلی زود (برای مثال پس از گذشت 6 ماه) حجیم می‌شود و تاثیر مشهودی را بر روی کارآیی سایت خواهد گذاشت. بنابراین، این سؤال مطرح می‌شود که چرا گوگل اینکار را برای ما انجام ندهد؟! هزینه بانک اس کیوال سرور بر روی هاست‌های اینترنتی بالا بوده و حجمی را هم که در اختیار قرار می‌دهند محدود است. در صورت نیاز به حجم‌های بالاتر باید هزینه بیشتری را پرداخت کرد. بنابراین هم از لحاظ قیمت و هچنین کارآیی سایت، استفاده از این سرویس واقعا مقرون به صرفه است. بعلاوه از تنوع آماری که ارائه می‌دهد نیز نمی‌توان چشم پوشی کرد. برای مثال کاربران چه واژه‌های کلیدی را در موتورهای جستجو وارد کرده‌اند تا به سایت شما رسیده‌اند؟ چند درصد کاربر وفادار دارید؟! (کاربرهای وفادار، منظور افرادی هستند که به صورت منظم به سایت سر می‌زنند) و امثال این. انصافا تهیه چنین ماژولی برای یک سایت از لحاظ برنامه نویسی شاید با برنامه نویسی کل یک سایت برابری کند.

اگر هم نیاز به یک برنامه سورس باز داشتید که هر روز به اکانت Google analytics شما سر بزند و اطلاعات آنرا استخراج کرده و در یک بانک SQL server ذخیره کند، می‌توانید به پروژه سی شارپ زیر مراجعه نمائید:

Google Analytics Data Extractor

البته باید دقت داشت که پس از ارائه API کامل Google analytics ، دیگر نیازی به این نوع روش‌های ابتکاری وجود نداشته و استخراج داده از آن بسیار ساده‌تر خواهد شد.

نظرات خوانندگان

نویسنده: سارا
تاریخ: ۱۳۹۱/۰۶/۰۲ ۲۰:۵۸

بسیار عالی بود.

عنوان: **مقایسه مجوزهای سورس باز**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۲۸ ۱۰:۵۲:۱۶
آدرس: www.dotnettips.info
برچسب‌ها: Tips

احتمالا همیشه برای شما سؤال بوده است که مجوزهای گوناگون سورس باز با هم چه فرقی دارند، یا اینکه اگر روزی خواستم پروژه‌ی خود را به صورت سورس باز ارائه کنم، کدامیک از مجوزهای موجود مناسب‌تر است و همچنین وقت مطالعه مقالات طولانی یا کتاب‌هایی چند صد صفحه‌ای در این مورد را نداشته‌اید.
جدول زیر کار مقایسه این مجوزها (موارد رایج‌تر) را به صورت مختصر و مفید و بر اساس سؤالات رایج کاربران، انجام می‌دهد:

نام مجوز	آیا به کار مشتق شده از پروژه اصلی، می‌توانم نامی دیگر بدهم؟	آیا باید حتما کار مشتق شده سورس باز باقی بماند؟	آیا می‌توانم برای کار مشتق شده مجوزی جدید انتخاب کنم؟	آیا می‌توانم کار مشتق شده را بفروشم و کسب درآمد کنم؟
Apache License 2.0	بله	خیر	بله	بله
Common Development and Distribution License (CDDL)	بله	خیر	بله (به مجوزهای سازگار دیگری از همین دست)	بله
GNU General Public License 2.0 (GPLv2)	بله، اما حتما باید لیست تغییرات انجام شده نسبت به پروژه اصلی را نیز ارائه بدهید.	بله	بله (به مجوزهای سازگار دیگری از همین دست یا توافق با نویسنده اصلی)	بله
GNU Library General Public License (LGPL)	بله	بله، اما امکان استفاده از کتابخانه‌های کامپایل شده یک پروژه سورس باز تحت این مجوز در یک پروژه سورس بسته نیز وجود دارد.	بله (به مجوزهای سازگار دیگری از همین دست)	بله
Microsoft Public License (Ms-PL)	بله، اما نمی‌توانید از علامت تجاری خود استفاده کنید.	خیر	خیر	بله
Microsoft Reciprocal License (Ms-RL)	بله، اما نمی‌توانید از علامت تجاری خود استفاده کنید.	بله	خیر	بله
Mozilla Public License 1.1 (MPL)	بله	خیر	خیر	بله
BSD License	بله	خیر	بله	بله
MIT License	بله	خیر	بله	بله

همچنین لازم به ذکر است که

مجوزهای کار اصلی و کار مشتق شده هر دو باید ذکر شوند.
پسندیده است که از نویسندگان کار اصلی، نامبرده شده و قدردانی گردد.
هیچکدام از این مجوزها مسؤولیتی را در قبال کار انجام شده نمی‌پذیرند!

جهت مطالعه بیشتر:

http://khason.net/blog/open-source-licenses-comparison-table/

http://developer.kde.org/documentation/licensing/licenses_summary.html

http://en.wikipedia.org/wiki/Comparison_of_free_software_licences

نظرات خوانندگان

نویسنده: SirAsad
تاریخ: ۱۳۸۷/۱۰/۲۸ ۱۷:۲۱:۰۰

ممنون آقای نصیری , بسیار استفاده بردم از بلاگتون .

شاید شما من رو به یاد نیآرید , من مدتی در سایت برنامه نویس در خدمتتون بودم , خوشحال که دوباره ملاقاتتون می کنم .

فکر کنم الان بیشتر از GNU استفاده میشه دیگه , نه؟

موفق باشید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۲۸ ۱۷:۴۷:۰۰

سلام،
شما لطفا دارید.
طبق مقاله زیر MPL محبوب‌ترین است:
The Mozilla Public License (MPL) is the most popular open source license that contains a weak copyleft provision

http://www.onlamp.com/pub/a/onlamp/2004/11/18/licenses.html

نویسنده: Anonymous
تاریخ: ۱۳۸۷/۱۰/۲۸ ۲۱:۳۹:۰۰

سلام. ممنون از این مطلب. سوالم این بود که در مجوز GPLv2 (GNU General Public License 2.0)
نوشته شده که حتما باید کار مشتق شده سورس باز باقی بماند و دیگر اینکه می توان کار خود را تحت مجوز دیگر منتشر کنیم
حالا اگر برای کار مشتق شده جدید مجوز ما از نوع CDDL (Common Development and Distribution License) انتخاب شود در
مورد سورس باز بودن کار با هم متناقض نمی شوند؟

نویسنده: salarblog
تاریخ: ۱۳۸۷/۱۰/۲۸ ۲۲:۳۹:۰۰

ممنون بسیار لازم بود
در ایران اکثرا کمتر در مورد تفاوت این موارد می دانند!

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۲۸ ۲۳:۱۷:۰۰

@ ناشناس
همانطور که ذکر گردید باید مجوز انتخاب شده با GPLv2 در یک گروه قرار داشته باشد (یا با نویسنده اصلی توافق صورت گیرد).
لیست موارد سازگار با آن‌را در این آدرس می‌توانید مشاهده نمائید و CDDL با آن سازگار نیست. بنابراین نمی‌توانید آنرا انتخاب کنید.

Compatible licenses
http://gplv3.fsf.org/wiki/index.php/Compatible_licenses

نویسنده: chalism
تاریخ: ۱۳۸۷/۱۱/۰۵ ۰۰:۱۵:۰۰

جالب بود
سپاس

نویسنده: مانی
تاریخ: ۱۳۸۷/۱۱/۰۶ ۲۲:۴۴:۰۰

واقعا ممنونم از این نوشته. به نظر من توی ویکیپدیای فارسی اگر قرارش بدید خیلی خوب میشه.

نویسنده: Nasser Man
تاریخ: ۱۳۸۹/۰۹/۳۰ ۱۵:۱۷:۴۴

در مورد
Open Source Initiative OSI - Common Public License Version 1.0:Licensing
[OSI Approved License]
(Common Public License Version 1.0 (CPL
چیزی ننوشته بودید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۳۰ ۱۷:۴۰:۳۲

در مورد CPL :
- آیا به کار مشتق شده از پروژه اصلی، می‌توانم نامی دیگر بدهم؟
بله.
- آیا باید حتما کار مشتق شده سورس باز باقی بماند؟
بله. اگر کار جدیدی است و از کدهای پروژه اول تحت مجوز CPL استفاده کرده.
خیر. اگر همان کار اولیه است و فقط کامپایل شده (فقط باید ذکر کنید سورس اصلی را از کجا می‌شود تهیه کرد).
در هر دو حالت، حتما باید ذکر کنید که کار جدید از چه کاری مشتق شده (به عبارتی تشکر فراموش نشود!)

- آیا می‌توانم برای کار مشتق شده مجوزی جدید انتخاب کنم؟
خیر. مگر با دریافت مجوز از صاحب اصلی کار. (و زمانیکه این مورد خیر شد یعنی نمی‌تونید از اینکار در یک پروژه جدید مثلا GPL استفاده کنید، مگر با نویسنده اصلی توافق کنید)

- آیا می‌توانم کار مشتق شده را بفروشم و کسب درآمد کنم؟
بله. (این بله در تمام مجوزها با توجه به شرایط قبل باید تفسیر شود. یعنی اگر در این مورد اخیر گفته شده بله، یعنی بله اگر سورس خودت رو در اختیار بقیه هم قرار دادی، چون در این مورد خاص صراحتا این مورد خواسته شده، اما مثلا مجوز MIT چنین چیزی رو نمی‌خواد)

ماخذ: (+)

نویسنده: میثم نوایی
تاریخ: ۱۳۸۹/۱۰/۰۵ ۱۱:۴۴:۵۰

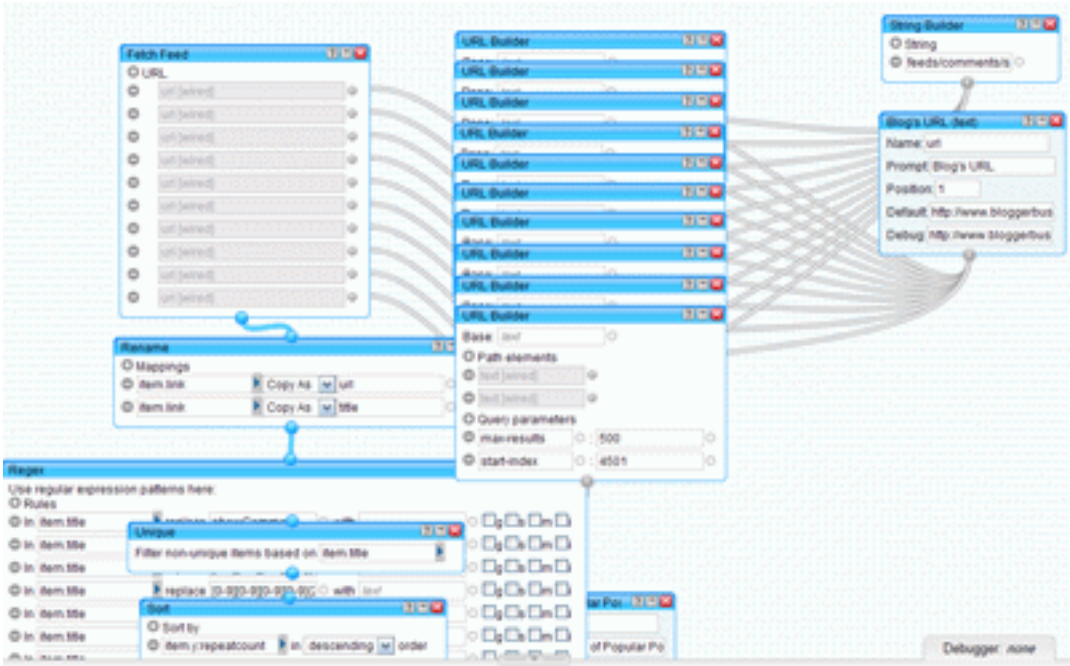
من خیلی وقت بود که در سایت های فارسی جای خالی چنین مطلبی را میدیدم.سپاس.

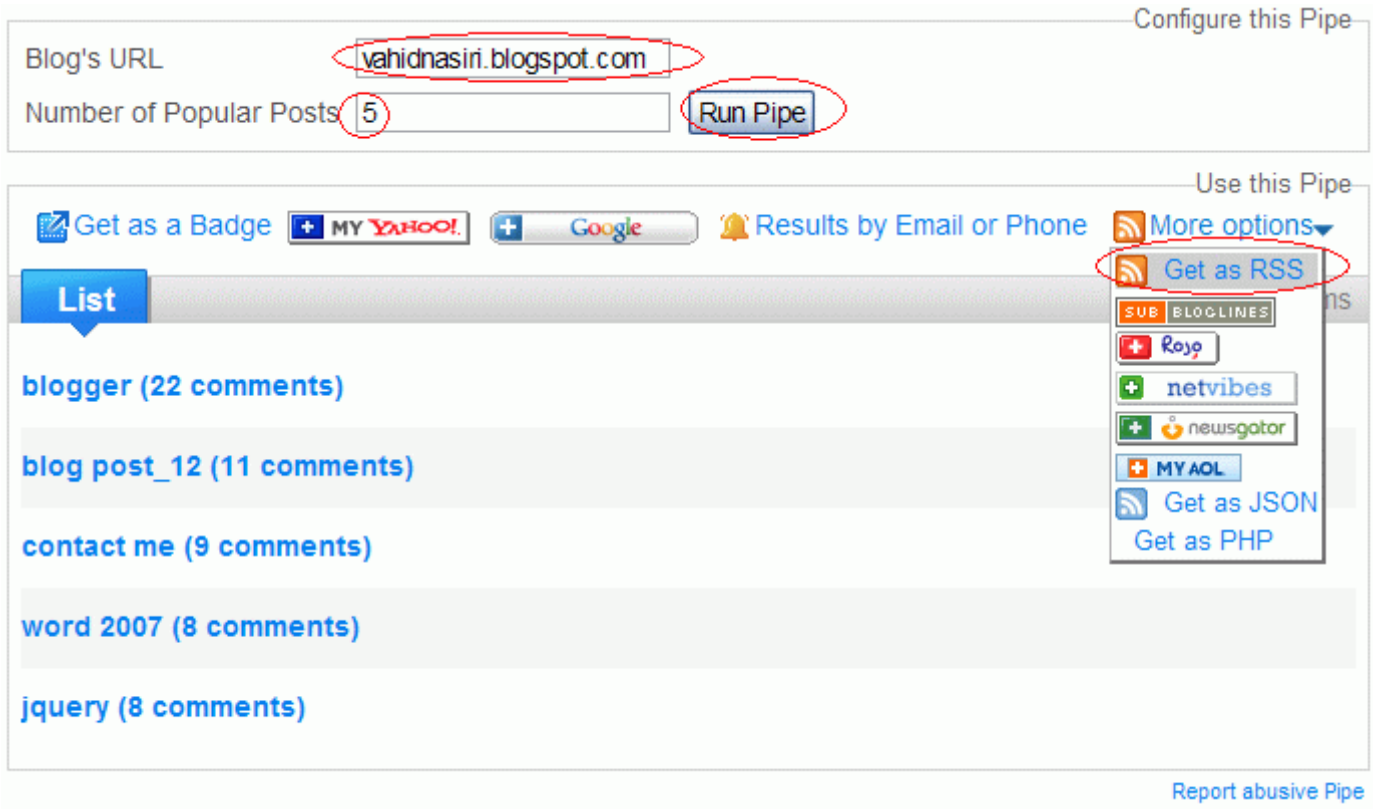
عنوان: **استفاده از لوله‌های یاهو در بلاگر!**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۲۹ ۱۷:۴۵:۰۵
آدرس: www.dotnettips.info
برچسب‌ها: Tips

داشتم به دنبال راهی برای نمایش محبوب‌ترین پست‌ها در وبلاگ جاری می‌گشتم، این جستجو به لوله‌های یاهو ختم شد!
یکی از پرکاربردترین ویجت‌های بلاگر، ویجت نمایش فید است (با نام "عناوین خبری" ترجمه شده است). برای مثال همین لیست آخرین نظرها در سایت، با استفاده از فید استاندارد کامنت‌های سایت درست شده است. 5 کامنت آخر سایت را نمایش می‌دهد که البته این یک ایراد هم هست و بیشتر از این تعداد را قبول نمی‌کند. یا دقیقا همان زمان ارسال کامنت به روز نمی‌شود. به نظر در ساعات مشخصی از روز این به روز رسانی صورت می‌گیرد.

جهت "نمایش لیست مطالبی با بیشترین کامنت" می‌توانید به آدرس زیر مراجعه کنید:
Popular Posts/Most Commented Widget for Blogger Blogs

آدرس بلاگ خود را وارد کنید (بدون http البته). عدد دلخواهی را که بیانگر تعداد رکورد بازگشت داده شده است نیز وارد نمائید (هر چند بلاگر فقط 5 آیتم را نمایش خواهد داد) و سپس بر روی دکمه run کلیک کنید. در همانجا روی more options کلیک کرده و لینک فید rss آن‌را دریافت کنید. در حقیقت با استفاده از لوله‌های یاهو یک سری پردازش روی فید کامنت‌های سایت صورت گرفته و آمار نهایی به صورت یک فید جدید به شما ارائه خواهد شد که از آن می‌توانید در ویجت مربوط به عناوین خبری یا همان فیدهای بلاگر، استفاده کنید.





و یا نمایش لیست برترین کامنت گذاران!

Top Commentators Widget for Blogger

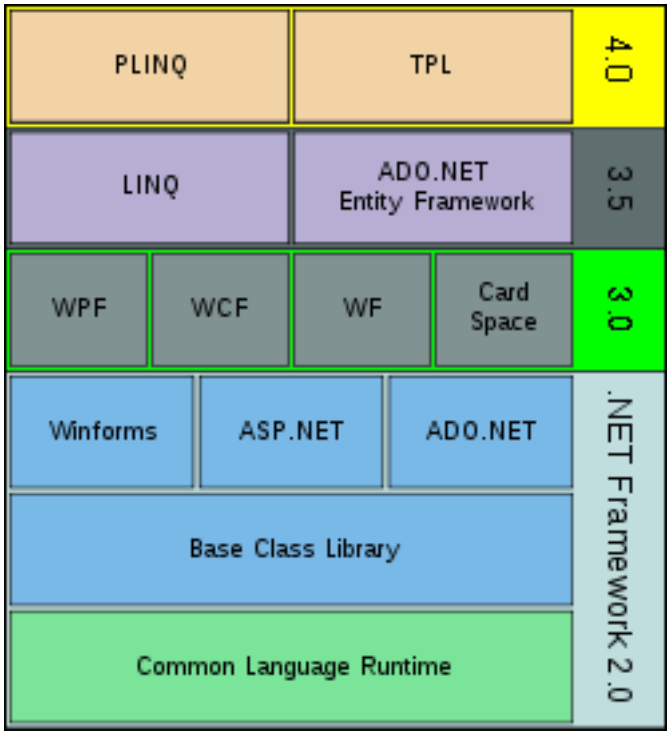
در اینجا روی لینک clone کلیک کنید تا یک نمونه مخصوص شما کپی شود (بهتر است به اکانت ایمیل خود در یاهو لاگین کرده باشید). اکنون می‌توانید آن را ادیت کرده و بر روی آن تغییرات دلخواه را اعمال کنید (وارد کردن لینک فید کامنت‌های سایت مطابق شکل). سپس بر روی دکمه ذخیره کلیک کرده و نهایتا فید rss آن‌را دریافت کنید.

موارد دیگری هم از همین دست در سایت لوله‌های یاهو موجود است:

http://pipes.yahoo.com/pipes/search?q=blogger&x=6&y=9

عنوان: **تاریخچه‌ی نگارش‌های مختلف دات نت فریم ورک**
نویسنده: وحید نصیری
تاریخ: ۱۸:۴۱:۱۶ ۱۳۸۷/۱۱/۱۸
آدرس: www.dotnettips.info
برچسب‌ها: Tips

حدود 8 سال از ارائه اولین نگارش دات نت فریم ورک می‌گذرد و در ادامه مرور سریعی خواهیم داشت بر عناوین کتابخانه‌های اضافه شده به این مجموعه:



دات نت فریم ورک 1.0

اولین ارائه عمومی آزمایشی آن در PDC 2000 صورت گرفت و در اوایل 2002 به عموم عرضه شد (2002/13/2).

عبارت کد مدیریت شده را به دنیا معرفی کرد (managed code) و شامل اجزای زیر بود:

- GC, JIT
- C#
- Coherent Framework
- XSP….ASP+…ASP.NET!
- WinForms

دات نت فریم ورک 1.1

در اوایل 2003 به همراه ویندوز سرور 2003 و VS 2003 ارائه شد.

- Mobile ASP.NET controls
- Built-in support for ODBC and Oracle databases.
- IPv6 support.

دات نت فریم ورک 2

در اواخر 2005 ارائه شد (2005/07/11) و شامل تازه‌های زیر بود:

```
• ASP.NET for the Masses
  ○ Application Building Blocks
    § Parts, Authentication, Role Management, etc
  ○ Visual Web Developer
• Client Development
• ClickOnce!
```

دات نت فریم ورک 3

در اواخر 2006 ارائه شد (2006/06/11) و موارد زیر را به این فریم ورک افزود:

```
• Windows CardSpace - Digital identity interface.
• Windows Presentation Foundation : WPF
  ○ Vector Graphics, Media and UI
  ○ Enters the age of UX
• Windows Communication Foundation : WCF
  ○ Unified messaging model
• Windows Workflow Foundation : WF
  ○ Coordinating work with durable applications
```

دات نت فریم ورک 3.5

در پایان 2007 ارائه شد (2007/19/11) و تازه‌های زیر را به همراه داشت:

```
• Linq
• Expression Trees and Lamda Methods
• Extension Methods
• Paging Support for ADO.NET
• Managed Wrappers for WMI and AD
• Enhancements to WCF and WF
• System.CodeDom namespace
• ASP.NET AJAX
• WCF/WF
  ○ REST Services
  ○ Workflow Services
• Client
  ○ Sync
  ○ Client app services
```

در همین زمان نیز دات نت فریم ورک سورس باز شد .

سرویس پک یک دات نت فریم ورک 3.5

در اواسط 2008 ارائه شد (2008/11/8) و به همراه تغییرات زیر بود:

```
• ASP.NET Dynamic Data
• ADO.NET
 ○ Entity Framework
 ○ Data Services (Astoria)
• WCF
 ○ AtomPub ServiceDocuments
• Client
 ○ Client Profile
 ○ Performance
   § Working set and startup time
• Silverlight 2
 ○ RTM end of 2008
 ○ Brings power of .NET to the web client
 ○ Media and RIA .NET platform
```

دات نت فریم ورک 4.0

نگارش بتای آن در دسترس است و احتمالا نگارش نهایی آن در سال آینده‌ی میلادی به همراه VS2010 ارائه می‌شود. این مجموعه تازه‌های زیر را به همراه خواهد داشت:

```
• Base Class Library Improvements
 o Managed Extensibility Framework
 o More Core Data Structures
 o I/O Improvements
• Parallel Computing
 o Task Parallel Library
 o Parallel Linq (PLINQ)
 o Coordination Data Structures (CDS)
• Client
 o WPF
  § Client Profile
  § Business Focused Controls
  § Win7 Advances (Multi-touch, etc)
 o ADO.NET
  § Entity Framework v2
    □ Code-First Development
    □ TDD Support
    □ Foreign-Key Support
 o ASP.NET
  § ASP.NET Dynamic Data Improvements
  § ASP.NET MVC
  § ASP.NET Dynamic Data for MVC
  § Extensible Caching Framework
 o WF & WCF
  § Fully Declarative Services
  § Workflow Enhancements
   □ New flowchart modeling
   □ Workflow Rules Integration
   □ … much more…
  § WCF Enhancements
   □ Durable Duplex
   □ WS-Discovery & UDP Channel
   □ In-Process Channel
  § RIA (Silverlight)
   □ Simplified N-tier development
   □ Business-focused framework
```

نظرات خوانندگان

نویسنده: ایمان
تاریخ: ۱۳۸۷/۱۱/۱۹ ۰۳:۴۷:۰۰

کار جالبی کردی جناب نصیری. توسعه دات نت فریم ورک بر خلاف دیگر محصولات مایکروسافت، به سرعت پیش میره. هنوز در حال یادگیری تکنولوژی هایی به کار رفته در نسخه 3 و 3.5 دات نت هستیم که نسخه 4 با این همه تغییرات عرضه میشه.

جناب نصیری اگر ممکنه در مورد PLINQ توضیحاتی در وبلاگتون بدین، ممنون میشم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۱/۱۹ ۰۹:۳۶:۰۰

ممنون. دات نت فریم ورک پایه و اساس محصولات جدید مایکروسافت است. به همین جهت توسعه‌ی آن برای آن‌ها خیلی اهمیت دارد.
WF برای مثال در شیرپوینت و بیزتاک سرور داره استفاده میشه. یا خیلی از قسمت‌هایی که به ASP.Net داره اضافه میشه به نظر من با هدفگیری این محصولات است که خیرش داره به بقیه هم میرسه.
management studio اس کیوال سرورهای 2005 به بعد با دات نت نوشته شده.
و خیلی از مواردی که مایکروسافت از لحاظ امنیتی با آن‌ها مشکل داشت با کوچ به دات نت فریم ورک به شدت امن شده‌اند چون برای مثال buffer overflow attacks در دات نت معنی ندارد و اگر کد شما فقط از managed code تشکیل شده باشد، کاملا کنترل شده است. به همین جهت اگر به لیست تعداد حملات صورت گرفته به اس کیوال سرورهای 2005 به بعد مراجعه کنید، کاهش محسوسی رو می‌تونید ملاحظه کنید.

نویسنده: بهروز راد
تاریخ: ۱۳۸۷/۱۱/۱۹ ۱۵:۵۷:۰۰

@ایمان
PLINQ یا همون Parallel LINQ، نسخه ای از LINQ هست که برای اجرا بر روی پردازنده های چند هسته ای بهینه شده. این کار به موازات TPL که در تصویر رشد دات نت که برادر نصیری گذاشته میسر شده.
من قبلاً در وبلاگم در موردش نوشتم:
http://brad.barnamenevis.org/?p=110

موفق باشید.

نویسنده: سعد
تاریخ: ۱۳۸۸/۰۱/۱۴ ۱۳:۲۷:۰۰

یک از تازه های دات نت 4.1 :
http://www.hanselman.com/blog/NET41PreviewNewBaseClassLibraryBCLExtensionMethodsRFC.aspx

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۱۴ ۱۳:۴۵:۰۰

دروغ سیزده بود :)

عنوان: **توصیه‌هایی در مورد overloading متدها**
نویسنده: وحید نصیری
تاریخ: ۱۷:۳۵:۰۸ ۱۳۸۷/۱۱/۲۷
آدرس: www.dotnettips.info
برچسب‌ها: Tips

مطلب زیر چکیده‌ای است از کتاب Framework Design Guidelines در مورد سربارگذاری توابع

الف) از یک نوع خروجی استفاده کنید

مثال زیر را در نظر بگیرید:

```
public User[] GetGroupMembers(int groupId)
public List<User> GetGroupMembers(string groupName)
```

هر دوی این متدها گروهی از کاربران را بازگشت می‌دهند. خروجی متد اول از نوع آرایه است و خروجی متد دوم از نوع یک لیست جنریک می‌باشد.
مشکل اینجا است که هنگام فراخوانی هر کدام، نحوه‌ی استفاده متفاوت خواهد بود. بنابراین باید از این نوع سربارگذاری پرهیز کرد.

ب) نام‌های آرگومان‌های توابع سربارگذاری شده شما باید یکسان باشند

در مثال زیر، از اولین آرگومان جهت دریافت شناسه‌ی یک گروه استفاده می‌شود:

```
public List<User> GetGroupMembers(int groupId)
public List<User> GetGroupMembers(int id, int pageIndex, int pageSize)
```

اما همانطور که ملاحظه می‌کنید در یکی از groupId استفاده شده و در دیگری از نام id . این روش مزموم است! از آن پرهیز کنید! (هر دو یا باید groupId باشند و یا id . این هماهنگی و یکسان بودن باید در توابع سربارگذاری شده‌ی شما حفظ شود)

ج) ترتیب آرگومان‌ها را در توابع سربارگذاری شده حفظ و رعایت نمائید

لطفا به مثال زیر دقت کنید:

```
public List<User> GetGroupMembers(int groupId)
public List<User> GetGroupMembers(int pageIndex, int pageSize, int groupId)
```

در اینجا شناسه‌ی گروه یکبار به عنوان آرگومان اول معرفی شده و بار دیگر به عنوان آرگومان سوم. این کار نیز مزموم است، زیرا برنامه نویس‌ها از روی عادت به اولین آرگومان توابع سربارگذاری شده‌ی شما، همان شناسه‌ی گروه را ارسال خواهند کرد و این مورد سبب بروز باگ‌هایی خواهد شد که ردیابی آن مشکل است. بنابراین اگر شناسه‌ی گروه به عنوان اولین آرگومان معرفی شده، در تمامی overload های این تابع نیز اولین آرگومان باید همان شناسه‌ی گروه باشد.

د) از call forwarding استفاده کنید

جهت توضیح بهتر call forwarding به مثال زیر دقت نمائید:

```
public List<User> GetGroupMembers(int groupId)
{
   return GetGroupMembers(groupId, 0, 10);
}

public List<User> GetGroupMembers(int groupId, int pageIndex, int pageSize)
{
   return GetGroupMembers(groupId, pageIndex, pageSize, SortOrder.Ascending);
}

public List<User> GetGroupMembers(int groupId, int pageIndex, int pageSize, SortOrder sortOrder)
{
   var query = new GroupQuery();
   query.GroupID = groupId;
   query.PageIndex = pageIndex;
   query.PageSize = pageSize;
   query.SortOrder = sortOrder;

   return GetGroupMembers(query);
}

public List<User> GetGroupMembers(GroupQuery groupQuery)
{
   // Actual implementation to get group members goes here
}
```

معنای call forwarding این است که هنگام پیاده سازی توابعی از این دست، کوچکترین تابع سربارگذاری شونده باید تابع بزرگتر بعدی خود را صدا بزند و همین‌طور الی آخر که در مثال فوق به خوبی آشکار است. در این مثال چهارمین تابع سربارگذاری شده، بیشترین حوضه‌ی تمرکز را در خود دارد و توابع دیگر می‌توانند از آن بهره جویند بدون اینکه پیاده سازی خاصی را ارائه دهند.

ه) زیاده روی نکنید!

آخرین موردی که توصیه شده این است که در تهیه توابع سربارگذاری شده زیاده روی نکنید. در این نوع موارد باید قانون 20/80 را در نظر گرفت. 2 تا 3 تابع سربارگذاری شده ارائه دهید که 80 درصد کار را انجام می‌دهند و سپس یک تابع سربارگذاری شده‌ی دیگر ارائه دهید که 20 درصد باقیمانده را پوشش دهد.

پ.ن.
در سی شارپ 4 ، با معرفی optional parameters ، شاید کمتر به سربارگذاری نیاز باشد.

عنوان: **آیا برنامه نویس‌های دات نت باید نگران دنیای 64 بیتی باشند؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۱/۳۰ ۱۸:۲۱:۱۲
آدرس: www.dotnettips.info
برچسب‌ها: Tips

جواب ساده و کوتاه: خیر!
کدمدیریت شده‌ی شما در هر دو پلتفرم 32 بیتی - x86 و x64 بدون نیاز به هیچگونه تغییری و بدون نگرانی اجرا خواهد شد.
گزیده‌ای از MSDN :
اگر کد شما 100 درصد مدیریت شده است (managed code ایی که به صورت خالص از دات نت فریم ورک استفاده می‌کند و هیچگونه وابستگی خارجی دیگری به کتابخانه‌های دیگر ندارد)، تنها با کپی شدن در یک محیط x64 دارای CLR ایی 64 بیتی (دات نت فریم ورک 64 بیتی)، بدون هیچگونه مشکلی اجرا خواهد شد.

سؤال: چرا و چگونه؟!
کامپایلرهای دات نتی (تفاوتی نمی‌کند که چه زبانی مورد استفاده باشد)، کد شما را به IL ترجمه می‌کنند و IL اساسا درکی از پروسسور ندارد. JIT است که در آخرین لحظه در این مورد تصمیم گیری می‌کند.

این نگرانی از کجا حاصل شده است؟
نگارش R2 ویندوز 2008 سرور، فقط 64 بیتی خواهد بود و ویندوز سرور 2008 فعلی، آخرین سروری از مایکروسافت است که هر دو نسخه‌ی 32 بیتی و 64 بیتی را دارد. بنابراین دیر یا زود تمام برنامه نویس‌های ویندوزی "مجبور" خواهند شد دنیای 64 بیتی را تجربه کنند. (البته اگر تاکنون آن‌را تجربه نکرده‌اند)
و البته هنوز یک سری از محیط‌های توسعه، کامپایلر مخصوص 64 بیتی ندارند (مانند دلفی که قرار است در طول سال جاری اولین تجربه‌ی 64 بیتی خود را ارائه دهد)

نکته:
در صفحه‌ی build ویژوال استودیو، شما می‌توانید نوع پلتفرم مورد نظر را نیز تعیین کنید:

پیش فرض آن بر روی Any CPU است و در این حالت کد کامپایل شده‌ی شما بدون مشکل بر روی پلتفرم‌هایی که مشاهده می‌کنید اجرا خواهد شد و تنها پیشنیاز اجرای آن، نصب نسخه‌ی دات نت فریم ورک مخصوص آن پلتفرم است، بدون اینکه نیاز باشد برنامه نویس نگران جزئیات خاصی در مورد خصوصیات ویژه‌ی آن پلتفرم ویژه باشد.

سؤال: اگر کد ما خالص نبود چطور؟ (منظور اینکه 100 درصد دات نتی نبود)

حالت الف) اگر از کامپوننت‌های خارجی استفاده می‌کنید (حتی اگر 100 درصد دات نتی هم باشند) حتما اطمینان حاصل کنید که برای پلتفرم خاصی کامپایل نشده‌اند (همان Any CPU مورد استفاده بوده)، زیرا کد شما که برای تمام CPU ها کامپایل شده، در محیط 64 بیتی، تنها توانایی بارگذاری اسمبلی‌های 64 بیتی را خواهد داشت (64 بیتی رفتار می‌کند) و با مواجه شدن با اسمبلی‌هایی که برای یک پروسسور خاص دیگر کامپایل شده‌اند، با خطای BadImageFormatException خاتمه می‌یابد.

حالت ب) استفاده از API ویندوز یا DLL های غیر دات نتی
باید با هماهنگی با تولید کننده‌ی مربوطه حتما از نگارش 64 بیتی استفاده شود و همچنین برنامه‌ی شما باید توانایی استفاده از اشاره‌گرهای 64 بیتی را داشته باشد. اندازه‌ی نوع داده‌ای IntPtr در یک محیط 32 بیتی 4 است و در یک محیط 64 بیتی 8 خواهد بود (IntPtr.Size). اگر در حین اجرای ترجمه‌ی API یک کتابخانه به اشتباه بجای استفاده از IntPtr از int استفاده شده باشد، ممکن است کد شما در یک محیط 32 بیتی سال‌ها بدون مشکل اجرا شود، اما در اولین اجرای خود در یک محیط 64 بیتی، کرش خواهد کرد. (بدلیل overflow حاصل)

IntPtr به اندازه‌ی کافی هوشمند است تا سایز خودش را مطابق پلتفرم تنظیم کند و مشکل ساز نشود.

مثال:

```
[DllImport("kernel32.dll")]
public static extern void GetSystemInfo([MarshalAs(UnmanagedType.Struct)] ref SYSTEM_INFO
lpSystemInfo);

[StructLayout(LayoutKind.Sequential)]
public struct SYSTEM_INFO
{
 internal _PROCESSOR_INFO_UNION uProcessorInfo;
 public uint dwPageSize;
 public IntPtr lpMinimumApplicationAddress;
 public int lpMaximumApplicationAddress;
 public IntPtr dwActiveProcessorMask;
 public uint dwNumberOfProcessors;
 public uint dwProcessorType;
 public uint dwAllocationGranularity;
 public ushort dwProcessorLevel;
 public ushort dwProcessorRevision;
}

[StructLayout(LayoutKind.Explicit)]
public struct _PROCESSOR_INFO_UNION
{
 [FieldOffset(0)]
 internal uint dwOemId;
 [FieldOffset(0)]
 internal ushort wProcessorArchitecture;
 [FieldOffset(2)]
 internal ushort wReserved;
}
```

در این مثال که از API ویندوز استفاده می‌شود، به اشتباه نوع lpMaximumApplicationAddress به صورت int تعریف شده است (بجای IntPtr). این کد بدون مشکل در یک برنامه‌ی 32 بیتی کار می‌کند اما همین برنامه در یک محیط 64 بیتی یا کرش خواهد کرد یا مقدار lpMaximumApplicationAddress منفی گزارش می‌شود.

عنوان: **زیباتر کد بنویسیم**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۰۵ ۱۲:۳۵:۲۸
آدرس: www.dotnettips.info
برچسب‌ها: Tips

داشتن آگاهی در مورد ساختارهای داده‌ها، الگوریتم‌ها و یا عملگرهای بیتی بسیار عالی است و یا تسلط بر نحوه‌ی کارکرد ابزارهایی مانند SharePoint و امثال آن این روزها ضروری است. اما باید در نظر داشت، کدی که امروز تهیه می‌شود شاید فردا یا ماه دیگر یا چند سال بعد نیاز به تغییر داشته باشد، بنابراین دانش زیبا نوشتن یک قطعه کد که خواندن آن‌را ساده‌تر می‌کند و در آینده افرادی که از آن نگهداری خواهند کرد زیاد "زجر" نخواهند کشید، نیز ضروری می‌باشد. (اگر کامنت‌های سایت را خوانده باشید یکی از دوستان پیغام گذاشته بود، اگر به من بگویند یک میلیون بگیرید و برنامه فعلی را توسعه دهید یا رفع اشکال کنید، حاضرم 10 هزارتومان بگیرم و آن‌را از صفر بنویسم! متاسفانه این یک واقعیت تلخ است که ناشی از عدم خوانا بودن کدهای نوشته شده می‌باشد.)

در ادامه یک سری از اصول زیبا نویسی کدها را بررسی خواهیم کرد.

1- سعی کنید میزان تو در تو بودن کدهای خود را محدود کنید.

لطفا به مثال زیر دقت نمائید:

```
void SetA()
{
  if(a == b)
  {
     foreach(C c in cs)
     {
        if(c == d)
        {
           a = c;
        }
     }
  }
}
```

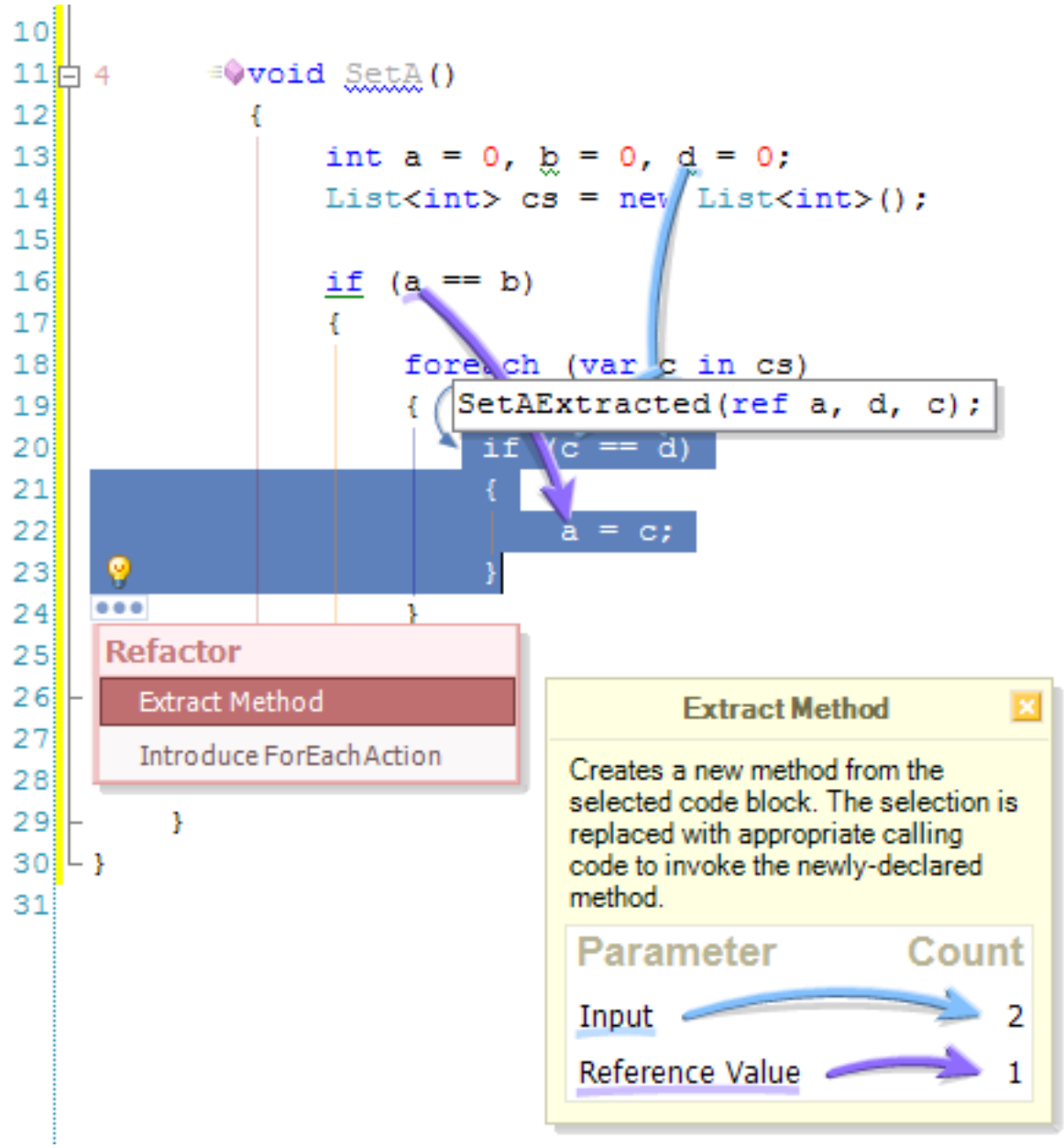
توصیه شده است فقط یک سطح تو در تو بودن را در یک تابع لحاظ کنید. تابع فوق 4 سطح تو رفتگی را نمایش می‌دهد (برای رسیدن به a=c باید چهار بار از tab استفاده کنید). برای کاهش این تعداد سطح می‌توان به صورت زیر عمل کرد:

```
void SetA()
{
   if(a != b)
       return;

  foreach(C c in cs)
       a = GetValueOfA(c);
}

TypeOfA GetValueOfA(C c)
{
  if(c == d)
      return c;

  return a;
}
```

خواناتر نشد؟!

افزونه‌های CodeRush و refactor pro مجموعه‌ی DevExpress از لحاظ مباحث refactoring در ویژوال استودیو حرف اول را می‌زنند. فقط کافی است برای مثال قطعه کد if داخلی را انتخاب کنید، بلافاصله سه نقطه زیر آن ظاهر شده و با کلیک بر روی آن

امکان استخراج یک تابع از آن‌را برای شما به سرعت فراهم خواهد کرد.



مثالی دیگر:

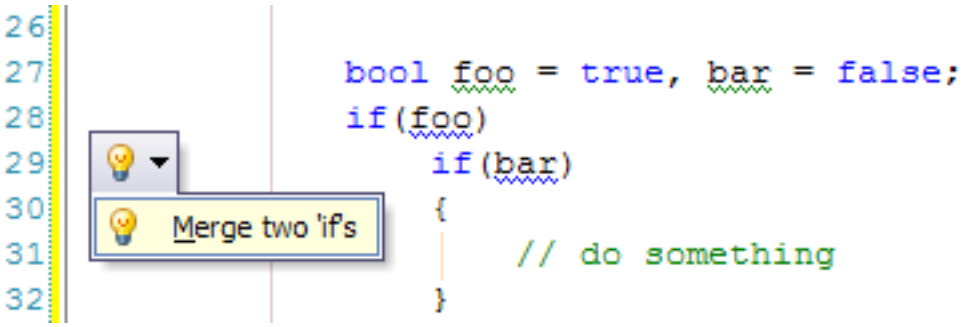
```
if (foo) {
if (bar) {
  // do something
}
}
```

به صورت زیر هم قابل نوشتن است (جهت کاهش میزان nesting):

```
if (foo && bar) {
// do something
```

```
}
```

افزونه‌ی Resharper امکان merge خودکار این نوع if ها را به همراه دارد.



و یا یک مثال دیگر:

میزان تو در تو بودن این تابع جاوا اسکریپتی را ملاحظه نمائید:

```
function findShape(flags, point, attribute, list) {
   if(!findShapePoints(flags, point, attribute)) {
       if(!doFindShapePoints(flags, point, attribute)) {
           if(!findInShape(flags, point, attribute)) {
               if(!findFromGuide(flags,point) {
                   if(list.count() > 0 && flags == 1) {
                         doSomething();
                   }
               }
           }
       }
   }
}
```

آن‌را به صورت زیر هم می‌توان نوشت با همان کارآیی اما بسیار خواناتر:

```
function findShape(flags, point, attribute, list) {
   if(findShapePoints(flags, point, attribute)) {
       return;
   }

   if(doFindShapePoints(flags, point, attribute)) {
       return;
   }

   if(findInShape(flags, point, attribute)) {
       return;
   }

   if(findFromGuide(flags,point) {
       return;
   }

   if (!(list.count() > 0 && flags == 1)) {
       return;
   }

   doSomething();
}
```

-2 نام‌های با معنایی را برای متغیرها وتوابع خود انتخاب کنید.
با وجود پیشرفت‌های زیادی که در طراحی و پیاده سازی IDE ها صورت گرفته و با بودن ابزارهای تکمیل سازی خودکار متن تایپ شده در آن‌ها، این روزها استفاده از نام‌های بلند برای توابع یا متغیرها مشکل ساز نیست و وقت زیادی را تلف نخواهد کرد. برای مثال به نظر شما اگر پس از یک سال به کدهای زیر نگاه کنید کدامیک خود توضیح دهنده‌تر خواهند بود (بدون مراجعه به مستندات موجود)؟

```
void UpdateBankAccountTransactionListWithYesterdaysTransactions()
//or?
void UpdateTransactions()
```

-3 تنها زمانی از کامنت‌ها استفاده کنید که لازم هستند.
اگر مورد 2 را رعایت کرده باشید، کمتر به نوشتن کامنت نیاز خواهد بود. از توضیح موارد بدیهی خودداری کنید، زیرا آن‌ها بیشتر سبب اتلاف وقت خواهند شد تا کمک به افراد دیگر یا حتی خود شما. همچنین هیچگاه قطعه کدی را که به آن نیاز ندارید به صورت کامنت شده به مخزن کد در یک سیستم کنترل نگارش ارسال نکنید.

```
//function thisReallyHandyFunction() {
//      someMagic();
//      someMoreMagic();
//      magicNumber = evenMoreMagic();
//      return magicNumber;
//}
```

زمانیکه از ورژن کنترل استفاده می‌کنید نیازی به کامنت کردن قسمتی از کد که شاید در آینده قرار است مجددا به آن بازگشت نمود، نیست. این نوع سطرها باید از کد شما حذف شوند. تمام سیستم‌های ورژن کنترل امکان revert و بازگشت به قبل را دارند و اساسا این یکی از دلایلی است که از آن‌ها استفاده می‌شود!
به صورت خلاصه جهت نگهداری سوابق کدهای قدیمی باید از سورس کنترل استفاده کرد و نه به صورت کامنت قرار دادن آن‌ها.

از کامنت‌های نوع زیر پرهیز کنید که بیشتر سبب رژه رفتن روی اعصاب خواننده می‌شود تا کمک به او! (خواننده را بی‌سواد فرض نکنید)

```
// Get the student's id
thisId = student.getId();
```

کامنت زیر بی معنی است!

```
// TODO: This is too bad. FIX IT!
```

اگر شخص دیگری به آن مراجعه کند نمی‌داند که منظور چیست و دقیقا مشکل کجاست. شبیه به افرادی که به فوروم‌ها مراجعه می‌کنند و می‌گویند برنامه کار نمی کند و همین! طرف مقابل علم غیب ندارد که مشکل شما را حدس بزند! به توضیحات بیشتری نیاز است.

-4 عدم استفاده از عبارات شرطی بی‌مورد هنگام بازگشت دادن یک مقدار bool:
مثال زیر را درنظر بگیرید:

```
if (foo>bar) {
```

```
return true;
} else {
return false;
}
```

آن‌را به صورت زیر هم می‌توان نوشت:

```
return foo>bar;
```

5- استفاده از متغیرهای بی مورد:

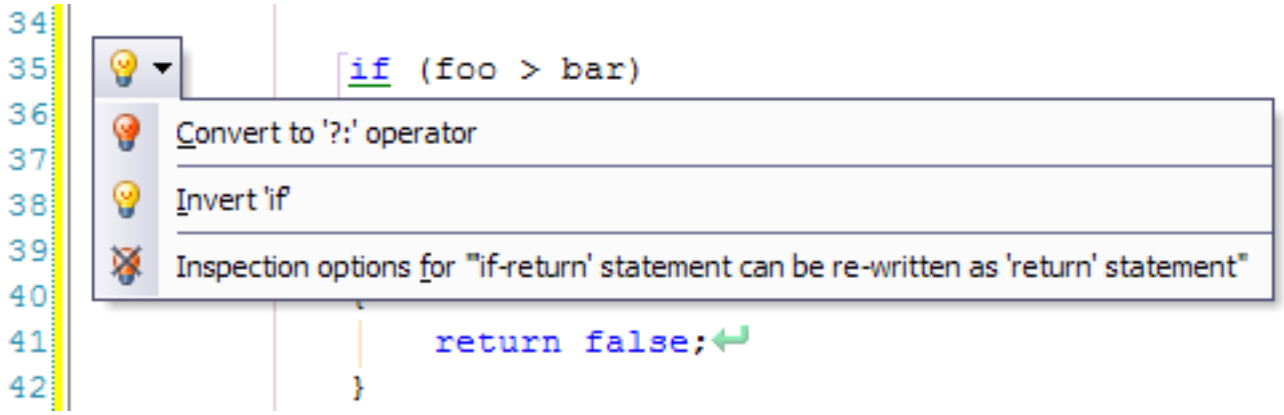
برای مثال:

```
Something something = new Something(foo);
return something;
```

که می‌شود آن را به صورت زیر هم نوشت:

```
return new Something(foo);
```

البته یکی از خاصیت‌های استفاده از افزونه‌ی Resharper ویژوال استودیو، گوشزد کردن و اصلاح خودکار موارد 4 و 5 است.



6- در نگارش‌های جدید دات نت فریم ورک استفاده از ArrayList منسوخ شده است. بجای آن بهتر است از لیست‌های جنریک استفاده شود. کدی که در آن از ArrayList استفاده می‌شود طعم دات نت فریم ورک 1 را می‌دهد!

7- لطفا بین خطوط فاصله ایجاد کنید. ایجاد فواصل مجانی است!

دو تابع جاوا اسکریپتی زیر را (که در حقیقت یک تابع هستند) در نظر بگیرید:

```
function getSomeAngle() {
   // Some code here then
   radAngle1 = Math.atan(slope(center, point1));
```

```
   radAngle2 = Math.atan(slope(center, point2));
   firstAngle = getStartAngle(radAngle1, point1, center);
   secondAngle = getStartAngle(radAngle2, point2, center);
   radAngle1 = degreesToRadians(firstAngle);
   radAngle2 = degreesToRadians(secondAngle);
   baseRadius = distance(point, center);
   radius = baseRadius + (lines * y);
   p1["x"] = roundValue(radius * Math.cos(radAngle1) + center["x"]);
   p1["y"] = roundValue(radius * Math.sin(radAngle1) + center["y"]);
   pt2["x"] = roundValue(radius * Math.cos(radAngle2) + center["y"]);
   pt2["y"] = roundValue(radius * Math.sin(radAngle2) + center["y");
   // Now some more code
}
```

```
function getSomeAngle() {
   // Some code here then
   radAngle1 = Math.atan(slope(center, point1));
   radAngle2 = Math.atan(slope(center, point2));

   firstAngle = getStartAngle(radAngle1, point1, center);
   secondAngle = getStartAngle(radAngle2, point2, center);

   radAngle1 = degreesToRadians(firstAngle);
   radAngle2 = degreesToRadians(secondAngle);

   baseRadius = distance(point, center);
   radius = baseRadius + (lines * y);

   p1["x"] = roundValue(radius * Math.cos(radAngle1) + center["x"]);
   p1["y"] = roundValue(radius * Math.sin(radAngle1) + center["y"]);

   pt2["x"] = roundValue(radius * Math.cos(radAngle2) + center["y"]);
   pt2["y"] = roundValue(radius * Math.sin(radAngle2) + center["y");
   // Now some more code

}
```

کدامیک خواناتر است؟

استفاده از فاصله بین خطوط در تابع دوم باعث بالا رفتن خوانایی آن شده است و این طور به نظر می‌رسد که سطرهایی با عملکرد مشابه در یک گروه کنار هم قرار گرفته‌اند.

-8 توابع خود را کوتاه کنید.

یک تابع نباید بیشتر از 50 سطر باشد (البته در این مورد بین علما اختلاف هست!). اگر بیشتر شد بدون شک نیاز به refactoring داشته و باید به چند قسمت تقسیم شود تا خوانایی کد افزایش یابد.

به صورت خلاصه یک تابع فقط باید یک کار را انجام دهد و باید بتوان عملکرد آن‌را در طی یک جمله توضیح داد.

-9 از اعداد جادویی در کدهای خود استفاده نکنید!

کد زیر هیچ معنایی ندارد!

```
if(mode == 3){ ... }
else if(mode == 4) { ... }
```

بجای این اعداد بی مفهوم باید از enum استفاده کرد:

```
if(mode == MyEnum.ShowAllUsers) { ... }
else if(mode == MyEnum.ShowOnlyActiveUsers) { ... }
```

-10 توابع شما نباید تعداد پارامتر زیادی داشته باشند

اگر نیاز به تعداد زیادی پارامتر ورودی وجود داشت (بیش از 6 مورد) از struct و یا کلاس جهت معرفی آن‌ها استفاده کنید.

نظرات خوانندگان

نویسنده: nima
تاریخ: ۱۳۸۷/۱۲/۰۵ ۱۴:۳۱:۰۰

سلام استاد نصیری
ممنون از مطلب مفیدتون
این افزونه های CodeRush رو براتون امکان داره جایی آپلود کنین؟
سپاسگزار

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۰۵ ۱۵:۱۰:۰۰

سلام،
نگارش کامل اون‌ها رو در فوروم board4all دات cz می‌تونید پیدا کنید ;)
تعداد ایرانی‌هایی را که در اون سایت می‌تونید پیدا کنید باعث تعجب شما خواهد شد!

نویسنده: Sirasad
تاریخ: ۱۳۸۷/۱۲/۰۵ ۲۰:۴۰:۰۰

ممنون آقای نصیری ...

فکر کنم یکی دیگر ار راههای استفاده مجدد از کد ها نوشتن Document برای کد ها و اجزای سیستم می باشد . من خودم به شخصه در شرکت با برنامه نویس هام تست کردم , خیلی خوب نتیجه داد .

موفق باشید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۰۵ ۲۱:۵۶:۰۰

بله مسلما همینطور است. مطلب فوق فقط بیانگر مواردی بود که از کامنت سوء استفاده شده است.

نویسنده: افشار محبی
تاریخ: ۱۳۸۷/۱۲/۰۷ ۰۸:۲۶:۰۰

خیلی خوب و کاربردی بود. البته شخصا بعضی وقت‌ها مجبور هستم موارد ۴ و ۵ را به خاطر راحتی debug رعایت نکنم.

نویسنده: hajloo
تاریخ: ۱۳۸۷/۱۲/۰۷ ۱۲:۱۳:۰۰

واقعا مفید بود. فق ای کاش لینک ابزارها رو هم برای دانلود می زاشتید.
در ضمن در صورت امکان یک مطلب در مورد استفاده بهینه از خود ویژوال استودیو بنویسید. مثلا اینکه چطور از پنجره های موجود استفاده بهتری بکنیم.
با سپاس فراوان

نویسنده: peyman naji
تاریخ: ۱۳۸۸/۰۷/۱۱ ۲۰:۳۴:۰۱

مهندس ممنون بسیار عالی بود

عنوان: **غلط یاب فارسی در دات نت با استفاده از امکانات open office**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۱۸ ۱۶:۵۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

مدتی است که محصور کننده‌ای سورس باز برای امکانات غلط یاب مجموعه‌ی open office در سایت code project ارائه شده است:

NHunspell - Hunspell for the .NET platform

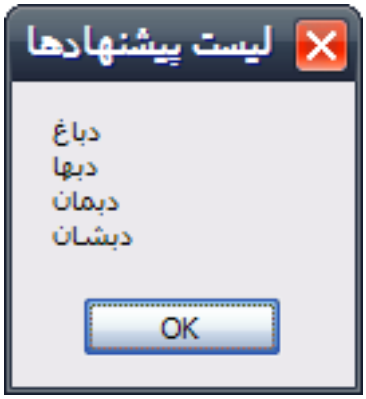
دریافت آخرین نسخه‌ی آن از source forge

خوشبختانه کتابخانه‌ی واژه‌های فارسی هم برای اپن آفیس مهیا است.

دریافت

پس از دریافت کتابخانه‌ی فوق و همچنین فایل‌های مربوط به زبان فارسی، فقط کافی است ارجاعی به اسمبلی NHunspell.dll در برنامه اضافه شود و سپس یک مثال ساده در مورد استفاده از آن به صورت زیر خواهد بود:

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;
using NHunspell;

namespace testWinForms87
{
   class CSpellCheck
   {
       public static void Test()
       {
           using (Hunspell hunspell = new Hunspell(@"fa_ir.aff", @"fa_ir.dic"))
           {
               bool correct = hunspell.Spell("دباق");
               if (correct)
                   MessageBox.Show("مشکلی نیست!");
               else
               {
                   List<string> suggestions = hunspell.Suggest("دباق");
                   string result = string.Empty;
                   foreach (string suggestion in suggestions)
                   {
                       result += suggestion + Environment.NewLine;
                   }

                   if (result != string.Empty)
                       MessageBox.Show(result,"لیست پیشنهادها");
               }
           }
       }
   }
}
```


لیست پیشنهادها
دباغ
دبها
دبمان
دبشان
OK

نظرات خوانندگان

نویسنده: peyman
تاریخ: ۱۳۸۸/۰۱/۱۱ ۱۸:۵۲:۰۰

خیلی جالب بود از لطف شما ممنون ...
موفق باشید .

عنوان: **اشتباهات متداول برنامه‌نویس‌های دات نت**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۱۹ ۲۰:۱۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

اشتباه 1:
استفاده از

```
throw ex;
```

بجای استفاده از

```
throw;
```

در حالت اول، تمام stack trace موجود تا نقطه‌ی فراخوانی دستور ذکر شده پاک خواهند شد، اما در حالت دوم stack trace حفظ شده و دیباگ کردن کد را ساده می‌کند.

اشتباه 2:
درک اشتباه عملکرد متد replace :

```
string s = "Take this out";
s.Replace("this", "that");  //wrong
```

بجای استفاده از :

```
s = s.Replace("this", "that");  //correct
```

اگر از fxCop استفاده کنید، اینگونه خطاها را (عدم استفاده از مقدار بازگشتی) گوشزد می‌کند.

اشتباه 3:
استفاده‌ی بی دقت از متغیرهای استاتیک در یک برنامه وب. دو مثال زیر را در نظر بگیرید:

```
public static string GetCookieName(Cookie c)
{
 return c.Name;
}
```

```
static List<string> cookieList = new List<string>();
public static void AddToCookieList(Cookie c)
{
 cookieList.Add(c.Name);

}
```

برنامه‌های وب ذاتا چند ریسمانی هستند و زمانیکه یک متغیر را از نوع استاتیک تعریف می‌کنید، این متغیر، هنگام مراجعه‌ی کاربران بین آن‌ها به اشتراک گذاشته می‌شود و امکان تخریب یا استفاده‌ی ناصحیح از مقادیر آن‌ها وجود خواهد داشت. در حالت اول نیازی

به مباحث همزمانی و قفل کردن منابع نیست زیرا متغیری که در متد استفاده می‌شود، thread safe است اما cookieList در مثال دوم خیر و حتما هنگام استفاده از آن باید مباحث قفل کردن منابع را درنظر داشت. (حتی اگر برنامه‌ی شما از نوع وبی هم نیست اما چند ریسمانی است این مطلب باز هم صادق می‌باشد)

اشتباه 4:

مقابله با خطاها به شکلی نادرست: (اصطلاحا خفه کردن خطاها!)

```
try
{
    //something
 File.Delete(blah);
}
catch{}
```

در این حالت اگر خطایی رخ دهد کاربر متوجه نخواهد شد دقیقا مشکلی وجود داشته یا خیر و علت آن چیست.
هرچند گاهی از اوقات اگر خطای حاصل برای ما اهمیتی نداشت می‌توان از آن استفاده نمود، در غیراینصورت باید حتما از این روش پرهیز کرد.

اشتباه 5:

ارائه‌ی برنامه‌های ASP.Net با گزینه‌ی پیش فرض Debug=true در web.config که پیشتر در مورد آن در این سایت بحث شده است .

اشتباه 6:

عدم استفاده از امکانات ویژه‌ی دات نت فریم ورک هنگام کار با رشته‌ها:

```
string s = "This ";
s += "is ";
s += "not ";
s += "the ";
s += "best ";
s += "way.";
```

```
StringBuilder sb = new StringBuilder();
sb.Append("This ");
sb.Append("is ");
sb.Append("much ");
sb.Append("better. ");
```

زمانیکه نیاز به کنار هم قرار دادن رشته‌های مختلف وجود داشت و تعداد آن‌ها نیز زیاد بود، مثال دوم توصیه می‌شود.
البته در مثال فوق که تعداد کمی رشته قرار است با هم جمع شوند، کامپایلر به اندازه‌ی کافی هوشمند خواهد بود که تمام آن‌ها را کنار هم قرار دهد و تفاوتی در کارآیی احساس نشود، حتی روش اول سریع‌تر از روش دوم خواهد بود، اما زمان استفاده از هزاران رشته، این تفاوت محسوس است.

اشتباه 7:

عدم استفاده از عبارت using هنگام استفاده از اشیایی از نوع Idisposable . مثالی در این مورد.

```
using (StreamReader reader=new StreamReader(file))
{
 //your code here

}
```

اشتباه 8:
فراموش کردن بررسی نال بودن یک شیء هنگام استفاده از آن.

```
string val = Session["xyz"].ToString();
```

این نوع کد نویسی یکی از اشتباهات متداول تمامی تازه واردان به ASP.Net است. حتما باید پیش از استفاده از متد ToString بررسی شود که آیا این سشن نال است یا نه. در غیراینصورت حاصل کار فقط یک exception خواهد بود. (استفاده از افزونه‌ی ری شارپر در این موارد کمک بزرگی است، زیرا به محض قرار گرفتن مکان نما روی شیءایی که احتمال نال بودن آن میسر است، یک راهنما را به شما ارائه خواهد کرد)

اشتباه 9:
بازگشت دادن یک property عمومی از نوع لیست‌های جنریک.
با توجه به اینکه این نوع لیست‌ها فقط خواندنی نیستند و امکان دستکاری اطلاعات آن توسط فراخوان وجود دارد، توصیه می‌شود از نوع جنریک IEnumerable استفاده شود. همچنین توصیه شده است هنگام انتخاب نوع پارامترهای ورودی یک متد نیز به این مورد دقت شود.

نظرات خوانندگان

نویسنده: کورش اسلام زاده
تاریخ: ۱۳۸۷/۱۲/۱۹ ۲۱:۰۰:۰۰

سلام

خبرخوان (تکست و گرافیکی)، لیست و فید وبلاگ‌های برنامه‌نویسی ارائه شده است که وبلاگ شما نیز جز آنها قرار گرفته است.

http://persianbloggers.blogspot.com/2009/03/programming-p.html

پرشین بلاگرز شما را به بازدید و استفاده از این خبرخوان و 23 خبرخوان تخصصی دیگر موجود دعوت می‌کند.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۱۹ ۲۱:۳۶:۰۰

با تشکر.
البته در تایتل لینک هم قرار گرفته که فید نهایی را ناخوانا کرده. لینک و تایتل باید مجزا تعریف شوند.
http://feeds2.feedburner.com/PersianBloggers/Programming

نویسنده: کورش اسلام زاده
تاریخ: ۱۳۸۷/۱۲/۲۰ ۱۹:۵۴:۰۰

سلام
ممنون از نظرتان. آیا شما فید را در گوگل ریدر یا هر خوراک خوان دیگر هم ملاحظه کرده اید؟ حرف شما درست که بصورت استاندارد باید لینک و تایتل جدا باشند اما هدف مشاهده در خوراک خوان است که گمان میکنم با روش موجود برتری های بیشتری هم بر فید مرسوم داشته باشد.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۲۰ ۲۰:۱۱:۰۰

سلام،
لطفا یکبار لینک فید فوق را در فایرفاکس و یکبار هم با IE ملاحظه کنید. متن تایتل‌ها به هم ریخته است (مخلوطی است از لینک مطلب و تایتل مطلب).

نویسنده: کورش اسلام زاده
تاریخ: ۱۳۸۷/۱۲/۲۰ ۲۰:۵۶:۰۰

سلام

یک مقایسه. تصویر بدون به هم ریختگی هم در ie و هم فایرفاکس
http://tinyurl.com/ahkw5u
http://tinyurl.com/asohbo
و تصویری که در ie کاملن سالم است و در فایرفاکس فقط یک کلمه آن جلو و عقب شده است
http://tinyurl.com/cf2yos

در هر حال در خوراک خوان عنوان با آبی مشخص شده است و در موارد به هم ریختگی در هر صورت مشخص است. در ضمن در مواردی که انگلیسی و فارسی در عنوان - با هم استفاده شده است در بعضی موارد مشکل بروز میکند که گمان نمیکنم خود فید اصلی هم همیشه و با یک رویه نشان داده شود. موارد بسیاری را شاهدیم که نام وبلاگ یا نویسنده گاهی در راست و گاهی در

چپ است و ...

به نظر من نمیرسد مشکل حادی در این رابطه وجود داشته باشد و شاید هم بنده کاملن مقصود شما را متوجه نشده ام که ممنون میشوم توضیح بدهید. در عین حال یک پست در رابطه با فید پرشین بلاگرز را در حال آماده کردن هستم که بزودی پست میکنم. در این پست مقایسه بیشتری به عمل خواهد آمد.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۲۰ ۲۱:۳۴:۰۰

به لطف isp ها، تصاویر بلاگر فیلتر شده ...

http://www.tinypic.info/files/9m8151hvn499d1megy21.png

نویسنده: Kourosh
تاریخ: ۱۳۸۷/۱۲/۲۰ ۲۱:۴۵:۰۰

واقعن اینجا راذ من دیوانه می‌شوم. واقعن نمیدانم کجا باید آپلود کنم.....

نویسنده: Kourosh
تاریخ: ۱۳۸۷/۱۲/۲۰ ۲۱:۴۶:۰۰

در این رابطه

http://weblogcrawler.blogspot.com/2009/03/feed-in-poersian-bloggers.html

عنوان: **پیش بینی وضعیت دنیای برنامه نویسی در 5 سال آینده**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۰۶ ۱۲:۲۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

در 5 سال آینده مواردی که در ادمه برشمرده خواهند شد، نقش بسیار مهمی را در دنیای برنامه نویسی و جهت گیری‌های آن ایفا خواهند کرد (برای مثال اگر برای شما این سؤال مطرح است که هدف از REST services ، WCF ، سیلورلایت 3 و غیره چیست، این مقاله‌ی کوتاه را مطالعه نمائید) :

الف) Object Relational Mapping
ORM یکی از بازیگرهای واضح خواهد بود. خصوصا پروژه‌ای مانند NHibernate Fluent با ویژگی‌های زیر:
سابقه‌ای 10 ساله (قسمت عمده‌ای از این سابقه به دنیای جاوا بر می‌گردد)
امکان استفاده از انواع و اقسام دیتابیس‌ها توسط آن
پشتیبانی از Linq
و ...

ب) نرم افزار به عنوان سرویس (Software as a Service یا SaaS)
نرم افزار به عنوان سرویس یک مفهوم تجاری است که در آن مصرف کننده بر اساس نیازهایش هزینه‌ی یک نرم افزار را خواهد پرداخت. بر این اساس برنامه نویسی در زمینه‌های طراحی و مدیریت دست خوش تغییرات عمده‌ای می‌شود. شاید نیازی به ذکر نباشد که حتی مایکروسافت نیز در حال برنامه ریزی برای این نوع از توسعه است.
پرداختن به SaaS نیازمند یک سری از ویژگی‌ها است:
سادگی توسعه و دستیابی: در این مدل تجاری، استفاده و دسترسی به نرم افزار مورد نظر باید بسیار ساده باشد. بر این اساس برنامه‌های تحت وب، یا برنامه‌های هاست شده توسط مرورگرها (مانند سیلورلایت) محبوبیت بیش از پیشی را خواهند یافت.
قابلیت تنظیم و ماژولار بودن برنامه‌ها: در این مدل نیاز است تا کاربر تنها هزینه‌ی ماژول‌هایی را بپردازد که به آن‌ها نیاز دارد و این امر سبب بازنگری در طراحی و توسعه‌ی برنامه‌های موجود خواهد شد.
نیاز به زیر ساخت بهینه و سریعی خواهد بود: از آنجائیکه کاربران بسیار ساده می‌توانند از یک برنامه به برنامه و شرکتی دیگر رجوع کنند، برای بقا باید جنگید! نیاز به زیر ساخت‌هایی وجود خواهد داشت که توسط آن‌ها بتوان نیازهای کاربران را در حداقل زمان ممکن برآورده کرد و این موارد نیاز به آموختن یکی از فریم ورک‌های مطرح موجود را خواهد داشت به همراه آموختن مباحث مدیریت پروژه، آشنایی با آزمون‌های واحد، کنترل کیفیت ، یکپارچگی مداوم و امثال آن.

ج) پردازش ابری
پردازش ابری شبیه به آن‌چیزی که مایکروسافت Azure ارائه می‌دهد، نیز یکی از نتایج مفهوم تجاری SaaS است. تمرکز پردازش ابری بر روی ارائه‌ی وب سرورها، مکان‌های ذخیره داده و امثال آن است. به این صورت شما دیگر درگیر تهیه و پرداخت هزینه جهت راه اندازی دیتاسنتر ویژه‌ی خود نخواهید شد و بسیاری از هزینه‌های شما کاهش خواهند یافت. بهره برداری تجاری گسترده از این روش با توجه به توسعه‌ی فریم ورک‌های ویژه‌ی این نوع پردازش‌ها، آموزش و غیره ، بین سال‌های 2010 و 2015 شروع خواهد شد.

د) اجرای موازی
پردازش ابری اثرات خاص خودش را بر روی دنیای نرم افزار و برنامه نویسی خواهد گذاشت. این طبیعت توزیع شده سبب خواهد شد که در آینده از برنامه نویسی‌های چند ریسمانی و مسایل همزمانی حاصل از آن‌ها بیشتر بشنوید و نهایتا معماری برنامه‌ها به سمت استفاده از روش‌های زیر سوق خواهند یافت:

```
REST services;
Message-based distributed architectures, i.e.: see NServiceBus, Mass Transit or Rhino Service Bus
```

ه) برنامه‌های غنی وب یا Rich Internet Applications

Rich Internet Applications یا RIA نقش مهمی را در SaaS بازی خواهند کرد و هدفگیری مایکروسافت در این باره ارائه Silverlight 3.0 و Microsoft .NET RIA Services است. هر چند این موارد راه طولانی (یکی دو ساله) را در پیش خواهند داشت تا به حد استانداردهای لازم برسند اما حرکت‌های مهمی در این زمینه به شمار می‌روند.

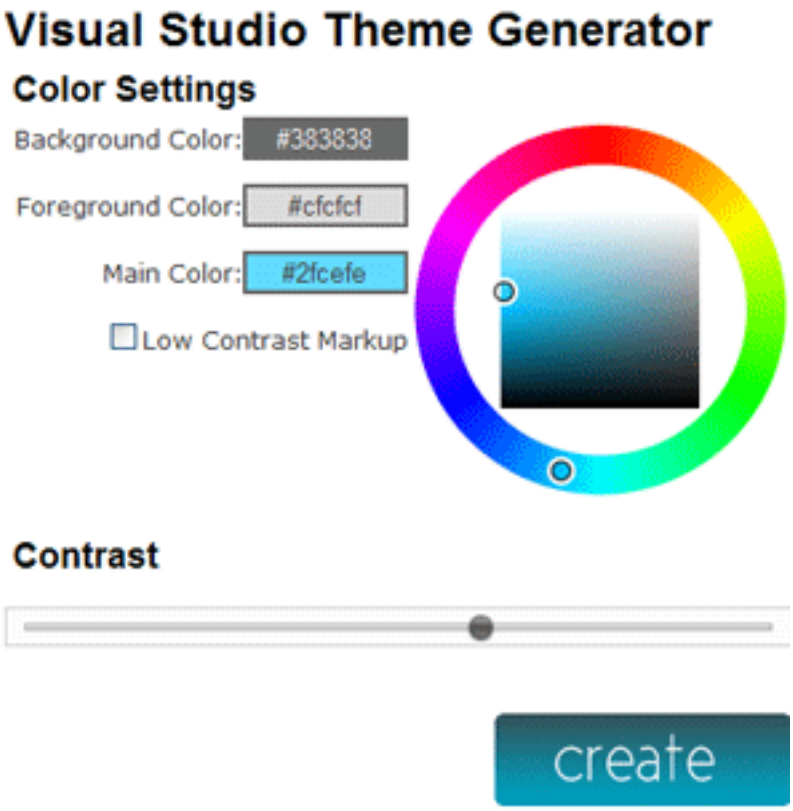
برداشتی آزاد از Development in 5 Years Would be Affected by

عنوان: **ایجاد رنگ و روی سفارشی برای ادیتور Visual Studio**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۰۷ ۱۲:۱۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

برنامه‌ی آنلاینی جهت تولید یک theme سفارشی جدید برای ویژوال استودیو ایجاد شده است که در آدرس زیر قابل استفاده است:

Visual Studio Theme Generator

ابتدا تنظیمات رنگ خود را انتخاب کنید، سپس بر روی دکمه refresh کلیک نمائید تا نمونه‌ای از نتیجه‌ی کار را بتوان مشاهده کرد و سپس بر روی دکمه create کلیک کنید.



و برای استفاده از فایل تولید شده از طریق منوی tools گزینه‌ی Import and Export Settings اقدام کنید.

عنوان: **مرجع سریع VSTT**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۱۲ ۱۱:۱۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

مرجع سریع Visual Studio Team Test 2008 (یکی از ویژگی‌های Visual Studio Team System 2008) را در طی 83 صفحه می‌توانید از آدرس زیر دریافت نمائید. بسیاری از سؤالات شما در مورد آزمایشات واحد و امثال آن در این راهنمای سریع توضیح داده شده‌اند.

VSTT 2008 Quick Reference Guide

این مجموعه موارد زیر را شامل می‌شود:

```
SETUP CONSIDERATIONS
WEB TEST CONSIDERATIONS
WEB SERVICE TEST CONSIDERATIONS
UNIT TEST CONSIDERATIONS
LOAD TEST CONSIDERATIONS
LOAD TEST RIG CONSIDERATION
PERFORMANCE DATA COLLECTION AND USAGE
LOAD TEST RESULTS STORE INFORMATION
TEST CUSTOMIZATION
ITEMS CHANGED OR FIXED IN VSTS 2008 SP1
GENERAL COMMANDS AND TRICKS
```

عنوان: مقایسه توانایی پردازش CSS توسط مرورگرهای مختلف
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۱۲ ۱۸:۳۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

CSS 2.1 selectors

Specification.

Contents of this table

See also the key to my compatibility tables.

Selector	IE 5.5	IE 6	IE 7	IE8 as IE7	IE8 as IE8	FF 2	FF 3.0	FF 3.1b	Saf 3.0 Win	Saf 3.1 Win	Saf 4.0b Win	Chrome 1	Chrome 2	Opera 9.62	Opera 10a	Konqueror 3.5.7
* selector Selects all elements	yes					yes			yes			yes		yes		yes
> selector Selects an element that is a child of another element	no	yes				yes			yes			yes		yes		yes
+ selector Selects an element that is a next sibling of another element	no	static		almost	static		yes		static		yes	static	yes	yes		yes

Static: the browser doesn't correctly update the styles of the original element when another element is placed before it.

IE8 as IE8 updates correctly, but only after you remove the focus from the test link.

CSS contents and browser compatibility

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۱/۱۳ ۱۷:۰۷:۰۰

سلام آقای نصیری.
وبلاگ بسیار جالب، آموزنده و مفیدی دارید.
میخواستم بدانم قالب وبلاگتان را از کجا تهیه کرده‌اید و اینکه چند قالب و وبجت جالب و مفید برای بلاگر به من معرفی نمایید.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۱۳ ۱۷:۰۹:۰۰

سلام،
برای تهیه قالب‌های فارسی بلاگر می‌تونید به سایت زیر مراجعه کنید:
http://baghbanbashi.blogspot.com

عنوان: **لیست شماره نگارش‌های دات نت فریم ورک تا این تاریخ**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۲۶ ۱۴:۱۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

از زمانیکه دات نت فریم ورک ارائه شده (حدودا 8 سال یا بیشتر اگر بتای آن‌را هم به حساب بیاوریم به سال 2000 بر می‌گردد)، نگارش‌های متفاوتی تا به امروز در اختیار عموم قرار گرفته اند.
جدول زیر این موارد را تا این تاریخ لیست کرده و شماره نگارش دقیق آن‌ها را نیز بر می‌شمارد:

Actual version	.NET version
3.5.30729.1	3.5 SP1
3.5.21022.8	3.5
3.0.4506.2152	3.0 SP2
3.0.4506.648	3.0 SP1
3.0.4506.30	3.0
2.0.50727.3053	2.0 SP2
2.0.50727.1433	2.0 SP1
2.0.50727.42	2.0
1.1.4322.2032	1.1 SP1
1.1.4322.2300	1.1 SP1 (in 32 bit version of Windows 2003)
1.1.4322.573	1.1
1.0.3705.6018	1.0 SP3
1.0.3705.288	1.0 SP2
1.0.3705.209	1.0 SP1

Actual version	.NET version
1.0.3705.0	1.0

برای بدست آوردن شماره نگارش‌های نصب شده بر روی یک کامپیوتر متاسفانه راه ساده‌ای وجود ندارد. امکاناتی هم که خود دات نت فریم ورک به صورت ذاتی ارائه می‌دهد به صورت زیر است:

```
class NetVersion
  {
      public static string Version
      {
          get
          {
              return Environment.Version + "\n" +
                     Environment.OSVersion;
          }
      }
  }
```

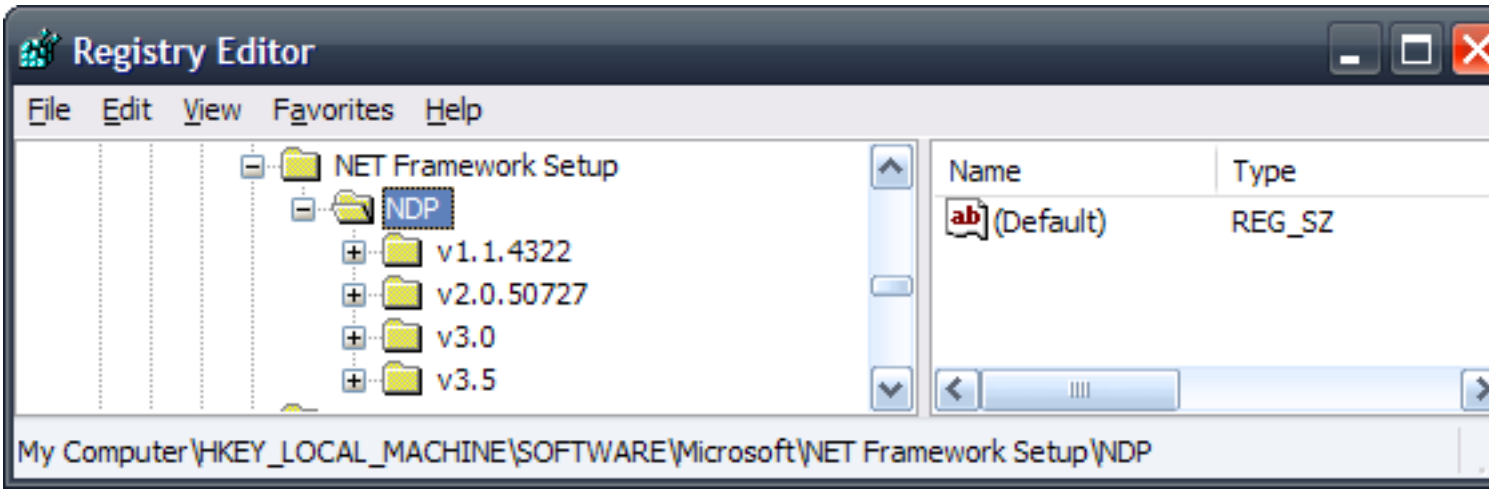
که خروجی آن فقط آخرین نگارش CLR را شامل می‌شود.
برای مثال روی ویندوز اکس پی سرویس پک 3 با دات نت فریم ورک سه و نیم، سرویس پک یک خواهیم داشت:

```
2.0.50727.3053
Microsoft Windows NT 5.1.2600 Service Pack 3
```

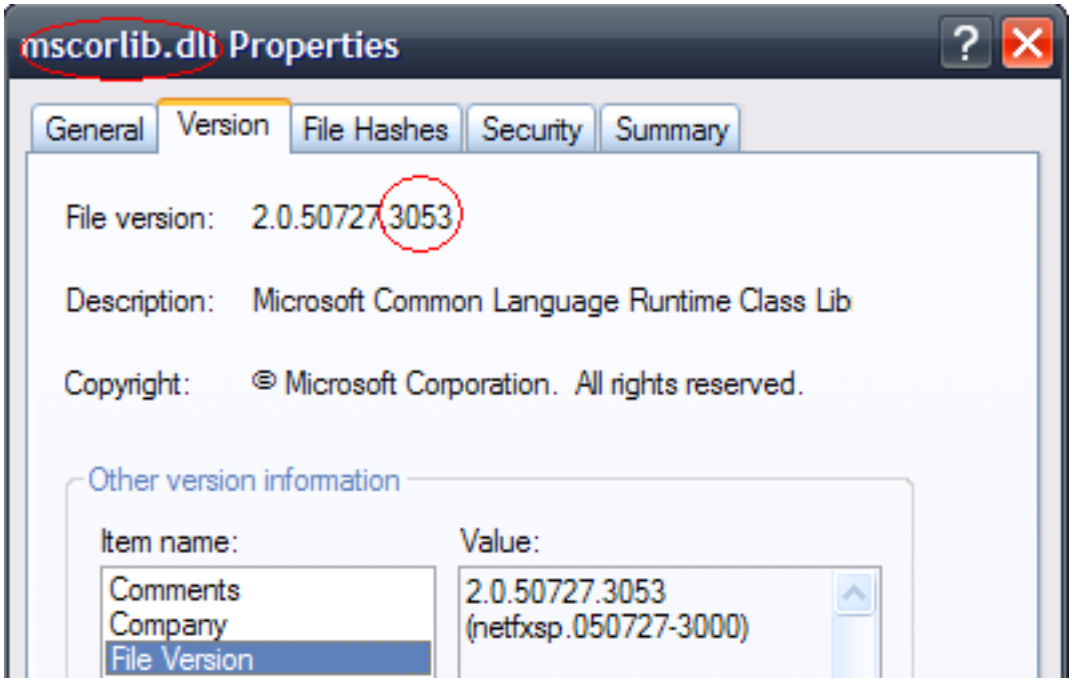
که عدد نگارش ارائه شده با دات نت فریم ورک 2 سرویس پک 2 تطابق دارد. به عبارت دیگر آخرین نگارش CLR هنوز همان 2 است و موارد دیگر (مثل wf و wcf و ...) فقط یک سری افزونه برای این هسته به شمار می‌روند.
خوبی این روش هم این است که اگر در یک هاست اینترنتی قصد داشتید شماره نگارش دات نت فریم ورک سرور را بررسی کنید، بدون مشکل پاسخ خواهد داد. برای مثال اگر به دات نت فریم ورک 2 سرویس پک 2 رسیدید، یعنی دات نت فریم ورک سه و نیم، سرویس پک یک حتما روی سرور نصب است، چون این دو با هم ارائه شده‌اند و به صورت مجزا ارائه نشده‌اند.

برای بدست آوردن لیست دات نت فریم ورک‌های مختلف باید به رجیستری ویندوز مراجعه کرد. مسیرهای:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\NET Framework Setup\NDP
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\User Agent\Post
Platform
```



این روش یا حتی لیست کردن فولدرهای نصب شد در مسیر C:\WINDOWS\Microsoft.NET\Framework نیز شماره نگارش کامل را ارائه نمی‌دهند. تنها راه باقیمانده مراجعه به فایل mscorlib.dll هر پوشه و بررسی نگارش آن است:



اینکار را با برنامه نویسی به صورت زیر می‌توان انجام داد:

```
public static string MscorlibVersion
      {
          get
          {
              //using System.Diagnostics;
              FileVersionInfo myFileVersionInfo = FileVersionInfo.GetVersionInfo(
                  Environment.GetEnvironmentVariable("windir") +
                  @"\Microsoft.NET\Framework\v2.0.50727\mscorlib.dll");
              return myFileVersionInfo.ProductVersion;
          }
      }
```

نظرات خوانندگان

نویسنده: Alex's Blog
تاریخ: ۱۳۸۸/۰۱/۲۹ ۰۷:۱۶:۰۰

جالب بود!

عنوان: **چگونه بررسی کنیم smtp server مورد نظر ما کار می‌کند؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۲۹ ۲۱:۲۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

گاهی برای عیب یابی عدم ارسال ایمیل نیاز هست بررسی کنیم آیا smtp server مورد نظر جواب می‌دهد؟ آیا در شبکه به آن دسترسی داریم؟ یا برای مثال آیا ISP ما این عملیات را بلاک نکرده است؟

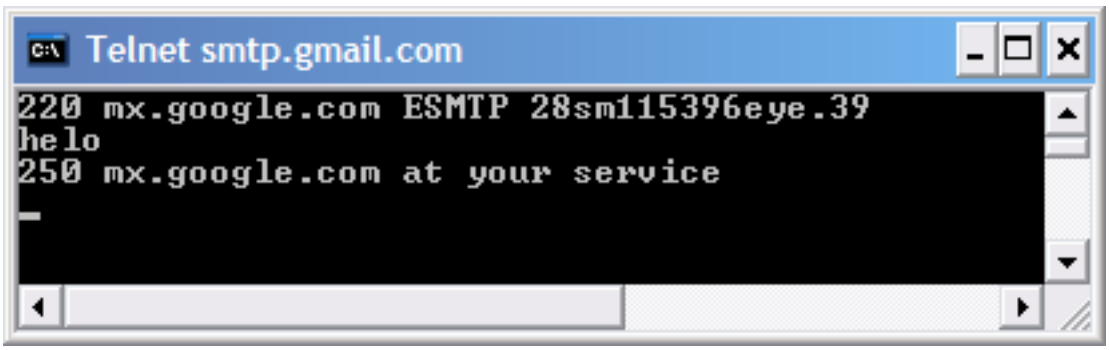
مثال الف) بررسی دسترسی به gmail
در خط فرمان تایپ کنید، telnet و سپس دکمه enter را فشار دهید.
در ادامه سطر زیر را وارد نمائید و enter کنید:

```
open smtp.gmail.com 587
```

اگر پیغام زیر را برای مدتی طولانی دریافت کردید، یعنی به میل سرور گوگل دسترسی ندارید و به هر علتی بلاک شده است:

```
Connecting To smtp.gmail.com...
```

در غیراینصورت یا یک صفحه مشکی خالی را دریافت خواهید کرد و یا یک پیغام معمولی؛ پس از آن مجددا در خط فرمان وارد کنید helo و سپس enter (دقت کنید helo یک l دارد اینجا).
صفحه زیر را باید مشاهده نمائید:



مثال ب) بررسی دسترسی به میل سرور لوکال
همانند مثال قبل است فقط قسمت open آن به شکل زیر تغییر خواهد کرد:

```
open localhost 25
```

و یا

```
open 127.0.0.1 25
```

نظرات خوانندگان

نویسنده: پرهام
تاریخ: ۱۳۸۸/۰۱/۳۰ ۱۳:۲۳:۰۰

سپاس فراوان از شما . . بالاخره برای من بیسواد یک چیزی پیدا شد در وبلاگ شما!!! :) وبلاگ شما را همیشه پیگیر هستم. . . .

نویسنده: mehdi@payervand.ir
تاریخ: ۱۳۸۸/۰۴/۲۸ ۱۶:۴۴:۱۴

سلام
از طریق برنامه نویسی چطور میشه آی پی کلایت رو که به اینترنت اتصل داره رو پیدا کرد.
ممنون از بابت سوالهای قبلی

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۲۸ ۱۶:۵۵:۳۴

به مثال این کتابخانه مراجعه کنید:
/http://dotras.codeplex.com

عنوان: 5 دلیل برای استفاده از یک ابزار ORM
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۰۱ ۱۴:۱۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

چرا باید از ابزارهای Object relational Mapper یا به اختصار ORM استفاده کرد؟ در اینجا سخن در مورد ORM خاصی نیست. هدف تبلیغ یک محصول ویژه هم نمی‌باشد و یک بحث کلی مد نظر است.
کار ابزارهای ORM خواندن ساختار دیتابیس شما بوده و سپس ایجاد کلاس‌هایی بر اساس این ساختار ، برقراری ارتباط بین اشیاء ایجاد شده و جداول، ویووها، رویه‌های ذخیره شده و غیره می‌باشد. همچنین این ابزارها امکان تعریف روابط one-to-one, one-to-many, many-to-one, و many-to-many بین اشیاء را نیز بر اساس ساختار دیتابیس شما فراهم می‌کنند.
در ادامه به فواید استفاده از ORM ها خواهیم پرداخت:

الف) یک ابزار ORM زمان تحویل پروژه را کاهش می‌دهد

اولین و مهم‌ترین دلیلی که بر اساس آن در یک پروژه، استفاده از ORM حائز اهمیت می‌شود، بحث بالا بردن سرعت برنامه نویسی و کاهش زمان تحویل پروژه به مشتری است. این کاهش زمان بسته به نوع پروژه بین 20 تا 50 درصد می‌تواند خود را بروز دهد.
بدیهی است ابزارهای ORM کار شگفت انگیزی را قرار نیست انجام دهند و شما می‌توانید تمام آن عملیات را دستی هم به پایان رسانید؛ اما اجازه دهید یک مثال کوتاه را با هم مرور کنیم.
برای پیاده سازی یک برنامه متداول با حدود 15 تا 20 جدول، حدودا به 30 شیء برای مدل سازی سیستم نیاز خواهد بود و برنامه نویسی این مجموعه بین 5000 تا 10000 سطر کد را به خود اختصاص خواهد داد. بدیهی است برنامه نویسی و آزمایش این سیستم چندین هفته یا ماه به طول خواهد انجامید.
اما با استفاده از یک ORM ، عمده وقت شما به طراحی سیستم و ایجاد ارتباطات بین اشیاء و دیتابیس در طی یک تا دو روز صرف خواهد شد. ایجاد کد بر اساس این مجموعه و با کمک ابزارهای ORM ، آنی است و با چند کلیک صورت می‌گیرد.

ب) یک ابزار ORM کدی با طراحی بهتر را تولید می‌کند

ممکن است شما بگوئید که کد نویسی من بی‌نظیر است و از من بهتر کسی را نمی‌توانید پیدا کنید! به تمامی زوایای کار خود مسلطم و نیازی هم به این‌گونه ابزارها ندارم!
عده‌ای از شما به طور قطع این‌گونه‌اید؛ اما نه همه. در یک تیم متوسط، همه نوع برنامه نویس با سطوح مختلفی را می‌توانید پیدا کنید و تمامی آن‌ها برنامه نویس‌ها و یا طراح‌های آنچنان قابلی هم نیستند. بنابراین امکان رسیدن به کدهایی که مطابق اصول دقیق برنامه نویسی شیء گرا نیستند و در آن‌ها الگوهای طراحی به خوبی رعایت نشده، بسیار محتمل است. همچنین در یک تیم زمانیکه از یک الگوی یکسان پیروی نمی‌شود، نتایج نهایی بسیار ناهماهنگ خواهند بود.
در مقابل استفاده از ORM های طراحی شده توسط برنامه نویس‌های قابل (senior (architect level) engineers) ، کدهایی را بر اساس الگوهای استاندارد و پذیرفته شده‌ی شیءگرا تولید می‌کنند و همواره یک روند کاری مشخص و هماهنگ را در یک مجموعه به ارمغان خواهند آورد.

ج) نیازی نیست تا حتما یک متخصص دات نت فریم ورک باشید تا از یک ORM استفاده کنید

قسمت دسترسی به داده‌ها یکی از اجزای کلیدی کارآیی برنامه شما است. اگر طراحی و پیاده سازی آن ضعیف باشد، کل برنامه را زیر سؤال خواهد برد. برای طراحی و پیاده سازی دستی این قسمت از کار باید به قسمت‌های بسیاری از مجموعه‌ی دات نت فریم ورک مسلط بود. اما هنگام استفاده از یک ORM مهمترین موردی را که باید به آن تمرکز نمائید بحث طراحی منطقی کار است و ایجاد روابط بین اشیاء و دیتابیس و امثال آن. مابقی موارد توسط ORM انجام خواهد شد و همچنین می‌توان مطمئن بود که پیاده سازی خودکار انجام شده این قسمت‌ها، بر اساس الگوهای طراحی شیءگرا است.

د) هنگام استفاده از یک ابزار ORM ، مدت زمان آزمایش برنامه نیز کاهش می‌یابد

بدیهی است اگر قسمت دسترسی به داده‌ها را خودتان طراحی و پیاده سازی کرده باشید، زمان قابل توجهی را نیز باید به بررسی و آزمایش صحت عملکرد آن بپردازید و الزامی هم ندارد که این پیاده سازی مطابق بهترین تجربیات کاری موجود بوده باشد. اما هنگام استفاده از کدهای تولید شده توسط یک ابزار ORM می‌توان مطمئن بود که کدهای تولیدی آن که بر اساس یک سری الگوی ویژه تولید می‌شوند، کاملا آزمایش شده هستند و همچنین صدها و یا هزارها نفر در دنیا هم اکنون دارند از این پایه در پروژه‌های موفق خود استفاده می‌کنند و همچنین بازخوردهای خود را نیز به تیم برنامه نویسی آن ابزار ORM ارائه می‌دهند و این مجموعه مرتبا در حال بهبود و به روز شدن است.

ه) استفاده از یک ابزار ORM ، کار برنامه نویسی شما را ساده‌تر می‌کند

توضیح این قسمت نیاز به ذکر یک مثال دارد. لطفا به مثال زیر دقت بفرمائید:

```
try {
       Employees objInfo = new Employees();
       EmployeesFactory objFactory = new EmployeesFactory();

       objInfo.EmployeeID = EmployeeID;
       objFactory.Load(objInfo);

       // code here to use the "objInfo" object
}
catch(Exception ex) {
      // code here to handle the exception
}
```

به نظر شما کار کردن با یک یا چند شیء تولید شده که نمایانگر ساختار دیتابیس شما هستند و با استفاده از اینترفیس عمومی آن‌ها می‌توان تمامی اعمال بارگذاری، درج و حذف و غیره را انجام داد، ساده‌تر است یا کار کردن با کوهی از دستورات ADO.Net ؟

برداشتی آزاد از Five Reasons for using an ORM Tool

نظرات خوانندگان

نویسنده: صدای سکوت
تاریخ: ۱۳۸۸/۰۳/۰۲ ۱۰:۵۳:۴۷

سلام
از مطالب خوبتون ممنون
من با اجازه شما رو به بلاگ خودم لینک کردم.
موفق باشید

نویسنده: افشار محبی
تاریخ: ۱۳۸۸/۰۳/۰۳ ۰۸:۴۴:۳۶

این را هم اضافه کنید: با استفاده از ORM نگهداری کدها و فهم آنها آسان‌تر می‌شود.
ما این موضوع در استفاده از NHibernate به خوبی درک کرده‌ایم.

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۳/۰۳ ۰۸:۵۰:۲۱

باسلام و تشکر بابت این پست
دوست جناب نصیری بهترین ORM tool موجود مال کدوم شرکته
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۰۳ ۱۰:۰۵:۱۴

NHibernate در حال حاضر پخته‌ترین ORM موجود برای دات نت فریم ورک است.

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۳/۰۵ ۰۷:۴۵:۲۶

آیا typed dataset ها در .net هم یک orm هستند؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۰۵ ۱۰:۲۴:۰۵

سؤال شما به موضوع بحث مرتبط نیست. آیا جایی در مقاله به typed dataset اشاره شده یا کار نقد orm های مختلف صورت گرفته؟ ... بگذریم.

در مورد typed dataset ها ، بله نسبتا تا حدودی و تا حد نازلی بله! شبه ORM هستند که این مشکلات را دارند:
- مشکل Synchronization بین آن‌ها و دیتابیس مساله ساز است که در یک ORM خوب باید این مساله حل شده باشد.
- join table queries در طراح آن کار نمی‌کند!
- query syntax استانداردی نداشته و هنگام کار با دیتابیس‌های مختلف (نوع‌های مختلف) این مورد مساله ساز می‌شود.
- typed dataset کمتر حال و هوای یک ORM واقعی و دنیای شیء گرایی با اشیایی که وابستگی کمی به دیتابیس دارند را ارائه می‌دهد.
- کلا dataset اشیایی با سربار بالا در دات نت فریم ورک مطرح هستند و زمانیکه کارآیی مطرح هست سعی می‌شود به روش‌هایی دیگری کوچ شود.
همچنین از لحاظ مباحث serialization هم بسیار ضعیف و کند عمل می‌کنند.
- زمانیکه از typed dataset استفاده می‌کنید عملا مدل رابطه‌ای دیتابیس خود را با business layer مخلوط کرده‌اید.

نویسنده: محمد امین شریفی
تاریخ: ۱۳۸۸/۰۷/۱۸ ۲۰:۰۴:۱۰

strongType ها و entity ها را می توان هم پای فناوری های مطرح ORM قرار داد؟

نویسنده: شاهین کیاست
تاریخ: ۱۳۸۹/۱۱/۰۸ ۱۵:۴۵:۳۷

سلام جناب نصیری.
آیا تجربه ای در ضمینه Telerik ORM دارا هستید؟
قابل رقابت با EF , nhibernate , llblgen هست؟
ایا شما nettiers که یک Template برای Codesmith هست را یک ORM به حساب می آورید؟
با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۰۸ ۱۷:۰۹:۲۷

سلام، به نظر من عمر یک برنامه نویس کوتاه است و بهتر است روی موردی وقت بگذارد که :
- امیدی به ادامه‌ی آن باشد (نگن امروز به به! فردا ... خوب دیگه ... تموم شد! صرف نمی‌کنه، دیگه توسعه نمی‌دیم! خیلی از سیاست‌های مایکروسافت همینطوری است. مثلا همون کاری که با LINQ to SQL کرد)
- چند هزار نفری پیرو و دنبال مباحث آن باشند (حداقل 2 تا فوروم رفع اشکال بتونید پیدا کنید)
- دو تا کتاب در موردش باشه
- 4 تا وبلاگ در موردش مطلب بنویسند.
و مسایلی از این دست.

به همین جهت یا روی EF یا NH سرمایه گذاری کنید.
به شخصه NH رو ترجیح می‌دم چون سورس باز است، به همین جهت مرگ برای آن معنی ندارد (این گروه نخواست ادامه بده ... گروه‌های دیگر هستند)، رایگان است، مجوزش اجازه استفاده در کارهای تجاری سورس بسته را می‌دهد. چندتا کتاب در موردش هست و ...
به EF شک دارم. نمی‌دونم مایکروسافت مثلا 4 سال دیگه آیا این را هم بازنشسته اعلام می‌کند یا نه.

نویسنده: mojtaba amrollahi
تاریخ: ۱۳۹۰/۰۲/۰۲ ۲۳:۰۸:۴۶

درود

آقای نصیری یه سئوال از خدمتتون داشتم:
ما این روزا می خایم استارت یه نرم افزار انترپرایس رو بزنیم ولی یکی از چالش های اساسی ما جنگ بر سر استفاده از EF یا NH است.
می خواستم ببینم می تونید اینجانب رو یه راهنمایی با دلایل منطقی بکنید.
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۲/۰۳ ۰۰:۱۱:۳۶

نگارش جدید EF (نگارش 4 و یک) یک ماهی هست که منتشر شده. مقایسه‌ای از اون رو با NH می‌تونید اینجا مطالعه کنید:
http://jfromaniello.blogspot.com/2011/03/entityframework-41-rc-code-first-review.html

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۲/۰۳ ۰۰:۱۴:۱۱

یک مقایسه دیگر هم اینجا است

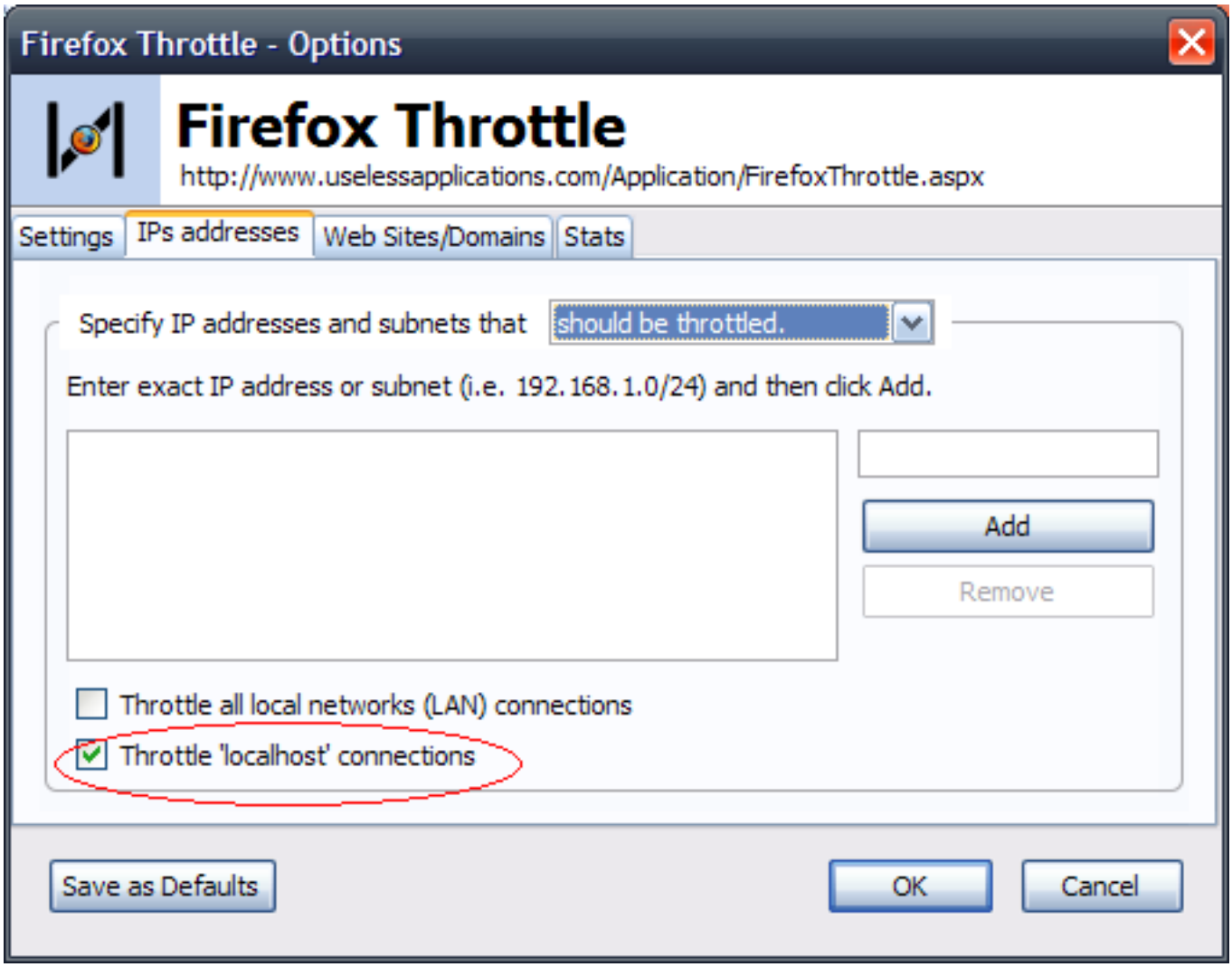
/http://farasun.wordpress.com/2011/01/26/entity-framework-vs-nhibernate

کامنت‌ها رو هم حتما بخونید

عنوان: **نحوه شبیه سازی سرعت‌های اتصال پایین جهت آزمایش یک وب سایت**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۰۲ ۱۸:۴۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

اکثر توسعه دهندگان وب، پیش از انتشار کار خود بر روی اینترنت، سایت خود را در یک محیط محلی آزمایش می‌کنند. بدیهی است سرعت بارگذاری سایت در این حالت از هر سرعت اتصال اینترنتی بالاتر می‌باشد و برای مثال یک توسعه دهنده، امکان تجربه‌ی وضعیت یک کاربر دایال آپ را پیش از انتشار سایت خود نخواهد یافت.
برای حل این مشکل، دو افزونه، برای فایرفاکس و IE تهیه شده‌اند که امکان تنظیم میزان پهنای باند دریافتی مرورگر وب را میسر می‌سازند. به این صورت می‌توان وضعیت‌های اتصالی مختلف را به سادگی و پیش از انتشار کار بر روی اینترنت، آزمود:

Firefox Throttle
IE Throttle



همانطور که در تصویر ملاحظه می‌کنید، توسط این افزونه می‌توان تنظیمات برگه settings را بر روی یک سری IP و یا سایت و یا

به سادگی بر روی کلیه ارتباط‌هایی که به `localhost` ختم می‌شوند، اعمال کرد.

نظرات خوانندگان

نویسنده: نیما
تاریخ: ۱۳۸۸/۰۳/۰۳ ۱۱:۲۴:۱۷

سلام جناب نصیری
این لینکها متاسفانه باز نمیشن
موفق باشید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۰۳ ۱۱:۵۹:۲۷

بله. دو حالت داره.
الف) مشکل DNS وجود دارد در ISP های ما
ب) سایت به نحو محترمانه‌ای فیلتر شده

هر دو مورد با روش‌های "متداول" قابل حل است.
و یا:
https://addons.mozilla.org/en-US/firefox/addon/5917

عنوان: **پرسش و پاسخ‌های متداول ایجاد یک وبلاگ بلاگری**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۱۱ ۱۹:۴۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

هر از چندگاهی دوستان پیغام می‌گذارند که چگونه تاریخ وبلاگم را فارسی کنم یا یک قالب فارسی خوب را از کجا می‌شود تهیه کرد و امثال آن. تمام این‌ها را به صورت یک FAQ کوچک در اینجا قرار می‌دهم تا برای عموم علاقمندان قابل استفاده باشد.

پرسش	پاسخ
برای ایجاد یک وبلاگ بلاگری از کجا شروع کنم؟	با داشتن یک اکانت جی میل، به آدرس زیر مراجعه کرده و هر تعداد که مایل بودید می‌توانید وبلاگ ایجاد نمائید: http://www.blogger.com/create-blog.g
قالب‌های پیش فرض گوگل چنگی به دل نمی‌زنند و بود و نبود آن‌ها احساس نمی‌شود! چه باید کرد؟	برای تهیه قالب‌های فارسی به آدرس زیر مراجعه نمائید: http://themes.blogger-fa.com
تاریخ فارسی را چه کنم؟	روش دوستان ، روش من
برنامه نویسم! کدهای من یا نمایش داده نمی‌شود یا فرمت آن‌ها هنگام ارسال به هم می‌ریزد. چه باید کرد؟	روش مورد علاقه‌ی من ، روش مایکروسافت (+ و + و افزونه‌ای برای این‌کار)
آخرین نظرات کاربران سایت را چگونه در ستون کناری سایت نمایش دهم؟	یکی از ویجت‌های پرکاربرد گوگل در بلاگر، ویجت فید است. هر بلاگ دارای دو فید مطالب و نظرات است. ویجت فید را اضافه کرده و سپس آدرس فید نظرات سایت خود را به آن معرفی کنید. همیشه آخرین 5 نظر ارسالی را نمایش می‌دهد. این ویجت کاربردهای قابل توجهی می‌تواند داشته باشد.
آمار مراجعان به سایت را باید در کجا ملاحظه کرد؟	جای پیش فرضی وجود ندارد! یک اکانت Analytics برای خود تهیه کرده و آن‌را به سایت اضافه کنید. همچنین پس از آن یک اکانت فید برنر را نیز برای خود تهیه کنید. فیدبرنر اکنون جزیی از گوگل است و پس از معرفی آدرس فید سایت خود به آن، یک آدرس جدید به همراه آیکونی برای نمایش در سایت به شما می‌دهد که می‌توانید تگ‌های آن تصویر را توسط ویجت html نمایش دهید. همچنین از همین طریق نیز می‌توان اشتراک از طریق ایمیل را به وبلاگ اضافه کرد.
کامنت‌های سایت من در بیرون از سایت باز می‌شوند. چرا؟	به این مطلب مراجعه نمائید: کادر نظرات در وبلاگهای بلاگر در ذیل مطالب، مانند وردپرس
چگونه برای بلاگ خود قسمت ارسال نظرات را تهیه نمایم؟	من از سرویس فرم ساز Zoho استفاده می‌کنم که در سال 2008 جزو برترین‌های وب شناخته شده بود. بعد از ایجاد فرم خود و دریافت کد مربوطه، یک مطلب جدید را ارسال کنید و در بدنه‌ی آن ، کدی را که Zoho به شما ارائه می‌دهد قرار دهید. سپس لینک این مطلب را توسط یک ویجت html به کنار سایت اضافه کنید تا همیشه در دسترس باشد.

نظرات خوانندگان

نویسنده: mohammad shams
تاریخ: ۱۳۸۸/۰۳/۱۲ ۱۵:۱۰:۴۷

جالب و مفید بود

نویسنده: محمد صاحب
تاریخ: ۱۳۸۸/۰۶/۰۱ ۰۹:۳۰:۱۸

پیرو این تاپیک
http://www.dotnettips.info/2009/08/blog-post_20.html
عنوان کردن این سوال برام خیلی سخت بود ولی هر کاری کردم نشد تمام حالت های ممکن رو بصروت سعی و خطا تست کردم ولی نشد

و اما مشکل
از روشی که شما برای تاریخ شمسی ارائه دادید استفاده کردم ولی خروجی به این شکله "انارام روز " رو می خوام حذف کنم

جمعه، ۳۰ مرداد ۱۳۸۸، انارام روز (۲۱ آگوست ۲۰۰۹)

ممنون از توجهتون

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۰۱ ۰۹:۴۵:۲۸

سلام
این "انارام روز " کار من نبوده :)
به عبارتی اگر از روش من استفاده کرده باشید
http://www.dotnettips.info/2008/12/blogger.html
این مورد را حذف کرده‌ام و لحاظ نشده که نمونه آن در سایت جاری مشهود است.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۰۱ ۰۹:۵۰:۲۴

در کل این موارد در فایل زیر تعریف شده
http://www.mozillafirefox.ir/js/persianDate.js
این فایل رو دانلود کنید و تغییرات لازم را در آن اعمال کنید. سپس یک جای دیگر آپلود کرده و لینک بدهید.
برای مثال، نمونه اصلاح شده من در آدرس زیر است:
http://vahidnasiri.persiangig.com/document/farsidate.js

نویسنده: محمد صاحب
تاریخ: ۱۳۸۸/۰۶/۰۱ ۱۲:۰۵:۴۰

خیلی ممنون که وقت گذاشتید

تاریخ قسمت نظرات درست نمایش داده میشه ولی هدر نه
http://mohammadsaheb.blogspot.com/2009/08/2-visual-studio.html

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۰۱ ۱۲:۳۳:۱۶

تاریخ بالای سایت شما از روش دوستان استفاده کرده.
به قسمت ویرایش html بلاگ مراجعه کنید. این مورد را باید دستی حذف کنید. مواردی که به قالب شما به صورت پیش فرض اعمال شده این‌جا توضیح داده شده است:
http://docs.blogger-fa.com/2008/09/blogger-persian-date.html

نویسنده: آراد
تاریخ: ۱۳۸۸/۰۷/۱۵ ۱۵:۲۷:۲۴

با سلام وخسته نباشید
دو تا سوال داشتم
سوال اول : توی قالب های جدید بلاگر (قالب هایی با فرمت جدید) ، وقتی روی
Edit HTML
می رم و متن داخل اون رو توی برنامه ی
Microsoft FrontPage
کپی می کنم تا بتونم ویرایشش کنم ، چیزی نشون نمیده و قالب رو به من نشون نمیده . مثلا می خوام واسه ی بک گراند قالبم از عکس استفاده کنم ولی محتویات قالب نشون داده نمیشه . مشکل از کجاست و چطوری باید حلش کنم تا بتونم یه قالب جدید با بک گراند مخصوص درست کنم؟
سوال دوم : توی قالب های کلاسیک بلاگر ، برای تبدیل تاریخ میلادی به شمسی از روشی که توی پست
http://www.dotnettips.info/2008/12/blogger.html
توضیح داد هبودین استفاده کردم . ولی دو تا مشکل وجود داره :
اولا اینکه فرمت و رنگ تاریخ متن به صورت مشکی و بزرگ در اومد . چطوری میتونم سایز و رنگ تاریخ رو عوض کنم ؟
دوم اینکه تاریخ آرشیو رو چطور می تونم به فارسی تبدیل کنم؟
وبلاگ آزمایشی من تو این ادرسه که با قالب کلاسیک ، تاریخش رو به شمسی تبدیل کردم :
http://arad-diarist2.blogspot.com
ممنون می شم اگه راهنمایی کنید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۱۵ ۱۵:۵۶:۰۵

قرار بود فقط دو تا سؤال بپرسید :)
- فرمت قالب‌ها xml است و با فرانت پیج قدیمی یا expression web جدید قابل ویرایش نیست و اصلا ادیتوری هم ندارد (یا جایی منتشر نشده تا جایی که اطلاع دارم). دستی باید روی آن کار شود.
- برای قرار دادن عکس در بک گراند، شما فقط کافی است عکس مورد نظر را در یک سایت دیگر آپلود کنید، سپس یک css تعریف کرده و آن‌را اعمال نمائید. به سلکتور body یا تگ body قابل اعمال است. اطلاعات بیشتر:
http://www.w3schools.com/css/pr_background-image.asp
- برای تعیین رنگ و اندازه تاریخ، باز هم باید css سایت را در همان قسمت ویرایش html سایت ویرایش کنید. برای مثال این موارد باید اضافه شود:

```
} h1, h2, h3
;font-family: Tahoma
;direction: rtl
;font-size:8pt
;color:green
;text-align: right
{
```

- تاریخ آرشیو هم در همان مقاله هست و باید اعمال شده باشد.

در کل شکل و شمایل این قالب‌ها با ویرایش css آن تا حد قابل قبولی قابل تغییر است (بدون نیاز به دستکاری قسمت‌های دیگر که مربوط به تعریف ویجت‌ها است)

نویسنده: آراد
تاریخ: ۱۳۸۸/۰۷/۱۵ ۱۷:۳۶:۵۹

وحید جان ممنون از جوابتون ، ولی من با توجه به همون مقاله ای که گفتین ، کد رو وارد کردم ولی تاریخ آرشیو عوض نشد و هم اینکه وقتی بیشتر از دو پست می نویسم ، تاریخ ناخوانا میشه و اون چیزی رو مینویسه که تو وبلاگ زیر مینویسه:
http://arad-diarist2.blogspot.com
اگه بخوام قالب های جدید رو مثل قالب وبلاگ آزمایشیم بکنم (یعنی عکس بک گراند بذارم و جای متن و ستون ها رو عوض کنم و تغییرات دلخواه رو بدم) چطور میتونم اینکارو بکنم وقتی که نمیشه کد قالب رو وقتی تو Microsoft FrontPage وارد میکنم ... ، قالب رو ببینم تا بخوام تغییرش بدم؟
ممنون میشم اگه باز راهنماییم کنی

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۱۵ ۱۸:۱۶:۱۳

برای ادیت این قالب‌ها شما باید به CSS کاملا مسلط باشید یا توصیه من این است که لطفا به آدرس زیر مراجعه کرده و از یکی از قالب‌های آماده آن استفاده کنید (تقریبا همه همین کار را می‌کنند):
http://themes.blogger-fa.com
برای تاریخ هم از این روش استفاده کنید:
http://docs.blogger-fa.com/2008/09/blogger-persian-date.html

نویسنده: آراد
تاریخ: ۱۳۸۸/۰۷/۱۵ ۲۲:۲۲:۰۵

وحید جان ، ممنون از راهنمایی

نویسنده: Hossein Moradinia
تاریخ: ۱۳۸۸/۱۱/۲۵ ۱۹:۵۳:۱۰

با سلام
استاد من یک weblog درست کردم اما هرچی فکر می کنم نمی دونم چطور میشه مثل وبلاگ شما نظرات اول به تایید مدیر برسن بعد نمایش داده بشن.

در جستجویی هم که انجام دادم گزینه ی خاصی رو در این زمینه پیدا نکردم
میشه راهنمایی کنید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۱/۲۵ ۱۹:۵۵:۴۰

سلام
در بلاگر
قسمت تنظیمات ... قسمت نظرات ... گزینه‌ی "بررسی نظرات" را روی همیشه قرار بدید.

ورد پرس هم شبیه به همین را دارد.

نویسنده: Hossein Moradinia

تاریخ: ۱۳۸۸/۱۱/۲۵ ۲۰:۵۱:۴۶

ممنون استاد
درست شد
خیلی لطف دارید
همیشه پیروز باشید

نویسنده: Hossein Moradinia
تاریخ: ۱۳۸۸/۱۱/۲۵ ۲۰:۵۸:۲۸

در ضمن به عنوان اولین پست وبلاگم وبلاگ شما رو معرفی کردم و هر جای دیگه هم بشه همین کار رو میکنم

سربلند و پیروز باشید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۱/۲۵ ۲۱:۱۴:۴۱

سلام
ممنون. شما لطف دارید. موفق باشید

نویسنده: fatemeh
تاریخ: ۱۳۸۹/۱۰/۲۵ ۱۹:۳۸:۰۳

سلام. من یه سوال کاملا حرفه ای داشتم.
میخوام برای وبلاگم در بلاگ اسپات که فقط برای نویسنده هاش بازه توی فیدبرنر فید نظرات درست کنم اما ارور میده. از یه آدم حرفه ای هم کمک گرفتم کاری از دستش بر نیمد!
خیلی خوشحال میشم اگه شما بتونید کمکم کنید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۵ ۲۰:۵۱:۵۹

کاری از دست کسی بر نمیاد! چون وبلاگ خصوصی، فید عمومی قابل دسترسی از بیرون ندارد. فیدبرنر یک سایت 100 درصد عمومی است و به سایت‌های خصوصی خدمات نمی‌دهد.

نویسنده: fatemeh
تاریخ: ۱۳۸۹/۱۰/۲۵ ۲۲:۲۰:۰۰

امااز پیامی که میده چنین چیزی فهمیده نمیشه:
The feed address you entered is password protected. You can specify a username and password in the URL like
.http://user:password@www.website.com/index.xml

نویسنده: fatemeh
تاریخ: ۱۳۸۹/۱۰/۲۵ ۲۲:۳۴:۴۹

چرا استقبال نسبت به بلاگر اینقدر کمه توی ایران؟ با وجود اینکه خیلی خوبه؟
آیا بلاگرهای دیگه ای رو میتونید به من معرفی کنید که در زمینه بلاگر بنویسند؟ آخه من هرچی توی گوگل برای بلاگر و مسائلش میگردم چیزی زیادی دستگیرم نمیشه، همش همون بلاگر فارسیه و یه چندتا وبلاگ دیگه!
تازه مثلا همین بلاگرفارسی که مطالبش خیلی قدیمیه و مثلا دیگه اینکه پوسته هاش هم خیلی محدود و قدیمین و من نتونستم یکی بپسندم! دیگه اینکه مثلا من نتونستم بر اساس دستور العملشون تاریخم رو هجری شمسی کنم چون اصلن کدی رو که گفته بودن نداشت!و سوالهای زیاد دیگه ای هم دارم که باید از یکی که با بلاگر کار کرده بپرسم!

نویسنده: fatemeh
تاریخ: ۱۳۸۹/۱۰/۲۵ ۲۲:۵۹:۲۸

شما چرا وبلاگتون فید نظرات نداره؟
یه فید بری نظرات همه پستها
منظورم از سوال اینه که بذارید براش!

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۶ ۰۰:۳۹:۰۷

- تمام بلاگ‌های بلاگر به صورت استاندارد فید نظرات را دارند: (+)
- آقای باغبانباشی در مورد تهیه قالب برای بلاگر فعال هستند: (+)

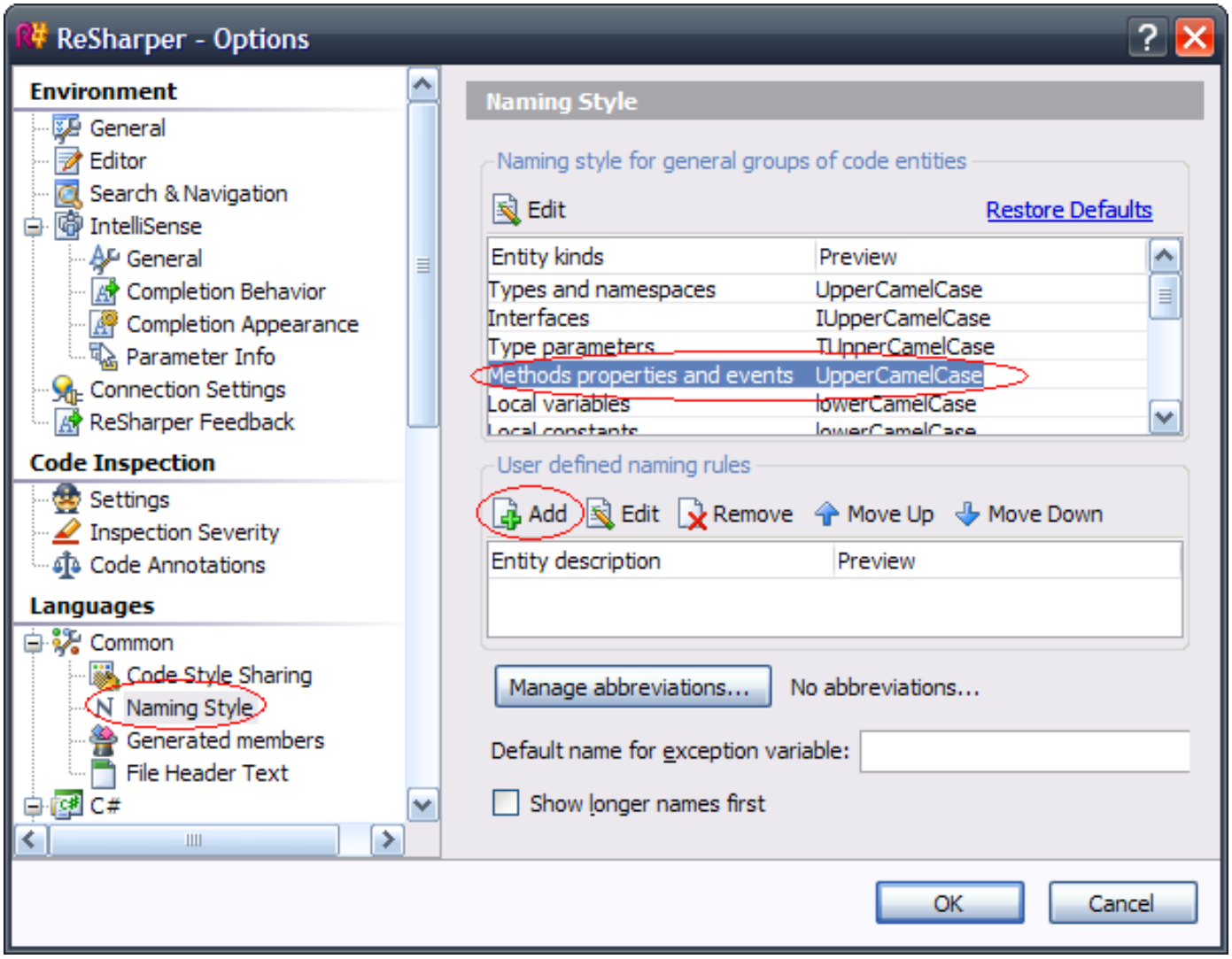
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۶ ۰۰:۴۳:۱۹

روش user:password@www.website.com با بلاگر کار نمی‌کند. سیستم امنیتی بلاگر خیلی فراتر از این روش‌های معمولی است.

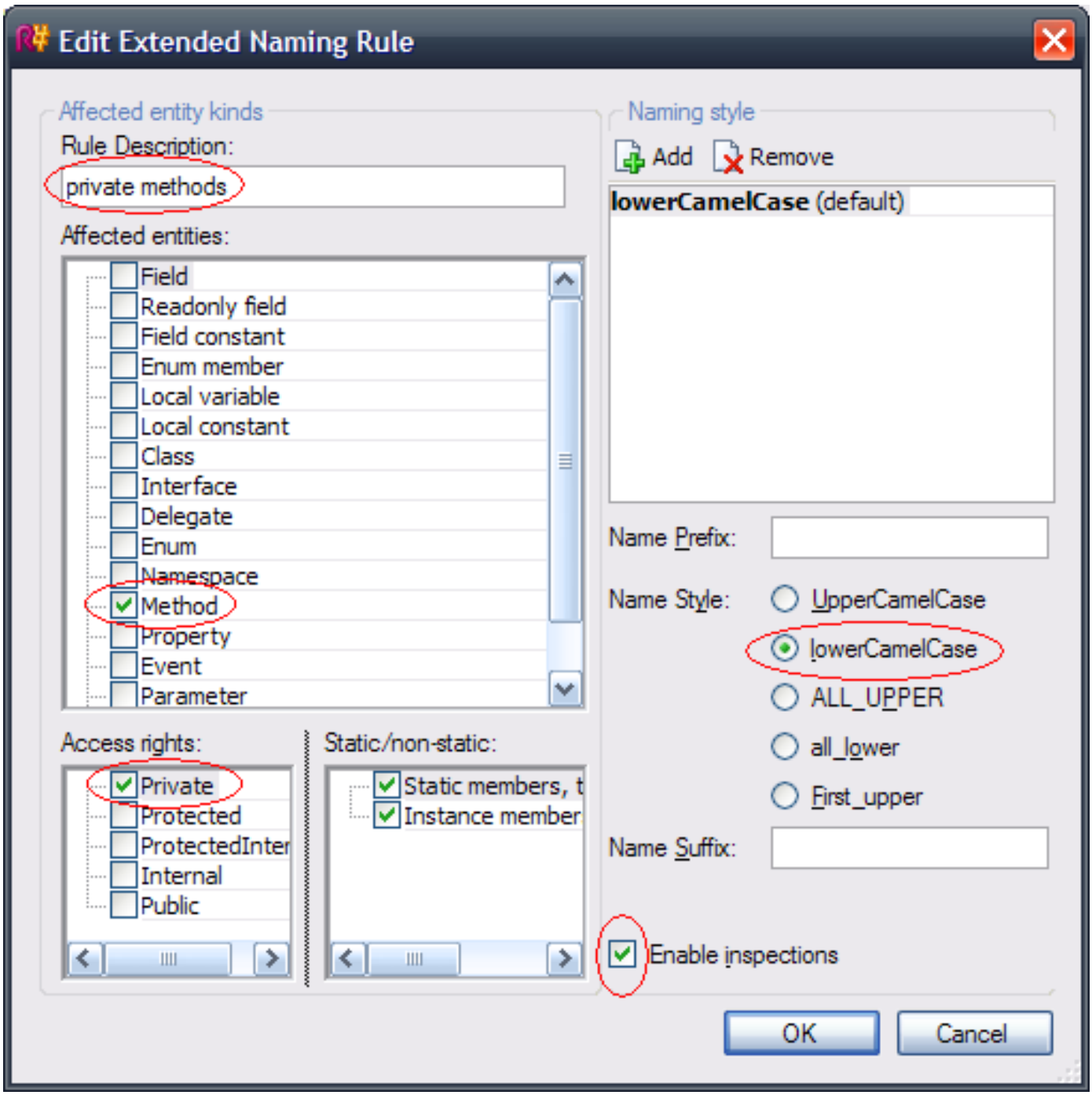
عنوان: اصلاح شیوه نامگذاری در ReSharper
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۲۰ ۲۱:۴۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

یکی از روش‌های متداول نام گذاری متدها در سی شارپ به این صورت است که متدهای خصوصی با حروف کوچک شروع شوند یا lower camel case و متدهای عمومی با حرف بزرگ.

ReSharper 4.5 که جزو ابزارهای واجب کاری است، گزینه Naming style را نیز اضافه کرده و اگر شما از اصول نامگذاری متدها، کلاس‌ها ، متغیرها و غیره پیروی نکنید، علایم راهنمایی را به شما ارائه خواهد کرد. در این نگارش تمامی متدها به یک صورت در نظر گرفته می‌شوند: Upper camel case .



برای اصلاح آن می‌توان به برگه گزینه‌های آن مراجعه کرده و در قسمت naming style بر روی دکمه add مربوط به user defined naming rules کلیک و تغییر زیر را اعمال نمود:



پس از اعمال آن اگر یک متد خصوصی را با حرف بزرگ شروع کنید، تصویر زیر نمایش داده خواهد شد:

```
public class CTest
{
    private int _sum;

    private void SubTest()
    {

    }
```

Name 'SubTest' does not match rule 'private methods'. Suggested name is 'subTest'.

نظرات خوانندگان

نویسنده: Novin
تاریخ: ۱۳۸۸/۰۳/۲۲ ۰۰:۴۴:۴۴

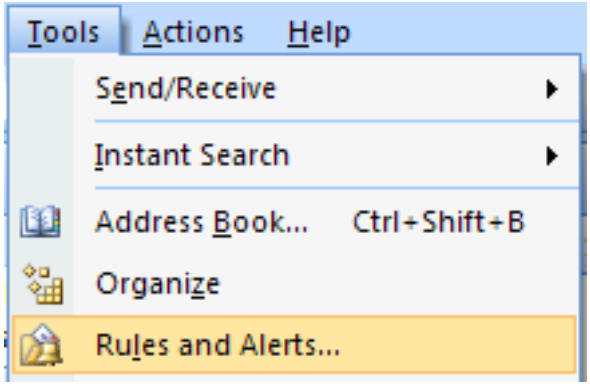
راهنمایی مفیدی بود.
مرسی.

عنوان: **نحوه به تاخیر انداختن ارسال ایمیل‌ها در آوت لوک**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۸ ۲۲:۰۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

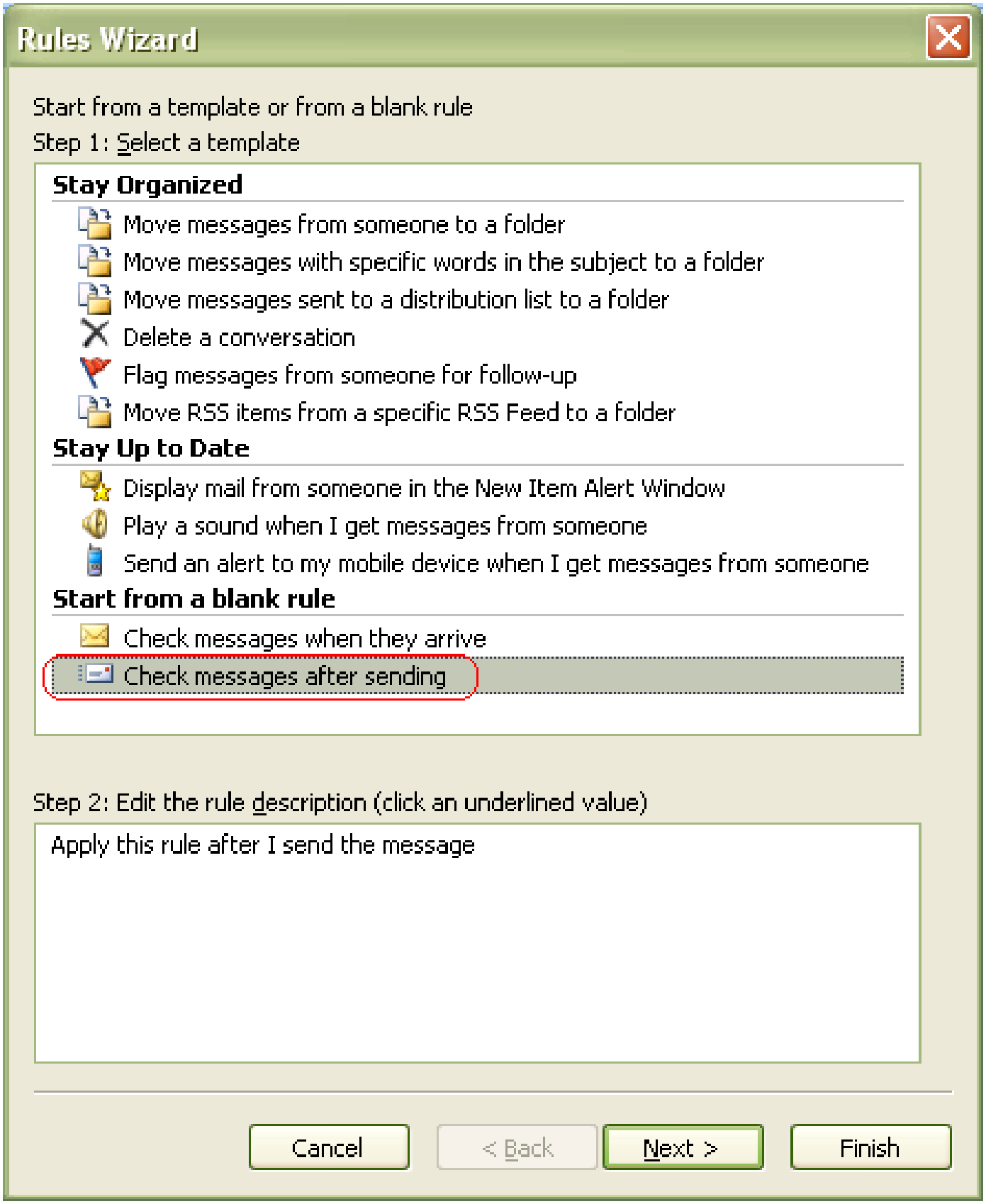
مطلب امروز به کنترل شخصی مرتبط است. به درد همه می‌خوره! :)
چگونه ارسال ایمیلی را که ممکن است 5 دقیقه بعد از ارسال آن به شدت پشیمان شویم، کنترل کنیم؟!

برای به تاخیر انداختن تمامی ایمیل‌های ارسالی از طریق آوت لوک می‌توان به صورت زیر عمل کرد:

به منوی tools گزینه rules and alerts مراجعه کنید.



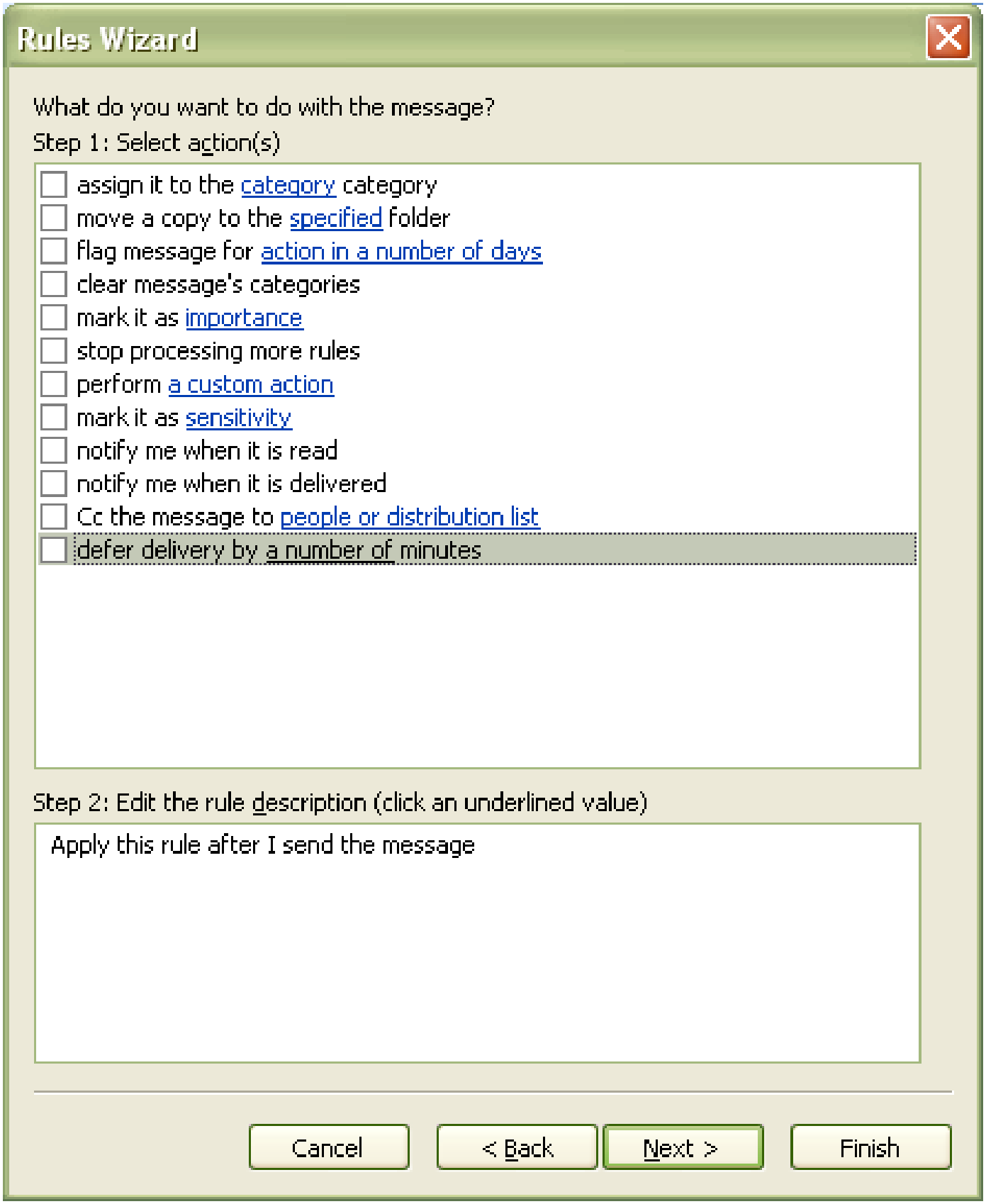
در صفحه باز شده بر روی دکمه new rule کلیک کنید.
در پنجره بعدی گزینه Check messages after sending را انتخاب کرده و بر روی دکمه next کلیک کنید.

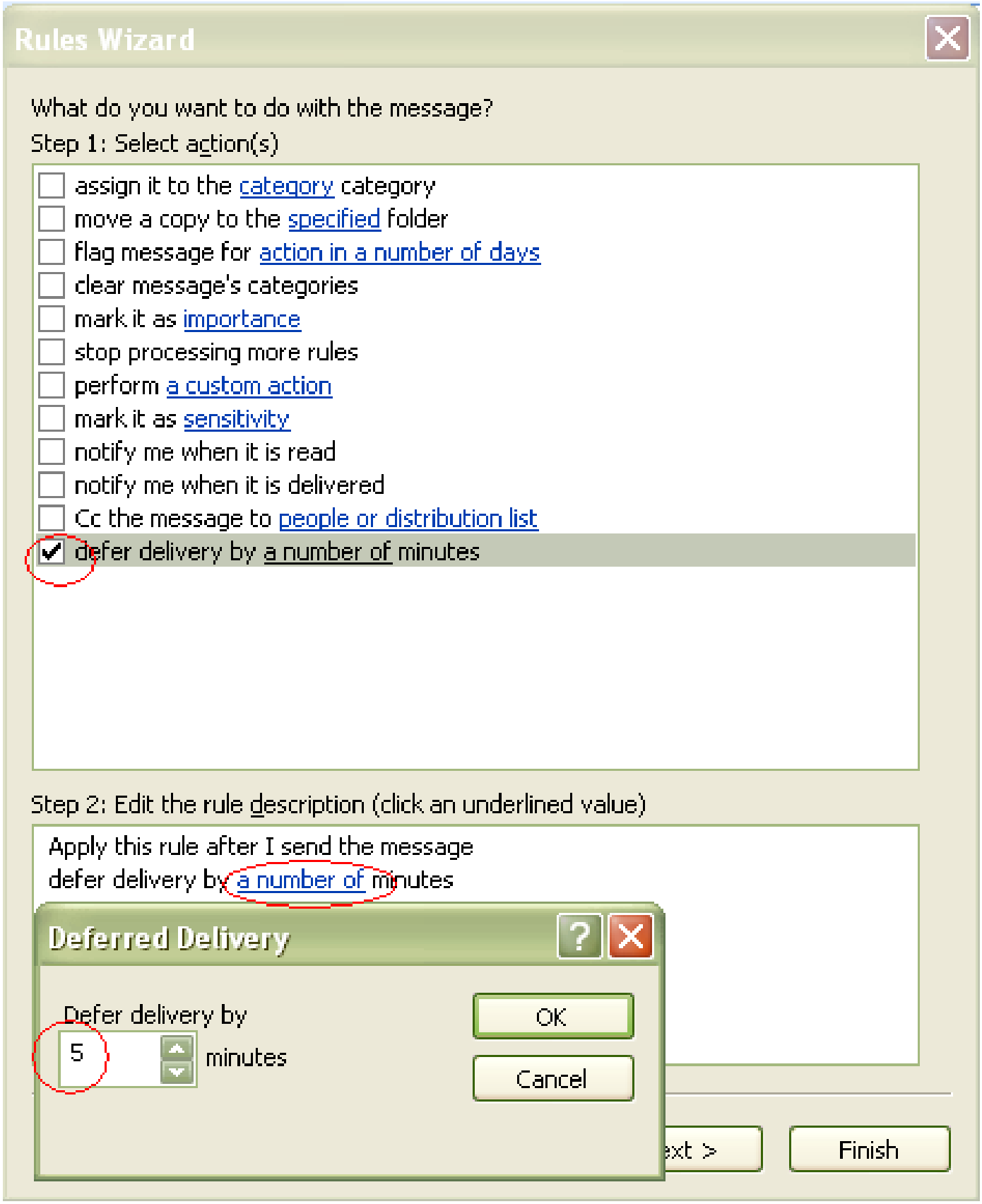


در صفحه بعد تنها بر روی گزینه next کلیک کنید (تا تنظیمات ما بر روی تمامی ایمیل‌های ارسالی اعمال شود).

سپس بر روی دکمه yes پیغام باز شده کلیک نمائید.

تنظیمات اصلی مطلب جاری مربوط به این صفحه است. در اینجا گزینه defer delivery by a number of minutes را تیک بزنید.


Rules Wizard
What do you want to do with the message?
Step 1: Select action(s)
assign it to the category category
move a copy to the specified folder
flag message for action in a number of days
clear message's categories
mark it as importance
stop processing more rules
perform a custom action
mark it as sensitivity
notify me when it is read
notify me when it is delivered
Cc the message to people or distribution list
defer delivery by a number of minutes
Step 2: Edit the rule description (click an underlined value)
Apply this rule after I send the message
Cancel
< Back
Next >
Finish



سپس بر روی لینک a number of کلیک کنید تا صفحه وارد کردن میزان زمان به تاخیر انداختن ارسال را بتوان وارد کرد. پس از وارد کردن یک عدد دلخواه و کلیک بر روی دکمه ok ، بر روی دکمه next کلیک نمائید.

در صفحه بعد نیز بر روی دکمه next کلیک کنید. (البته در اینجا می‌توان مشخص کرد که برای مثال اگر عنوان ویژه‌ای بکار برده شد یا به گروه خاصی ایمیل ارسال گردید، این محدودیت برداشته شود)
و در آخرین صفحه، نامی دلخواه را وارد کرده و بر روی دکمه‌ی خاتمه کلیک نمائید.
در صفحه‌ی اصلی rules & alerts نیز بر روی دکمه apply کلیک کنید، تا تنظیمات اعمال گردد.

از این پس هر ایمیل ارسالی شما مدتی در outbox معطل شده و سپس ارسال می‌گردد.
هنوز تا 5 دقیقه دیگر فرصت هست! با مراجعه به outbox می‌توان ایمیل مورد نظر را در صورت منصرف شدن حذف یا ویرایش کرد. (بنابر تجربه 3 دقیقه کافی است!)

مطلب ارسالی فوق برای آوت لوک 2007 تنظیم شد. اگر آوت لوک شما 2003 است لطفا به آدرس زیر مراجعه کرده و قسمت Delay delivery of all messages را مطالعه نمائید.

http://office.microsoft.com/en-us/outlook/HP052427901033.aspx

نظرات خوانندگان

نویسنده: مهدی پایروند
تاریخ: ۱۳۸۸/۰۵/۱۹ ۰۹:۴۴:۱۴

ممنون جالب و کاربردی بود.

عنوان: آشنایی با Defensive programming
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۳۱ ۱۳:۴۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

تصادف برای یک راننده حتی در صورت داشتن بیمه نامه‌ای معتبر، گران تمام خواهد شد (از لحاظ جانی/مادی/...). بنابراین صرف نظر از اینکه شرکت بیمه کننده چه میزان از خسارت راننده را جبران خواهد کرد، باید تا حد ممکن از تصادفات بر حذر بود (defensive driving).

در برنامه نویسی، استثناءها (Exceptions) مانند تصادفات هستند و مدیریت استثناءها (exception handling)، همانند بیمه خودرو می‌باشند. هر چند مدیریت استثناءها جهت بازگردان برنامه شما به ادامه مسیر مهم هستند، اما جایگزین خوبی برای Defensive programming به شمار نمی‌روند. استثناءها و مدیریت آن‌ها برای برنامه گران تمام می‌شوند (خصوصا از لحاظ میزان مصرف منابع سیستمی و سربارهای مربوطه). بنابراین در برنامه باید توجه خاصی را به این موضوع معطوف داشت که چه زمانی، چگونه و در کجا ممکن است استثنائی رخ دهد و علاج واقعه را پیش از وقوع آن نمود.

اصل اول Defensive programming : همیشه ورودی دریافتی را تعیین اعتبار کنید
به مثال زیر دقت بفرمائید:

```
public void LogEntry(string msg)
    {
        string path = GetPathToLog();
        using (StreamWriter writer = File.AppendText(path))
        {
            writer.WriteLine(DateTime.Now.ToString(CultureInfo.InstalledUICulture));
            writer.WriteLine("Entry: {0}", msg);
            writer.WriteLine("--------------------");
        }
    }
```

قرار هست رخدادهای برنامه را توسط این متد، لاگ کنیم. اکنون لحظه‌ای دقت نمائید که این تابع در چه مواقعی ممکن است دچار مشکل شود:
path می‌تواند یک رشته خالی باشد.
path می‌تواند نال باشد.
path می‌تواند حاوی کاراکترهای غیرمجازی باشد.
path می‌تواند فرمت نادرستی داشته باشد.
path می‌تواند به محلی ناصحیح اشاره نماید.
path می‌تواند اصلا وجود نداشته باشد.
فایل مورد نظر ممکن است readonly باشد.
برنامه ممکن است دسترسی لازم را برای نوشتن در مسیر ذکر شده، نداشته باشد.
فایل مورد نظر ممکن است توسط پروسه‌ای دیگر قفل شده باشد.
ممکن است در لحظه نوشتن یا خواندن بر روی فایل، هارد دیسک دچار مشکل گردد.
و ...

رخ دادن هر کدام از موارد ذکر شد منجر به بروز یک استثناء خواهد شد.

چگونه این وضعیت را بهبود بخشیم؟
فرض کنید متد GetPathToLog قرار است مسیر ذخیره سازی لاگ‌ها را از کاربر در یک برنامه ASP.Net دریافت کند. برای این

منظور باید حداقل دو مورد را منظور کرد.

```
<asp:TextBox ID="txtPath" runat="server" MaxLength="248" />

<asp:RequiredFieldValidator ID="reqval_txtPath" runat="server" ControlToValidate="txtPath"
ErrorMessage="Path is required." />

<asp:RegularExpressionValidator ID="regex_txtPath" runat="server" ControlToValidate="txtPath"
ErrorMessage="Path is invalid." ValidationExpression='^([a-zA-
Z]\:)(\\{1}|((\\{1})[^\\]([^/:*?<>”|]*(?<![ ]))+)$' />
```

برای تکست باکس ارائه شده، ابتدا یک RequiredFieldValidator در نظر گرفته شده تا مطمئن شویم که کاربر حتما مقداری را وارد خواهد کرد. اما این کافی نیست. سپس با استفاده از عبارات باقاعده و RegularExpressionValidator بررسی خواهیم کرد که آیا فرمت ورودی صحیح است یا خیر.

تا اینجا 4 مورد اول مشکلاتی که ممکن است رخ دهند (موارد ذکر شده فوق)، کنترل می‌شوند بدون اینکه احتمال رخ دادن این استثناءها در برنامه وجود داشته باشد. Defensive programming به این معنا است که طراحی برنامه باید به گونه‌ای باشد که در اثر استفاده‌ی غیر قابل پیش بینی از آن، در عملکرد برنامه اختلالی رخ ندهد.

نظرات خوانندگان

نویسنده: افشار محبی
تاریخ: ۱۳۸۸/۰۶/۰۱ ۰۹:۱۴:۰۱

ولی بعضی وقت‌ها اعتبارسنجی مقادیر ورودی آنقدر زمان‌گیر هستند و پیچیده که بهتر است بگذاریم خطا به وجود بیاید و وقتی به وجود آمد آن را «مدیریت» کنیم.

نویسنده: SirAsad
تاریخ: ۱۳۸۸/۰۶/۰۱ ۱۰:۰۰:۲۱

ولی بعضا نمیشه کل خطاها رو کنترل کردبه خصوص در برنامه های Enterprise چونکه اگر بخوای مدیریت کنی باید به قول امریکن ها Rest of you life ات رو IF بنویسی .

عنوان: آشنایی با Defensive programming - قسمت دوم
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۰۱ ۱۷:۵۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

در ادامه یک سری از خط مشی‌های متداول در defensive programming را با هم مرور خواهیم کرد:

1- بررسی نال بودن اشیاء
سعی در استفاده از اشیاء نال، به یک NullReferenceException منتهی خواهد شد. اگر به هر دلیلی امکان نال بودن یک شیء وجود داشت، پیش از استفاده از آن، حتما این وضعیت را بررسی نمائید.
بهترین ابزاری هم که برای این منظور می‌توان استفاده کرد، نگارش جدید افزونه‌ی ReSharper است که زیر شیءایی را که احتمال نال بودن آن می‌رود یک خط آبی رنگ می‌کشد.

```
using (SqlDataReader myReader = sqlCommand.ExecuteReader())
{
    while (myReader.Read())
        {   Possible 'System.NullReferenceException'
```

2- بررسی آرگومان‌های دریافتی
برای نمونه اگر متد شما تاریخی را بر اساس DateTime دریافت می‌کند، حتما حدود آن‌را بررسی نمائید. برای مثال دریافت تاریخ 31 اسفند از کاربر، به یک ArgumentOutOfRangeException منتهی خواهد شد. بنابراین آرگومان‌های دریافت شده باید انتظارات مربوطه را برآورده کنند و پیش از استفاده حتما بررسی گردند تا بتوان مشکلات را به کاربر به صورت واضحی گوشزد کرد. (خطای ArgumentOutOfRangeException برای کاربر نهایی بی‌معنی است)

3- وضعیت اشیاء را بررسی کنید
برای مثال بستن یک کانکشن به دیتابیس در صورت بسته بودن آن، به یک InvalidOperationException منتهی می‌شود. بنابراین بهتر است ابتدا وضعیت این شیء بررسی شده و سپس عملیات خاصی بر روی آن صورت گیرد.

4- هنگام کار با آرایه‌ها دقت کنید
اگر اندیس فراخوانی شده کمتر از صفر یا بیشتر از اندازه‌ی آرایه باشد به یک IndexOutOfRangeException بر خواهید خورد. بنابراین همواره بهتر است که این بررسی پیش از بروز واقعه انجام شود.

5- مراقب الگوریتم‌های بازگشتی باشید
هر چند متدهای بازگشتی در بعضی از موارد کار راه انداز هستند اما اگر بدون دقت از آن‌ها استفاده شود ممکن است سبب ایجاد یک حلقه‌ی بی نهایت شده و نهایتا برنامه با یک StackOverFlowException خاتمه می‌یابد (این مورد در دات نت فریم ورک تنها حالتی است که با try و catch قابل مهار نیست).

6- هنگام تبدیل اشیایی از نوع object مراقب باشید
اگر قصد تبدیل یک شیء را به نوعی داشته باشید که با مقدار ذخیره شده در آن مطابقت ندارد، یک InvalidCastException حاصل خواهد شد. بنابراین در اینگونه موارد بهتر است از اپراتورهای as و یا is استفاده کنید. هنگام استفاده از as اگر عملیات

تبدیل با موفقیت صورت نگیرد، حاصل عملیات تنها یک شیء نال خواهد بود و استثنایی رخ نخواهد داد.

-7 بجای متد Parse از TryParse استفاده کنید
برای مثال در دات نت جهت تبدیل یک رشته به مقداری عددی می‌توان از int.Parse و یا int.TryParse استفاده کرد. در حالت اول اگر عملیات تبدیل میسر نباشد حتما یک FormatException رخ خواهد داد اما در حالت دوم در صورت موفقیت آمیز بودن عملیات تبدیل، خروجی true خواهد بود و حاصل اصلی را با یک پارامتر از نوع out در اختیار شما قرار می‌دهد.

و به صورت خلاصه
- ورودی‌های کاربر را محدود کرده (مثلا اگر قرار است عددی را وارد کند، از یک تکست باکس عددی (masked edit controls) استفاده کنید) و یا آن‌ها را دقیقا بررسی نمائید تا احتمال بروز خطاهای بعدی را کاهش دهید.
- زمانیکه می‌توان از بروز یک exception جلوگیری کرد، بهتر است مدیریت آن‌را به قسمت catch بلاک try/catch واگذار نکرد.
- و هنگامیکه با برنامه نویسی نمی‌توانید تمامی خطاهای ممکن را پیش بینی کرده و آن‌ها را مدیریت کنید، برای مدیریت استثناءها برنامه داشته باشید.

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۶/۰۱ ۲۳:۵۶:۱۷

سلام
خسته نباشید من تازه از جاوا به c# اومدم (یه ماهی هست شروع کردم) وبلاگ خیلی پرمحتوا و فعالی دارین
می خواستم اگر ممکن باشه در مورد masked edit controls یکم توضیح بدین.
مرسی و ممنون.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۰۲ ۰۰:۰۳:۴۸

سلام
یک نمونه masked edit control با استفاده از یکی از پلاگین‌های jQuery برای ASP.Net درست کرده‌ام که می‌تونید شرح آن‌را در آدرس زیر ملاحظه کنید
http://www.dotnettips.info/2008/11/jquery-aspnet.html

عنوان: **رفع مشکل initializing toolbox هنگام باز کردن یک پروژه وب**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۰۳ ۱۴:۲۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

بعد از نصب مجموعه AJAX Control Toolkit و همچنین نمونه‌های مشابه موجود برای سیلورلایت و WPF دو مشکل برای VS2008 من رخ داد :

1- از کار افتادن نوار ابزار کنترل‌ها (کار نمی‌کرد! نمی‌شد روی کنترلی کلیک کرد)
2- به ازای باز کردن هر صفحه aspx و امثال آن در IDE ، حدود یک دقیقه در پایین صفحه، سمت چپ نوشته می‌شد، initializing toolbox و بعد هم IDE تقریبا هنگ می‌کرد.

نحوه رفع مشکل:
روی tool box کلیک راست کرده و گزینه reset toolbox را انتخاب کنید.
حدود 10 دقیقه با مصرف CPU نزدیک به 100 درصد همه چیز مرتبط و مثل روز اول می‌شود.
بعد از این، باز کردن فایل‌های پروژه فقط با یک کلیک امکان پذیر شده و مشکل هنگ IDE برطرف می‌شود.

البته VS 2008 خام با پروژه‌های وب خیلی مشکل دارد و عملا بعد از نصب SP1 آن کاملا روان شده و درست کار می‌کند.

نظرات خوانندگان

نویسنده: MDP
تاریخ: ۱۳۸۸/۰۶/۰۳ ۱۵:۱۲:۴۰

این مشکل رو من قبلا از طریق خود ماکروسافت دنبال کردم.

نتیجه ای نداشت و منم به همین روش رسیدم ولی خودم حدس می زنم دلیلش اینه که وقتی توی پروژه های اپن سورس هرکسی تول باکس رو یه جور کاستو مایز می کنو این مشکل پیش میاد و VS موقع لود گیج میزنه.

تازه مشکل بعدی هم توی اینه که آبجکت هایی که ساخته میشه باید رندر بشن توی Toolbox جا داده بشن.

عنوان: **دریافت کتاب از Google books**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۰۳ ۱۳:۵۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

پیرو مطلب " آیا نمودارهای UML هنوز هم استفاده‌ی صنعتی گسترده‌ای دارند یا خیر؟ " در کامنت‌های این مطلب، اکثرا عنوان می‌کردند که از CRC cards استفاده می‌کنند. کتاب معروف CRC cards هم به نام The CRC card book مربوط به سال 1997 است و در کتاب فروشی‌های این دور و اطراف یافت نشد (یا حداقل من نیافتم). اما این کتاب در Google books موجود است .

برنامه رایگان و سورس بازی برای این منظور در CodePlex موجود است که پس از دریافت آدرس کتاب، کل آن‌را از Google books دریافت کرده و یک خروجی pdf و یا تصویری ارائه می‌دهد.

Google book downloader

روش استفاده
نکته‌ای برای غیر ساکنین آمریکا

آنچنان سرعت بالایی برای دریافت یک کتاب ندارد اما کار می‌کند (برای دریافت یک کتاب شاید نصف روز معطل شوید).

نظرات خوانندگان

نویسنده: Mohammad Shams Javi
تاریخ: ۱۳۸۸/۰۷/۰۳ ۱۵:۳۴:۲۵

سلام
جالب بود مواجه شدن با محدودیت، برای هیچ کس جالب نیست.

ضمنا قابلیت انتخاب فرمت نمایش تاریخ به ShamsiXp 0.25 اضافه شده و سورس کد آن نیز پیوست شده است.
(http://mohammadshams.blogspot.com/2009/09/shamsi-xp-025-source-code.html)

برای ادامه این کار به شکل صحیح و اضافه کردن Calendar Type فارسی به استانداردهای ویندوز، تاپیکی در بخش امنیت نرم‌افزار برنامه‌نویس اضافه شد.
امیدوارم با همکاری دیگر دوستان بتوان این "محدودیت" را هم از بین برد.
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۰۳ ۱۶:۰۱:۴۱

آقای شمس،
ممنون. کار بزرگی انجام دادید، خصوصا انتشار سورس آن.
دید صرفا تجاری به این مساله که به نوعی ملی هم محسوب می‌شود سبب شده از زمان ویندوز 95 تا الان، ما هنوز درگیر این مساله باشیم و هر شرکتی برای خودش یک پکیج ارائه داده و ...

نویسنده: محمد امین شریفی
تاریخ: ۱۳۸۸/۰۷/۰۳ ۱۷:۰۵:۰۳

اگر خواستید قسمتی از کتاب را از گوگل دریافت کنید و یا فصلی از کتاب و یا همه آن،راه حل ساده آن اینست که وارد temporary مرورگر شود(جایی که اطلاعات کش مرورگر ذخیره میشود و بعد از مدتی پاک می شود).و آن را در قالب تصویر دریافت کنید.
تازه می توانید کیفیت کتاب را هم مشخص کنید،به این دلیل که زوم کتاب سمت سرور می باشد،کیفت و حجم آن هم قابل تنظیم می باشد.
در ضمن در یاد داشته باشید که هر فایلی که بخواهد در مرورگر به نمایش در بیاید،قابل ذخیره کردن است :D

نویسنده: محمد امین شریفی
تاریخ: ۱۳۸۸/۱۰/۱۱ ۲۲:۴۹:۳۹

سلام،
جناب مهندس نصیری،پروژه ِ ی فوق الذکر از codeplex حذف شده است،در صورت امکان برنامه را آپلود کنید.

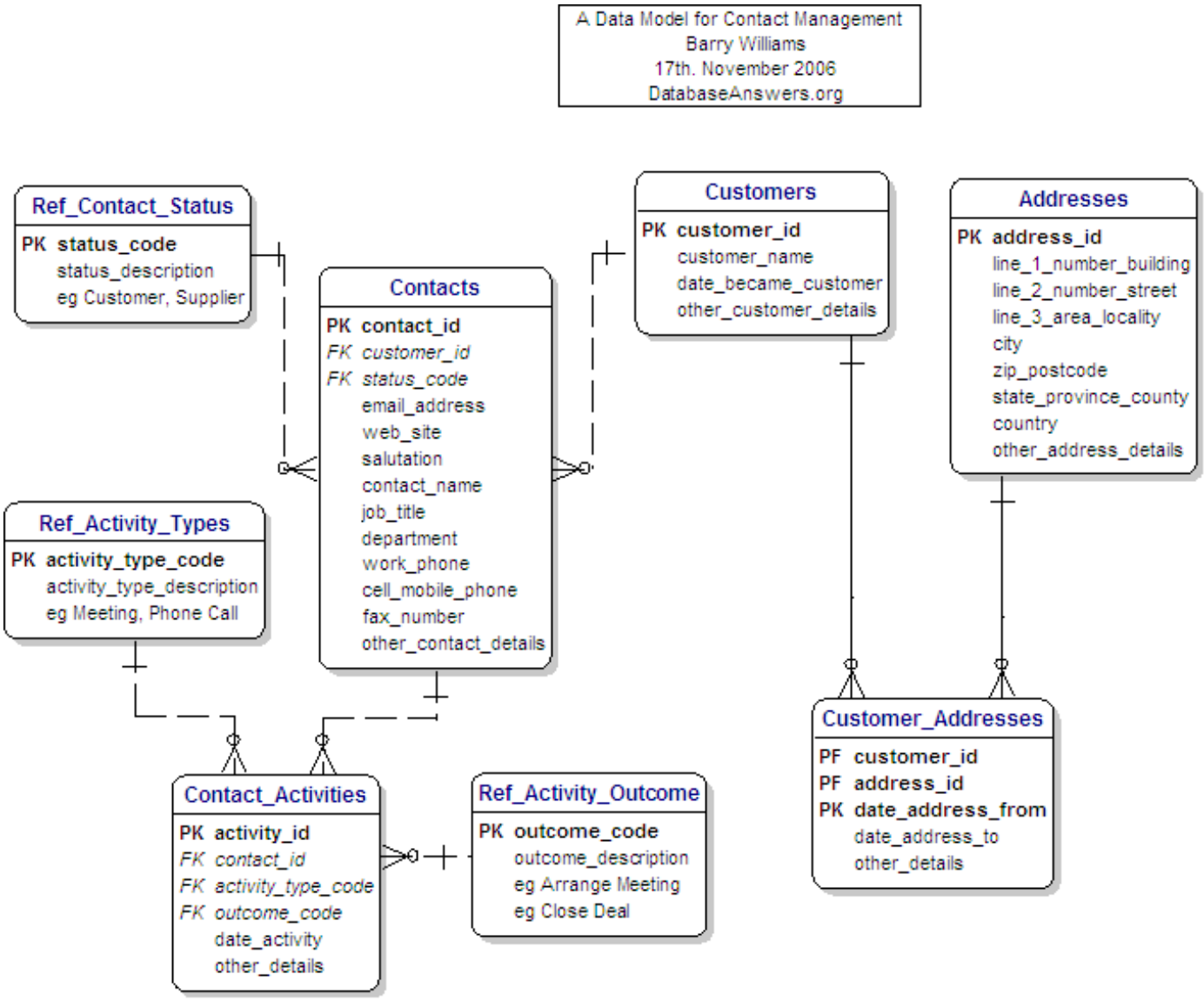
با سپاس،

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۰/۱۱ ۲۳:۵۳:۴۳

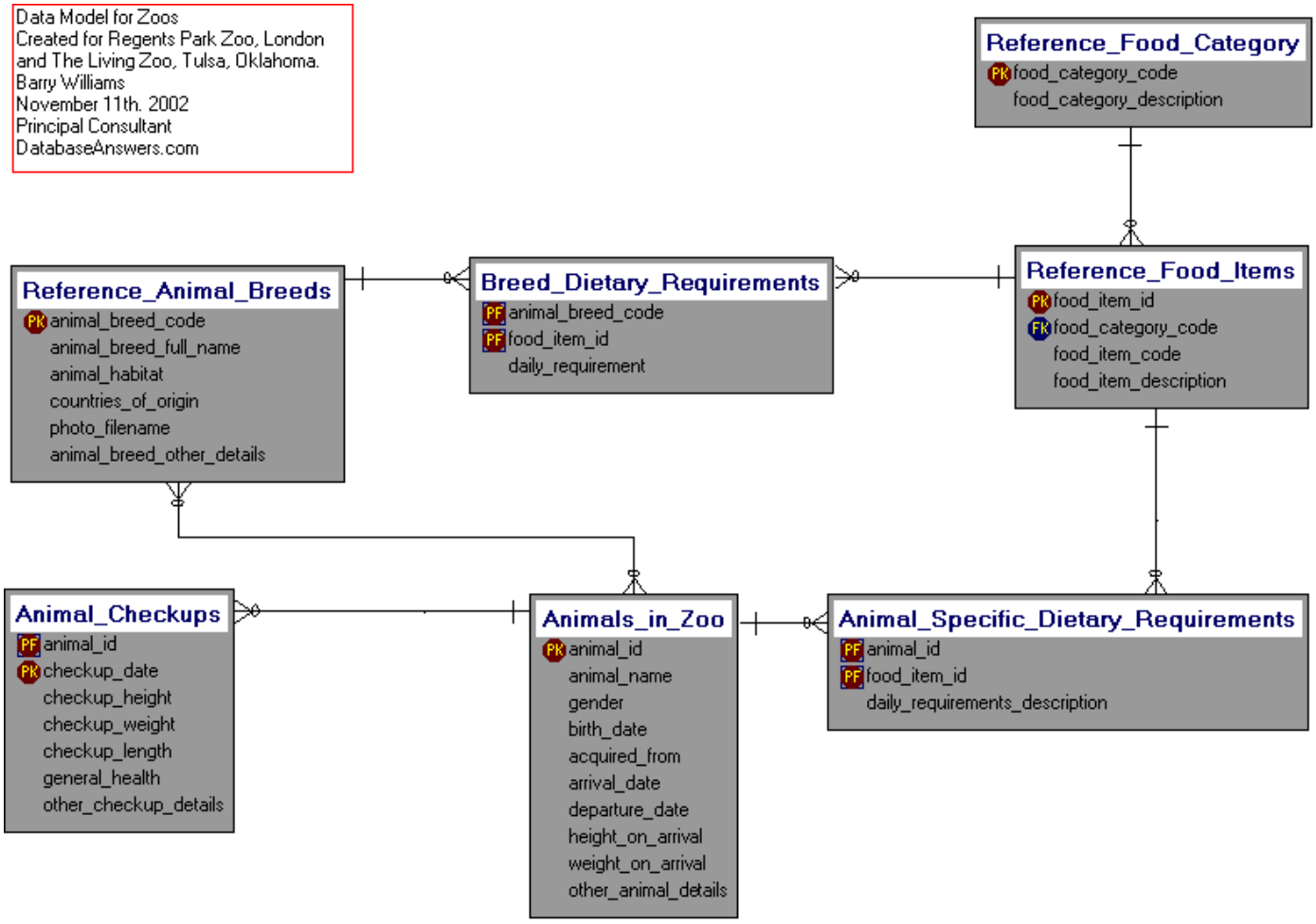
سلام
آپلود شد:
http://rapidshare.com/files/328924251/GBDownloader-0.6.9.33532.rar

سایت databaseanswers.org قسمتی را دارد تحت عنوان Data models که طراحی دیتابیس بیش از 600 سیستم را جمع آوری کرده است.

از data model سیستم مدیریت ارتباطات



تا data model یک باغ وحش



مشاهده سایت databaseanswers.org

نظرات خوانندگان

نویسنده: Mohammad Shams Javi
تاریخ: ۱۳۸۸/۰۸/۲۷ ۲۳:۱۹:۴۸

خیلی مفید بود. متشکرم.

آیا اینچنین مجموعه جامعی در مورد استراتژیهای طراحی GUI برنامه، ایجاد تعامل مفید بین کاربر و واسط کاربری و از این دست مطالب سراغ ندارید؟؟ (مسلما با توجه به نوع برنامه و سطح مخاطب آن، طراحی کاربر باید متفاوت باشد)

فکر میکنم لازم باشد که خودمان، اینچنین مجموعه‌ای از تصاویر واسطهای کاربری، در برنامه‌های موفق (فارسی) تهیه و در دسترس عموم قرار بدهیم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۸/۲۸ ۰۰:۰۶:۳۲

سلام
بله. این سایت واقعا مفید است. خیلی ایده می‌شود از آن استخراج کرد.
و یک سری سایت هستند که به صورت تخصصی کار UI انجام می‌دن. برای مثال:
/http://www.uxbooth.com/view/blog
که چندی قبل در مورد آینده رابطهای کاربری یک سری مطلب جالب منتشر کرده بود:
/http://www.uxbooth.com/blog/the-future-of-interface-design

نویسنده: Mohammad Shams Javi
تاریخ: ۱۳۸۸/۰۸/۲۸ ۱۰:۱۳:۴۰

بله قبلا با UX-Booth آشنا شده بودم.
مدتی است که در فکر جمع‌آوری نمونه‌های GUI موفق در برنامه‌های فارسی و تهیه دستورالعمل مناسبی برای طراحی آن هستم.
راستش از آنجا که از گرافیک سررشته دارم، همیشه با دیدی انتقادی به نما و UI برنامه‌ها نگاه میکنم و متاسفانه فکر میکنم که اکثر GUIهای فارسی، نمونه‌های عینا "راست به چپ" شده GUI های خارجی هستند.
اصولا شکل قلم و تایپوگرافی حروف فارسی، ترکیب‌رنگ، اندازه و شکل متفاوتی را در طراحی رابط میطلبد.

با توجه به تجربه و سابقه شما، میخواستم خواهش کنم موفق‌ترین و مفیدترین UIهای فارسی که تا کنون دیده‌اید به بنده معرفی کنید. اگر تصاویری از آنها در اختیار داشته باشید که فبها.

با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۸/۲۸ ۱۱:۴۲:۵۰

سلام
کلا طراحی رابط کاربر صرفنظر از ایرانی و غیر ایرانی بودن یک سری اصول کلی دارد که فکر نمی‌کنم آنچنان تفاوتی کند و نسخه خاصی برای نمونه ایرانی داشته باشد و اگر نمونه ایرانی هم از اصول جهانی آنها پیروی کند به عنوان نمونه دارای اشکال در نظر گرفته شود.
یک وبلاگ ایرانی هم دیدم که در این زمینه فعالیت می‌کند. لطفا با ایشان هم مشورت کنید:
http://tweex.uxiran.com

عنوان: **استفاده از API ترجمه گوگل**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۹/۰۶ ۱۵:۲۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

مطابق
Ajax API ترجمه گوگل، برای ترجمه یک متن باید محتویات آدرس زیر را تحلیل کرد:

```
http://ajax.googleapis.com/ajax/services/language/translate?v=1.0&q={0}&langpair={1}|{2}
```

که در آن پارامتر اول، متن مورد نظر، پارامترهای 1 و 2 زبان‌های مبدا و مقصد می‌باشند. برای دریافت اطلاعات، ذکر ارجاع دهنده الزامی است (referrer)، اما ذکر کلید API گوگل اختیاری می‌باشد (که هر فرد می‌تواند کلید خاص خود را از گوگل دریافت کند). بنابراین برای استفاده از آن تنها کافی است این URL را تشکیل داده و سپس محتویات خروجی آن‌را آنالیز کرد. فرمت نهایی دریافت شده از نوع JSON است. برای مثال اگر hello world! را به این سرویس ارسال نمائیم، خروجی نهایی JSON دریافت شده به صورت زیر خواهد بود:

```
//{\"responseData\": {\"translatedText\":\"سلام جهان!\"}, \"responseDetails\": null,
\"responseStatus\": 200}
```

در کتابخانه‌ی System.Web.Extensions.dll دات نت فریم ورک سه و نیم، کلاس JavaScriptSerializer برای این منظور پیش بینی شده است. تنها کافی است به متد Deserialize آن، متن JSON دریافتی را پاس کنیم:

```
GoogleAjaxResponse result =
          new JavaScriptSerializer().Deserialize<GoogleAjaxResponse>(jsonGoogleAjaxResponse);
```

برای اینکه عملیات نگاشت اطلاعات متنی JSON به کلاس‌های دات نتی ما با موفقیت صورت گیرد، می‌توان خروجی JSON گوگل را به شکل زیر نمایش داد:

```
//ResponseData.cs file
public class ResponseData
{
  public string translatedText { get; set; }
}

//GoogleAjaxResponse.cs file
using System.Net;

/// <summary>
/// کلاسی جهت نگاشت اطلاعات جی سون دریافتی به آن
/// </summary>
public class GoogleAjaxResponse
{
  public ResponseData responseData { get; set; }
  public object responseDetails { get; set; }
  public HttpStatusCode responseStatus { get; set; }
}
```

با این توضیحات، کلاس نهایی ترجمه گوگل ما به شکل زیر خواهد بود:

```
using System;
using System.Globalization;
using System.IO;
using System.Net;
using System.Web;
using System.Web.Script.Serialization;
```

```
//{\"responseData\": {\"translatedText\":\"سلام جهان!\"}, \"responseDetails\": null,
\"responseStatus\": 200}

public class CGoogleTranslator
{
  #region Fields (1)

  /// <summary>
  /// ارجاع دهنده
  /// </summary>
  private readonly string _referrer;

  #endregion Fields

  #region Constructors (1)

  /// <summary>
  /// مطابق مستندات نیاز به یک ارجاع دهنده اجباری می‌باشد
  /// </summary>
  /// <param name="referrer"></param>
  public CGoogleTranslator(string referrer)
  {
      _referrer = referrer;
  }

  #endregion Constructors

  #region Properties (2)

  /// <summary>
  /// ترجمه از زبان
  /// </summary>
  public CultureInfo FromLanguage { get; set; }

  /// <summary>
  /// ترجمه به زبان
  /// </summary>
  public CultureInfo ToLanguage { get; set; }

  #endregion Properties

  #region Methods (2)

  // Public Methods (1)

  /// <summary>
  /// ترجمه متن با استفاده از موتور ترجمه گوگل
  /// </summary>
  /// <param name="data"></param>
  /// <returns></returns>
  public string TranslateText(string data)
  {
      //ساخت و انکدینگ آدرس مورد نظر
      string url =
          string.Format(

"http://ajax.googleapis.com/ajax/services/language/translate?v=1.0&q={0}&langpair={1}|{2}",
             HttpUtility.UrlEncode(data), //needs a ref. to System.Web.dll
          FromLanguage.TwoLetterISOLanguageName,
          ToLanguage.TwoLetterISOLanguageName
      );

      //دریافت اطلاعات جی سون از گوگل
      string jsonGoogleAjaxResponse = fetchWebPage(url);

      //needs a ref. to System.Web.Extensions.dll
      //نگاشت اطلاعات جی سون دریافت شده به کلاس مرتبط
      GoogleAjaxResponse result =
          new JavaScriptSerializer().Deserialize<GoogleAjaxResponse>(jsonGoogleAjaxResponse);

      if (result != null && result.responseData != null && result.responseStatus == HttpStatusCode.OK)
      {
          return result.responseData.translatedText;
      }
      return string.Empty;
  }
  // Private Methods (1)

  /// <summary>
  /// دریافت محتویات جی سون بازگشتی از گوگل
```

```
  /// </summary>
  /// <param name="url"></param>
  /// <returns></returns>
  string fetchWebPage(string url)
  {
      try
      {
          var uri = new Uri(url);
          if (uri.Scheme == Uri.UriSchemeHttp || uri.Scheme == Uri.UriSchemeHttps)
          {
              var request = WebRequest.Create(uri) as HttpWebRequest;
              if (request != null)
              {
                  request.Method = WebRequestMethods.Http.Get;
                  request.Referer = _referrer;
                  request.UserAgent = "Mozilla/5.0 (Windows; U; Windows NT 5.0; ; rv:1.8.0.7)
Gecko/20060917 Firefox/1.9.0.1";
                  request.AllowAutoRedirect = true;
                  request.Timeout = 1000 * 300;
                  request.KeepAlive = false;
                  request.ReadWriteTimeout = 1000 * 300;
                  request.AutomaticDecompression = DecompressionMethods.GZip |
DecompressionMethods.Deflate;

                  using (var response = request.GetResponse() as HttpWebResponse)
                  {
                      if (response != null)
                      {
                          using (var reader = new StreamReader(response.GetResponseStream()))
                          {
                              return reader.ReadToEnd().Trim();
                          }
                      }
                  }
              }
          }
          return string.Empty;
      }
      catch (Exception ex)
      {
          Console.WriteLine(String.Format("fetchWebPage: {0} >> {1}", ex.Message, url), true);
          return string.Empty;
      }
  }

  #endregion Methods
}
```

مثالی در مورد نحوه‌ی استفاده از آن برای ترجمه یک متن از انگلیسی به فارسی:

```
string res = new CGoogleTranslator("http://www.dotnettips.info/")
  {
     FromLanguage = CultureInfo.GetCultureInfo("en-US"),
     ToLanguage = CultureInfo.GetCultureInfo("fa-IR")
   }.TranslateText("Hello world!");
```

نظرات خوانندگان

نویسنده: Parham,پرهام
تاریخ: ۱۳۸۸/۰۹/۰۷ ۱۶:۰۷:۴۰

از بخش برنامه نویسی آن که جای تشکر بسیار داره و عالی بود که بگذریم. یک بار هم نشده با این مترجم گوگل کار کنم و ترجمه خوب بگیرم.

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۷/۱۶ ۱۳:۴۶

امروز خواستم از این API برای برنامه ای که قبلا بوده استفاده کنم متوجه شدم که پولی شده، متاسفانه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۱۶ ۱۳:۵۴

مترجم مایکروسافت زبان فارسی رو هم مدتی است که اضافه کرده:

http://www.bing.com/translator

http://msdn.microsoft.com/en-us/library/ff512404.aspx

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۷/۲۰ ۹:۱۳

مترجم مایکروسافت هم روی کلمات فارسی خطا میده و میگه که پارامتر دوم که fa هست رو درست وارد نکردی در حالی که من این پارامتر رو از روی زبانهای در دسترسی که یکی از فانکشنهای خود api برمیگردونه انتخاب میکنم.
نکته اینجاست که وقتی پارامتر دوم رو مثلا با عربی ar عوض میکنم مثل قرقی کار ترجمه رو انجام میده.
نکته آخر اینه که برای ترجمه تا دومیلیون کاراکتر برای لایسنس رایگانش در ماه محدودیت داره.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۲۰ ۱۲:۲۲

شما برای استفاده از مترجم گوگل یا بینگ حتما نیازی به استفاده از API آن‌ها ندارید. خیلی‌ها از طریق یک HttpWebRequest و سپس آنالیز خروجی مثلا با Regular expressions و یا نمونه بهتر استفاده از html agility pack اقدام می‌کنند. این روش رسمی یا توصیه شده‌ای نیست. اما محدودیتی هم ندارد. اگر می‌خواهید سیستم آن‌ها را مهندسی معکوس کنید می‌شود با برنامه Fiddler این کار رو کرد. تمام رفت و برگشت‌ها را ثبت می‌کند. بعد بر این اساس، اقدام به آنالیز سیستم و تهیه یک برنامه سفارشی کنید.
برای نمونه این لینک بینگ در آزمایش امروز بود: (^)

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۷/۲۰ ۲۳:۱۶

ممنون بابت لینکی که داده بودید.

عنوان: **مشکل فایل‌های ANSI-Windows-1256 با VS.Net در ویندوز 7**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۹/۲۰ ۱۳:۰۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

در ویندوز XP زمانیکه زبان سیستم و همچنین کشور جاری به ایران تنظیم شود، VS.Net فایل‌های ANSI را از نوع ANSI-Windows-1256 (یا همان ANSI-Arabic) در نظر می‌گیرد و مشکلی هم برای ذخیره داده‌های یونیکد در این نوع فایل‌های ANSI ویژه نخواهد بود (الزامی وجود ندارد که این فایل‌ها حتما به فرمت UTF8 ذخیره شوند). اما در ویندوز 7 با همان تنظیمات، VS.Net این فایل‌ها را با encoding از نوع windows-1252 تشخیص می‌دهد و پس از کامپایل برنامه‌ای که قبلا مشکل نداشت، این‌بار همه چیز به همه ریخته خواهد بود. شاید اینطور به نظر برسد که این فایل‌ها خراب شده‌اند، ولی خیر. مشکلی وجود ندارد؛ فقط فرمت encoding خواندن آن‌ها باید windows-1256 باشد (و نه 1252) و گرنه تخریب شده به نظر می‌رسند.

```
protected void Page_Load(object sender, EventArgs e)
{
    Page.Title = "ÊÚÑíÝ ãÏíÑÇä";
```

تعداد فایل‌ها هم زیاد است و نیاز به یک روش سریع برای رفع این مشکل وجود داشت.
بنابراین سه عملیات باید صورت گیرد:
لیست کردن تمام فایل‌های مورد نظر (فایل‌های cs و aspx و امثال آن فقط)
پیدا کردن encoding جاری فایل‌ها : کدامیک از آن‌ها با فرمت UTF-8 ذخیره نشده‌اند؟
تبدیل به یونیکد: خواندن این فایل‌های غیر یونیکد یافت شده با فرمت windows-1256 و سپس ذخیره مجدد با فرمت UTF-8

که خلاصه روش انجام کار به صورت زیر است:

الف) آیا فایل جاری مورد نظر با فرمت UTF-8 with signature ذخیره شده است؟
این signature در مورد فایل‌های UTF-8 به سه بایت اول فایل بر می‌گردد که اصطلاحا byte-order mark یا BOM گفته می‌شود و باید مساوی EFBBBF باشد. چون فایل‌های ANSI این امضا را ندارند، در یک سیستم ممکن است 1256 خوانده شوند و در یک سیستم دیگر 1252 یا نوع‌های ANSI دیگر بسته به تنظیمات جاری سیستم و مشکل اصلی از VS.Net نیست.

```
/// <summary>
    /// آیا فایل مورد نظر با فرمت یونیکد دارای امضا ذخیره شده است؟
    /// </summary>
    /// <param name="filePath">فایل ورودی</param>
    /// <returns>بله یا خیر</returns>
    public static bool IsUTF8(string filePath)
    {
        using (FileStream file = new FileStream(filePath,
            FileMode.Open, FileAccess.Read, FileShare.Read))
        {
            if (file.CanSeek)
            {
                byte[] bom = new byte[4]; // Get the byte-order mark, if there is one
                file.Read(bom, 0, 4);
                if ((bom[0] == 0xef && bom[1] == 0xbb && bom[2] == 0xbf)) // utf-8
                {
                    return true;
                }
                else
                {
                    return false;
```

```
                }
            }
            else
            {
                //احتمالا فایل باینری است
                return false;
            }
        }
    }
```

ب) خواندن یک فایل ANSI عربی با فرمت windows-1256 بدون تخریب اطلاعات آن و سپس ذخیره سازی با فرمت UTF-8

```
/// <summary>
    /// تبدیل یک فایل انسی عربی به یونیکد دارای امضاء
    /// </summary>
    /// <param name="path">مسیر ورودی</param>
    public static void FixWindows1256(string path)
    {
        try
        {
            //باز کردن فایل با فرمت انسی عربی و تبدیل به یونیکد
            string data = File.ReadAllText(path, Encoding.GetEncoding("windows-1256"));
            //نوشتن حاصل یونیکد در جای قبلی با فرمت مربوطه
            File.WriteAllText(path, data, Encoding.UTF8);
        }
        catch (Exception ex)
        {
            //دسترسی وجود ندارد یا امثال آن
            Console.WriteLine(ex.ToString());
        }
    }
```

پ.ن.

جالب اینجا است که این نوع فایل‌های ANSI عربی در وب زیاد پیدا می‌شوند. برای مثال اینجا کلیک کنید. تمام این نوع فایل‌ها را با متد فوق می‌توان بدون تخریب اطلاعات به فرمت UTF-8 دارای امضاء اصلاح کرد.

نظرات خوانندگان

نویسنده: Hossein Moradinia
تاریخ: ۱۳۸۸/۰۹/۲۰ ۱۷:۳۱:۱۸

بازم تشکر به خاطر زحمت زیادی که می کشید

نویسنده: MDP
تاریخ: ۱۳۸۸/۰۹/۲۰ ۱۹:۱۶:۴۸

سلام خسته نباشید.

منم 3 روز سویچ کردم روی ویندوز سون.با فارسی یکم مشکل داره.

خوشحال میشم راه حل های مشکلاتی که بر می خورید رو بلاگ کنید.صد در صد استفاده های زیادی خواهد داشت.

عنوان: **چند نکته کوتاه و عمومی در مورد قبل و بعد از نصب ویندوز 7**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۹/۲۳ ۱۷:۳۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

- فایل ISO ایی که از سایت‌های این دور و اطراف قابل دریافت است، یک DVD از نوع bootable به شما ارائه نخواهد داد. بنابراین نیاز است تا این قابلیت را فراهم کرد. خود مایکروسافت برنامه سورس بازی را در آدرس زیر جهت تهیه یک bootable DVD یا حتی bootable flash drive از روی فایل ISO دریافت شده ارائه داده است:

Windows 7 USB/DVD Download Tool

- نیاز است تا کلیه افزونه‌ها، کلمات عبور ذخیره شده و غیره‌ی فایرفاکس را بتوان به سیستم جدید انتقال داد. ابزار رایگان و سورس باز MozBackup این‌کار را به بهترین نحو ممکن میسر می‌سازد: +

- پس از نصب، نیاز به صفحه کلید فارسی مرغوب می‌باشد : + و +

- پس از نصب ویندوز 7 ، درایو ویندوز قبلی موجود شما ناپدید شده است! نحوه‌ی رفع مشکل : +

- هر چند در پیدا کردن درایورها از اینترنت مشکلی ندارد، اما بد نیست یکبار هم سیستم را با ابزار رایگانی جهت به روز بودن درایورها بررسی کرد: +

- اگر هنگام کار با Virtual PC مایکروسافت هنگام اتصال به VPN با خطای 721 متوقف شدید، نگران نباشید. با انتخاب گزینه‌ی run as admin این مشکل برطرف می‌شود.

- تنظیم DNS جدید گوگل نیز توصیه می‌شود: +

و یک گجت تقویم فارسی هم کفایت می‌کند : + و یا +

نظرات خوانندگان

نویسنده: مهدی پایروند
تاریخ: ۱۳۸۸/۰۹/۲۴ ۱۲:۵۲:۲۱

سلام برای اجرای Vitual Pc احتیاج به نصب Windows Xp Mode پیدا نکردید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۹/۲۴ ۱۳:۴۵:۰۴

سلام
نه. بدون مشکل نصب می‌شود و از آن دارم استفاده می‌کنم. فقط برای ویندوز 64 بیتی حتما نسخه‌ی 64 بیتی را لازم دارد و نسخه‌ی 32 بیتی آن نصب نمی‌شود:
http://www.microsoft.com/downloads/details.aspx?FamilyId=28C97D22-6EB8-4A09-A7F7-F6C7A1F000B5&displaylang=en

نویسنده: Kianoosh
تاریخ: ۱۳۸۸/۱۰/۰۹ ۱۷:۱۲:۳۸

من به تازگی یک ویندوز 7 از یکی از شرکت های داخلی خریداری کرده ام که دارای مشکلات زیر است:
- هنگام restart کردن هنگ میکند و صفحه آبی رنگ ظاهر میشود
- هنگام نصب یک نرم افزار درایو c را به عنوان درایو پیش فرض معرفی میکند در حالی که من ویندوز 7 را در درایو d و در کنار ویندوز xp نصب کرده ام.

شما ویندوز 7 را دانلود کرده اید؟ اگر اینطور است و مشکلی ندارد آدرس سایت را معرفی نمائید تا بتوان آنرا دانلود نمود.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۰/۰۹ ۱۸:۱۶:۳۶

- شرکت‌های داخلی فقط کار بسته بندی رو انجام می‌دن و پخش. بنابراین زیاد نگران دستکاری در اصل محصول نباشید (و ویندوز من و شما فرقی ندارد)، البته شاید یک سری برنامه یا آپدیت را پس از نصب به صورت خودکار هم نصب کنند.
- در مورد صفحات آبی، عمده‌ی مشکل، درایورهای قدیمی یا نصب برنامه‌های قدیمی و ناسازگار است.
در کل برای تشخیص آن نیاز به memory dump سیستم است. در ویندوز هفت به صورت پیش فرض در این مسیر تشکیل می‌شود:
SystemRoot->MEMORY.DMP
فایل MEMORY.DMP را جستجو و آپلود کنید تا بشود آنالیز کرد.
در ویندوزهای قبلی یک پوشه به نام minidump داخل ویندوز بود که این فایل‌های دامپ در آن تشکیل می‌شد.

- درایوی که ویندوز در آن نصب می‌شود درایو سی ممکن است تشخیص داده شود و مشکلی هم نیست.

نویسنده: Kianoosh Alesheikh
تاریخ: ۱۳۸۸/۱۰/۱۰ ۰۸:۱۳:۴۹

ممنون از راهنمائی تون

عنوان: **فایرفاکس 3.5.6 و غیرفعال شدن افزونه‌ها**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۹/۲۶ ۱۱:۴۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

امروز فایرفاکس 3.5.6 به صورت خودکار نصب شد؛ پس از نصب هم هیچکدام از افزونه‌های نصب شده ظاهر نشدند. به عبارتی به نظر همه‌ی آن‌ها غیرفعال شده بودند. اگر هم قرار باشد از فایرفاکس بدون افزونه استفاده کرد، استفاده از IE8، هم از نظر میزان مصرف حافظه و هم از نظر تعداد باگ‌های امنیتی کمتر گزارش شده (مطابق آمار) ارجحیت بالاتری دارد.
پس از اندکی جستجو مشخص شد که کاربران دیگری هم به این مشکل دچار شده‌اند.

راه حل ساده‌ای هم دارد:
فایرفاکس را بسته و پوشه‌ی Profiles مربوط به فایرفاکس را پیدا کنید. برای مثال:
C:\Users\Vahid\AppData\Roaming\Mozilla\Firefox\Profiles
در این پوشه و زیر پوشه‌های آن چهار فایل زیر را پیدا کرده و به یک پوشه دیگر منتقل کرده و یا حذف کنید:
extensions.cache
extensions.ini
extensions.rdf
compatibility.ini

اکنون با اجرای فایرفاکس این فایل‌ها مجددا تولید شده و مشکلات مربوطه هم برطرف می‌شود.
پس از تولید خودکار مجدد این فایل‌ها، مجددا افزونه‌ها ظاهر شدند و همه چیز مثل قبل شد!

نظرات خوانندگان

نویسنده: محمد حسین ربیعی
تاریخ: ۱۳۸۸/۰۹/۲۶ ۱۲:۲۷:۱۶

خوبه ولی برای من این طوری نشد

نویسنده: kh
تاریخ: ۱۳۸۸/۰۹/۲۷ ۱۲:۱۹:۲۵

با سلام
در این مطلب در مورد آماری از میزان مصرف حافظه کمتر ie نسبت به فایرفاکس صحبت شده است. لطفاً منبع این آمار را نیز معرفی کنید.
با تشکر. رامین.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۹/۲۷ ۱۳:۴۴:۴۲

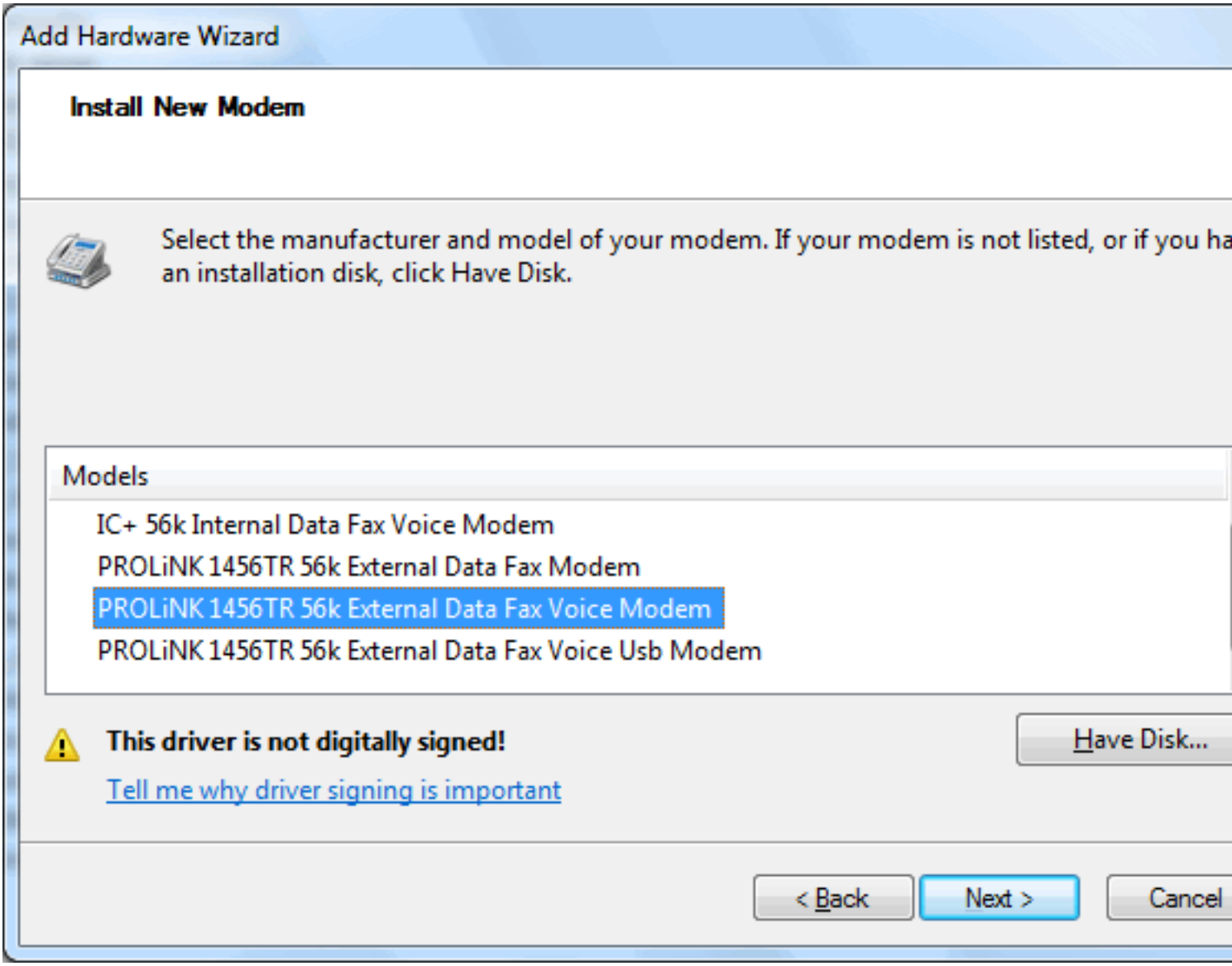
منبع این مورد میزان مصرف حافظه، تجربیات شخصی بنده در کار با IE8 و همچنین فایرفاکس است.
+ Firefox Memory Hog را در گوگل جستجو کنید.

عنوان: **سخت افزارهای قدیمی و ویندوز 7**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۰/۲۳ ۱۴:۰۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

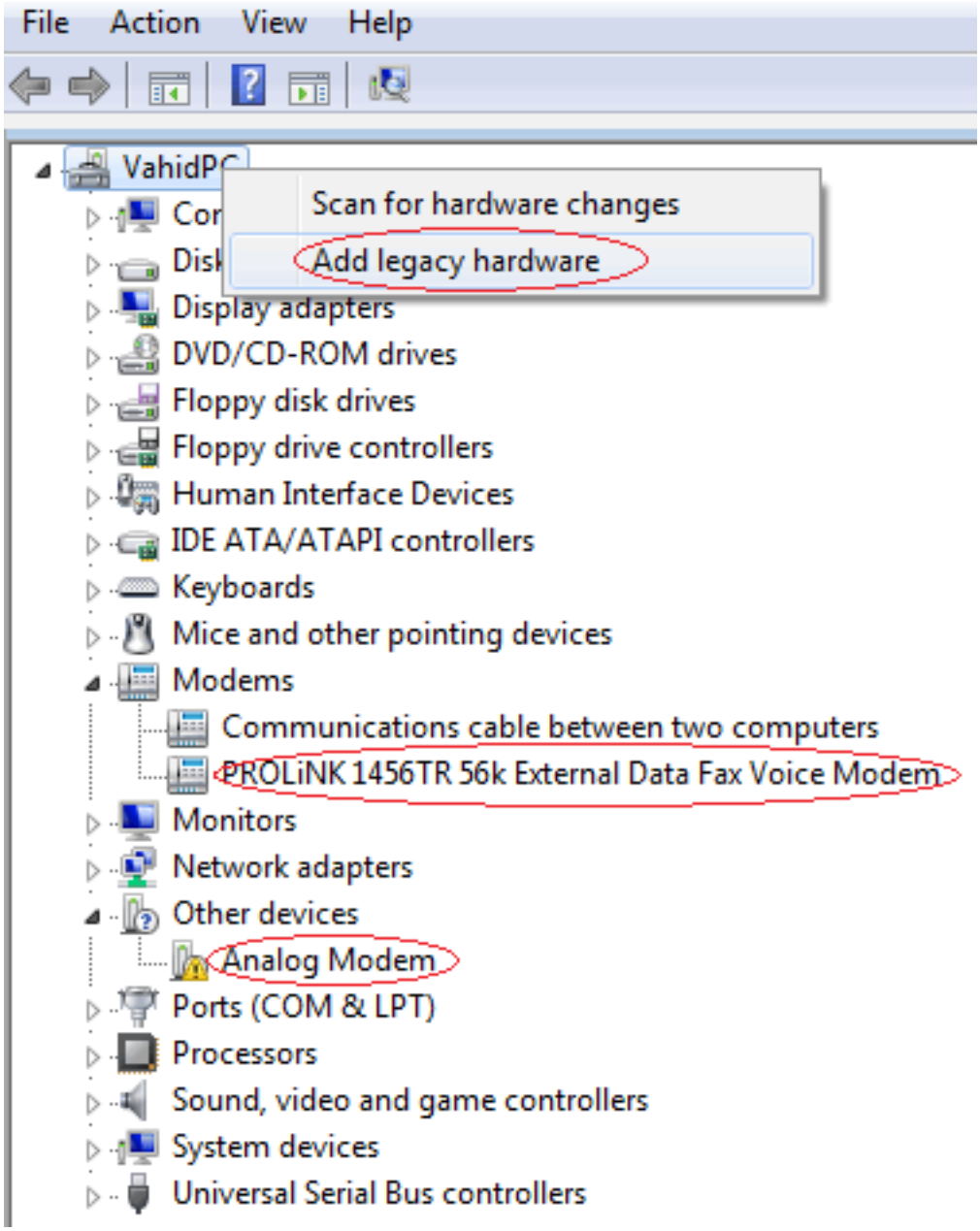
یک مودم d-link قدیمی external دارم (DFM-560E) که آخرین درایور آن مربوط به ویندوز XP، آن هم 32 بیتی است.
با نصب ویندوز 7 64 بیتی این مودم به صورت یک analog modem غیرقابل استفاده در device manager شناسایی شد. هر چند ویندوز 7 توانایی یافتن آخرین درایورهای مرتبط را از اینترنت و سایت مایکروسافت دارد اما چون مطابق جستجویی که انجام دادم این نوع مودم d-link صرفا برای خارج از آمریکا تهیه و تولید شده، دیگر درایوری برای آن تهیه نخواهد شد. خوب، خدا را شکر!
این مودم، مستعمل ماند تا اینکه در یکی از سایت‌ها، مقاله‌ی زیر را دیدم:

How to Configure Legacy Devices to Run on Windows 7

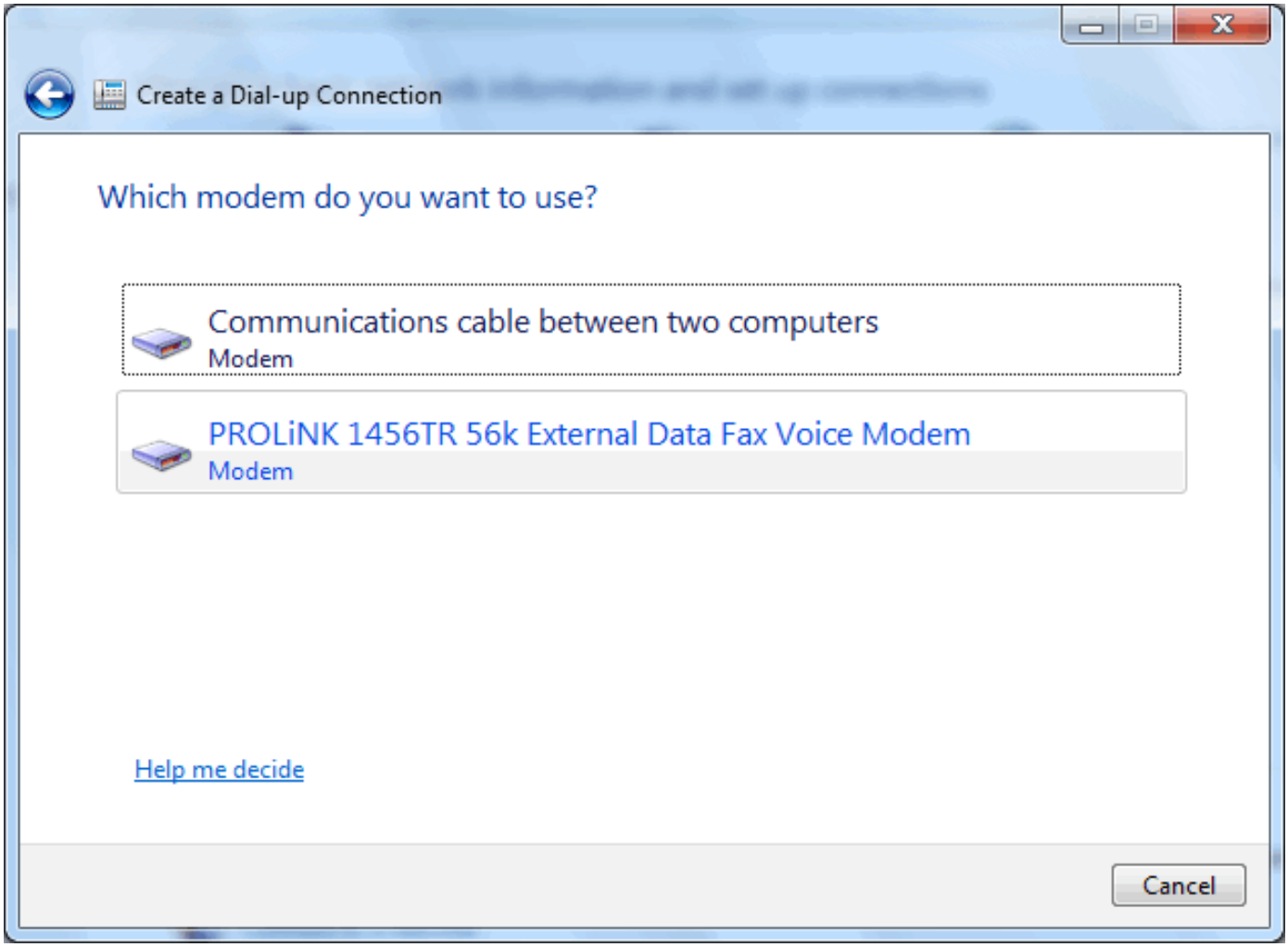
این مقاله در مورد همان گزینه‌ی add new hardware ویندوزهای قبلی که در اینجا به نام add legacy hardware تغییر نام یافته، صحبت می‌کند.
در device manager ویندوز 7 از منوی Actions گزینه‌ی add legacy hardware را انتخاب کنید تا همان ویزاد آشنای add new hardware ویندوز XP ظاهر شود. تمام مراحل آن هم مانند قبل است.
در مرحله‌ی have disk مربوط به مودم، فایل زیر را که مربوط به معرفی درایور مودم ویندوز 64 بیتی است به آن معرفی کردم و مشکل حل شد (احتمالا با تمام مودم‌هایی با chipset از نوع Conexant باید کار کند).
دریافت فایل



الان هنوز همان analog modem شناسایی نشده باقی است اما مودم فوق نیز شناسایی شد و بدون مشکل کار می‌کند.



هنگام ساخت کانکشن دایال آپ هم این مودم جدید شناسایی شده و قابل استفاده است.


Create a Dial-up Connection
Which modem do you want to use?
Communications cable between two computers
Modem
PROLiNK 1456TR 56k External Data Fax Voice Modem
Modem
Help me decide
Cancel

نظرات خوانندگان

نویسنده: Kianoosh Alesheikh
تاریخ: ۱۳۸۸/۱۰/۲۴ ۰۰:۴۴:۴۵

مرسی از مقاله جالبتون.
من هم یک عدد از این مودم های موتورالاغ (موتورولای سابق) دارم که امیدوارم از این روش بتونم راه اندازی اش کنم.

ممنونم آقای نصیری عزیز

عنوان: **چرا در سازمان‌ها برنامه‌های وب جایگزین برنامه‌های دسکتاپ شده‌اند (یا می‌شوند)؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۰/۲۹ ۱۹:۴۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

- برنامه‌های وب نیازی به نصب بر روی تک تک کلاینت‌ها و همچنین به روز رسانی مداوم کلاینت‌ها را ندارند. به این صورت مدیریت چند صد کاربر در یک سازمان ساده‌تر از قبل خواهد بود. دیگر نگران این نخواهید بود که آیا فلان کاربر آخرین به روز رسانی‌ها را نصب کرده (دریافت کرده) یا خیر.
- امکان دسترسی از راه دور، برای مثال از طریق اینترنت یا VPN یا RRAS و خطوط دایال آپ (برای مثال دسترسی ساده‌تر دفاتر مختلف یک سازمان به اطلاعات یکدیگر یا امکان داشتن کارکنانی که از راه دور برای شما کار می‌کنند).
- امکان ذخیره سازی داده‌ها در سازمانی دیگر (هاست کردن این برنامه‌ها در محیط‌های ابری(!) (cloud computing) هزینه‌های تهیه و نگهداری سخت افزارهای یک سازمان را نیز کاهش می‌دهند).
- کاهش هزینه‌های سازمان با توجه به اینکه اگر از سرورهای قدرتمندی استفاده شود؛ از یک برنامه‌ی وب چندین هزار یا چند میلیون کاربر می‌توانند استفاده کنند بدون اینکه نگران تامین هزینه مجوز استفاده از برنامه‌ی تهیه شده به ازای هر کاربر باشید.
- امکان دسترسی به برنامه‌ی وب تهیه شده در انواع و اقسام سیستم عامل‌هایی که تنها مجهز به یک مرورگر وب هستند (نتیجه نهایی قابل استفاده مستقل از سکو است). برای مثال این روزها به کمک Silverlight ، Adobe AIR و یا کتابخانه‌های اسکریپتی مانند jQuery و ASP.Net Ajax، بسیاری از توانایی‌های نمایشی برنامه‌های دسکتاپ را در وب نیز می‌توان شاهد بود با این خصوصیت که نتیجه‌ی نهایی مستقل از سکو است.
- در این حالت کلاینت‌ها نیازی به داشتن سخت افزارهای قوی ندارند (که در کاهش هزینه‌های یک سازمان مؤثر است). همچنین این برنامه‌ها مشکلات ناسازگاری با سخت افزارها و نگارش‌های مختلف سیستم عامل‌ها را نیز ندارند. بنابراین یک سازمان می‌تواند بودجه‌ی خود را صرف تهیه‌ی سرورهای بهتری کند.
- کلاینت‌ها با توجه به محدود بودن دسترسی‌های امنیتی اعمالی توسط مرورگرها، امنیت بیشتری خواهند داشت. به همین ترتیب کاربران برای استفاده از این برنامه‌ها نیز نیازی به دسترسی بالا در یک سازمان برای اجرای مرورگر خود نخواهند داشت (کمتر جمله‌ی "من دسترسی ادمین می‌خواهم" را خواهید شنید).
- امکان مونیتور کردن ساده‌تر فعالیت کاربران در برنامه.
- در صورت محافظت از سرور، کدهای شما از خطر دزدیده شدن مصون(تر) هستند.
- مدیریت ساده‌تر و مجتمع اطلاعات تولیدی با توجه به اینکه همه چیز باید بر روی سرور ذخیره شود. به این صورت مدیریت نقل مکان کاربران از یک کامپیوتر به کامپیوتری دیگر نیز ساده‌تر می شود؛ زیرا چیزی را قرار نیست جابجا کنند (نه اطلاعات و نه برنامه را). اگر یکی از کامپیوترهای کلاینت‌ها قابل استفاده نباشد، به سادگی می‌تواند از کامپیوتری دیگر در شبکه استفاده کند، بدون اینکه معطل تیم فنی شود تا برنامه‌ای را برای او نصب و راه اندازی کنند. به علاوه تهیه پشتیبان از اطلاعات سرورها نیز همیشه ساده‌تر است از تهیه پشتیبان از 100 ها کامپیوتر موجود در شبکه.
- اگر خروجی برنامه‌ی وب شما تنها از صفحات وب و جاوا اسکریپت تشکیل شده باشد، امکان دسترسی آن در دستگاه‌های موبایل به سادگی میسر است.

نظرات خوانندگان

نویسنده: peyman naji
تاریخ: ۱۳۸۸/۱۱/۰۱ ۱۴:۴۵:۱۷

سلام مهندس . جالب بود و در اکثر کشور ها با توجه به افزایش سرعت اینترنت به این سمت گرایش داشتند . در آلمان در ایستگاه قطار یا آژانس هواپیمایی و ... به همین شکل هست و یا حداقل از یک وب سرویس بهره میبرند حتی اگر برنامه ویندوز داشته باشند . در آمریکا هم تا اونجا که میدونم silverlight و wpf به این منظور (جهت همسان سازی محیط اینترنت و ویندوز) استفادش رو به رشد هست . راستی آقای نصیری شما رو silverlight مانور نمیدید . یه مدتی هست که از Jquery چیزی نمیگید . منتظر مطالب جدیدتون هستم . موفق باشید .

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۱/۰۱ ۲۳:۵۵:۱۸

سلام
ممنون.
در این موارد کلان، مباحث مقیاس پذیری، caching ، clustering ، mirroring ، load balancing و امثال آن هم حائز اهمیت می‌شود که باز هم برنامه‌های وب و امکانات مهیای برای آن‌ها در این زمینه حرف اول را می‌زنند.

نویسنده: Hossein Moradinia
تاریخ: ۱۳۸۸/۱۱/۰۲ ۱۶:۴۳:۱۶

سلام استاد

من برنامه نویس ویندوز هستم.البته نه خیلی حرفه ای ولی دوست دارم یادگیری خودم رو در این زمینه ادامه بدم.

به نظر شما مانور دادن روی برنامه های ویندوز و برنامه نویسی windows application اشتباهه و باید به سمت برنامه نویسی Web حرکت کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۱/۰۲ ۱۷:۲۶:۳۲

نه. هر کدوم جایگاه خودشون رو دارند.
بحث من مدیریت ساده‌تر بیش از 200 کاربر و کامپیوتر در یک شبکه بود. ساده‌تر شدن به روز رسانی‌ها، مدیریت و تهیه پشتیبان از اطلاعات، دسترسی ریموت به برنامه‌ها، امکان دسترسی به برنامه از سیستم عامل‌های مختلف و کلیه مواردی که ذکر شد. در اینجاها windows application نویسی مقرون به صرفه نیست.
و گرنه نوشتن windows service یا برنامه‌های تک کاربره یا حتی چندکاربره در مقایس کم مثلا یک شرکت 5 نفره ... و امثال آن لزومی ندارد که با وب اپلیکیشن باشد.

عنوان: **تفاوت‌های یک برنامه نویس کارمند با یک برنامه نویس علاقمند**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۱/۲۱ ۱۲:۳۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

اگر در یک محیط کاری به برنامه نویس‌ها دقت کنید دو گروه را به وضوح می‌توان تمایز داد. کسانی که برنامه نویسی می‌کنند تا اموراتشان بگذرد و کسانی که واقعا علاقمند به کارشان و دنیای برنامه نویسی هستند. به گروه اول می‌توان IT worker نام داد و گروه دوم را Software developer نامید.
جدول ذیل تفاوت‌های این دو گروه را بر می‌شمارد:

IT Workers	Software developers
عموما 5 تا 9 ساعت در یک شرکت کار می‌کنند.	عموما 5 تا 9 ساعت در یک شرکت کار کرده و پس از مراجعت به منزل بر روی پروژه‌های شخصی کار می‌کنند.
با اینکه هنوز در همان شرکت مشغول به کار است همیشه مشغول نق زدن است. احتمالا شاید بتواند همان موقعیت کاری را در یک شرکت دیگر نیز کسب کند.	تا زمانیکه شغل فعلی برای او جذابیت دارد به آن ادامه خواهد داد و ترسی از حضور در شرکت‌های دیگر ندارد.
تنها محل یادگیری او همان پروژه‌هایی است که در شرکت وجود دارند یا مشغول به کار بر روی آن‌ها است. دید کاری و آموزشی او تنها به همین موارد خلاصه می‌شود.	به صورت مداوم مشغول خواندن بلاگ‌ها، کتاب‌های جدید و فراگیری نحوه‌ی استفاده از برنامه‌های جدید می‌باشد.
عموما و اکثریت آن‌ها فقط به خاطر کلاس کاری به این رشته روی آورده‌اند و نه اصل کار مربوطه.	به شدت علاقمند به بهبود روش‌های توسعه کاری و همچنین بهبود وضعیت خویش هستند.
اگر احتمالا بلاگی داشته باشند تنها به توضیح همان نق زدن‌های رایج در محیط کار می‌پردازند.	از بلاگ خود در جهت توضیح تجارب کاری و کمک به ارتقای سایر همکاران خود استفاده می‌کنند.
اگر دانشی را کسب می‌کنند تنها محل عرضه‌ی آن جهت پز دادن پیش مدیر پروژه خواهد بود.	بسیار با معلومات اما افتاده حال هستند.
از تغییرات مداوم دنیای IT که در آن قرار دارند هراسان هستند. مدام نق می‌زنند که مگر فاکس پروی 2.6 چه مشکلی دارد که باید از NHibernate استفاده کنند؟! این نوع افراد همیشه می‌گویند که وقت ندارند مطالب جدید را بیاموزند و میل به تحجر و مقاومت در برابر تغییرات در آن‌ها بسیار زیاد است.	در تغییرات روی داده در دنیای IT سهیم بوده و جزئی از آن هستند.
زمانیکه قرار است یک قطعه کد اس کیوال را نمایش دهند از یک برچسب ساده یا یک تکست باکس استفاده می‌کنند. در حدی که فقط به قولی برنامه "کار کند". در همان حدی کار می‌کنند که به آن‌ها حقوق می‌دهند. نه بیشتر.	چند روز وقت می‌گذارند و با روش‌های مختلف syntax highlighting و نمایش زیبای کد آشنا می‌شوند تا کاری را که ارائه می‌دهند مزه‌ی غذای مانده‌ی چند روز قبل را ندهد.

برای مطالعه بیشتر
+ و + و +

نظرات خوانندگان

نویسنده: zamanphp
تاریخ: ۱۳۸۸/۱۱/۲۱ ۱۳:۵۵:۴۴

با این وجود پس من یه Software developers هستم.
فکر کنم فقط مرگ من رو از برنامه نویسی جدا کنه.

نویسنده: کاوه یزدی نژاد
تاریخ: ۱۳۸۸/۱۱/۲۱ ۱۴:۳۴:۴۸

خیلی مطلب جالبی بود. واقعا لذت بردم

نویسنده: Amin
تاریخ: ۱۳۸۸/۱۱/۲۱ ۱۴:۵۵:۰۲

عالی بود

نویسنده: Zoghal
تاریخ: ۱۳۸۸/۱۱/۲۱ ۱۶:۲۹:۴۲

عالی بود . دسته بندی درستی هست

نویسنده: ooghry
تاریخ: ۱۳۸۸/۱۱/۲۱ ۲۲:۳۷:۳۴

خیلی جالب بود

نویسنده: Meysam
تاریخ: ۱۳۸۸/۱۱/۲۲ ۰۹:۱۹:۳۲

به نکته جالبی اشاره کردید، داشتن انگیزه برای یک برنامه نویس، خیلی مهمه

نویسنده: shr6557
تاریخ: ۱۳۸۸/۱۱/۲۳ ۰۹:۵۱:۰۰

می تونستی این مطلب رو هم اضافه کنی
it worker : جنسیت آن می توان مونث یا مذکر باشد
software developer: جنسیت آن نمی تواند مونث باشد

نمی دونم ما که ندیدیم!

نویسنده: Alshimo
تاریخ: ۱۳۸۸/۱۱/۲۴ ۱۰:۱۴:۰۹

shr6557 این چه حرفیه میزنی؟
خانوما خیلی برنامه نویسای خوبی هستن...
من خودم یه نفر رو میشناسم که 1 ساله دات نت کار میکنه... هفته پیش روش اتصال به دیتابیس با SQLConnection رو کشف کرد.

نویسنده: وحید نصیری

تاریخ: ۱۳۸۸/۱۱/۲۴ ۱۰:۲۰:۱۲

در مورد آقایان و خانم‌های شاغل در صنعت IT واقعیت‌ها را در مطلب زیر مطالعه بفرمائید:
http://weblog.radmanitd.com/archives/000378.html

نویسنده: Ali
تاریخ: ۱۳۸۸/۱۱/۲۵ ۱۱:۵۵:۳۶

آقا این لیست شما وسط نداره؟ نه این ور نه اون ور... فقط وسط

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۱/۲۵ ۱۴:۰۲:۳۶

سمت چپ این لیست به نحوی که نوشته شده حد وسط هست. در کل این رشته از تمام رشته‌های دیگر مشکل‌تر است. هر روز باید خود را به روز کرد. هر روز مطالعه ... هر روز تحقیق. و گرنه زود بازنشسته خواهید(خواهیم) شد.

نویسنده: Hossein Moradinia
تاریخ: ۱۳۸۸/۱۱/۲۵ ۱۶:۰۸:۰۲

سلام استاد
به نظر شما تا چند سالگی میشه این مطالعه مداوم را همیشه حفظ کرد؟
البته اینو میدونم که علاقه بسیار تاثیر داره

نویسنده: homa
تاریخ: ۱۳۸۸/۱۲/۰۱ ۲۰:۰۰:۳۹

من بااینکه خانومم ولی فکر میکنم یه چیزی توی مایه های software developer ام البته یه مقداری هم IT worker...

در هر دو گروه به نظرمن مرد و زن هست...

عنوان: **دیتابیس سایت Stack Overflow**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۱/۳۰ ۱۱:۵۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

مدتی است که سایت Stack Overflow دیتابیس سؤال و جواب‌های خود را به صورتی سخاوتمندانه در اختیار عموم قرار داده است . مجوز استفاده از این اطلاعات cc-wiki است، به این معنا که مجاز هستید این اطلاعات را به اشتراک بگذارید (کپی، توزیع و امثال آن)، مجاز هستید این اطلاعات را با اطلاعاتی دیگر ترکیب کرده و کار جدیدی را ارائه دهید؛ با این شرط که فراموش نکنید از بانیان اصلی این کار یاد نموده و همچنین کار تولیدی شما نیز باید بر مبنای همین مجوز cc-wiki باقی بماند.

این دیتابیس که با فرمت xml ارائه شده در حقیقت دامپ اطلاعات عمومی سایت بوده که اطلاعات شخصی کاربران از آن حذف شده است و شامل فایل‌های زیر می‌باشد:

1. badges.xml
2. comments.xml
3. posts.xml
4. users.xml
5. votes.xml

این اطلاعات شامل سایت‌های همخانواده‌ی Super User ، Server Fault، Stack Overflow وMeta Stack Overflow نیز می‌شود.

جهت دریافت آخرین نگارش این مجموعه به آدرس زیر می‌توان مراجعه نمود:
cc-wiki-dump

حجم این اطلاعات چند گیگ می‌شود که جهت آشنایی با نحوه‌ی import آن به SQL Server می‌توان به مقالات زیر مراجعه نمود:
Understanding the StackOverflow Database Schema
How to Import the StackOverflow XML into SQL Server
Interesting StackOverflow Database Queries

عنوان: **سه مطلب کوتاه**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۲/۲۷ ۱۱:۲۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

مشکل فایرفاکس با سایت‌های msdn و codeplex

هر از چند گاهی در بلاگ‌های msdn و یا سایت codeplex با خطای زیر از طرف سایت مواجه می‌شوم:

```
Bad Request - Request Too Long
HTTP Error 400. The size of the request headers is too long.
```

اگر با این مشکل مواجه شدید، تمامی کوکی‌های مربوط به سایت‌های مذکور را یافته و حذف کنید.
به نظر باگی در فایرفاکس در این زمینه سبب می‌شود که کوکی‌های تمام زیر سایت‌های فوق با هم ترکیب شده و رشته‌ای بسیار طولانی بجای کوکی اصلی آن زیر سایت به هاست ارسال شود.
که البته عکس العمل سایت‌های مایکروسافت از دیدگاه امنیتی هم جالب توجه است (برای برنامه نویس‌های وب).

IE8 و ارائه‌ی fakepath بجای آدرس فایل

کنترل استاندارد آپلود فایل در مرورگرهای جدید، دیگر آدرس محلی فایل را حتی در اختیار اسکریپت‌های سمت کاربر نیز قرار نمی‌دهند. فایرفاکس مدت زیادی است که این مورد را پیاده سازی کرده. اما IE بجای اینکه یک رشته‌ی خالی را بازگشت دهد مسیر c:\fakepath را ارائه خواهد داد (fakepath جزو استاندارد html 5 است). اگر احیانا با این مورد برخورد داشتید، با استفاده از تنظیم زیر می‌توان مانند سابق مسیر کامل را نیز دریافت کرد:

اهمیت این مورد هم برای من این است که IE، یعنی همان مرورگر کاری در اکثر شرکت‌ها (و این مورد فوق را به سادگی از طریق گروپ پالیسی می‌توان به تمام کامپیوترها اعمال کرد).

علت تایم آوت و باز نشدن یک سری از سایت‌ها در ایران

مدت زیادی این سؤال برای من وجود داشت که وجه مشترک سایت‌هایی مانند summerofnhibernate.com ، dotnetkicks.com و امثال آن چیست که از این طرف باز نمی‌شوند؟ lessthandot.com
اگر به آدرس‌های فوق که به سایت domaintools.com ختم می‌شوند، مراجعه کنید و سپس برگه‌ی Server Stats آن‌ها را ملاحظه نمائید، همگی توسط Godaddy.com Inc هاست می‌شوند. این شرکت غیرمحترم، IP های ایرانی را بسته است (مطلب جدیدی هم نیست).

نظرات خوانندگان

نویسنده: حسین مرادی نیا
تاریخ: ۱۳۸۸/۱۲/۲۷ ۱۲:۴۲:۰۲

حالا از کجا بفهمیم که این شرکت ip های ایران رو بسته؟
نکته خاصی در این زمینه وجود داره یا همین شواهد موجود و همین اشتراکات؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۲/۲۷ ۱۳:۲۰:۲۵

more info

نویسنده: peyman naji
تاریخ: ۱۳۸۸/۱۲/۲۷ ۱۵:۳۳:۰۷

با سلام .
مهندس من هم یه مشکلی برخورد کردم که هنگام رفتن به سایت هایی که با Https کار میکنن google chrome میگه The server's security certificate is not yet valid!
آیا isp بنده اینکا رو کرده ؟ مشکل چی میتونه باشه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۲/۲۷ ۱۶:۴۴:۳۵

هر ssl certificate حتما دارای یک تاریخ انقضاء است. احتمالا سایت مورد نظر شما مجوز خودش را به روز نکرده است.
سایت‌های دست چندم ممکن است از این نوع مشکلات داشته باشند.

عنوان: **با کمک link blog ها در زمان خود صرفه جویی کنید**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۲۰ ۲۱:۱۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

این مطلب در حقیقت خلاصه و چکیده‌ی RSS خوانی‌های دو سال اخیر بنده است.
برخلاف فوروم‌های ایرانی که سعی می‌کنند نقش تمام رسانه‌ها را به صورت یکجا بازی کنند، برای مشاهده مقالات جدید یا اخبار به روز فناوری‌های مرتبط، راهی بجز وبلاگ خواندن وجود ندارد (یک استثناء اینجا وجود دارد و آن‌هم مباحث وارز است. در این یک مورد هم بیشتر فوروم‌های خارجی فعال هستند تا وبلاگ‌های آن‌ها).
خوب، در ابتدا هر سایتی که به نظر با کیفیت است به لیست RSS reader ما اضافه می‌شود و همینطور هر روز این لیست بیشتر و بیشتر خواهد شد. ابتدا روزی نیم ساعت RSS خوانی خواهید داشت بعد می‌شود روزی 2 ساعت و بعد از یک مدتی هم از کنترل خارج می‌شود.
در لابلای تمام این سایت‌هایی که من الان جمع آوری کرده‌ام و تعدادشان به چند هزار(!) می‌رسد، چند سایت (حدود 30 سایت) هستند که اصطلاحا link blog نام دارند و به صورتی پیوسته، خلاصه‌ی مفید وبلاگ‌های روز را ارائه می‌دهند (که پشتکار آن‌ها قابل ستایش است).
اکنون کار من بجای روزی چند ساعت زیر و رو کردن مطالب RSS feed وبلاگ‌هایی که ثبت کرده‌ام، صرفا به مرور سریع این link blog ها خلاصه می‌شود و انصافا گنجینه‌ای با ارزش هستند (خصوصا از این جهت که کار دست بوده و خودکار تهیه نمی‌شوند).
برای import ساده‌ی این لیست، فایل opml آن‌ها را از آدرس زیر می‌توانید دریافت کنید.

دریافت فایل OPML

این مجموعه برای راه اندازی یک سایت خبری کفایت می‌کند ...

نظرات خوانندگان

نویسنده: علی رضا میرزایی
تاریخ: ۰۷:۴۰:۲۸ ۱۳۸۹/۰۱/۲۱

سلام
از شما بی نهایت تشکر میکنم

نویسنده: Amin
تاریخ: ۰۸:۳۰:۵۴ ۱۳۸۹/۰۱/۲۱

ممنون. من قبلا هم از OPML های شما استفاده میکردم. اینم که عالیه.

نویسنده: Mostafa
تاریخ: ۲۳:۴۵:۰۶ ۱۳۸۹/۰۱/۲۱

عمرا اگه این بار دانلودش کنم !
:-s

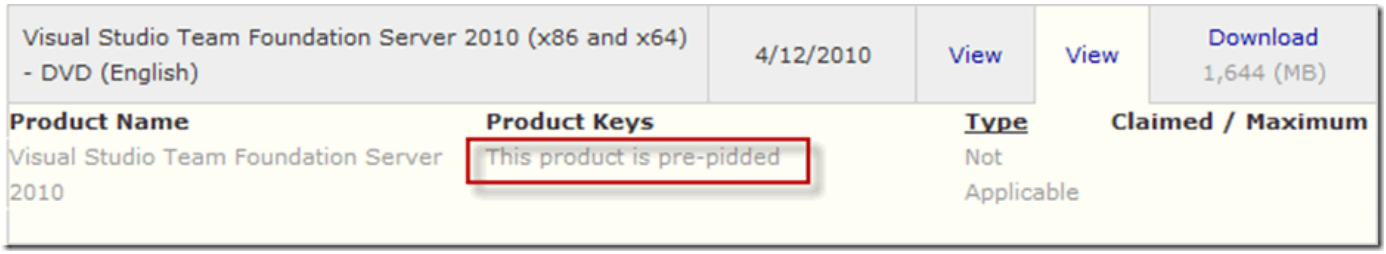
عنوان: انتشار VS2010
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۲۵ ۱۰:۱۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

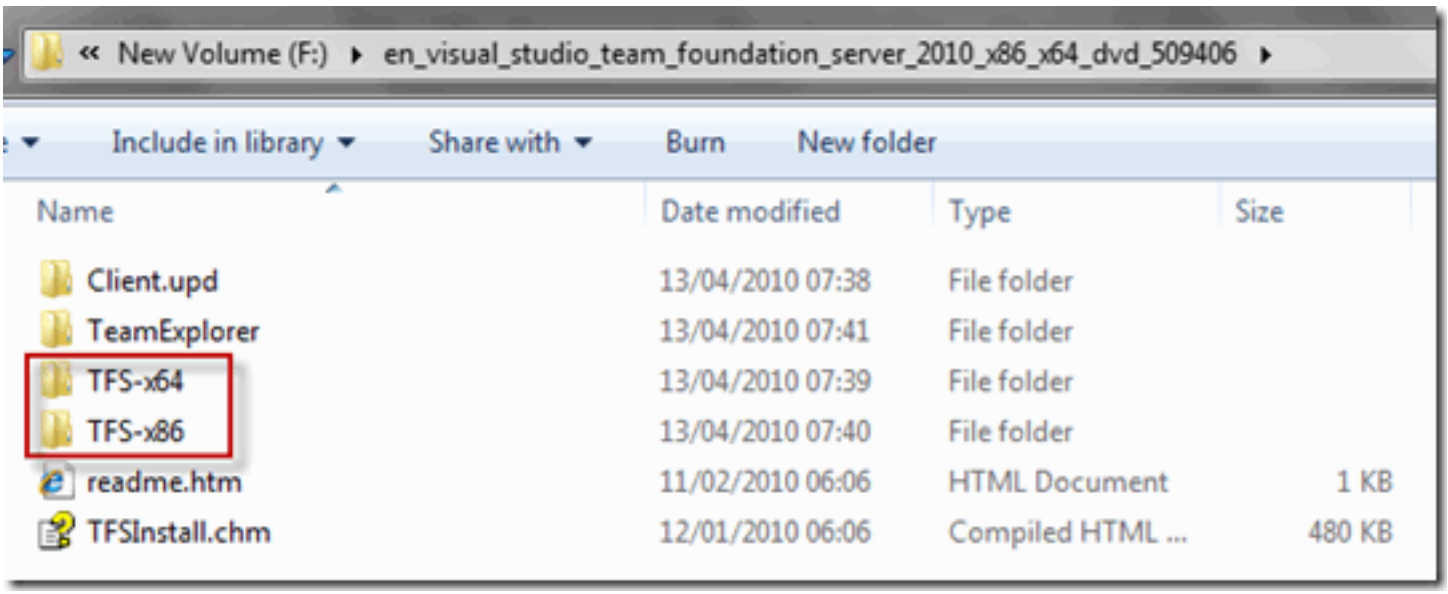
یکی دو روزی هست که VS2010 منتشر شده و در این مطلب خلاصه‌ای از لینک‌های مفید مرتبط را جمع آوری کرده‌ام که در ادامه ملاحظه خواهید کرد:

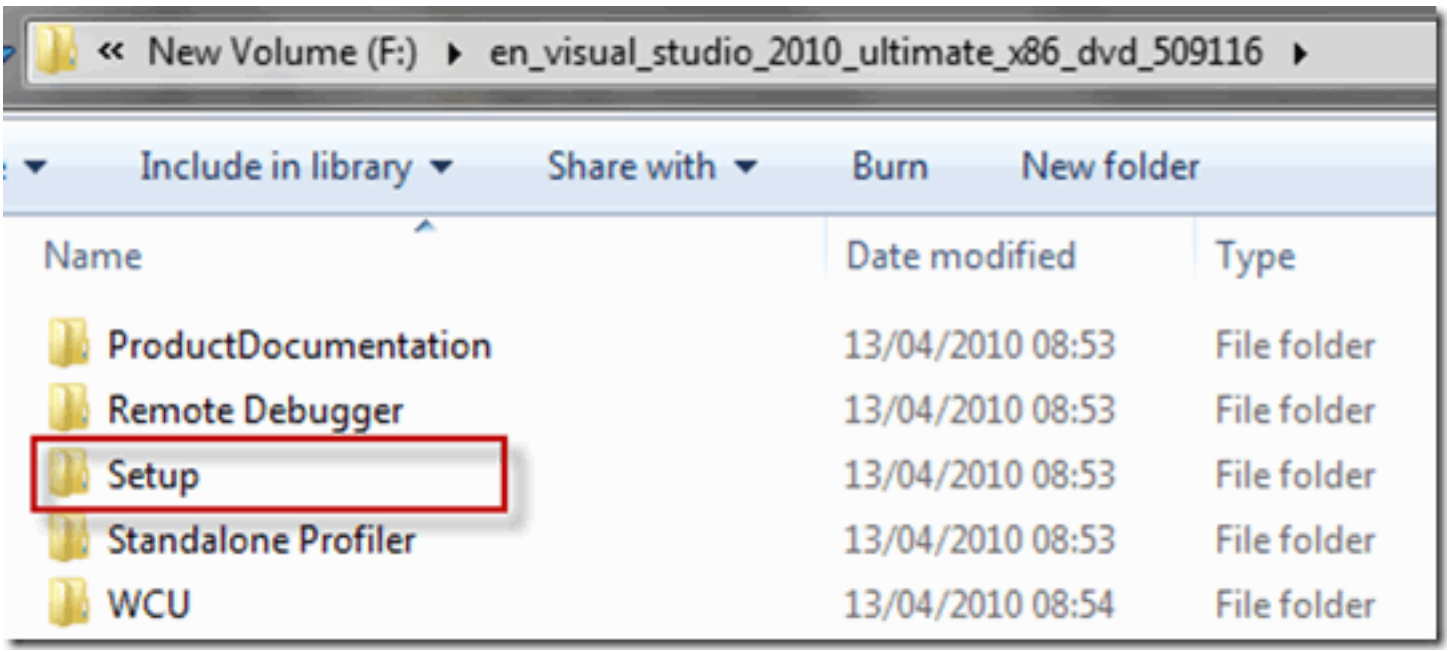
دریافت فایل ISO اصلی

- نسخه‌ی Ultimate
- سایر نسخ
- و همچنین دات نت فریم ورک 4

این فایل ISO منتشر شده، نسخه‌ی نهایی است و تنها تفاوت آن با نگارش اصلی منتشر شده برای دارندگان اکانت‌های MSDN ، عدم درج سریال در فایل setup.sdb آن است. تصاویر زیر متعلق است به دارندگان اکانت‌های MSDN و همانطور که مشاهده می‌کنید، محل وارد کردن سریالی که یافته‌اید، در فایل setup.sdb است. پس از آن نگارش Trial به صورت کامل نصب خواهد شد.






« New Volume (F:) ▸ en_visual_studio_2010_ultimate_x86_dvd_509116 ▸
Include in library
Share with
Burn
New folder
Name
Date modified
Type
ProductDocumentation
13/04/2010 08:53
File folder
Remote Debugger
13/04/2010 08:53
File folder
Setup
13/04/2010 08:53
File folder
Standalone Profiler
13/04/2010 08:53
File folder
WCU
13/04/2010 08:54
File folder

```
[Event Log]
dd_install_VS_VSTSCORE_100.txt

[Suite Requirements]
vsbasereqs.dll, GFN_CID VS Baseline Requirements

[Product Key]
[redacted]  <- Remove This Line

[Dialog Page Order - AdminMaintenance]
COMPage,[vs7Opgui.dll]-[F7AFBAD4-E116-4dbe-9965-AD718BA07CAE] : Maintenance Page
COMPage,[vs7Opgui.dll]-[9FE307C0-3646-11d3-A508-00C04F5E0BA5] : Options Page
COMProgPage,[vs7Opgui.dll]-[BB81D811-3F9B-11D3-A50C-00C04F5E0BA5] : Install Page
COMPage,[vs7Opgui.dll]-[1920CC5D-5BE5-45d4-9C1C-3513D334C71C] : Install Page

[ClientInstall]
GFN_Client_MSI

[Install Action Overrides]
GFN_Client_MSI,Action,5
```

خلاصه‌ای از تازه‌ها و موارد مرتبط با VS2010 و دات نت 4

مثال‌های VS2010 برای برنامه نویسان در مورد زبان‌های مختلف (CS ، VB و ...) و همچنین فناوری‌های مرتبط (WPF ، پردازش موازی و ...)

تازه‌های WPF4

تازه‌های Entity framework 4

Visual Studio 2010 Resource Kit

بهبود کارآیی WPF در دات نت 4

تغییراتی در ASP.Net 4.0 که برنامه نویسان مربوطه باید از آن مطلع باشند

اف شارپ 2 به همراه VS2010 ارائه شد

کد نویسی بهتر با VS2010

ویدیوهای VB 2010

ویدیوهای CS 2010

ویدیوهای VS 2010

ویدیوهای VSTO 2010

و بسیاری موارد دیگر

راستی! ReSharper 5.0 هم در همین زمان منتشر شده است.

نظرات خوانندگان

نویسنده: Amin
تاریخ: ۱۳۸۹/۰۱/۲۵ ۱۲:۳۳:۴۸

ممنون از لینکهاتون.
یه سوال: آقای نصیری این نسخه 90 روزه هستش؟ یا محدودیتی نداره؟ بخاطر این میپرسم که بدونم و بعد دانلود کنم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۲۵ ۱۲:۵۱:۱۷

منهای لینک‌هایی که در اختیار دارندگان اکانت‌های msdn است (که همانطور که عرض کردم سریال آن‌ها در فایل setup.sdb درج شده) مابقی لینک‌ها محدودیت زمانی دارند. البته از تمام این نگارش‌های trial پس از نصب با داشتن سریال معتبر، رفع محدودیت زمانی خواهد شد یا اگر در ابتدای نصب سریال خود را در فایل setup.sdb وارد کنید (مطابق تصویر آخر)، به صورت full و غیر trial نصب می‌شوند.

نویسنده: Amin
تاریخ: ۱۳۸۹/۰۱/۲۵ ۱۳:۳۶:۰۷

بازم ممنون.

نویسنده: بهزادیان
تاریخ: ۱۳۸۹/۰۱/۲۶ ۰۰:۰۲:۵۴

یعنی باید سریال داشته باشیم (که نداریم) یا اینکه باید خط مزبور را پاک کنیم و همین کار باعث نصب نسخه اصلی می شود؟
لطفا بیشتر توضیح بدید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۲۶ ۰۰:۵۲:۲۱

نه. اون خط رو باید برای انتشار عمومی پاک کنند نه اینکه شما پاک کنید. شما باید اضافه کنید (چون در نگارش trial وجود ندارد؛ بدون سریال است).
روش دوم فعال سازی VS2010 از بلاگ‌های MSDN به شرح زیر است:
http://bit.ly/aIpp9l

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۲۶ ۰۰:۵۸:۵۰

یک دوره آموزشی رایگان VS2010 ازAppDev
/http://launchoffer.appdev.com

نویسنده: بهزادیان
تاریخ: ۱۳۸۹/۰۱/۲۶ ۰۹:۰۶:۵۶

سلام
من دانلودش کردم، توی فایل مذکور سریال نامبر نوشته شده بود، بعد بدون تغییر نصب کردم، الان نسخه آزمایشی نصب شده است
الف) آیا باید قبل از نصب اون سریال رو بر می داشتم؟ و اگر بر دارمش، آیا نیاز به سریال اصلی هست؟
ب) الان آیا می شود سریال اصلی را به دست آورد و مطابق توضیح وبلاگی که لینکش را فرستاده اید، Activate کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۲۶ ۱۰:۲۹:۵۲

شما در هر زمانی مطابق لینک http://bit.ly/aIpp9l می‌تونید سریال جدید و غیر trial را وارد کنید و نگارش خودتون رو ارتقاء بدید.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۲۶ ۱۰:۴۴:۳۲

خلاصه‌ی ساده‌ی مطالب این صفحه:

1- بدون داشتن سریال اصلی و غیر trial مطالب خوانده شده در حد اطلاعات عمومی است.

2- یک روش ورود این سریال اصلی وارد کردن آن به فایل setup.sdb است (پیش از شروع به نصب).

3- یک روش دیگر وارد کردن این سریال اصلی مراجعه به کنترل پنل ویندوز (پس از نصب نسخه مدت دار موجود)، add/remove programs و سپس کلیک بر روی دکمه change/remove برنامه VS2010 است. در اینجا در صفحه‌ی maintenance می‌تونید سریال برنامه رو در هر زمانی عوض کنید.

نویسنده: بهزادیان
تاریخ: ۱۳۸۹/۰۱/۲۶ ۱۱:۰۸:۴۲

مهندس جان، پس مشکل سریال اصلی است!
از کجا میشه گیر آورد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۲۶ ۱۳:۱۸:۴۳

به فوروم exetools دات کام مراجعه کنید قسمت General > Software Release ... همین روش ویرایش فایل sdb و وارد کردن سریال به همراه سریال مربوطه ذکر شده ... که به شخصه امروز تست کردم و کار می‌کنه.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۲۶ ۱۸:۱۵:۴۸

آقای حاج لو تمام سریال‌ها و پسوردهای مرتبط را در کامنت‌های مطلب ذیل لیست کردن که می‌تونید استفاده کنید:
http://bit.ly/as9vK6

نویسنده: بهزادیان
تاریخ: ۱۳۸۹/۰۱/۲۷ ۱۱:۳۲:۲۵

دست همه دوستان درد نکند، هم نصب شد و هم فعال.
کسی خبر از MSDN ندارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۲/۰۱ ۰۱:۵۵:۰۱

iso نسخه 2010 کتابخانه msdn هنوز (تا این تاریخ) منتشر نشده (از نسخه آنلاین استفاده کنید).
iso نسخه 2008 کتابخانه msdn موجود است:
http://www.microsoft.com/downloads/details.aspx?FamilyID=7bbe5eda-5062-4ebb-83c7-d3c5ff92a373&DisplayLang=en

نویسنده: مهدی پایروند
تاریخ: ۱۳۸۹/۰۲/۰۱ ۰۸:۱۴:۵۹

من هم استفاده کردم، از شما هم آقای مهندس ممنون!

نویسنده: Amir
تاریخ: ۱۳۸۹/۰۴/۱۸ ۱۳:۱۹:۳۶

من وقتی میخوام VS2010 Ultimate Trial رو نصب کنم خطای زیر داده میشه

Error 1935.An error occurred during the installation of assembly 'Microsoft.VisualStudio.Debugger.Runtime,version="10.0.0.0",publicKeyToken="b03f5f7f11d50a3a",processorArchitecture="MSIL",fileVersion="10.0.30319.1",culture="neutral"'. Please refer to Help and Support for more information. HRESULT: 0x8002802F. assembly interface: , function: CreateAssemblyCache, component: {813438C3-{8069-495F-9AE3-CA6499E3021B

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۴/۱۸ ۱۳:۳۲:۵۸

دات نت فریم ورک 4 را یکبار از طریق کنترل پنل حذف کنید. سیستم را ری استارت کنید و بعد مجددا دات نت فریم ورک 4 کامل را که لینکش در بالا هست نصب کنید مشکل حل می‌شود.

نویسنده: Amir
تاریخ: ۱۳۸۹/۰۴/۲۰ ۱۱:۳۰:۵۷

من دات نت فریم ورک 4 رو حذف و بعد از ری استارت دوباره نصب کردم ولی باز هم موقع نصب خطا داد و نصب نشد!؟ من حتی نسخه اکسپرس رو هم نتونستم نصب کنم!

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۴/۲۰ ۱۲:۲۱:۰۱

در این مورد اینجا بحث شده. یک سری تداخل را باید رفع کنید (که مورد قبل یکی از آن‌ها بود):

http://bit.ly/a7ro3v

+

http://support.microsoft.com/kb/970652/en-us?p=1

+

http://blogs.msdn.com/b/astebner/archive/2004/11/10/255346.aspx

عنوان: **املای صحیح فناوری‌های مرتبط با مایکروسافت**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۲/۱۴ ۲۰:۱۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

به نظر شما .net درست است یا .Net یا ... ؟
Asp.net چطور؟ آیا ASP.Net درست است یا Asp.net یا ... ؟!
Direct X درست است یا Direct-X یا ... ؟
Linq درست است یا linq یا ...؟

پاسخ: هیچکدام!
املای تمامی موارد ذکر شده نادرست است. جدولی را یکی از کارکنان مایکروسافت در این زمینه تهیه کرده است که در آدرس ذیل می‌توانید مشاهده نمائید:

Correcting Grammar for Microsoft Products and Technology

نظرات خوانندگان

نویسنده: peyman naji
تاریخ: ۱۳۸۹/۰۲/۱۵ ۱۹:۳۴:۰۳

جالب بود ممنون .

نویسنده: zamanphp
تاریخ: ۱۳۸۹/۰۲/۲۰ ۲۰:۲۲:۴۰

با اون که از سال 84 .NET رو کنار گذاشته ام و به سمت OPEN SOURCE رفته ام ولی برای شخص بنده خیلی جالب بود.
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۲/۲۰ ۲۰:۳۲:۱۱

دات نت هم سورس باز است.

releasing-the-source-code

عنوان: **لیستی از بانک‌های اطلاعاتی قابل استفاده در دات نت**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۲/۲۴ ۱۸:۳۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

بد نیست لیست تعدادی از بانک‌های اطلاعاتی مهم قابل استفاده در دات نت به همراه درایورهای ADO.NET آن‌ها را با هم مرور نمائیم.

بانک‌های اطلاعاتی قابل استفاده در دات نت فریم ورک						
ردیف	بانک اطلاعاتی	سایت مرجع	درایور ADO.NET	امکان استفاده از LINQ	مجوز استفاده	توضیحات
1	SQL Server 2000/2005/2008/2008 R2	+	توکار (به صورت پیش فرض در دات نت فریم ورک موجود است)	بلی . به کمک LINQ to SQL ، NHibernate ، Entity Framework و بسیاری از ORM های دیگر	رایگان - تجاری	نسخه‌های Express آن رایگان است.
2	Microsoft SQL Azure	+	بلی : +	بلی. به کمک LINQ to SQL و Entity Framework	تجاری	
3	SQL Server Compact					

بانک‌های اطلاعاتی قابل استفاده در دات نت فریم ورک
+
بلی :
+
بلی. به کمک LINQ to SQL و
Entity Framework
رایگان
4
Advantage Database Server
+
قابل دریافت از سایت اصلی:
+
بلی. به کمک Entity framework و
Telerik OpenAccessORM
تجاری
5
SQL Anywhere
+
قابل دریافت از سایت اصلی:
+
بلی. به کمک Entity framework
و TelerikOpenAccess ORM
رایگان - تجاری
Web Edition آن رایگان است.
6
MySQL
+
قابل دریافت از سایت اصلی :
+
بلی . به کمک NHibernate ،
LightSpeed
، DbLinq و تعدادی دیگر از
ORM's
رایگان - تجاری
7
Oracle

بانک‌های اطلاعاتی قابل استفاده در دات نت فریم ورک
+
پشتیبانی توکار آن به زودی حذفخواهد شد اما از سایت اصلی قابل دریافت است : +
بلی . به کمک NHibernate ، LightSpeed ، DbLinq و تعدادی دیگر از ORM's
رایگان - تجاری
نسخه‌ی Express آن رایگان است.
8
Access
+
توکار
بلی. به کمک ALinq ، NHibernate و یا LINQ toDataSets
تجاری
اگر از دات نت فریم ورک سه و نیم، سرویس پک یک استفاده کنید، امکان استفاده از LINQ to SQL جهت کار با بانک‌های اطلاعاتی اکسس نیز مهیا است: +
9
SQLite
+
مهیا به صورت سورس باز : +
بلی. درایور ADO.NET آن پشتیبانی از Entity Framework را نیز اضافه می‌کند. همچنین NHibernate ، ALinq و سایر ORM's را باید به این لیست اضافه کرد.
رایگان
10
Firebird
+
قابل دریافت از سایت اصلی: +
بلی. توسط ALinq ، NHibernate و موارد دیگر.

بانک‌های اطلاعاتی قابل استفاده در دات نت فریم ورک
رایگان
11
PostgreSQL
+
قابل دریافت از سایت اصلی: +
بلی. توسط DBLinq ، NHibernate و موارد دیگر
رایگان
12
DB2 UDB
+
قابل دریافت از سایت اصلی: +
بلی. توسط NHibernate
تجاری
13
ScimoreDB
+
قابل دریافت از سایت اصلی: +
محدود. توسط LINQ to DataSets
رایگان
14
MongoDB
+
معرفی شده در سایت اصلی : +
بلی. درایور ADO.NET معرفی شده به همراه پروایدر LINQ نیز می‌باشد.
رایگان
15
CouchDB

بانک‌های اطلاعاتی قابل استفاده در دات نت فریم ورک
+
معرفی شده در سایت اصلی :
+
محدود
رایگان
16
VistaDB
+
اساسا برای دات نت نوشته شده است.
بلی. به کمک Entity framework
تجاری

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۲/۲۵ ۰۰:۵۶:۳۰

دو مورد دیگر

db4o

RavenDB

نویسنده: mironline
تاریخ: ۱۳۸۹/۰۲/۲۶ ۰۰:۱۷:۱۴

سلام . چند وقت پیش دنبال یک embedded database میگشتم .یک دیتابیس به نام berkeley db پیدا کردم . توی sourceforge ، آداپتور دات نت اش موجود هست .

http://sourceforge.net/projects/libdb-dotnet

نویسنده: Meysam
تاریخ: ۱۳۸۹/۰۲/۲۷ ۱۰:۰۶:۵۹

واسه نرم افزارهایی که یه DB کوچیک با سرعت خوب میخوان SQLite بهترین انتخابه.

نویسنده: Pourya
تاریخ: ۱۳۸۹/۰۲/۲۸ ۱۸:۲۷:۲۱

با تشکر از مطالب مفیدتون،
اگه امکان داره لینکی برای دانلود LightSpeed که تو لیست بالا اومده معرفی کنید(نسخه کاملش)

ظاهرا همه چیز رو پشتیبانی میکنه!

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۲/۲۸ ۲۱:۵۸:۳۶

- من ندیدم ...
- از NHibernate استفاده کنید که رایگان است و پخته‌تر. نگارش سوم آن که تا چند وقت دیگری نهایی می‌شود تقریبا دیگر جایی برای بقیه نخواهد گذاشت.

نویسنده: Pourya
تاریخ: ۱۳۸۹/۰۲/۲۹ ۰۸:۴۵:۳۱

کار با NHibernate سخته، Wizard برای Generate کردن Mapping دیتابیس نداره و همه چیز را خودمان باید تولید کنیم، ولی ابزار هایی مثل OpenAccess همه چیزهارو خودشان تولید میکنند واز Forward Mappingو Reverse Mapping هم پشتیبانی می کنند و کلی زمان تولید کد رو کم می کنند.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۲/۲۹ ۰۹:۵۵:۰۱

- البته روشی که به صورت پیش فرض در NHibernate استفاده می‌شود سیستم اول کد و طراحی و بعد دیتابیس است که Entity framework 4 برای نمونه به این سمت هم گرایش پیدا کرده.
- ضمنا یک Visual NHibernate تجاری هم موجود است:

Visual NHibernate

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۳/۰۴ ۰۸:۵۴:۵۰

یک مورد دیگر

Perst - An open source, object-oriented embedded database

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۳/۰۴ ۱۸:۰۹:۲۱

چند مورد مخصوص Silverlight

Siaqodb - object database for .NET, Mono and Silverlight

Silverlight Database

EffiProz Database for Silverlight

#ported SQLite to C

نویسنده: Mohagheghi
تاریخ: ۱۳۸۹/۰۳/۰۸ ۱۵:۱۰:۳۵

من از LLBL Gen استفاده می کنم چیز فوق العاده ای است.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۳/۱۱ ۰۰:۵۶:۱۷

- تعریفش رو شنیدم.
- یک دیتابیس دیگر:

Eloquera Database

عنوان: **درسی از بروسلی در توسعه‌ی نرم افزار!**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۳/۰۹ ۱۵:۰۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

نقل قولی از بروسلی:

```
The style doesn’t make the fighter, the fighter makes the style.
```

معادل آن در دنیای نرم افزار:

```
The process doesn’t make the developer, the developer makes the process.
```

معنا:
پیروی کورکورانه از روش‌ها و شیوه‌های جدید از شما برنامه نویس نخواهد ساخت! "شما" هستید که به کار معنا خواهید داد.
تمرکز بیش از حد بر روی یک روش کاری، دیدگاه شما را محدود ساخته و از بسیاری از موقعیت‌ها و امکانات مهیای دیگر غافل خواهید شد.

ادامه مطلب :

Bruce Lee on Software Development

عنوان: **چگونه یک ایمیل مفید خودکار را طراحی کنیم؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۴/۱۲ ۲۰:۵۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

از آنجائیکه مدتی قسمتی از کارم مرتبط بود به طراحی ایمیل‌های خودکار برای برنامه‌های تهیه شده (مثلا، ایمیل‌های مرحله به مرحله یک گردش کاری ... اطلاع رسانی‌های خودکار از وضعیت داده‌ها، گزارشاتی از برنامه‌ها که به صورت خبرنامه‌های ایمیلی در بازه‌های زمانی مشخصی به اشخاص مشخص شده ارسال می‌شد و غیره)، لازم می‌دونم خلاصه‌ای از تجربیات برخورد با کاربران را در این مورد در ادامه ذکر کنم، شاید مفید باشد.

1) حتما در انتهای ایمیل خودکار ارسالی، ساعت و تاریخ شمسی ارسال پیام را نیز ذکر کنید.
عموما از آنجائیکه سیستم استاندارد ارسال ایمیل بر اساس تاریخ میلادی است و تقریبا تمام کلاینت‌های دریافت ایمیل موجود نیز توانایی شمسی سازی تاریخ دریافت و ارسال ایمیل را ندارند (مگر با یک سری افزونه و یا دستکاری در سیستم عامل که آنچنان خوشایند و مرسوم نیست)، ذکر تاریخ شمسی در انتهای پیام بسیار مفید خواهد بود و در اکثر اوقات استناد به ایمیل‌های دریافت شده بر اساس تاریخ دریافت آن‌ها است.

2) سعی کنید از بکارگیری عناوین (subject) ثابت جهت ارسال ایمیل‌های خودکار پرهیز کنید.
دقیقا یادم میاد زمانیکه برای مدیر عامل شرکتی سه بار پشت سرهم ایمیلی با یک عنوان ارسال شده بود بنده را بازخواست کردند که چرا برنامه‌ی شما ایمیل تکراری ارسال می‌کند!
بله، سعی می‌کنند محتوا را از روی عنوان ایمیل حدس بزنند و زمانیکه یک عنوان ثابت را برای ایمیل‌های خودکار خود انتخاب کردید، تکراری به نظر خواهند رسید یا حتی ممکن است به اشتباه پیش از خوانده شدن حذف شوند.
برای مثال فرض کنید ایمیل ارجاع کاری را قرار است به صورت خودکار ارسال کنید. انتخاب عنوان ثابت برای مثال "ارجاع کار جدید" اشتباه است! این عنوان باید بر اساس نوع کار هر بار به صورت پویا متغیر باشد؛ مثلا: "ارجاع کار جدید: از طرف : ... ، موضوع: ... ، درجه اهمیت: ..." که این سه نقطه‌ها باید توسط برنامه هر بار پر شوند.

3) هر چه می‌توانید اطلاعات بیشتری را توسط یک ایمیل خودکار منتقل کنید.
مورد قبل را در نظر بگیرید. ذکر "ارجاع کار جدید ..." در عنوان و سپس مجددا ذکر همین عنوان به عنوان بدنه‌ی ایمیل خودکار به زودی ایمیل‌های شما را تبدیل به نوعی Spam آزار دهنده خواهد کرد. کار جدیدی ارجاع شده است؟ آیا می‌توان خلاصه‌ای از این کار را به همراه ایمیل نیز ارسال کرد تا کاربر حتما برای مشاهده‌ی ریز جزئیات کار به برنامه مراجعه نکند و این ایمیل واقعا ارزش مطالعه را داشته باشد و سبب تسریع در انجام کارها شود؟
برای مثال ذکر کلی این مورد که درخواست مرخصی جدیدی را باید تائید یا رد کنید، کافی نیست. ریز جزئیات مرخصی را هم به همراه ایمیل ارسال کنید.

4) ایمیل شما باید حاوی لینکی جهت باز کردن برنامه‌ی تحت وب مرتبط نیز باشد.
کاری ارجاع شده است؟ بهتر است لینک پویایی را جهت هدایت کاربر به صفحه‌ی مرتبط رسیدگی به همان کار ارجاعی ارسال کنید. به این صورت زحمت او را کمتر کرده و یک مرحله گزارش گیری را حذف خواهید کرد. یا حداقل یک محل مراجعه‌ی کلی بعدی را به این صورت می‌توان ارائه داد.

5) از بکارگیری قسمت from ایی مانند DoNotReply@Site.Com خودداری کنید.
کاربر دریافت کننده‌ی ایمیل باید بداند که در صورت وجود مشکل باید به کجا مراجعه کند؟ چه کسی این ایمیل را ارسال کرده؟ هرچند برنامه به صورت خودکار تمام قسمت‌های این ایمیل ارسالی را تهیه می‌کند اما اگر خبرنامه‌ی تنظیم شده‌ای نیست، حتما شخص ارسال کننده‌ای دارد. یا حداقل یک ایمیل عمومی را برای این مورد تنظیم کنید (ایمیلی که وجود خارجی داشته و هر از چندگاهی بررسی می‌شود).

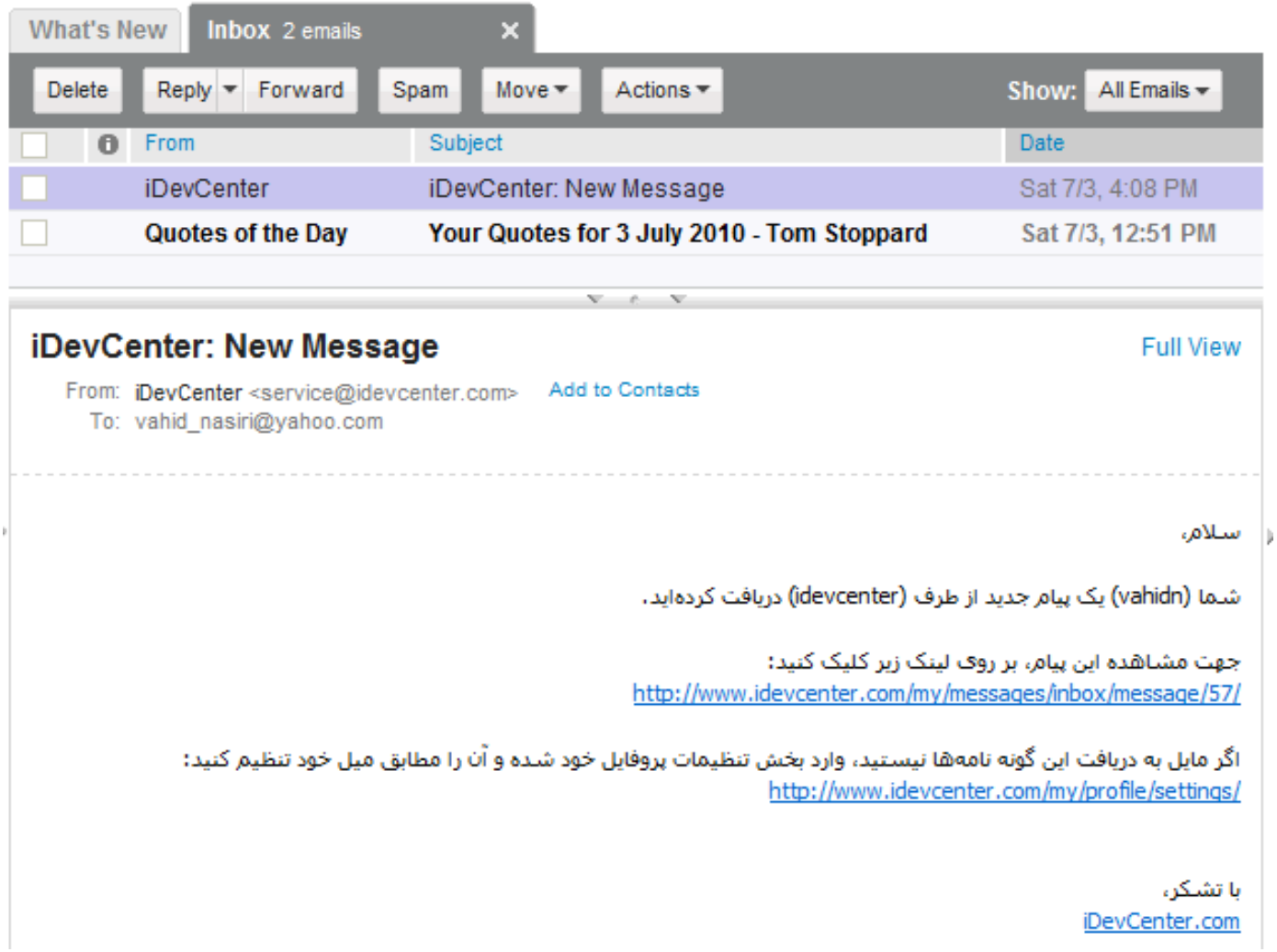
6) رنگ زمینه و اندازه‌ی قلم مناسبی را انتخاب کنید.
دقیقا برای هر کدام از موارد ذکر شده چندین بار مشکل داشته‌ام! عموما کسانی که ایمیل‌ها را دریافت می‌کنند سن و سال دار هستند. بنابراین انتخاب فونت tahoma با اندازه‌ی 8 یا 7 pt سبب توبیخ زود هنگام شما خواهد شد!
همچنین هر چه ساده‌تر بهتر. دقیقا مشکلات از زمانی آغاز می‌شوند که طرحی را انتخاب کنید یا رنگی را برای زمینه بکار ببرید. اینجا است که هر روز یک سلیقه‌ی تحمیلی را باید پذیرا باشید.

7) دقیقا مشخص کنید که ایمیل دریافتی آیا رونوشت است یا خیر!
همان مبحث ارجاع کار را در نظر بگیرید. پس از اینکه سیستم راه اندازی شد، مدیر یکی از قسمت‌ها چند روز بعد این درخواست را "حتما" ارسال خواهد کرد: رونوشت تمام کارهای ارجاعی به کلیه پرسنل بخش و همچنین ریز اقدامات آن‌ها باید برای بنده نیز ارسال شود.
در اینجا تنها افزودن قسمت CC به ایمیل‌های خودکار کفایت نمی‌کند. حتما به صورت درشت در بالای ایمیل، قبل از شروع بدنه ذکر کنید که ایمیل دریافتی یک رونوشت است. در غیر اینصورت باید پاسخگوی علت دریافت ایمیل‌هایی باشید که به درخواست خودشان CC شده است!

8) از ایمیل‌های خودکار برنامه log تهیه کنید.
بارها به این مساله برخورد کرده‌ام که اشخاص برای شانه خالی کردن از انجام کار محوله، سعی در تخریب کار شما خواهند داشت. خیلی ساده عنوان می‌کنند که ایمیلی را دریافت نکرده‌اند. حالا شما بیاید ثابت کنید که اگر سیستم مشکل داشت کلا برای هیچ کسی ایمیل ارسال نمی‌شد، نه فقط برای شما. در اینگونه مواقع وجود یک لاگ از ایمیل‌ها (ثبت در بانک اطلاعاتی) و ارجاع به آن‌ها بسیار راه گشا است.

9) راهی را برای خلاص شدن از شر دریافت ایمیل‌های خودکار نیز پیش بینی کنید!
همان مورد 7 را در نظر بگیرید. دو روز اول خیلی ذوق خواهند کرد! روز سوم وقتی انبوهی از ایمیل‌ها را دریافت کردند، مشکل شما هم شروع خواهد شد. بنابراین امکان تنظیم دریافت یا عدم دریافت ایمیل را حتما در برنامه قرار دهید. یا حداقل نحوه‌ی ایجاد یک پوشه جدید و فیلتر کردن ایمیل‌های رسیده و هدایت خودکار آن‌ها به این پوشه‌ی جدید را آموزش دهید.

خوب! حالا به نظر شما این ایمیل خودکار ارسالی سایت IDevCenter که اخیرا اضافه شده است چه نمره‌ای را کسب می‌کند؟



- تاریخ شمسی در انتهای ایمیل ندارد.
- عنوان‌ها ثابت هستند.
- هیچ جزئیاتی ارائه نشده است.
- لینک مرتبط دارد.
- قسمت from مناسبی دارد.
- ساده است؛ خوب است! فقط اندازه قلم آن بهتر است یک شماره بزرگتر شود.
- بحث رونوشت اینجا مورد ندارد.
- بحث لاگ ... شخصی است.
- امکان تنظیم دریافت ایمیل پیش بینی شده است.

نمره از 7 : 3.5

نظرات خوانندگان

نویسنده: Amir
تاریخ: ۱۳۸۹/۰۴/۱۳ ۱۰:۰۵:۲۵

بسیار مطلب مفیدی بود. مرسی ...

نویسنده: Afshar Mohebbi
تاریخ: ۱۳۸۹/۰۴/۱۴ ۱۷:۳۴:۳۲

مفید و به موقع بود. درست همین روزها من هم باید دست به کار ساختن یک سیستم ایمیل خودکار شوم. البته دوست داشتم کمی راجع به مسائل فنی و روش‌های جلوگیری از اسپم شناخته شدن ایمیل‌های خودکار هم می‌نوشتید.

عنوان: **نحوه‌ی خاتمه‌ی سشن‌های کاربران، از راه دور**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۵/۰۲ ۱۰:۴۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

یکی از مواردی رو که بعضی از ادمین‌ها هیچ وقت یاد نمی‌گیرند این است که لطفا پس از اتمام کار ریموت، logoff کنید و سشن را باز نگه ندارید. اگر چندین سشن به همین ترتیب باز بمانند پس از مدتی با پیغام حداکثر تعداد کانکشن‌های همزمان به یک سرور مواجه خواهیم شد و دیگر امکان اتصال نخواهد بود مگر اینکه یکی از سشن‌های باز خاتمه پیدا کند (همچنین مسایل امنیتی را هم در نظر بگیرید). باز بودن یک سشن هم همانطور که عنوان شد به معنای فعال بودن کاربر نیست. عموما به معنای عدم logoff است.

خوب الان نیاز است که یک اتصال ریموت جهت به روز رسانی برنامه‌ای برقرار شود اما نمی‌شود! کسانی هم که سشن باز بر روی سرور دارند هم اکنون در دسترس نیستند. چکار باید کرد؟

عموما این کار رو می‌کنند: فلانی برو دکمه‌ی ریست سرور رو بزن! بعد هم که سرور بالا اومد دیدن دیتابیس suspect شده واقعا لذت بخش خواهد بود!

راه بهتری هم هست:

برنامه‌ای به نام psexec از SysInternals موجود است که آن‌را می‌توان از آدرس ذیل دریافت کرد:

PSTools

توسط این برنامه می‌توان به خط فرمان سرور ریموت دسترسی یافت (البته بدیهی است که پیشتر باید این دسترسی را داشته باشید) و سپس امکان لیست کردن سشن‌های باز و همچنین خاتمه‌ی آن‌ها میسر خواهد شد؛ به شرح زیر:

الف) دسترسی به خط فرمان سرور از راه دور

```
psexec \\x.x.x.x -u domain\user -p password cmd
```

برای مثال:

```
psexec \\192.168.1.16 -u domain\admin -p password cmd
```

ب) لیست کردن سشن‌های باز

پس از اجرای موفقیت آمیز دستور فوق، خط فرمان سرور در اختیار شما خواهد بود. اکنون دستور qwinsta را وارد کنید. در لیست ارائه شده شماره Id ها مهم است.

ج) بر اساس Id های قابل مشاهده با استفاده از دستور زیر می‌توان سشن را خاتمه داد:

```
logoff [id# of session to quit] /v
```

برای مثال:

```
logoff 2 /v
```

با حذف سشن‌های باز امکان استفاده از امکانات ریموت دسکتاپ مجددا برقرار خواهد شد.

نظرات خوانندگان

نویسنده: Yasser
تاریخ: ۱۳۸۹/۰۵/۰۲ ۲۱:۵۷:۴۷

این دیتابیس Suspect شدن رو بسیار عالی اشاره کردی؛ دوست زحمتکش من

نویسنده: مهدی پایروند
تاریخ: ۱۳۸۹/۰۵/۰۳ ۱۲:۱۱:۱۸

مشکل لاگ-آف نکردن یوزرها یه طرف بعضی‌هاشون که از چند تا سیستم لاگ-این میکنن اونم همون طرف.

نویسنده: محمد صاحب
تاریخ: ۱۳۸۹/۰۵/۰۴ ۱۵:۵۲:۳۰

با سلام و تشکر از شما
روشی که من استفاده میکنم
اجرای برنامه Remote Desktop Connection با سویچ admin
mstsc /admin
ضمنا یوزری که قراره با اون ریموت بشیم باید عضو گروه Administrators سرور مقصد باشه

عنوان: **به روز رسانی‌های مهم هفته دوم شهریور 89**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۱۲ ۱۲:۱۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

نسخه‌ی جدید برنامه Resharper ارائه شده به همراه بهبودهایی در کارآیی آن.
لیست موارد برطرف شده : +
دریافت : +

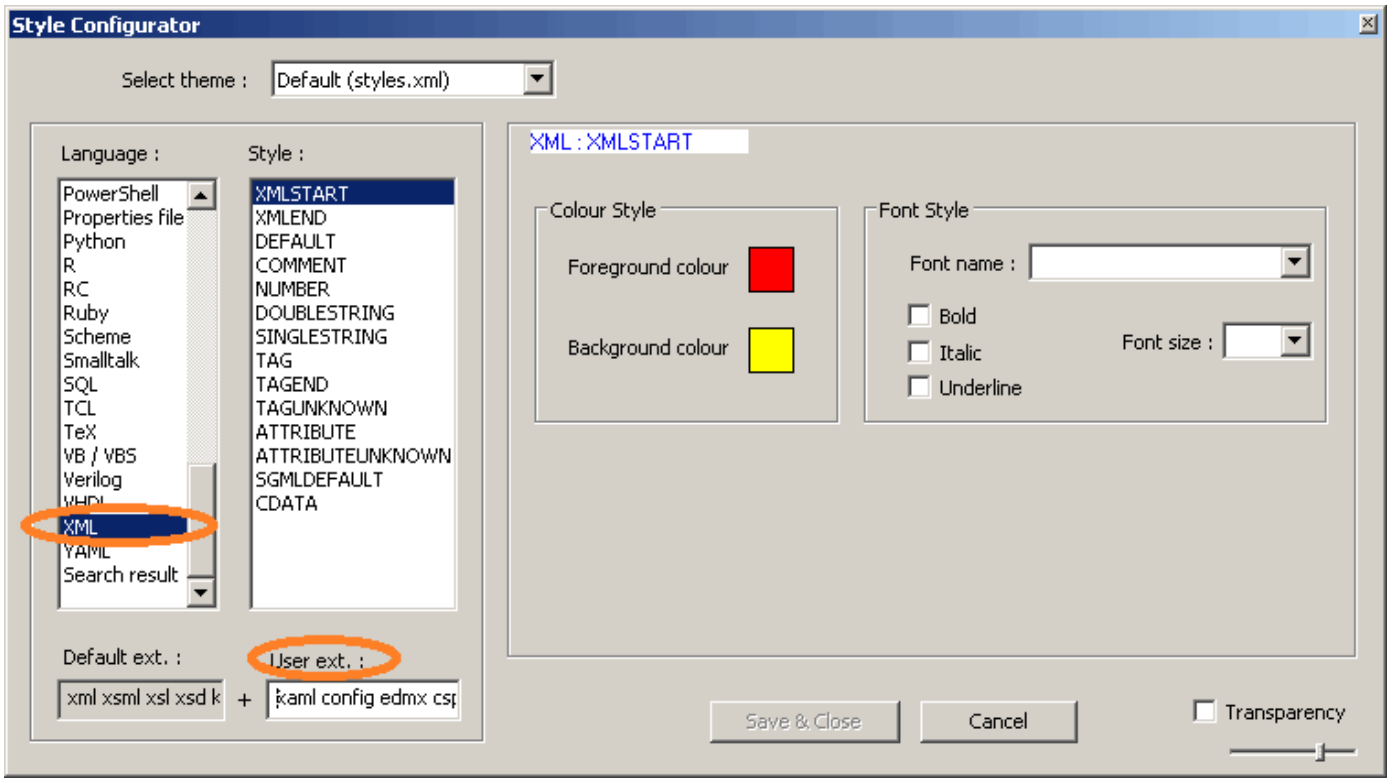
به روز رسانی‌هایی هم در مورد سیلورلایت 4 ارائه شده و اگر آپدیت ویندوز شما روشن بوده باشد، حتما حداقل runtime آن‌را به صورت خودکار دریافت کرده‌اید و از آنجائیکه visual studio LightSwitch هم مبتنی بر سیلورلایت 4 و WCF RIA Services است؛ این به روز رسانی‌ها شامل حال این برنامه نیز می‌گردد.

دریافت SDK جدید: +
دریافت Runtime جدید: +
توضیحات بیشتر در مورد موارد فیکس شده: +

عنوان: **برنامه‌ی Notepad ++ و syntax highlighting فایل‌های دات نتی**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۰۸ ۱۷:۳۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

برنامه‌ی Notepad ++ به عنوان یک ادیتور رایگان، سبک و بسیار سریع به همراه ارائه‌ی syntax highlighting عالی، یکی از ویرایشگرهای محبوب برنامه نویس‌ها (در هر گرایش و صنفی) به شمار می‌رود. اکثر فایل‌های مرتبط با دات نت فریم ورک هم منهای فایل‌های سورس‌ها با پسوند cs و vb و امثال آن، دقیقا از نوع XML هستند؛ برای مثال: xaml, config, edmx, csproj و بسیاری موارد دیگر. این پسوندها به صورت پیش فرض در برنامه‌ی Notepad ++ جهت اعمال syntax highlighting تعریف نشده‌اند و برای اضافه کردن آن‌ها کافی است به صورت زیر عمل کرد:

به منوی Settings گزینه‌ی Style configurator مراجعه کنید. سپس در قالبی که آن‌را انتخاب کرده‌اید، زبان XML را انتخاب کرده و فیلد User ext را تکمیل کنید (شکل زیر):



نظرات خوانندگان

نویسنده: RaminMjj
تاریخ: ۱۳۸۹/۰۷/۱۱ ۱۷:۴۴:۴۸

برای ویرایش XML بهتر از Expression Web هستش مخصوصاً اگر فایل خیلی بزرگ هستش

عنوان: **پایان پشتیبانی از دات نت فریم ورک 3 و سه و نیم**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۲۴ ۰۹:۴۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

پشتیبانی از دات نت فریم ورک‌های سه و سه و نیم دقیقا در تاریخ " سه‌شنبه 23 فروردین 1390 " پایان خواهد یافت. توصیه مایکروسافت ارتقاء برنامه‌های موجود به دات نت فریم ورک سه و نیم، سرویس پک یک و یا دات نت 4 است. طول عمر دات نت فریم ورک سه و نیم، سرویس پک یک به صورت Component محاسبه می‌شود؛ به این معنا که اگر دقت کرده باشید این نگارش به صورت پیش فرض جزئی از ویندوز 7 یا ویندوز سرور 2008 است. بنابراین تا زمانیکه این دو سیستم عامل توسط مایکروسافت پشتیبانی می‌شوند، اجزای آن‌ها نیز همانند دات نت فریم ورک سه و نیم، سرویس پک یک، پشتیبانی خواهند شد.
جدول کامل طول عمر نگارش‌های مختلف دات نت فریم ورک را در اینجا می‌توانید ملاحظه کنید.

عنوان: **وضعیت فناوری‌های مرتبط با دات نت از دیدگاه مرگ و زندگی!**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۱۴ ۲۲:۴۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

مطلبی که در ذیل آورده شده صرفا یک برداشت شخصی است بر اساس نقل قول‌ها و بررسی وضعیت اعضای تیم‌های مرتبط با فناوری‌های مختلف بکار گرفته شده در دات نت فریم ورک و ... نه رسمی!

DataTable ، DataSet ، ADO.NET و امثال آن: مرده! (مرده به معنای اینکه دیگر توسعه‌ی جدی نخواهد یافت)
ADO.NET اولین فناوری دسترسی به داده‌ها در دات نت فریم ورک بود/است. مدل طراحی آن هم بر اساس امکانات زبان‌های آن زمان (زمان شروع به کار دات نت) بود (و تا دات نت 4 هم تغییر عمده‌ای نکرده). برای مثال در زمان ارائه اولین نگارش آن خبری از Generics نبود (در دات نت 2 اضافه شد)؛ یا LINQ وجود نداشت (در دات نت سه و سه و نیم اضافه و تکمیل شد). به همین جهت طراحی آن در حال حاضر (با وجود دات نت 4) بوی ماندگی می‌دهد (مانند استفاده از دیتاست و دیتاتیبل) و با ORM های مایکروسافت جهت استفاده از امکانات Generics و LINQ جایگزین شده است.
البته این مورد تنها مورد مرده‌ای است که "باید" یاد گرفت؛ مهم نیست که ORMs ارائه شده‌اند. مهم این است که زیر ساخت تمام ORM های نوشته شده برای دات نت همین ADO.NET خام است.

LINQ to SQL : مرده!
مایکروسافت با این فناوری ORM های خودش را شروع کرد اما بعد از مدتی دید که بهتر است یک نسخه‌ی عمومی‌تر با پشتیبانی از بانک‌های اطلاعاتی دیگر مانند اوراکل، MySQL و غیره را نیز ارائه دهد. همینجا بود که آن‌را خیلی ساده با Entity framework جایگزین کرد و در roadmap ارائه شده صراحتا EF به عنوان راه حل توصیه شده دسترسی به داده‌های مایکروسافت اعلام شده است (+). حالا این وسط دیگر مهم نیست شما پروژه نوشته بودید یا هر چی. دیگر منتظر تغییرات خاصی در LINQ to SQL نباشید. فقط یک سری رفع باگ و نگهداری پروژه را شاهد خواهید بود. البته در همان زمان خیلی زود تکذیب کردند که LINQ to SQL مرده اما برای نمونه آقای Damien که عضو اصلی تیم LINQ to SQL بودند، اکنون در تیم XBOX مشغول به کار هستند! (+) تو خود شرح مفصل بخوان از این مجمل!

ضمنا این رو هم در نظر داشته باشید که LINQ != LINQ to SQL ؛ به عبارتی LINQ به خودی خود فقط یک language feature است.

Windows Forms یا به اختصار WinForms : مرده!
به نظر مظلوم‌تر از این یکی در دات نت یافت نمی‌شود! همین چند وقت پیش یکی از اعضای مایکروسافت این نظر سنجی رو برگزار کرده بود که "ما چکار کنیم که شما راحت‌تر از WinForms به WPF مهاجرت کنید؟!" (+)
در قاموس WPF ، ویندوز فرمز یعنی Canvas panel ؛ به عبارتی صلب‌ترین حالت طراحی رابط کاربری و این انتقال و مهاجرت هرچند برای کسانی که عمری را روی آن گذاشته‌اند، دردناک خواهد بود اما با وجود توانایی‌های WPF چیزی را از دست نخواهند داد. سیستم Layout (طرح بندی) در WinForms و همچنین دلفی، بر اساس قراردهی اشیاء در مختصات مطلقی در صفحه است. مثلا این دکمه‌ی خاص در آن نقطه‌ی معلوم قرار می‌گیرد و همین. این روش طرح بندی یکی از چندین روش طرح بندی در WPF است که اصطلاحا Canvas نام دارد. توصیه اکید و مطلق در WPF آن است که از Canvas فقط برای طراحی اشیاء گرافیکی پیچیده استفاده کنید و نه طراحی رابط کاربر. چرا؟ چون برای مثال در Silverlight که برادر کوچکتر WPF محسوب می‌شود، رابط کاربری آن باید بتواند همانند HTML انعطاف پذیر باشد و با اندازه‌های مختلف مرورگر یا اندازه‌ی قلم‌های بزرگتر هماهنگ شده و مقاومت کند، بدون اینکه از ریخت بیفتد و این مورد را سایر سیستم‌های طرح بندی رابط کاربر (منهای Canvas یا همان سیستم طرح بندی WinForms) ارائه می‌دهند. مورد دیگری که در WPF و Silverlight بسیار حائز اهمیت است ، Content Controls می‌باشد. چقدر خوب می‌شد بتوان داخل یک کنترل، کلا یک سیستم طرح بندی را تعریف کرد و اشیاء دلخواهی را داخل آن قرار داد. مثلا ToolTip پیش فرض وجود دارد. بسیار هم خوب. بنده علاقه دارم، متن عنوان آن ضخیم باشد. کنار آن یک تصویر کوچک و در سمت چپ آن متن قرار گیرد. برای انجام اینکار در WPF لازم نیست تا شما منتظر نگارش بعدی دات نت باشید که دست اندرکاران مربوطه با افتخار در یک

سند 50 صفحه‌ای توضیح دهند که چگونه می‌توان اینکار را انجام داد. یک سیستم طرح بندی را اضافه کنید. موارد ذکر شده را در آن تعریف کنید. بدون استفاده از هیچ نوع کامپوننتی یا بدون منتظر ماندن تا نگارش بعدی این محصولات، به مقصود خود رسیده‌اید.

ASP.NET Web forms : داره نفس‌های آخرش رو می‌کشه!

از زمانیکه ASP.NET MVC آمد نسخه‌ی Web forms تقریبا فراموش شد. به وبلاگ اسکات گاتری یا Haacked و سایر اعضای اصلی دات نت که مراجعه کنید در سه سال اخیر در حد تعداد انگشتان یک دست هم در مورد Web forms مطلب ننوشته‌اند. تمام تمرکز و نوآوری‌ها بر روی MVC متمرکز شده و حتی در نسخه‌ی 4 دات نت هم فقط یک سری صافکاری مختصر را در Web forms شاهد بودیم مثلا نام کنترل‌ها را خودتان هم در زمان رندر نهایی می‌توانید تعیین کنید! یا لطفا کردند و قسمتی از url routing موجود در ASP.NET MVC را به ASP.NET web forms 4 هم قرض دادند (این مورد شاید مهم‌ترین تغییر قابل ذکر در ASP.Net web forms 4 است).

البته این رو هم اضافه کنم که ASP.NET MVC واقعا قابل احترام است؛ هدف از آن جدا سازی لایه‌های برنامه با الگوهای استاندارد صنعتی (و نه هر روش برنامه نویسی چند لایه من درآوردی)، ترویج کد نویسی بهتر، ترویج Unit testing ، رفع یک سری مشکلات ASP.NET Web forms (مثلا از ViewState های حجیم دیگر خبری نیست) و امثال آن است.

برای نمونه توجه شما را به مطلبی که آقای Dino Sposito در مورد ASP.NET Webforms نوشته جلب می‌کنم: (+)

به صورت خلاصه ایشان عنوان کرده زمان طراحی ASP.NET Webforms در 10 سال قبل، هدف ما انتقال ساده‌تر برنامه نویس‌های VB به محیط وب بود. به همین جهت دست به اختراع viewState ، postback ، کنترل‌های سمت سرور و غیره زدیم. (بنابراین تاکید تمام این‌ها این است که webforms فناوری دهه قبل "بود" و الان بنابر نیازهای جدید دست به طراحی مجدد زده‌ایم)

در مورد "پایان پروژه ASP.NET Ajax Control Toolkit" هم قبلا مطلب نوشته بودم و این یکی واقعا official است!

و در پایان باید متذکر شد که فلان فناوری مرده یا داره نفس‌های آخرش رو می‌کشه اصلا مهم نیست. هنوز هم هستند سازمان‌هایی که برنامه‌های نوشته شده با ASP کلاسیک (نگارش قبل از ASP.NET Web forms) خود را دارند و خیلی هم از آن راضی هستند!

این موارد رو از این جهت نوشتم که اگر می‌خواهید تازه به این جمع وارد شوید دقیقا بدانید باید روی چه مواردی بیشتر وقت بگذارید و یادگیری کدامیک صرفا اتلاف وقت خواهند بود (مثلا EF بر LINQ to SQL ارجح است و اگر امروز می‌خواهید شروع کنید با EF شروع کنید، یا دیگر کم کم با وجود WPF ، بازار کاری برای WinForms نخواهد بود).

نظرات خوانندگان

نویسنده: مصطفی دیندار
تاریخ: ۱۳۸۹/۱۰/۱۴ ۲۳:۳۲:۱۱

جناب نصیری , مبتنی بودن شیرپوینت بر اساس وب فرم ها نمیتونه یک دلیل باشه که وب فرمها هنوز زنده بمانند ؟

نویسنده: علی اقدم
تاریخ: ۱۳۸۹/۱۰/۱۵ ۰۰:۰۶:۰۱

درود بر شما جناب نصیری به مقوله خوبی پرداختید

امروزه در مورد استفاده از تکنولوژی های جدید در سیستم های جدید نظرات زیاد و متناقصی هست مثلا به نظر شما تو یه سیستم که تراکنش اون بسیار بالا در حدود مثلا 1 میلیون تراکنش در روز آیا استفاده از EF عاقلانه است یا نه؟

تکنولوژی های جدید به سمت راحتی برنامه نویسان و خوانایی کد سوق داده شده اند ولی در عوض به اتلاف Performance می انجامند!

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۱۵ ۰۰:۰۷:۱۵

مایکروسافت عادت داره دست به تغییرات اساسی بزنه. نمونه‌اش exchange server یا نمونه‌ی دیگر isa server که اگر به سیر تکاملی این‌ها دقت کنید می‌بینید که نگارش‌های جدید با نگارش‌های 10 سال قبل از زمین تا آسمان فرق کرده‌اند.
در کل باید منتظر نگارش بعدی شیرپوینت بود. یکی از ایراداتی که به شیرپوینت وارد هست سنگین بودن آن برای محیط خارج از اینترانت است (حجم بالای view state).

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۱۵ ۰۰:۲۰:۱۳

همین الان سایت stack overflow مبتنی بر LINQ to SQL و ASP.NET MVC است (مطابق آخرین آماری که منتشر کرده‌اند) و تعداد تراکنش‌های بسیار بالایی هم دارد. بنابراین، بله. عاقلانه است و دارد استفاده می‌شود : (+)

نویسنده: shahin
تاریخ: ۱۳۸۹/۱۰/۱۵ ۰۸:۱۶:۳۳

سلام.
ASP.NET MVC برای پروژه های درون سازمانی و سیستم های اتوماسیون اداری مناسب هست؟
یک مقدار درباره بازار کار آن در خارج از ایران توضیح می فرمایید نسبت به WEB Forms ?
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۱۵ ۰۸:۴۵:۰۷

سلام، من داخل ایران زندگی می‌کنم ولی در کل یک سری ویدیو در مورد ASP.NET MVC هست که اکثر تهیه کنندگان آن‌ها افرادی هستند که مشغول تولید نرم افزار برای دولت آمریکا هستند و الان کارشون رو به MVC ارتقاء دادند (از لابلای صحبت‌ها قابل استخراج است): (+)

نویسنده: Meysam
تاریخ: ۱۳۸۹/۱۰/۱۵ ۱۰:۱۷:۳۰

L2S تو SO جوابش رو پس داده ولی به قیمت بسیار تخصصی! Jeff چند تا پست در مورد Performance واسه L2S داره که نشون میده قضیه چقدر بغرنج شده!
http://www.codinghorror.com/blog/2010/03/compiled-or-bust.html
بله موافقم، مایکروسافت داره به سمت ASP.Net MVC میره.
نوشتن اتوماسیون با ASP.Net دیگه کار درستی نیست! سیلورلایت بهترین انتخابه(فقط از MVVM استفاده نکنید، اگر مشکل زمان دارید)

نویسنده: ebraminio
تاریخ: ۱۳۸۹/۱۰/۱۵ ۱۰:۴۹:۱۳

ممنون از شما، مفید بود

نویسنده: A
تاریخ: ۱۳۸۹/۱۰/۱۵ ۲۳:۴۱:۴۵

اسکات هنسلمن در یک فیلمی که در مورد MVC صحبت می‌کند، MVC را به موتور سیکلت و WebForms را به یک ماشین ون تشبیه می‌کند. این تشنیه به علت وجود خصایص کامپوزیتی در WebForms است که تشبیه بسیار خوبیست.

MVC خیلی خوب است اما برای مثال شما در آن نمی‌توانید یک کامپوننت ایجاد کرده و به راحتی در پروژه‌های بعدی استفاده کنید بدون اینکه به کدتان دست بزنید. (فقط از یک بُعد به کامپوننت نگاه نکنید. مثلا نه فقط بر پایه jQuery و این قبیل و یا فقط کد و ...) این می‌تواند نقطه ضعف بزرگی باشد. به نظر من ASP.NET عقب افتاده است اما MVC نمی‌تواند جایگذین 100% آن باشد. چیز دیگری باید به وجود آید ولی آن MVC نیست. البته باید این را هم در نظر گرفت که شاید سیاستمداران مایکروسافت به این فکر کرده‌اند که وقتی با PHP وب سایت‌های معتبری را راه می‌اندازند چه نیازی به کامپوننت گرایی و ...؟!

البته اگر از حق نگذریم وب محیطی متفاوت است و رفتار با آن مانند محیط‌های دیگر، به هر حال باعث بروز سرباره می‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۱۶ ۰۱:۱۱:۵۰

بیشتر مطالعه کنید. یک نمونه از مجموعه‌های افزودنی‌های جالب ASP.NET MVC توسط شرکت معظم Telerik به صورت سورس باز و رایگان در این آدرس قابل دریافت است: (+)
هدف از ASP.NET MVC پیاده سازی مفاهیم مهندسی نرم افزار مانند unit testing ، فریم ورک‌های IOC ، جداسازی صحیح لایه‌های برنامه از یکدیگر، دور ریختن مفاهیم web forms که اساسا تحمیل مایکروسافت به وب است و نه روند طبیعی کارکردن آن و موارد شبیه به آن است.

هدف من از ارائه‌ی این مطالب چی بود؟ تلنگری به کسانی که خوابند! مایکروسافت داره به این سمت میره. رفته ... من و شما خوشمون بیاد یا نیاد.
همین کار رو با ASP کلاسیک کرد (سرش رو برید گذاشت یک گوشه جون بده). همین کار رو با Vb6 کرد، همین کار رو با فاکس پرو کرد و ....
این یک روند تکامل تدریجی است.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۱۶ ۰۱:۴۲:۴۴

یک سری آموزشی مفصل ASP.NET MVC اینجا است: (+) (از آقای راب کانری نویسنده ORM ایی به نام ساب سونیک) که بعدها به سایت ASP.NET هم منتقل شد: (+)

نویسنده: Armin

تاریخ: ۱۳۸۹/۱۰/۲۸ ۱۳:۴۴:۱۲

نظرتون درباره Silverlight و آیندش چی هستش؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۸ ۱۴:۲۵:۲۹

روی سیلورلایت 5 خیلی دارند کار می‌کنند:

- API سه بعدی
- شتاب دهنده‌های مبتنی بر GPU
- ارائه‌ی نسخه‌ی 64 بیتی

و ...

جزئیات رو می‌تونید اینجا ملاحظه کنید: (+)

همچنین تیم سیلورلایت داره مجددا استخدام می‌کنه! : (+)

و سیلورلایت 4 به عنوان بهترین فناوری RIA سال از طرف InfoWorld انتخاب شده: (+)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۲۰ ۹:۲۷

نسخه رسمی و به روز شده «وضعیت فناوری‌های مرتبط با دات نت از دیدگاه مرگ و زندگی!» از طرف مایکروسافت:

Summary - .NET Technology Guide for Business Applications

The .NET Technology Guide for Business Applications

عنوان: **یافتن نام رشته‌ای کامل یک کلاس در دات نت**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۰ ۰۹:۳۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

دو تنظیم زیر را در نظر بگیرید:

```
<add key="nhibernate-logger" value="NHibernate.Helper.Logging.LoggerFactory, NHibernate.Helper" />
```

و یا

```
<add name="StaticContentCacheModule" type="StaticContentCacheModule.StaticCache,
StaticContentCacheModule"/>
```

این نوع موارد را در فایل‌های app.config و یا web.config زیاد می‌توان یافت.
الان فرض کنید کلاس StaticCache مربوط به StaticContentCacheModule فرضی فوق را به صورت دستی به برنامه‌ی خود اضافه کرده‌اید. همچنین سطر فوق را نیز بدون هیچ تغییری در قسمت http modules مربوط به web.config برنامه معرفی نموده‌اید. برنامه را اجرا می‌کنید، اما ماژول ذکر شده کار نمی‌کند! چرا؟
چون نام رشته‌ای متناظر با کلاس StaticCache ایی که اکنون به پروژه‌ی خود اضافه کرده‌اید، با توجه به فضاهای نام پروژه‌ی جدید، کاملا دگرگون شده است. بنابراین، سؤال مهم اینجا است که این نام را بر اساس تنظیمات پروژه‌ی جاری چگونه می‌توان یافت؟
خوشبختانه دات نت فریم ورک، ابزاری توکار را برای تولید این نام رشته‌ای، به همراه دارد:

```
class Test
{
  static void Main()
  {
      string name = typeof(System.Data.DataView).AssemblyQualifiedName;
      Console.WriteLine(name);
  }
}
```

خاصیت AssemblyQualifiedName ذکر شده در مثال فوق، دقیق‌ترین نامی است که می‌توانید در پروژه‌ی خود استفاده نمائید.
خروجی این مثال جهت نمایش نام رشته‌ای معادل کلاس System.Data.DataView به صورت زیر است:

```
System.Data.DataView, System.Data, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
```

سؤال: از کجا متوجه شوم که رشته‌ی فوق واقعا کار می‌کند؟
مقدار متغیر name مثال فوق باید پس از بکارگیری در متد Type.GetType ، حاصلی غیر null را بازگشت دهد.

```
var name = typeof(System.Data.DataView).AssemblyQualifiedName;
var type = Type.GetType(name);
```

اگر حاصل نال بود، یعنی همان مشکلی که در ابتدای مطلب ذکر شد: ماژول مشخص شده در web.config برنامه توسط رشته‌ی مورد نظر کار نخواهد کرد.

نکته: اگر قصد معرفی اسمبلی دیگری را به برنامه دارید و این اسمبلی امضای دیجیتال دارد (strong name signature)، باید تمام اطلاعات حاصل را ذکر کنید (مانند مثال فوق که شامل Public key token ، Version و غیره است). در غیر اینصورت (عدم وجود

امضای دیجیتال) ذکر دو قسمت اول خروجی خاصیت `AssemblyQualifiedName` کافی خواهند بود.

عنوان: **OneNote و مصرف بالای RAM و CPU**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۰۵ ۱۰:۲۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

از برنامه OneNote زیاد استفاده می‌کنم. عموما برای یادداشت برداری از سایت‌ها؛ فقط کافی است یک صفحه از مرورگر خودتون را با Ctrl+A انتخاب و با Ctrl+C در حافظه کپی کنید. سپس با Ctrl+V در OneNote ، کل صفحه با همان فرمت اصلی و تمام تصاویر، جداول و غیره ذخیره خواهد شد. همچنین در OneNote 2010، دریافت تصاویر از سایت‌ها به صورت asynchronous است (برخلاف نگارش 2007 آن) و حین دریافت تصاویر برنامه متوقف نمی‌شود.

اما اگر سرویس Windows search که کار indexing را انجام می‌دهد خاموش باشد:
احتمالا OneNote شما بالای 400 مگ رم مصرف خواهد کرد.
مرتبا هنگ می‌کند.
عموما CPU Usage ایی بالای 50 درصد به صورت مداوم خواهید داشت.
جستجوی آن دیگر درست کار نمی‌کند و بسیار بسیار کند خواهد بود.
و اگر سرویس یاد شده روشن باشد، همه چیز عادی است، از مصرف رم تا CPU و غیره.

عنوان: **محصولات آموزشی سایت Tekpub**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۲/۰۵ ۱۷:۲۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

سایت Tekpub محصولات آموزشی خودش رو به مدت فقط 24 ساعت به صورت رایگان در دسترس علاقمندان قرار داده. شامل دوره‌های کامل Entity framework ، ASP.NET MVC و غیره. بشتابید!

Tekpub's 24 Hour Open House

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۲/۰۵ ۱۹:۴۵:۱۶

نکته‌ای برای دریافت این ویدیوها:
چون به صورت استریم پخش می‌شوند نیاز به برنامه‌ی مخصوص برای دریافت دارند. بهترین برنامه برای اینکار IDM است چون لینک دانلود را هم تشخیص می‌دهد: (+)
مشکل سایت tekPub برای ما ها ... کیفیت HD پیش فرض این ویدیوها است. هر کدام حدود 500 مگ. در صورتیکه در سایت ذکر شده، برای سه کیفیت iphone ، HD و ipad هم ویدیوها مهیا هستند. برای اینکار نیاز است تا با user agent مربوط به iphone وارد شد. برای این منظور از این افزونه استفاده کنید: (+)
سپس با IDM به سادگی نگارش‌های کم حجم‌تر را می‌شود دریافت کرد. (حجم نسخه‌های مربوط به iphone کمتر از نصف HD است)

نویسنده: شاهین کیاست
تاریخ: ۱۳۸۹/۱۲/۰۵ ۲۰:۵۲:۱۹

سلام.
من رفتم قسمت EF
Episode اولی 3 4 دقیقه صبر کردم Icon دانلود IDM برام پدیدار نشد.
در صورتی که با IDM از YouTube ویدیو دریافت می کنم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۲/۰۵ ۲۰:۵۹:۵۰

به نظر مربوط به اختلالات عمدی اینترنت در این طرف است ... مثلا فایل‌های با پسوند mp4 رو نمی‌تونید الان دریافت کنید. مشکل اینجا است.

عنوان: **رادار فناوری اطلاعات**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۱/۳۰ ۱۰:۱۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

احتمالا یک سری نمودار مانند این را دیده باشید که هر از چندگاهی برای زبان‌های برنامه نویسی از دیدگاه محبوبیت تعیین رتبه می‌کنند. البته این آمار با آنچه که در سایت پر بازدید stackoverflow در جریان است عموما در تناقض است؛ از این لحاظ که برنامه نویسی با سؤال مواجه نمی‌شود که کار نمی‌کند!

بر این اساس عده‌ای آمار و اطلاعات سایت stackoverflow را هر از چندگاهی آنالیز می‌کنند تا متوجه شوند هم اکنون کدام زبان‌ها یا فریم ورک‌ها بیشتر مورد استفاده هستند؛ برای مثال:

کدامیک از فریم ورک‌های وب بیشتر مورد استفاده هستند؟ (+)

کدامیک از زبان‌های برنامه نویسی بیشتر محبوب هستند؟ (+) و (+) (محبوبیت در اینجا به معنای تعداد سؤال پرسیده شده است نه علاقه شخصی)

همین سوال بر اساس مسابقه هوش مصنوعی گوگل: (+) و (+)

این‌ها یک روی سکه هستند؛ روی دیگر آن هم به این صورت است که "الان واقعا اون ور آب‌ها چه خبره؟!" صرف نظر از علاقه شخصی من که بخواهند بر اساس آن نمودار رسم کنند یا تعداد سؤالات پرسیده شده در یک انجمن، نظر کارشناسان امر در این باره چیست؟ بر همین اساس شرکت ThoughtWorks مدتی است یک سری آمار را بر اساس وضعیت شرکت‌های بزرگ، بازار و علاقمندی‌های جاری آن‌ها منتشر می‌کند که از آدرس زیر قابل دریافت هستند:

ThoughtWorks Technology Radar

نظرات خوانندگان

نویسنده: Sina Jahan
تاریخ: ۱۳۹۰/۰۱/۳۰ ۱۷:۴۰:۱۲

پستت از گودر منو کشوند تا اینجا که یه تشکر خشک و خالی ازتون بکنم حداقل

عنوان: **ظهور میکرو ORMs**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۲/۰۲ ۱۶:۳۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

پس از " معرفی Microsoft.Data.dll یا WebMatrix.Data.dll " که یک کتابخانه‌ی سورس بسته و همچنین مخصوص وب ماتریکس می‌باشد، این ایده توسط سایر برنامه نویس‌ها دنبال و تبدیل به ORMs جدیدی با کمتر از 400 سطر کد شده است که به Micro ORMs هم شهرت یافته‌اند.
در اینجا شما هنوز هم کاملا با SQL سر و کار دارید اما با امکان استفاده بسیار ساده‌تر از پارامترها و همچنین بکارگیری قابلیت‌های جدید dynamic معرفی شده در دات نت 4 . برای مثال:

Dapper

```
var guid = Guid.NewGuid();
var customer = connection.ExecuteMapperQuery<customer>("select Age = @Age, Id = @Id", new { Age =
(int?)null, Id = guid });
```

Massive

```
var tbl = new Products();
var products = tbl.All(where: "CategoryID = @0 AND UnitPrice &gt; @1", orderBy: "ProductName", limit:
20, args: 5,20);
```

Massive
توسط آقای راب کانری که قبلا ORM دیگری را به نام ساب سونیک ایجاد کرده بود، تهیه شده و Dapper توسط تیم سایت StackOverflow جهت مواردی خاصی که استفاده از ORMs (از LINQ to SQL استفاده می‌کنند) هزینه زیادی داشته، مورد استفاده قرار می‌گیرد. در همان صفحه اصلی پروژه، یک سری آمار و ارقام از دید مقایسه کارآیی با سایر ORMs نیز ذکر شده‌اند.
حتی اگر قصد استفاده از آن‌ها را هم نداشته باشید مطالعه کد آن‌ها از دیدگاه کاربردهای عملی قابلیت‌های پویای زبان، بسیار آموزنده هستند.

نظرات خوانندگان

نویسنده: Majid
تاریخ: ۱۳۹۰/۰۲/۰۳ ۱۹:۴۹:۲۹

منون مطلب جالبی بود

نویسنده: Hamed
تاریخ: ۱۳۹۰/۰۲/۰۴ ۰۷:۲۰:۵۱

با درود
استاد با توجه به روش‌های مختلفی که برای پرس‌وجوی داده‌ها وجود داره و شما بسیاری از آن‌ها را در این‌جا معرفی کرده اید؛ خودتان چه روشی را می‌پسندید؟
خود من معمولاً پرس‌وجوهامو با اس‌کیو‌ـال (منظورم استور پروسیجر) می‌سازم که هم راحت‌تره و هم امکانات خوبی داره و برنامه فقط یک نوع استفاده کننده است. که البته لایه‌ی دسترسی به داده‌ها هم مهمه که شما چه روشی را پیشنهاد می‌کنید.

می‌خوام بدونم روش خوبی استفاده می‌کنم یا روش‌های دیگه با توجه به آماری که در لینک روش دپر داده شده بود سرعت و کارآیی بهتری داره؟

سپاس‌گزارم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۲/۰۴ ۰۸:۱۶:۲۶

سلام،
نظر من در مورد رویه‌های ذخیره شده در کامنت این مطلب اومده
http://www.dotnettips.info/2010/08/sql-wcf-ria-services.html

روش مورد استفاده فعلی هم در کامنت این مطلب
http://www.dotnettips.info/2009/05/5-orm.html

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۲/۰۵ ۰۰:۵۲:۱۵

جهت تکمیل مطلب جاری
A light-weight, dynamic data access component for .NET 4.0

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۰/۰۲/۰۵ ۰۸:۴۱:۴۳

سلام.
چرا اسم من که در پست زیر در نظرات عوض شده؟!
شده صدای سکوت !
کامنت هشتم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۲/۰۵ ۰۹:۳۵:۰۱

اصلش درسته ... مراجعه کنید به فایل سی اچ ام سایت

این یک باگ در یسکاس هست در مورد کامنت‌های قبل از یکپارچه شدن با آن
بهشون گزارش دادم ... حالا تا کی بخواهند درستش کنند

عنوان: **مجوز WTFPL**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۳/۱۲ ۲۱:۱۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

در بین مجوزهای سورس باز، یکی از اون‌ها که اتفاقا مورد پذیرش FSF هم هست ، عنوان جالبی داره که ترجمه‌اش به فارسی می‌شود: "برو هر غلطی که دلت می‌خواد باهاش بکن!" یا WTFPL = Do What The F.u.c.k You Want To Public License
نگارش یک این مجوز توسط Banlu Kemiyatorn نویسنده برنامه Window maker در سال 2000 ارائه شده و در سال 2007 توسط مدیر پروژه تیم Debian نگارش دوم آن ارائه گردیده است!
این مجوز به شما اجازه هر نوع تغییر یا هر روش توزیعی را در مورد برنامه‌ی مورد نظر می‌دهد.
ترجمه این مجوز هم به زبان فارسی به صورت زیر است:

"
مجوز برو هر غلطی که دلت می‌خواد بکن!
نگارش 2، دسامبر 2004

هر کسی مجاز است این مستند را کپی یا توزیع کند با این شرط که اگر تغییری در اصل آن داده شد، نامش را تغییر دهد.
شروط اصلی این مجوز به شرح ذیل اعلام می‌گردد:
0- فقط برو هر غلطی که دلت می‌خواد باهاش بکن

"

البته شاید این سؤال پیش بیاد که این موارد به چه دلیلی اضافه شده؟ احتمالا شاید شنیده باشید که عده‌ای GPL رو یک نوع سرطان می‌دونند؛ از این لحاظ که اگر طرف اون رفتید باید کل برنامه خودتون رو سورس باز ارائه بدید. به همین جهت کسانی که کار تجاری انجام می‌دهند از طرف سورس‌های پروژه‌های مبتنی بر GPL رد هم نمی‌شوند. در مقابل آن مجوزهایی مانند BSD یا MIT ملاحظات GPL را ندارند (+). در کل GPL تا به امروز لینوکس را زنده نگه داشته است.

نظرات خوانندگان

نویسنده: SMAH1
تاریخ: ۱۳۹۰/۰۳/۱۴ ۱۲:۵۹:۰۰

سلام
تا اونجایی که من می دونم،میشه به جای GPL از LGPL استفاده کرد تا هم بحث تجاریش حل بشه و هم بحث Opensource بودن برنامه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۳/۱۴ ۱۳:۱۹:۲۹

بله. برای مثال NHibernate مجوز LGPL دارد؛ به این معنا که منعی در مورد استفاده از کتابخانه‌های باینری کامپایل شده آن در یک پروژه تجاری و سورس بسته وجود ندارد.

عنوان: **به روز رسانی جدید دات نت 4**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۳/۱۷ ۰۹:۱۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

به روز رسانی جدیدی جهت رفع 35 مشکل در دات نت 4 ، روز قبل ارائه شده است که از آدرس ذیل قابل دریافت می‌باشد. این به روز رسانی شامل کلیه موارد برطرف شده در به روز رسانی‌های قبلی نیز می‌باشد:

دریافت KB2468871

ریز جزئیات KB2468871

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۳/۱۸ ۱۰:۰۲:۴۳

اگر پیغام زیر رو دریافت کردید

.KB2468871v2 does not apply, or is blocked by another condition on your computer

به این معنا است که این آپدیت مرتبط با سیستم شما نیست.

در حقیقت این آپدیت پس از اعمال سرویس پک یک ویژوال استودیوی 2010 معنا پیدا می‌کند

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۳/۱۸ ۱۶:۱۰:۵۷

لینک‌های مستقیم

Download x86 Version

Download x64 Version

Download ia64 Version

عنوان: **مشکلات نصب به روز رسانی‌های اخیر**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۳/۲۶ ۱۷:۳۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

مشکل نصب به روز رسانی‌های دات نت فریم ورک
اگر احتمالا آپدیت‌های اخیر دات نت فریم ورک را نتوانسته‌اید نصب کنید و هربار پس از ری استارت کامپیوتر مجددا سر و کله برنامه نصاب پیدا می‌شود و نهایتا پیغام failed را دریافت می‌کنید، متاسفانه فقط یک راه حل دارد. حذف موارد موجود و نصب مجدد نگارش جدید آن‌ها. برای حذف کلیه نگارش‌های دات نت فریم ورک از برنامه زیر استفاده کنید:

NetFx Cleanup tool.

سپس بهتر است از نصاب نگارش وب دات نت فریم ورک استفاده کنید. از این لحاظ که تمام به روز رسانی‌ها را به همراه دارد و متفاوت است با نسخه‌ی آفلاین آن.

مشکل نصب به روز رسانی‌های سیلورلایت
اگر هنگام نصب به روز رسانی جدید سیلورلایت هر بار پیغام silverlight.msi File Missing را دریافت می‌کنید، مسیر زیر را از رجیستری ویندوز حذف کنید. سپس به روز رسانی سیلورلایت بدون مشکل نصب خواهد شد:

HKEY_CLASSES_ROOT / installer / products / D7314F9862C648A4DB8BE2A5B47BE100

پ.ن.
من هر دو مشکل را با ویندوز سرور 2003 داشتم.

نظرات خوانندگان

نویسنده: Mehran Zand
تاریخ: ۱۳۹۰/۰۳/۲۷ ۱۰:۵۳:۵۳

این لینکی برای دانلوذ شما قرار دادید نسخه 32 بیتی اون بود که مشکل من با دانلود نسخه 64 بیتی حل شد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۳/۲۷ ۱۲:۴۲:۱۴

- اون برنامه حذف کننده که با تمام نگارش‌ها کار می‌کند.
- نصاب وب هم خودش تشخیص می‌ده سیستم شما 32 بیتی است یا 64 بیتی و حجم کمتری هم اتفاقا داره (یا دریافت می‌کنه) نسبت به نسخه‌ی آفلاین.

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۰/۰۴/۰۵ ۰۰:۳۱:۴۸

سلام من با مشکل زیر در سیلورلایت روبه رو شدم =>

Error: Unhandled Error in Silverlight Application Code: 1001 Category: RuntimeError Message: AG_E_UNKNOWN_ERROR

بعد از ایجاد پروژه و موقع اجرا (هیچ کدی هم ننوشتم)

به نظر شما مشکل از کجاست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۰۵ ۰۰:۵۱:۳۲

به روز رسانی‌های سیلورلایت دو نسخه دارند. یکی مخصوص کاربران نهایی، یکی هم مخصوص برنامه نویس‌ها. حجم این‌ها هم متفاوت است:

Silverlight Developer Run time

همچنین اخبار مرتبط با نگارش‌های متفاوت سیلورلایت همیشه اینجا هست:

release history

پس از این به روز رسانی، این شماره جدید باید در فایل‌های aspx و html ایی که object مربوط به سیلورلایت را اضافه کرده‌اند هم به روز شوند.

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۰/۰۴/۰۵ ۱۸:۳۹:۱۷

سلام

بنده این دو بسته رو نصب کرده بودم و با توجه به توضیحاتتون شماره جدید رو هم در html قرار داده و قبلی رو عوض کردم (برای پارامتر minRuntimeVersion مقدار 5.0.60401.0 رو قرار دادم؛ اگه اشتباه می کنم لطفا راهنمائیم کنید) اما مشکل همچنان وجود داره

به نظر شما مشکل از کجاست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۰۵ ۱۹:۵۴:۴۷

- لینک فوق همیشه آخرین نگارش مخصوص برنامه نویس‌ها را دارد و از این لحاظ بسیار عالی است (به صورت خودکار به جدیدترین نگارش ری دایرکت می‌شود). پس از نصب آن باید شماره نگارش 4.0.60531.0 در سیستم شما موجود باشد. این

شماره رو از طریق صفحه About مربوط به اطلاعات سیلورلایت (all programs---silverlight) می‌تونید مشاهده کنید.
- ضمنا خطای AG_E_UNKNOWN_ERROR عموما مرتبط می‌شود به وجود مشکلی در XAML تولیدی. بررسی کنید آیا فایل‌ها درست تولید شده‌اند؟ آیا فضای نامی فراموش نشده؟ آیا تمام ارجاعات به اسمبلی‌های مورد نیاز تعریف شده؟ آیا پس از به روز رسانی جدید سیلورلایت، پروژه مجددا Re build شده؟
در کل این خطای 1001 (+) فوق العاده عمومی است. برای مثال همانطور که می‌دونید در سیلورلایت دسترسی به یک سری منابع با پروتکل file://URLs میسر نیست (یعنی اگر برنامه سیلورلایت با یک صفحه html معمولی از روی هارد باز شده) و حتما باید پروژه شما به همراه وب سایت ASP.NET ایجاد شود تا منابع مورد نظر از طریق پروتکل http://URLs قابل دسترسی شوند (به دلایل امنیتی).

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۰۵ ۲۰:۴۸:۵۴

ضمنا به نظر نگارش 5 را نصب کردید. این مورد بدون داشتن silverlight tools مخصوص نگارش 5 در VS.NET قابل استفاده نیست. اگر SP1 نگارش 2010 رو نصب کردید فقط از نسخه‌ی 4 می‌تونید استفاده کنید.

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۰/۰۴/۰۵ ۲۳:۱۱:۱۶

من پروژه رو که ایجاد می کنم بدون هیچ عملی اونرو ران می کنم و با این مشکل برخورد می کنم
اما در مورد فایل XAML حق با شما هستش به محض اینکه هر کاری انجام میدم مثلا یک باتون به XAML اضافه می کنم با ارور زیر رو به رو می شم =>

System.InvalidOperationExceptionLayout measurement override of element 'Microsoft.Windows.Design.Platform.SilverlightViewProducer+SilverlightContentHost' should not return PositiveInfinity as its DesiredSize, even if Infinity is passed in as available size

و دیزاینر از بین می ره و اینکه در برخی موارد ویژوال استودیو کرش می کنه
هنوز نفهمیدم مشکل از کجاست چون نگارش های 3 و 4 هم دچار همین مشکل هستند !!!!!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۰۶ ۰۰:۱۸:۰۷

پیشنهاد من نصب مجدد VS2010 و سپس SP1 آن است. چیز دیگری هم نصب نکنید. نیازی نیست (تمام موارد با SP1 ارائه شده).
بعد از اون حداکثر یک sl-toolkit نصب کنید و نه بیشتر. نگارش 5 هم هنوز بتا است و باید صبر کنید تا نسخه‌ی پایدار آن ارائه شود.

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۰/۰۴/۰۶ ۱۶:۵۴:۵۲

ممنون از وقتی که گذاشتید آقای نصیری
این کار رو انجام میدم و نتیجه رو همینجا قرار میدم
ممنون

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۰/۰۴/۰۶ ۱۸:۱۶:۴۸

نگارش 5 رو از control panel حذف کردم و پس از یکبار restart و نصب sl toolkit4 و یک restart مجدد (واسه محکم کاری) مشکل برطرف شده و همه چیز ok هست
به نظر بتا بودن نسخه 5 و ناپایدار بودنش دلیل مشکلات بود
آقا وحید ممنون از وقتی که گذاشتید

یا علی

عنوان: **ردیابی واژه‌های کلیدی در سایت‌های اجتماعی**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۰۷ ۱۸:۴۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

فرض کنید می‌خواهید واژه KeyWord را در یک سری سایت اجتماعی از طریق Google reader تحت نظر داشته باشد که کلا اوضاع از چه قرار است و علاقمندی‌های عموم در این زمینه چیست؟ آخرین مطالب به اشتراک گذاشته شده کدام‌ها هستند و در این زمینه خاص، دنیا دست کیست؟

فید تحت نظر قرار دادن یک واژه کلیدی در
کلیه وبلاگ‌های سایت wordpress.com :

```
http://en.wordpress.com/tag/
```

KeyWord

```
/feed/
```

سایت twitter.com: (پیدا کردن فید این یکی آنچنان واضح نیست چون به نظر اخیرا توئیتر فیدها رو حذف یا مخفی کرده)

```
http://search.twitter.com/search.atom?q=
```

KeyWord

سایت delicious.com :

```
http://feeds.delicious.com/v2/rss/tag/
```

KeyWord

```
?count=15
```

سایت friendfeed.com :

```
http://friendfeed.com/search?q=
```

KeyWord

```
&format=atom
```

سایت stackoverflow.com :

```
http://stackoverflow.com/feeds/tag/
```

KeyWord

شما هم اگر مورد خاصی را مد نظر دارید که در بالا ذکر نشده، لطفا مطرح کنید.

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۰۹ ۱۱:۴۴:۲۹

فید یک واژه کلیدی در کلیه وبلاگ‌های سایت ASP.NET

http://weblogs.asp.net/search/Searchrss.aspx?q=&tag=KeyWord&o=DateDescending

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۲۱ ۰۹:۰۰:۱۱

فید یک واژه کلیدی در کلیه بلاگ‌های GeeksWithBlogs

http://geekswithblogs.net/tags/KeyWord/rss.aspx

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۲۷ ۲۲:۰۱:۱۵

فید یک واژه کلیدی در کلیه ویدیوهای یوتیوب

http://gdata.youtube.com/feeds/base/videos?q=KeyWord&alt=rss

نویسنده: mohammadsaheb
تاریخ: ۱۳۹۰/۰۴/۲۸ ۰۱:۱۵:۳۷

فید یک واژه کلیدی در کلیه وبلاگ‌های سایت msdn

http://blogs.msdn.com/search/searchrss.aspx?q=KeyWord

فید یک واژه کلیدی در کلیه وبلاگ‌های سایت sqlblog

http://sqlblog.com/search/Searchrss.aspx?q=KeyWord

نویسنده: mohammadsaheb
تاریخ: ۱۳۹۰/۰۴/۲۹ ۱۵:۱۰:۵۶

فید یک واژه کلیدی در کلیه وبلاگ‌های سایت msmvps

http://msmvps.com/search/Searchrss.aspx?q=KeyWord&o=Relevance

عنوان: **بررسی علت CPU Usage بالای برنامه در حال اجرا**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۳۱ ۰۰:۲۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

فرض کنید به یک سرور مراجعه کرده‌اید و شکایت از CPU Usage مربوط به پروسه w3wp.exe یا همان IIS Worker Process است که بالای 90 درصد می‌باشد. بر روی این سرور هم هیچ چیز دیگری نصب نیست و مطابق مقررات موجود، قرار هم نیست که برنامه‌ای نصب شود. اکنون سؤال این است که چطور تشخیص می‌دهید، کدام قسمت یکی از برنامه‌های دات نتی در حال اجرا (در اینجا یکی از برنامه‌های ASP.NET هاست شده)، سبب بروز این مشکل شده است؟ کدام ترد بیشترین زمان CPU را به خود اختصاص داده است؟ چطور باید خطایابی کرد؟

اولین کاری که در این موارد توصیه می‌شود مراجعه به برنامه‌ی معروف process explorer و بررسی برگه‌ی threads آن است. در اینجا حتی می‌توان call stacks مرتبط با یک ترد را هم مشاهده کرد. اما ... این برگه در مورد پروسه‌ها و تردهای دات نتی، اطلاعات چندانی را در اختیار ما قرار نمی‌دهد.

خوشبختانه امکان دیباگ پروسه‌های دات نتی در حال اجرا توسط کتابخانه‌ی MdbgCore.dll پیش بینی شده است. این فایل را در یکی از مسیرهای ذیل می‌توانید پیدا کنید:

```
C:\Program Files\Microsoft SDKs\Windows\vXYZ\bin\MdbgCore.dll
C:\Program Files\Microsoft SDKs\Windows\vXYZ\bin\NETFX 4.0 Tools\MdbgCore.dll
```

در ادامه می‌خواهیم توسط امکانات این کتابخانه، به stack trace تردهای در حال اجرای یک برنامه دات نتی دسترسی پیدا کرده و سپس نام متدهای مرتبط را نمایش دهیم:

```
using System;
using System.Collections;
using System.Diagnostics;
using Microsoft.Samples.Debugging.MdbgEngine;

namespace CpuAnalyzer
{
   class Program
   {
       static void Main(string[] args)
       {
           var engine = new MDbgEngine();

           var processesByName = Process.GetProcessesByName("MyApp");
           if (processesByName.Length == 0)
               throw new InvalidOperationException("specified process not found.");
           var processId = processesByName[0].Id;

           var process = engine.Attach(processId);
           process.Go().WaitOne();

           foreach (MDbgThread thread in (IEnumerable)process.Threads)
           {
               foreach (MDbgFrame frame in thread.Frames)
               {
                   if (frame == null || frame.Function == null) continue;
                   Console.WriteLine(frame.Function.FullName);
               }
           }

           process.Detach().WaitOne();
       }
   }
}
```

در اینجا در ابتدا نیاز است تا pid یا process-id مرتبط با برنامه در حال اجرا یافت شود. سپس از این pid جهت اتصال (engine.Attach) به پروسه مورد نظر استفاده خواهیم کرد. در ادامه کلیه تردهای این پروسه در حال دیباگ لیست شده و سپس

MDbgFrameهای این ترد بررسی می‌شوند و نام متدهای مرتبط در کنسول نمایش داده خواهند شد.
خوب در مرحله بعد شاید بد نباشد که این متدها را بر اساس CPU usage آن‌ها مرتب کنیم. به این ترتیب بهتر می‌توان تشخیص داد که کدام متد مشکل ساز بوده است. برای این منظور باید به API ویندوز و تابع GetThreadTimes مراجعه کرد و اولین پارامتر ورودی آن، همان thread.CorThread.Handle اولین حلقه مثال فوق می‌باشد. هر کدام که جمع KernelTime + UserTime بیشتری داشت، همان است که مشکل درست کرده است.

```
[DllImport("kernel32.dll", CharSet = CharSet.Auto, SetLastError = true)]
public static extern bool GetThreadTimes(IntPtr handle, out long creation, out long exit, out long
kernel, out long user);
```

این مورد را به عنوان تمرین بررسی کرده و ادامه دهید! همچنین بهتر است جهت دستیابی به اطلاعاتی معتبر، اولین حلقه برنامه فوق، حداقل 10 بار اجرا شود تا اطلاعات آماری بهتری را بتوان ارائه داد. البته در این حالت نکته‌ی زیر باید رعایت شود:

```
for (int i = 0; i < 10; i++)
{
   foreach (MDbgThread thread in (IEnumerable)process.Threads)
   {
     //...
   }
   process.Go();
   Thread.Sleep(1000);
   process.AsyncStop().WaitOne();
}
```

در کل این مثال جای کار زیاد دارد. برای مثال طراحی یک رابط کاربری برای آن و نمایش جزئیات بیشتر. به همین منظور حداقل سه پروژه مشابه را می‌توان نام برد:

Managed Stack Explorer

CLR Stack Explorer

CPU-analyzer

نظرات خوانندگان

نویسنده: Nima
تاریخ: ۱۳۹۰/۰۵/۰۱ ۰۸:۲۰:۱۵

سلام آقای نصیری
بخاطر آموزش بسیار مفیدتون ممنون.یه مطلب جالب بگم بارها شده دقیقا چیزی که باهاش به مشکل خوردم رو شما در پست بعدی وبلاگتون آموزش دادین D:
یهع سوال از خدمت شما داشتم اینکه چند وقته بطور اتفاقی CUP Usage سرور ما برای 2 3 بار در روز به 100 درصد میرسه و این حالت حدود 5 دقیقه ادامه داره. برنامه های ما به صورت Service Oriented نوشته شده و سرویسهای مختلفی(WCF) روی IIS هاست شدن. اگر بخوام کد بالا زمانی که CPU Usage مثلا از 80% بیشتر بشه فعال بشه چکار باید بکنم؟آیا لازمه یک ویندوز سرویس بنویسم؟
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۰۱ ۰۸:۲۷:۰۴

سلام، نه. تمام سرویس‌ها و تمام برنامه‌ها هم در نهایت یک پروسه در ویندوز هستند. شما در کد بالا بجای MyApp ، نام پروسه دات نتی مورد نظر را قرار دهید (حتی اگر پروسه مثل w3wp باشد (یعنی یک سری برنامه دات نتی رو هاست کرده) هم کار می‌کند). سپس می‌تونید شروع به دیباگ کنید. البته قسمت اندازه گیری KernelTime + UserTime را که ذکر شد خودتون باید لحاظ کنید. به این ترتیب (به کمک مرتب سازی بر اساس KernelTime + UserTime) می‌تونید نام متدی که الان سبب بروز این CPU usage بالا شده رو دقیقا تشخیص بدید.

نویسنده: Nima
تاریخ: ۱۳۹۰/۰۵/۰۱ ۰۸:۳۱:۱۸

ممنون آقای نصیری
از کجا میشه درصد استفاده از CPU رو متوجه شد؟ وقتی CPU Usage ما به 100 درصد میرسه نه میشه به سرور کانکت شد و نه اینکه وقتی کانکتیم میذاره برنامه ای رو اجرا کنیم. بخاطر همین مجبوریم وقتی درصد از حدی بالاتر رفت بلافاصله یه سری اطلاعات را ثبت کنیم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۰۱ ۰۸:۴۵:۲۱

در یک چنین حالتی بله. می‌تونید سرویس ویندوز ان اتی بنویسید برای بررسی خودکار. مطلب فوق یک بررسی دستی است. یک تایمر داخل این سرویس تعریف کنید سپس به کمک WMI ((+)) هر از چند لحظه یکبار، وضعیت CPU usage کل سیستم را بررسی کنید. سپس اگر بالا بود می‌شود کل پروسه‌های ویندوز را لیست کرد و نکته فوق را (بر اساس مرتب سازی اطلاعات با توجه به KernelTime + UserTime) به آن‌ها اعمال کرد. نهایتا یک گزارش متنی یا یک ایمیل می‌شود از این برنامه گرفت.

نویسنده: masud
تاریخ: ۱۳۹۰/۰۵/۰۱ ۱۲:۵۲:۴۰

با سلام خدمت آقای نصیری
چند وقتی است روی یکی از سرورهی ما مش out of memory به وجود آمده
مشخصات سرور ویندوز 2003 سی دو بیتی رم بالای 4 گیگابایت که با سویچ /PAE در فایل بوت ... این حافظه به ویندوز معرفی شده در
ضمن اینکه Lock Page in memory برای یوزری که سرویس IIS را Start کرده تنظیم شده در این حالت باز ما خطا را میگیرم
چیزی که به ذخن من میرسه ما در برنامه از کریستال ریپورت استفاده کردیم که داخل اون یک library ار نوع com وجود داره که کار

تبدیل تاریخ میلادی به شمسی رو انجام میده فکر میکنم به خاطر اینکه این کتابخانه به شکل unmanage code است
سوالی که دارم اینه
چه طور می توانم مطمئن بشم مشکل از این کتابخانه است و یا خیر

لازم میدونم اشاره کنم تعداد کاربرای سایت بالا و همچنین این سرور مختص iis است و به هیچ عنوان SQL روی اون نصب نمی باشد

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۰۱ ۱۳:۱۶:۳۲

سوال شما مرتبط با بحث CPU Usage نیست...
- در سرور 32 بیتی شما نمی‌تونید استفاده مناسبی از سخت افزاری که تهیه کردید بکنید. نیاز است به سرور 64 بیتی نقل مکان کنید. یعنی هزینه کردید اما ... از آن استفاده بهینه‌ای نمی‌کنید.
- تابع SQL ، برای تبدیل تاریخ میلادی به شمسی وجود دارد (بگردید در انجمن‌ها هست). از همان تابع در کوئری‌های خودتون قبل از ارسال به گزارشات استفاده کنید. یعنی دیتا از همان اول تاریخ شمسی داشته باشد. به این صورت نیازی به استفاده از افزونه دیگری نیست.
- IIS یک قسمت دارد به نام Application pool . اولا به ازای هر برنامه بهتر است یک Application pool جدا درست کنید. این حالت دقیقا مثل اجرای هر سایت در یک پروسه جدا است. مثلا IE8 رو دیدید که هر tab آن در یک پروسه جدا اجرا می‌شود، یا مثلا مرورگر کروم هم به همین صورت است. Application pool در IIS هم دقیقا همین معنا را دارد. عادت متداول، استفاده از یک Application pool به ازای تمام سایت‌ها است که غلط است. یعنی زمانیکه این توانایی هست چرا استفاده نمی‌کنید؟ ثانیا در همین Application pool می‌تونید تنظیم کنید که اگر میزان مصرف حافظه برنامه منتسب، مثلا به 600 مگ رسید لطفا خودت به صورت خودکار اون رو recycle کن تا مشکلات کمبود حافظه رخ ندهد.
- از برنامه‌های dotTrace و Ants memory profiler هم می‌تونید استفاده کنید. این‌ها قابلیت دیباگ پروسه‌های مرتبط با IIS رو هم دارند و دقیقا گزارشات آماری پیشرفته‌ای می‌تونند از وضعیت مصرف حافظه قسمت‌های مختلف برنامه در اختیار شما قرار بدن: (+) و (+)

نویسنده: Nima
تاریخ: ۱۳۹۰/۰۵/۰۱ ۱۵:۲۴:۱۱

سلام مجدد آقای نصیری
من پیرو این مطلب مفید شما یک برنامه نوشتم اما با مشکلاتی مواجه شدم که در اینجا سوال پرسیدم :

http://stackoverflow.com/questions/6799007/check-cpu-usage-and-log-processes-that-have-up-to-30-cpu-usage-in-windows

قضیه اینه که CPU من 4 هسته داره و من فکر میکنم چیزی که WMI به من میده مال یکی از هسته هاست. یعنی اصلا نمیخونه عددی هر 100 میلی ثانیه میخونم با عددی که در Task Manager میبینم. درضمن من فکر کردم که پراسسی که ماکزیمم استفاده از CPU رو داره رو اطلاعاتش رو بگیرم ولی متاسفانه هر چی گشتم چیز مناسب و بدرد بخوری پیدا نکردم. ممنون میشم من رو کمی راهنمایی کنید.
باتشکر

نویسنده: Hossein Raziee
تاریخ: ۱۳۹۰/۰۵/۰۱ ۱۶:۰۵:۳۶

با درود
جناب نصیری برام سوالی مطرح شد. شما مخالف ثبت تاریخ به صورت میلادی در دیتابیس هستید؟ و یا در این مورد خاص که دوستمون گفتند این موضوع رو پیشنهاد میکنید؟
از این که سوال ارتباطی با موضوع پست نداره پوزش میخوام ولی برام جالب بود بدونم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۰۱ ۱۶:۴۵:۵۲

استفاده از تاریخ میلادی در دیتابیس خوبه. مثلا آمارگیری از تاریخ تا تاریخ یا اعمال بسیاری از امکانات توکار بانک‌های اطلاعاتی. اما یکجا مشکل ساز می‌شود و آن هم گروه بندی بر اساس ماه‌های شمسی است. مثلا گزارش جمع حقوق کارکنان را بر اساس ماه‌های شمسی یک سال تهیه کنید. این گزارش 12 سطر دارد (به ازای هر ماه) و 2 ستون (نام ماه و جمع حقوق پرداختی). اینجا است که کوئری SQL آن اصلا شکل قشنگی پیدا نمی‌کند که هیچ (چون ماه‌های تاریخ میلادی تطابقی با ماه‌های شمسی ندارد)، بسیار هم غیربهینه می‌شود. به همین جهت یکی از سربارهایی که می‌شود از آن چشم پوشی کرد، نگهداری تاریخ شمسی و میلادی با هم است.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۰۱ ۱۷:۲۵:۵۹

- بله. هر بار در طی حلقه شما CPU Usage یک هسته رو دریافت می‌کنید. یک روش دیگر هم اینجا ذکر شده: ((+)). که باید دوبار اطلاعات دریافت شود و بعد بین آن‌ها هم یک sleep کوچک نیاز است.
- لیست تمام پروسه‌های ویندوز با این متد قابل دریافت است: Process.GetProcesses (در فضای نام System.Diagnostics) که یک سری اطلاعات در مورد TotalProcessorTime دارد.
همچنین این پروژه هم در راستای مطلب جاری است:
How to get CPU usage of processes and threads
البته این پروژه اطلاعاتی در مورد stack trace تردها (مثل روش فوق) نمی‌دهد فقط یک سری اطلاعات کلی است.

نویسنده: Nima
تاریخ: ۱۳۹۰/۰۵/۰۲ ۰۸:۲۰:۲۲

با سلام مجدد جناب نصیری
با تشکر از وقتی که گذاشتین باید عرض کنم من این لینک (Stackoverflow)رو موقع سرچ دیدم ولی انگار این نسخه از دات نت که کد رو بر اساسش نوشتن با نسخه 2010 تفاوتهای زیادی داره چون اصلا کدها رو نمیشناسه .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۰۲ ۰۹:۰۳:۲۴

جهت کامپایل سورس "How to get CPU usage of processes and threads" ، در پروژه CpuUsageAPI ، پوشه properties آن خالی است. یک سری فایل نیست. مهم هم نیستند. این‌ها را از پروژه حذف کنید. تست شد با دات نت 4 هم کامپایل می‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۲۳ ۲۱:۱۴:۱۵

نگارش بعدی ASP.NET امکانات بیشتری روی کنترل CPU usage دارد:
CPU Throttling for ASP.NET Asynchronous Scenarios in vNext

عنوان: **هزینه استفاده از دات نت فریم ورک چقدر است؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۱۵ ۱۲:۱۳:۰۰
آدرس: www.dotnettips.info
گروه‌ها: Tips

سؤالی از طریق ایمیل از من پرسیده شده که ترجیح می‌دهم آن‌را به صورت باز در اینجا پاسخ دهم. اگر فرض کنیم همین فردا مجبور شویم برای عمده‌ی کارهای خود لایسنس تهیه کنیم، آیا می‌توان از ابزارهای موجود دات نت در یک شرکت تازه تاسیس (startup) استفاده کرد؟ آیا هزینه‌ی کار با ویندوز واقعا بالا است؟ آیا ...
همچنین عموم تازه واردان به این جمع هم از لیست امکانات رایگان مهیا که فقط پس از خرید یک لایسنس اولیه ویندوز در اختیار آن‌ها خواهند بود، بی‌اطلاع هستند. بنابراین بد نیست این‌ها را با هم لیست کنیم.

سؤال: هزینه استفاده از دات نت فریم ورک چقدر است؟
پاسخ: رایگان است! از زمان ارائه ویندوز سرور 2003 به بعد، دات نت فریم ورک به عنوان یکی از کامپونت‌های اصلی ویندوز عرضه می‌شود و هر شخصی که مجوز اصلی استفاده از ویندوز را خریده باشد، به صورت خودکار مجوز استفاده از دات نت فریم ورک را هم خواهد داشت و نیازی نیست بابت آن هزینه‌ی خاصی را متقبل شود. برای مثال ویندوز سرور 2003 ، دات نت 1.1 سرخود است و ویندوز 7 و ویندوز سرور 2008 ، دات نت سه و نیم سرخود هستند.

سؤال: آیا برای توسعه‌ی دات نت حتما نیاز است تا ویژوال استودیوی چند هزار دلاری را خرید؟!
پاسخ: خیر! ویژوال استودیو، نسخه‌های مختلفی دارد و حتما نیازی نیست تا از نسخه‌ی ultimate آن استفاده کنید. برای مثال نسخه‌ی Express آن که توسط خود مایکروسافت ارائه شده، رایگان است (+). مهم‌ترین تفاوت آن با نسخه‌ی ultimate در این است که افزونه پذیر نیست و این مورد شاید برای خیلی‌ها اصلا اهمیتی نداشته باشد چون عموما افزونه‌های بد نوشته شده، باعث ناپایداری IDE می‌شوند یا مثلا نسخه‌ی ultimate به همراه MSTests جهت انجام ساده‌تر unit testing ارائه شده که در نسخه‌ی Express وجود ندارد، این هم مهم نیست چون فریم ورک‌های سورس باز آزمون واحد دیگری مانند MBUnit ، Nunit و غیره هم وجود دارند که اصلا نیازی به هیچ IDE خاصی جهت کار ندارند و مسایلی از این دست. یا برای سورس کنترل می‌شود از Git ، SVN ، مرکوریال و غیره هم همیشه استفاده کرد. این‌ها هم مستقل هستند از نوع IDE مورد استفاده.

همچنین یک گروه مستقل، IDE دیگری را به نام SharpDevelop تهیه کرده‌اند که بسیار با کیفیت بوده و از همه مهم‌تر سورس باز است و رایگان. خیلی‌ها از کنترل‌های این IDE در پروژه‌های خودشون استفاده می‌کنند (مثل syntax highlighting همراه آن و غیره)

در کنار تمام این‌ها، هیچ وقت دقت کرده‌اید که در پوشه دات نت نصب شده در ویندوز چه چیزهایی یافت می‌شوند؟! مثلا به آدرس C:\Windows\Microsoft.NET\Framework\v4.0.30319 مراجعه کنید. فایل csc.exe همان کامپایلر خط فرمان سی شارپ است. کاری هم که IDE ها انجام می دهند این است که لیست فایل‌های شما رو به صورت آرگومان به این برنامه ارسال می‌کنند. این اتوماسیون به صورت دستی هم میسر است. IDE شما می‌شود مثلا برنامه رایگان notepad++ ایی که syntax highlighting تمام زبان‌های دات نتی را پشتیبانی می‌کند. ساده‌ترین سیستم build شما یک فایل bat خواهد بود که csc.exe را مدیریت می‌کند.

علاوه بر تمام این‌ها، MonoDevelop را هم اضافه کنید. این IDE ، نسخه‌ی ویندوزی هم دارد.

سؤال: من شنیدم دات نت فقط با SQL Server کار می‌کنه. این هم که گرونه؟!
پاسخ: خیر! نسخه‌ی رایگان SQL server به نام SQL Server express هم موجود است و در نسخه‌ی 2008 R2 آن محدودیت حجم بانک اطلاعاتی آن به 10 گیگ رسیده که برای اکثر کارها تا چند سال کافی است (+).
ضمنا قبلا در مورد لیست بانک‌های اطلاعاتی قابل استفاده توسط دات نت فریم ورک مطلب نوشتم و در سایت موجود است.
بنابراین دات نت اصلا به SQL Server محدود نیست (+).

سؤال: دات نت سورس بسته است! نه این اصلا خوب نیست؛ آینده نداره!
پاسخ: خیر. سورس کامل دات نت فریم ورک تحت مجوزهای MS-PL و MS-RSL چندسالی هست که منتشر شده و این مجوز (MS-PL) جزو مجوزهای معتبر و پذیرفته شده سورس باز است (+).
لینک دریافت

خبر مرتبط

ضمنا کارهای سورس باز مایکروسافت به این یک قلم ختم نمی‌شود. برای مثال به سایت CodePlex مراجعه کنید تا سورس کامل ASP.NET MVC نگارش 3 را دریافت کنید(+)، همینطور مجموعه کنترل‌های WPF و Silverlight و غیره‌ای که تهیه کرده‌اند (+) و بسیاری موارد دیگر.

سؤال: دات نت که فقط روی ویندوز اجرا میشه. نه؛ این خوب نیست!
پاسخ: خیر! پروژه‌ای سابقا تحت عنوان Mono وجود دارد/داشت (+) که توسط شرکت ناول اداره می‌شد و کار آن انتقال دات نت به لینوکس و سایر سکوهای کاری بود. پس از فروخته شدن ناول، این پروژه به ظاهر متوقف شد، اما تیم اصلی آن تحت نام دیگری به نام زاماریان (+) دوباره شروع به کار کرده و جالب اینجا است که تا دات نت 4 را هم تحت لینوکس پشتیبانی می‌کند(+).
جهت اطلاع ASP.NET رو به صورت کامل می‌تونید تحت لینوکس اجرا کنید (+). همچنین سایر موارد پشتیبانی شده رو.

سؤال: وضعیت بازار کار دات نت در ایران چطور است؟
پاسخ: آقای محبی برنامه‌ای رو درست کرده‌اند به نام کارویس که من مدتی است مشترک فید آن هستم تا حدودا از نیازهای عمومی بازار کار برنامه نویسی ایران اطلاع کسب کنم (+). نکته‌ی جالبی که در فید این برنامه مشخص است، درخواست بالای نیاز به برنامه نویس دات نت در کشور است. بالای 80 درصد آگهی‌ها نیاز به برنامه نویس دات نت دارند.

در کنار تمام این مباحث، بحث « **هزینه یادگیری و آموزش** » را هم اضافه کنید. شاید یکی بیاد بگه CPP فلان، اون یکی بهمان! ولی این سؤالات هم برای من نوعی که تیم گوگل کروم نوشته شده با CPP رو مدیریت نمی‌کنم (و به قول اونطرف آبی‌ها یک Average Joe هستم!) مهم هستند:
- چندتا کتاب فارسی آموزشی خوب در این زمینه در کشور موجود است. من یادم میاد اولین کتاب VC منتشر شده در کشور پس از 450 صفحه به شما یاد می‌داد چطور با MFC دکمه به صفحه اضافه کنید، چطور منو درست کنید و همین! این شد برنامه نویسی CPP!
- چند عدد انجمن رفع اشکال فعال را که تاریخ آخرین به روز رسانی و پرسش و پاسخ آن‌ها مثلا دیروز بوده را در مورد زبان xyz می‌توان یافت؟
- اگر به مشکل برخوردم، کسی هست به داد من برسه؟ چه زمانی؟! مثلا هفته بعد یا تا آخر امروز؟
- چندتا وبلاگ فعال در این زمینه موجود است؟ آیا هستند کسانی که در این زمینه‌ها فعالانه مطلب منتشر کنند؟ اطلاع رسانی کنند؟
- اگر کارمند برنامه نویس شرکتی امروز قهر کرد، مدیر بخش توسعه تا چه زمانی می‌تونه یک نفر رو جایگزین اون کنه؟ مثلا تا آخر ماه یا تا آخر سال؟! اون هم با چه کیفیتی؟ با چه دستمزدی؟
و اینجا است که وضعیت دات نت در ایران بسیار مطلوب‌تر از موارد مشابه به نظر می‌رسد. از نظر تعداد کتاب فارسی مهیا، تعداد انجمن، تعداد وبلاگ فعال و غیره.

نظرات خوانندگان

نویسنده: afsharm
تاریخ: ۱۳۹۰/۰۶/۱۵ ۱۳:۳۴:۳۷

سلام مجدد، تازگی‌ها به ذهنم خطور کرده بود که حساب کتاب کنم ببینم در یک شرکت ۲۰ نفره نرم افزاری ایرانی اگر نخواهیم خیلی سخت بگیریم چقدر باید هزینه مجوز کنیم و آیا این هزینه نسبت به بقیه هزینه‌های یک شرکت چیز زیادی می‌شود یا نه. و به این ترتیب بتوان کمی دوستان را ترغیب کرد به رعایت حق تالیف. جالب بود که شما بخش زیادی از این موضوع رو پوشش دادید.

یک نکته در مورد مونو. من چند وقت پیش خیلی دنبال این موضوع بودم که آیا ما ویندوز/ویژوال استودیو کارها می‌تونیم به لینوکس/مونو مهاجرت کنیم و چیز زیادی رو از دست ندیم؟ و دست آخر به این نتیجه رسیدم که اقلاً در مورد ASP.NET مقداری کمبود و کاستی وجود دارد. البته هنوز بررسی‌ها و آزمایشاتم تمام نشده. به اینجا نگاه کنید:
http://programmers.stackexchange.com/questions/96154/can-i-use-linux-mono-instead-of-windows-visual-studio-without-any-problem

نویسنده: پرهام
تاریخ: ۱۳۹۰/۰۶/۱۵ ۱۸:۰۳:۵۳

ممنون، اطلاعات جالبی بود.یک خواهش دیگه مطرح میکنم!! سوال من درباره مدارک Microsoft هست. MCTS , MCPD و خیلی چیزهای دیگه که هر کدوم صد شاخه میشن! (با کمی مبالغه) این مدارک برای چه کسانی مفید هستند؟ آیا باید همه course ها را گذراند؟ در کل امکانش هست در این باره مطلبی بنویسید؟ یا اینکه نوشته اید؟!

نویسنده: Hadi Eskandari
تاریخ: ۱۳۹۰/۰۶/۱۵ ۲۱:۳۰:۲۹

سلام ، چند تا مورد را خواستم متذکر بشم. درست است که ورژن VisualStudio Express مجانی است ، ولی همین که امکان اجرای Addin نداره باعث محدودیت خیلی زیادی میشه. اکثر کسانی که از Test Runner ها استفاده میکنند از نوعی Addin برای اجرای اونها استفاده میکنند بطوری که بدون ساپورت Addin استفاده از این نسخه برای این افراد غیر ممکنه میشه.

نکته دومی که بهش اشاره نکردید ، این است که در خود اروپا و امریکا شرکت های نرم افزاری Microsoft Partner پولی بابت License برای VisualStudio نمیدند. شرکت هایی هم که پول میدهند بابت MSDN Subscription میدهند که مثلا با ده هزار دلار تا یک سال تقریبا به همه محصولات دسترسی دارند و جدا جدا پولی پرداخت نمیکنند. شرکت هایی هم که Microsoft Partner باشند معمولا یک هزینه خیلی پایین (در حد پنج هزار دلار) در سال میدهند بابت حدود صد نفر برنامه نویس که این شامل هزینه سرور ها ، دیتابیس ، ویندوز و ... میشود. البته در حال حاظر در ایران به علت تحریم عملا این روش قابل استفاده نیست.

نویسنده: A.Karimi
تاریخ: ۱۳۹۰/۰۶/۱۵ ۲۱:۳۳:۳۷

سورس دانت به صورت Shared Source توزیع شده و به عبارت دیگر سورس پشت شیشه است (فقط جهت Reference و از این قبیل). تا آنجا که می‌دانم این مجوز توسط جامعه متن باز پذیرفته شده نیست. متن باز امکان تغییر را نیز می‌دهد. و یا شاید منظور شما از سورس باز، متن باز نبوده!

در مورد قیمت VS یکی از بچه‌ها صحبت خوبی داشت؛ اگر قرار بود ما VS را خریداری کنیم هرسال یا هر چند سال یکبار این کار را می‌کردیم و مایکروسافت با توجه به Optionهای خوبی که برای اینطور خرید کردن دارد (Renew و ...)، می‌توان VS Ultimate که اگر اشتباه نکنم 11 هزار دلار است را با 3 هزار دلار تهیه کنیم. وقتی همه حقوق مولفین رعایت شود مطمئناً 3 هزار دلار برای یک برنامه نویس متوسط به بالا مبلغ بسیار مناسبی خواهد بود. مشترک MSDN بودن هم که شامل سیستم‌عامل‌ها و ... هم خواهد شد.

نویسنده: وحید نصیری

تاریخ: ۲۱:۴۰:۴۹ ۱۳۹۰/۰۶/۱۵

مجوز رو می‌تونید اینجا مطالعه کنید:
http://www.opensource.org/licenses/ms-pl.html
و زمانیکه در این سایت مطرح شده یعنی پذیرفته شده و معتبر است.
توضیحات بیشتر هم در این سؤال و جواب:
http://programmers.stackexchange.com/questions/85301/understanding-the-microsoft-public-license-ms-pl

نویسنده: A.Karimi
تاریخ: ۲۲:۰۶:۰۳ ۱۳۹۰/۰۶/۱۵

البته که در مورد MS-PL و باز بودن آن کاملاً حق با شماست. اما سورس دات نت (حتی BCL) تحت این مجوز ارائه نشده (در مورد تکنولوژی‌هایی مثل LINQ و WCF و ... حتی هنوز استاندارد نشده که بتوان پیاده‌سازی قانونی از آنها داشت).

مجوز دات نت در ویکیپدیا "MS-EULA, BCL under Microsoft Reference Source License" عنوان شده.

به همین خاطر برنامه‌نویسان مونو حق دیدن سورس‌های مایکروسافت را ندارد.

خلاصه مجوز MS-RSL:
http://en.wikipedia.org/wiki/Shared_source#Microsoft_Reference_Source_License_.28Ms-RSL.29

و توضیحات کاملتر درباره استاندارد شدن دات نت:
http://en.wikipedia.org/wiki/.NET_Framework#Standardization_and_licensing

نویسنده: وحید نصیری
تاریخ: ۲۲:۳۰:۵۴ ۱۳۹۰/۰۶/۱۵

- هر کسی می‌تونه برای کار خودش مجوز سورس باز انتخاب کنه، و مهم هم نیست که آن کار استاندارد باشد یا نباشد. مثل هزاران کاری که تابحال دیگران انجام دادن.
- در مورد مجوز فقط خواندنی MS-RSL حق با شما است؛ هر چند مثلا ASP.NET MVC3 مجوز MS-PL دارد. ولی ... مهم نیست. مهم این است که سورس آن در دسترس همه است. همین در دسترس بودن سبب شده مونو الان پیاده سازی خوبی رو از ASP.NET MVC 3 ارائه بده:
http://www.mono-project.com/Release_Notes_Mono_2.10#ASP.NET_MVC3_Support

نویسنده: Jobfa Com
تاریخ: ۱۰:۲۰:۰۳ ۱۳۹۰/۰۶/۱۶

با اجازه شما من این مطلب را عینا در وبلاگ خودم منتشر کنم

نویسنده: وحید نصیری
تاریخ: ۱۱:۱۱:۲۰ ۱۳۹۰/۰۶/۱۶

سلام، خواهش می‌کنم.
فقط قسمت مجوز سورس دات نت کمی باید ویرایش شود. مثلا MVC مجوز MS-PL دارد، مابقی مجوز فقط خواندنی MS-RSL. ولی در کل سورس آن از لینک داده شده قابل دانلود، بررسی و مطالعه کامل است.

نویسنده: کوروش
تاریخ: ۱۴:۰۸:۵۸ ۱۳۹۰/۰۶/۱۶

سلام دوست عزیز بنده قصد تخریب یا نفی گفته های شما رو ندارم فقط نکاتی که به نظرم میرسه بیان می کنم:

نسخه express ویژوال استودیو به گفته خود مایکروسافت هم در سطحی نیست که توسعه دهنده های حرفه ای نرم افزار ازش استفاده کنن و بیشتر برای علاقه مندان و دانشجو ها و یا تست پروژه کاربردش توصیه شده. برای مثال می تونید از همین شرکت های ایرانی فعال در نرم افزار بپرسیم که کدومشون می تونن از نسخه express بدون مشکل استفاده کنن. IDE های دیگر هم برای دات نت به پای ویژوال استودیو نمی رسن اکثرا هم این مورد رو موافقن که بهترین IDE برای دات نت همون ویژوال استودیو هست. اون مورد هم که به صورت دستی کامپایل و دیباگ رو انجام بدیم که به این موضوع بر می گرده که اصلا از IDE استفاده بکنیم یا نه؟ هدف تو دات نت اینه که توسعه دهنده فکر و نیروش روی طراحی و منطق برنامه بگذاره و تا حد امکان درگیر این مسایل نشه بنابراین کار با IDE برای هماهنگی با این موضوع ضروری هست. اگه توسعه دهنده ها بخوان زحمت این ریزه کاری ها رو تحمل کنن خیلی بهتره که به جای دات نت سراغ یه پلتفرم و زبان دیگه برن چون عملا از خیلی از مزایاش استفاده نمی کنن.

در مورد پروژه مونو هم بیشتر میشه گفت این تعریف ها اسم هست تا واقعی. برای مثال میشه به این نگاه کرد که چه پروژه ی بزرگی تا به حال با مونو پیاده سازی شده؟ به جز چند برنامه متن باز و پروژه های کوچیک دیگه شرکتی پیدا نمیشه که بخواد روی مونو سرمایه گذاری کلانی کنه. یکی از دلایل این عدم استقبال از مونو مربوط میشه به ترس از این که امکان داره مایکروسافت با شکایت علیه مونو تمام مجوز های استفاده از اون رو سلب کنه و اون وقته که هر چی رشته بودن پنبه میشه. حتی پروژه های متن باز هم سعی می کنن تا جایی که امکان داره از مونو دوری کنن. واقعا به طور عملی نمیشه فعلا روی مونو حسابی باز کرد مخصوصا از نوع تجاری.

این موارد رو فقط برای این گفتم که با دید بازتری به مساله نگاه کنیم نه این که حرف های شما رو نقض کنم. ولی باید این مورد هم در نظر داشت که با وجود این که سری رایگان محصولات express مایکروسافت و پروژه مونو وجود دارن آیا در یک محیط کاری و تجاری میشه از اون ها استفاده کرد یا خیر. چیزی که تو تقریبا تمام شرکت ها مشاهده می کنیم که که عملا همه از نسخه های پولی ویژوال استودیو و SQL server استفاده می کنن و خبری هم از مونو نیست.

حرف آخرم اینه منظورم این نیست که بگیم باید از مایکروسافت دوری کرد. مایکروسافت یه پلتفرم درست کرده برای یه عده از توسعه دهنده ها که تونسته موفق هم باشه و خیلی ها هم جذب کنه. اتفاقا خیلی هم خوبه که در دات نت توی ایران پر بار هستیم. ولی آیا زبان ها و پلتفرم های دیگه هم به اون جایگاهی که دارن در ایران رسیدن؟ بازار کار دات نت در ایران خوبه اما آیا این تک قطبی شدن بازار به نفع توسعه دهنده یا مشتری هست؟ منابع غیر دات نت در ایران کم هست اما این دلیل میشه که زبان های دیگرو نفی کنیم و همه منابع انگلیسی و دیگر اون ها رو نادیده بگیرم؟ اگه فقط بخوایم به چیزی که جا افتاده هست مهر تایید بزنیم و بقیه رو نادیده بگیریم خب حرفی باقی نمی مونه اما اگه می خوایم دانش کامپیوتر رو گسترش بدیم به بقیه هم باید همون قدر بها بدیم.
با احترام

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۱۶ ۱۴:۳۶:۵۲

- نسخه express منهای بحث افزونه پذیری و مواردی که ذکر شد، چیزی کمتر از نسخه‌ی ultimate ندارد. توسعه دهنده‌ی حرفه‌ای هم که نمی‌دونه مثلا بجای MSTest می‌تونه از NUnit استفاده کنه، خوب کمی این واژه‌ی حرفه‌ای برازنده‌ی اون نیست.
- اون مورد کامپایل دستی هم به این دلیل ذکر شد که کمی بیشتر با امکانات موجود آشنا بشیم. کمی شناخت بیشتری از زیر ساخت‌ها پیدا کنیم.
- در مونو که عرض کردم دیگری پروژه‌ی شرکت ناول وجود ندارد. الان شده زاماریان. در همان زمان هم جالب است بدونید مایکروسافت کمک‌های مادی زیادی به ناول کرد که لینوکسی‌ها خیلی به این قضیه مشکوک شده بودند که چرا. در مورد برنامه‌های پیاده سازی شده با آن هم لطفا به لیستی که در سایت آن‌ها هست مراجعه کنید. ضرورتی به تکرار آن‌ها اینجا نیست.
- بله می‌شود در محیط تجاری از آن‌ها استفاده کرد و خیلی‌ها اینکار رو می‌کنند.
- اینکه بقیه کم اطلاع هستند خوب می‌شود اطلاع رسانی کرد. به همین دلیل این مطلب نوشته شده. هدف دیگری در کار نبود.
- اینکه زبان‌های دیگه به این حد نرسیدن مشکل طرفداران آن‌ها هست. بروند کتاب چاپ کنند، زحمت بکشند، تلاش کنند. مثلا فکر می‌کنید سرپا نگه داشتن همین بلاگ کار ساده‌ای است؟ برای نمونه مطلبی رو که در مورد تبدیل html to pdf چند روز قبل منتشر کردم یک هفته کار برده بود تا نحوه‌ی اعمال فونت فارسی رو به اون بتونم تکمیل کنم.

نویسنده: Arash
تاریخ: ۱۳۹۰/۰۶/۱۶ ۱۵:۰۹:۳۱

اینجوری که من این متن رو خوندم فکر نکنم منظورشون نادیده گرفتن زبانهای دیگر بود.آقا کوروش درباره منابع هم باید در نظر داشته باشی که خیلی ها نمیتونن از منابع انگلیسی بخوبی استفاده کنن.در ضمن من خودم برای مطالبی که مربوط به زبان جاوا بود به انگلیسی سرچ کردم ولی خب نتایج اصلا قابل مقایسه باجستجو در زبانهای دات نتی نبود.پس فقط استفاده از ویژوال استودیو در ایران رایج نیست.اینکه زبانهای دیگر به جایگاه اصلی خود نرسیدن فقط در ایران صادق نیست.اون هم بخاطر باگ ها و عدم راحتی کار با این نرم افزارها در مقایسه با دات نت هست. در ضمن من نسخه اکسپرس ویژوال استودیو رو با آی دی ای مثل نت بینز مقایسه کردم.باز هم به نظر من ویژوال استودیو سر تر هست.البته من در سطح خیلی حرفه ای از ویژوال استودیو استفاده نکردم ولی خب مشکلاتی در نت بینز وجود داره که هیچوقت در ویژوال استودیو دیده نشده.
الان یک لحظه قیمت ویژوال استودیو در سرچ کردم:نسخه استاندارد 178$-نسخه حرفه ای=600$-نسخه تیم سیستم=2000$-
من که مطمعنا 178$ برای ویژوال استودیو میدادم.در لینک زیر هم میتونین تفاوتهای نسخه های متفاوت رو ببینید:

http://www.hallogram.com/vstudio/compare.html

بااحترام

نویسنده: کوروش
تاریخ: ۱۳۹۰/۰۶/۱۶ ۱۶:۲۳:۴۰

- نسخه express تو نسخه های مختلفش محدودیت های مختلف داره مثلا یه جا تعداد breakpoint ها محدود هست در CPP کامپایلر ۶۴ بیتی وجود ندارد و چیزای دیگه. حالا اگه یه کسی با این چیزا مشکلی نداشت که می تونه ولی به همین راحتی هم نمیشه پیش بینی کرد که به چی در آینده احتیاج پیدا می کنی یا نه. خود مایکروسافت بهتر می دونه که چی کار کنه که به درآمدش لطمه نخوره؛ اگه کامپایلر ها و برنامه های متن باز توسعه تحت ویندوز نبودن از express رایگان هم خبری نبود.
- ناول خودش توسعه دهنده مونو بود معلومه ازش پشتیبانی می کرد منظور من شرکت های دیگه بود. من هنوز برنامه بزرگ یا تجاری ندیدم که با مونو پیاده سازی بشه جز یه چند تا برنامه پخش موسیقی یا برداشتن نوت و دیدن عکس و از این قبیل. اگر شما جایی چیز دیگه ای دیدید لطفا آدرسش رو بذارین.
- شکی نیست که شما دارید زحمت می کشید و منم منکر این نشدم. فقط نظرم رو برای کامل کردن مطلب گفتم.

یه نکته جالب که گفتنش خالی از لطف نیست. سایت جاوا دسترسی رو برای کاربران ایرانی میبنده و همین طور سایت های openoffice و netbeans و sourceforge و google code و... ولی سایت مایکروسافت و محصولاتش همیشه برای ایران باز و قابل دسترس بوده.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۱۶ ۱۶:۴۵:۰۹

- من نمی‌دونم محدود بودن تعداد breakpoints چه اهمیتی داره، یا اینکه چندتا مثلا شما رو راضی می‌کنه. یا اینکه مثلا کامپایلر CPP چه ربطی به بحث دات نت داره. خوشبختانه پروژه سورس باز CodeBlocks موجود است و کامپایلر GCC لینوکس رو در ویندوز به صورت یکپارچه به شما ارائه می‌ده (http://www.codeblocks.org/). بحث ما اینجا در مورد دات نت است در مورد کدهای مدیریت شده آن.
- من در مورد مایکروسافت عرض کردم نه در مورد ناول. در مورد قراردادهای مایکروسافت با ناول و پشتیبانی مایکروسافت از ناول. در مورد استفاده کنندگان از مونو مراجعه کنید به کامنت‌های لینکی که در مورد شرکت زاماریان داده شد. لینکش در متن فوق هست.
- به این اصطلاحا می‌گن توهم توطئه ... (+)
و صد البته اگر در رابطه با کاهش هزینه‌های توسعه با دات نت باشد بسیار نیکو و پسندیده است و در راستای بحث جاری. اون شرکت معروف سان هم اگر کارش درست بود الان سرپا بود. هنوز نتونسته جاوای 7 رو جمع کنه. سیستم JIT اون باگ داره.
کسی جرات نمی‌کنه ازش در محیط توسعه استفاده کنه : (+)

من فقط خواستم ثابت کنم با حداقل هزینه و با یک مجوز اولیه ویندوز راحت می‌شود با دات نت برنامه توسعه داد. هزینه زیادی هم ندارد. لطفا بحث رو به حاشیه نکشید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۱۷ ۰۰:۴۱:۰۱

یک IDE خوب دیگر را هم در متن فوق فراموش کردم که بنویسم و اون هم mono develop است. امروز نگارش جدیدی از اون منتشر شده:

MonoDevelop 2.6 is out

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۱۷ ۱۶:۲۳:۲۰

کسی که نسخه‌ی ultimate ویژوال استودیو رو نصب کرده اما از ابزارهای UML Modeling projects ، unit testing ، بررسی معماری پروژه جاری و امثال آن استفاده نمی‌کند، همانند کسی است که Photoshop را نصب کرده اما دارد با آن عکس Resize می‌کند! :)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۲۷ ۱۹:۴۰:۰۵

علاوه بر مطالبی که دوستان در مورد اشتراک MSDN و فواید آن گفتند ... برنامه‌ی دیگری به نام bizspark از طرف مایکروسافت برای شرکت‌های تازه تاسیس و کمک به آن‌ها وجود دارد. به این ترتیب به مدت سه سال مشترک MSDN خواهید شد و تنها 100 دلار در پایان سه سال باید پرداخت کنید. در طی این مدت دسترسی قانونی به تمام محصولات مهم مایکروسافت را دارید (VS2010, Office2010, MS Axapta, SQL SERVER, SERVER 2008, Windows 7-8 و ...)

http://www.microsoft.com/bizspark/

نویسنده: afsharm
تاریخ: ۱۳۹۰/۰۶/۲۸ ۰۰:۱۸:۵۹

در رابطه با BizSpark ببینید:

http://blog.afsharm.com/2011/09/msdn-bizspark.html

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۲۸ ۱۲:۲۲:۵۴

راستی، ویندوز 8 دارای دات نت نگارش «4 و نیم» سرخود است. این هم رایگان است برای خریداران ویندوز و همچنین توسعه دهنده‌ها هم نگرانی از توزیع آن نخواهند داشت و یک قدم مثبت است در جهت ساده‌تر کردن کارها.

نویسنده: Farhad Yazdan-Panah
تاریخ: ۱۳۹۰/۰۶/۲۸ ۱۹:۰۰:۰۲

البته جسارتا گاهی مشاهده شده که این ادواتیکه روی نسخه ی Ultimate و ..و نصبه نسبت به برنامه های موجود (کا کارشون اینه) خیلی کم داره.
مثلا آنچه در VS به نام UML Diagram ... موجوده در مقایسه با EA یا حتی برنامه های Free خیلی ساده و پیش پا افتادست.
یا خیلی پیچیدست (مثل TFS در مقابل SVN).

نویسنده: Farhad Yazdan-Panah
تاریخ: ۱۳۹۰/۰۶/۲۸ ۱۹:۰۲:۱۶

بله گویا خیلی هم با XAML هماهنگه.

عنوان: **تبدیل عدد به حروف**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۲۸ ۱۵:۰۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

به طور قطع توابع و کلاس‌های تبدیل عدد به حروف، در جعبه ابزار توابع کمکی شما هم پیدا می‌شوند. روز قبل سعی کردم جهت آزمایش، عدد 3000,000,000,000,000 ریال را با کلاسی که دارم تست کنم و نتیجه overflow یا اصطلاحا ترکیدن سیستم بود! البته اگر مطالب این سایت را دنبال کرده باشید پیشتر در همین راستا مطلبی در مورد نحوه‌ی صحیح بکارگیری توابع تجمعی SQL در این سایت منتشر شده است و جزو الزامات هر سیستمی است (تفاوتی هم نمی‌کند که به چه زبانی تهیه شده باشد). اگر آن‌را رعایت نکرده‌اید، سیستم شما «روزی» دچار overflow خواهد شد.

در کل این کلاس تبدیل عدد به حروف را به صورت ذیل اصلاح کردم و همچنین دو زبانه است؛ چیزی که کمتر در پیاده سازی‌های عمومی به آن توجه شده است:

```
using System.Collections.Generic;
using System.Linq;

namespace NumberToWordsLib
{
 /// <summary>
 /// Number to word languages
 /// </summary>
 public enum Language
 {
  /// <summary>
  /// English Language
  /// </summary>
  English,

  /// <summary>
  /// Persian Language
  /// </summary>
  Persian
 }

 /// <summary>
 /// Digit's groups
 /// </summary>
 public enum DigitGroup
 {
  /// <summary>
  /// Ones group
  /// </summary>
  Ones,

  /// <summary>
  /// Teens group
  /// </summary>
  Teens,

  /// <summary>
  /// Tens group
  /// </summary>
  Tens,

  /// <summary>
  /// Hundreds group
  /// </summary>
  Hundreds,

  /// <summary>
  /// Thousands group
  /// </summary>
  Thousands
 }

 /// <summary>
```

```
/// Equivalent names of a group
/// </summary>
public class NumberWord
{
 /// <summary>
 /// Digit's group
 /// </summary>
 public DigitGroup Group { set; get; }

 /// <summary>
 /// Number to word language
 /// </summary>
 public Language Language { set; get; }

 /// <summary>
 /// Equivalent names
 /// </summary>
 public IList<string> Names { set; get; }
}

/// <summary>
/// Convert a number into words
/// </summary>
public static class HumanReadableInteger
{
 #region Fields (4)

 private static readonly IDictionary<Language, string> And = new Dictionary<Language, string>
 {
  { Language.English, " " },
  { Language.Persian, " و " }
 };
 private static readonly IList<NumberWord> NumberWords = new List<NumberWord>
 {
  new NumberWord { Group= DigitGroup.Ones, Language= Language.English, Names=
   new List<string> { string.Empty, "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight",
"Nine" }},
  new NumberWord { Group= DigitGroup.Ones, Language= Language.Persian, Names=
   new List<string> { string.Empty, "نه", "هشت", "هفت", "شش", "پنج", "چهار", "سه", "دو", "یک"
}},

  new NumberWord { Group= DigitGroup.Teens, Language= Language.English, Names=
   new List<string> { "Ten", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen",
"Seventeen", "Eighteen", "Nineteen" }},
  new NumberWord { Group= DigitGroup.Teens, Language= Language.Persian, Names=
   new List<string> { "هجده", "هفده", "شانزده", "پانزده", "چهارده", "سیزده", "دوازده", "یازده", "ده
نوزده" }},

  new NumberWord { Group= DigitGroup.Tens, Language= Language.English, Names=
   new List<string> { "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninety" }},
  new NumberWord { Group= DigitGroup.Tens, Language= Language.Persian, Names=
   new List<string> { "نود", "هشتاد", "هفتاد", "شصت", "پنجاه", "چهل", "سی", "بیست" }},

  new NumberWord { Group= DigitGroup.Hundreds, Language= Language.English, Names=
   new List<string> {string.Empty, "One Hundred", "Two Hundred", "Three Hundred", "Four Hundred",
    "Five Hundred", "Six Hundred", "Seven Hundred", "Eight Hundred", "Nine Hundred" }},
  new NumberWord { Group= DigitGroup.Hundreds, Language= Language.Persian, Names=
   new List<string> {string.Empty, "هفتصد", "ششصد", "پانصد", "چهارصد", "سیصد", "دویست", "یکصد
نهصد" , "هشتصد" }},

  new NumberWord { Group= DigitGroup.Thousands, Language= Language.English, Names=
    new List<string> { string.Empty, " Thousand", " Million", " Billion"," Trillion", " Quadrillion",
" Quintillion", " Sextillian",
   " Septillion", " Octillion", " Nonillion", " Decillion", " Undecillion", " Duodecillion", "
Tredecillion",
   " Quattuordecillion", " Quindecillion", " Sexdecillion", " Septendecillion", " Octodecillion", "
Novemdecillion",
   " Vigintillion", " Unvigintillion", " Duovigintillion", " 10^72", " 10^75", " 10^78", " 10^81", "
10^84", " 10^87",
   " Vigintinonillion", " 10^93", " 10^96", " Duotrigintillion", " Trestrigintillion" }},
  new NumberWord { Group= DigitGroup.Thousands, Language= Language.Persian, Names=
    new List<string> { string.Empty, " تریلیون", " میلیارد", " میلیون", " هزار", " Quadrillion", "
Quintillion", " Sextillian",
   " Septillion", " Octillion", " Nonillion", " Decillion", " Undecillion", " Duodecillion", "
Tredecillion",
   " Quattuordecillion", " Quindecillion", " Sexdecillion", " Septendecillion", " Octodecillion", "
Novemdecillion",
   " Vigintillion", " Unvigintillion", " Duovigintillion", " 10^72", " 10^75", " 10^78", " 10^81", "
10^84", " 10^87",
   " Vigintinonillion", " 10^93", " 10^96", " Duotrigintillion", " Trestrigintillion" }},
 };
```

```
private static readonly IDictionary<Language, string> Negative = new Dictionary<Language, string>
{
 { Language.English, "Negative " },
 { Language.Persian, "منهای " }
};
private static readonly IDictionary<Language, string> Zero = new Dictionary<Language, string>
{
 { Language.English, "Zero" },
 { Language.Persian, "صفر" }
};

#endregion Fields

#region Methods (7)

// Public Methods (5)

/// <summary>
/// display a numeric value using the equivalent text
/// </summary>
/// <param name="number">input number</param>
/// <param name="language">local language</param>
/// <returns>the equivalent text</returns>
public static string NumberToText(this int number, Language language)
{
 return NumberToText((long)number, language);
}


/// <summary>
/// display a numeric value using the equivalent text
/// </summary>
/// <param name="number">input number</param>
/// <param name="language">local language</param>
/// <returns>the equivalent text</returns>
public static string NumberToText(this uint number, Language language)
{
 return NumberToText((long)number, language);
}

/// <summary>
/// display a numeric value using the equivalent text
/// </summary>
/// <param name="number">input number</param>
/// <param name="language">local language</param>
/// <returns>the equivalent text</returns>
public static string NumberToText(this byte number, Language language)
{
 return NumberToText((long)number, language);
}

/// <summary>
/// display a numeric value using the equivalent text
/// </summary>
/// <param name="number">input number</param>
/// <param name="language">local language</param>
/// <returns>the equivalent text</returns>
public static string NumberToText(this decimal number, Language language)
{
 return NumberToText((long)number, language);
}

/// <summary>
/// display a numeric value using the equivalent text
/// </summary>
/// <param name="number">input number</param>
/// <param name="language">local language</param>
/// <returns>the equivalent text</returns>
public static string NumberToText(this double number, Language language)
{
 return NumberToText((long)number, language);
}

/// <summary>
/// display a numeric value using the equivalent text
/// </summary>
/// <param name="number">input number</param>
/// <param name="language">local language</param>
/// <returns>the equivalent text</returns>
public static string NumberToText(this long number, Language language)
{
```

```
    if (number == 0)
    {
     return Zero[language];
    }

    if (number < 0)
    {
     return Negative[language] + NumberToText(-number, language);
    }

    return wordify(number, language, string.Empty, 0);
   }
   // Private Methods (2)

   private static string getName(int idx, Language language, DigitGroup group)
   {
    return NumberWords.Where(x => x.Group == group && x.Language == language).First().Names[idx];
   }

   private static string wordify(long number, Language language, string leftDigitsText, int thousands)
   {
    if (number == 0)
    {
     return leftDigitsText;
    }

    var wordValue = leftDigitsText;
    if (wordValue.Length > 0)
    {
     wordValue += And[language];
    }

    if (number < 10)
    {
     wordValue += getName((int)number, language, DigitGroup.Ones);
    }
    else if (number < 20)
    {
     wordValue += getName((int)(number - 10), language, DigitGroup.Teens);
    }
    else if (number < 100)
    {
     wordValue += wordify(number % 10, language, getName((int)(number / 10 - 2), language,
DigitGroup.Tens), 0);
    }
    else if (number < 1000)
    {
     wordValue += wordify(number % 100, language, getName((int)(number / 100), language,
DigitGroup.Hundreds), 0);
    }
    else
    {
     wordValue += wordify(number % 1000, language, wordify(number / 1000, language, string.Empty,
thousands + 1), 0);
    }

    if (number % 1000 == 0) return wordValue;
    return wordValue + getName(thousands, language, DigitGroup.Thousands);
   }

   #endregion Methods
  }
}
```

دریافت پروژه کامل به همراه Unit tests مرتبط

نظرات خوانندگان

نویسنده: Farhad Yazdan-Panah
تاریخ: ۱۳۹۰/۰۶/۲۸ ۱۸:۵۲:۲۸

بسیار طراحی جالبی داشت.
البته به نظر من اگر اون ثوابت رو (معادل ها) از یک فایل (XML یا ...) بخونه تا حدودی بهتر میشه (برای زبان جدید).
البته جسارت نشه.فقط کاش برای اعداد ممیز شناور (Floating point) هم یه فکری می کردید.چون گاهی لازم میشه اعداد (امتیاز و ...) رو به حروف نوشت.

نویسنده: Farhad Yazdan-Panah
تاریخ: ۱۳۹۰/۰۶/۲۸ ۱۹:۵۳:۳۶

گویا موتور گزارشگیری Stimulsoft در نسخه 2011.2 یک تابع مشابه به موضوع رو به کنابانه خودش اضافه کرده است.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۲۸ ۲۳:۱۲:۱۲

چند بحث کلی اینجا هست:
- چون عموما از عدد به حروف در گزارشات مالی استفاده می‌شود، اعداد همه int و big int هستند. بنابراین آنچنان کاربرد دنیای واقعی ندارد سایر حالت‌ها.
- بحث اضافه کردن سایر زبان‌ها ... خوب، بستگی به تسلط به زبان‌های مختلف هم دارد. مثلا در انگلیسی می‌گویند Three hundred اما در فارسی مرسوم نیست که کسی بگه «سه صد». به همین جهت یک قسمت اضافه‌تر برای معرفی سیصد و امثال آن در کد فوق وجود دارد. به احتمال زیاد زبان‌های دیگر هم ریزه‌کاری‌های خاص خودشان را دارند.
- بحث سرعت را هم در نظر بگیرید. در این نوع الگوریتم‌ها به علت استفاده مکرر، ترجیح داده می‌شود که از کالکشن‌های تشکیل شده در حافظه (بجای خواندن از فایل) جهت سرعت بالاتر دسترسی به اطلاعات و سربار کمتر استفاده شود.

نویسنده: Farhad Yazdan-Panah
تاریخ: ۱۳۹۰/۰۶/۲۹ ۰۰:۳۳:۱۱

ممنون از توضیحاتتون.
- در مورد کاربرد بیشتر تبدیل اعداد صحیح موافقم، ولی در سیستم های آکادمیک (حیطه کاری من)، همانند: سیستم های پژوهشی، آموزشی، و ... معمولا حالت ممیز شناور نیز دارای کاربردهای زیادیه. (امتیاز، گرنت، معدل، و ...)- در مورد سایر زبان ها حق با شماست و هر زبونی برای خودش یه داستانی داره!!
- منظور من از خواندن از فایل بیشتر برای مقدار دهی اولیه بود (استفاده از الگوی singleton) و جدا کردن دادگان از الگوریتم بود. چون بعیده که کسی بخواد مقادیر رو در زمان اجرا عوض کنه. مسلما سرعت در درجه بالایی از اهمیته و مقیم بودن همیشگی در حافظه یک راه حل مناسبه.

عنوان: **بیشترین کاربرد دات نت فریم ورک تابحال در کجا بوده است؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۷/۰۱ ۱۱:۲۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

برخلاف تصور عموم، کاربرد اصلی دات نت فریم ورک در طی این چندین و چند سالی که از ارائه آن می‌گذرد، در توسعه‌ی گسترده برنامه‌های دسکتاپ نبوده است. عمده کاربرد آن در تهیه برنامه‌های وب است. برای نمونه می‌توان به آمارگیری زیر سیستم‌های مورد استفاده دات نت در بین برنامه نویس‌ها در سال 2010 مراجعه کرد [^] و کاربردهای وب آن را حداقل باید در جمع استفاده از Ajax ، WebForms و MVC جستجو کرد (البته اگر WCF و ASMX را ندید بگیریم که آن‌ها هم عمده کاربردشان در پروژه‌های وب است). این اعداد و ارقام سال 2010 را اگر بخواهیم از بیشترین به کمترین لیست کنیم، حاصل آن به صورت زیر درخواهد آمد:

```
01 - WebForms
02 - Ajax
03 - WCF
04 - Linq to SQL
05 - MVC
06 - WinForms
07 - ASMX
08 - Silverlight
09 - WPF
10 - ADO DataSets
11 - Entity-Framework (EF)
12 - Workflow
13 - ADO.NET Data Services
14 - DynamicData
15 - CardSpace
```

مورد دیگری که شاید برای خیلی‌ها جالب توجه باشد، آمار تعداد سایت‌هایی است که از ASP.NET استفاده می‌کنند، در مقابل تعداد سایت‌هایی که بر پایه PHP تهیه شده‌اند. مطابق آمار این سایت [^] و [^] در حال حاضر در بین یک میلیون سایت برتر دنیا (سایت‌هایی که بیشترین ترافیک وب را به خود اختصاص داده‌اند) حدود 216 هزار سایت از ASP.NET و 394 هزار سایت از PHP استفاده می‌کنند. از مابقی وب سایت‌های موجود در وب، حدود 27 میلیون سایت از ASP.NET و 26 میلیون سایت از PHP استفاده می‌کنند. این اعداد و ارقام از این جهت حائز اهمیت هستند که مدت زمان ارائه ASP.NET کمتر از PHP است و همچنین بیشترین کاربرد ASP.NET در سرورهای ویندوزی است، برخلاف PHP که علاوه بر ویندوز، در بین سرورهای لینوکسی نیز گزینه‌ی بسیار محبوبی محسوب می‌شود.

عنوان: **محاسبه مجدد میزان مصرف حافظه‌ی برنامه‌های دات نت**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۸/۰۷ ۲۲:۵۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips

اگر به میزان مصرف حافظه اولیه‌ی برنامه‌های دات نت دقت کنیم، نسبت به مثلا یک برنامه‌ی MFC چند برابر به نظر می‌رسند و ... این علت دارد:

زمانیکه یک برنامه‌ی مبتنی بر دات نت اجرا می‌شود، ابتدا JIT compiler شروع به کار کرده و شروع به کامپایل برنامه می‌کند. این بارگزاری هم در همان پروسه‌ی اصلی برنامه انجام می‌شود. به همین جهت میزان مصرف حافظه‌ی برنامه‌های دات نت عموما بالا به نظر می‌رسد.

اکنون سؤال اینجا است که آیا می توان این حافظه‌ای را که دیگر مورد استفاده نیست (و توسط JIT compiler اخذ شده) به سیستم بازگرداند و محاسبه‌ی مجددی را در این مورد انجام داد. پاسخ به این سؤال را در متد ReEvaluateWorkingSet زیر می‌توان مشاهده کرد:

```
using System;
using System.Diagnostics;

namespace Toolkit
{
    public static class Memory
    {
        public static void ReEvaluateWorkingSet()
        {
            try
            {
                Process loProcess = Process.GetCurrentProcess();
                //it doesn't matter what you set maxWorkingSet to
                //setting it to any value apparently causes the working set to be re-evaluated and
excess discarded
                loProcess.MaxWorkingSet = (IntPtr)((int)loProcess.MaxWorkingSet + 1);
            }
            catch
            {
                //The above code requires Admin privileges.
                //So it's important to trap exceptions in case you're running without admin rights.
            }
        }
    }
}
```

در این متد ابتدا پروسه جاری دریافت شده و سپس MaxWorkingSet به یک عدد دلخواه تنظیم می‌شود. مهم نیست که این عدد چه چیزی باشد، زیرا این تنظیم سبب می‌شود که در پشت صحنه به شکل حساب شده‌ای حافظه‌ای که مورد استفاده نیست به سیستم بازگردانده شود و سپس عددی که در task manager نمایش داده می‌شود، مجددا محاسبه گردد. همچنین باید دقت داشت که این کد تنها با دسترسی مدیریتی قابل اجرا است و به همین دلیل وجود این try/catch ضروری است.

نحوه استفاده از متد ReEvaluateWorkingSet در برنامه‌های WinForms :

فایل Program.cs را یافته و سپس در روال رویداد گردان Idle برنامه، متد ReEvaluateWorkingSet را فراخوانی کنید (مثلا هر زمان که برنامه minimized شد اجرا می‌شود):

```
//Program.cs
namespace MemUsage
{
    static class Program
    {
        /// <summary>
```

```
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            //...

            Application.Idle += applicationIdle;
        }

        static void applicationIdle(object sender, EventArgs e)
        {
            Memory.ReEvaluateWorkingSet();
        }
    }
}
```

نحوه استفاده از متد ReEvaluateWorkingSet در برنامه‌های WPF :

فایل App.xaml.cs را یافته و سپس در روال رویدادگردان Deactivated برنامه، متد ReEvaluateWorkingSet را فراخوانی کنید:

```
//App.xaml.cs

public App()
{
    this.Deactivated += appDeactivated;
}

void appDeactivated(object sender, EventArgs e)
{
     Memory.ReEvaluateWorkingSet();
}
```

تاثیر آن هم قابل ملاحظه است (حداقل از لحاظ روانی!). تست کنید!

نظرات خوانندگان

نویسنده: Ameer Taghavi
تاریخ: ۱۳۹۰/۰۸/۰۸ ۱۲:۱۱:۳۸

تست کردم از 7524K شد 1604K
واقعا تاثیر داره!

نویسنده: Mohsen
تاریخ: ۱۳۹۰/۰۸/۰۸ ۱۴:۵۲:۰۱

مهندس در ویرایش 2 دات نت از چه متدی می توان برای اینکار استفاده کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۸/۰۸ ۱۷:۱۴:۵۳

با دات نت 2 هم کار می‌کنه. مطابق مستندات MSDN کلاس پروسس از زمان دات نت یک اضافه شده:
Process Class Supported in: 4, 3.5, 3.0, 2.0, 1.1, 1.0
زمان دات نت 2 با توجه به اینکه WPF نبوده بنابراین بحث WinForms مطرح است و رویداد Idle هم از زمان دات نت یک وجود داشته:
Idle EventSupported in: 4, 3.5, 3.0, 2.0, 1.1, 1.0

نویسنده: مهمان
تاریخ: ۱۳۹۰/۰۸/۱۱ ۱۲:۲۴:۳۵

برای ASP.NET هم کاربرد داره؟
Silverlight چطور؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۸/۱۱ ۱۲:۳۱:۰۳

خیر. چون هر برنامه‌ای تحت ویندوز حتما باید تحت مجوز یک کاربر اجرا شود. برنامه‌های ASP.NET یا سیلورلایت هم از این مطلب مستثنی نیستند. برای نمونه کاربر پیش فرض ASP.NET یا همان Network service (مثلا)، دسترسی مدیریتی ندارد. بنابراین قادر به اجرای کد فوق نیست.

عنوان: **نگهداری رشته ها (String) در حافظه به صورت Encrypt**
نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۱/۰۳/۳۱ ۴:۹
آدرس: www.dotnettips.info
برچسب‌ها: Security, Tips

در همین سایت در بخش لینک‌های ارسالی ، لینکی توسط آقای امیر هاشم زاده به اشتراک گذاشته شده بود با عنوان "چرا هکرها نوع داده String را دوست دارند" ؛ مقاله ای بود در سایت CodeProject که در آن روش هایی که هکرها توسط آن می‌توانند اطلاعات حساس نرم افزار را که در قالب String در حافظه ذخیره شده اند را بررسی نمایند.
اصل مطلب را می‌توانید اینجا مطالعه کنید.

در دات نت فریم ورک کلاسی با عنوان SecureString وجود دارد که توسط آن می‌توان عبارات رشته ای که دارای اطلاعات حساس می‌باشند را به صورت رمز گذاری شده در حافظه ذخیره نمود.

نمونه ای از استفاده این تابع را در زیر مشاهده میکنید:

```
public class Example
{
    public static void Main()
    {
        .
        SecureString securePwd = new SecureString();
        ConsoleKeyInfo key;

        Console.Write("Enter password: ");
        do {
           key = Console.ReadKey(true);

           // بررسی میشود که کلید فشرده شده جزو حروف الفبا می‌باشد یا کلید دیگری است
           if (((int) key.Key) >= 65 && ((int) key.Key <= 90)) {
              // کاراکتر مربوط به کلید فشرده شده به انتهای متغیر سکوراسترینگ اضافه می‌شود
              securePwd.AppendChar(key.KeyChar);
              Console.Write("*");
           }
        // خروج از حلقه در صورت فشردن کلید اینتر
        } while (key.Key != ConsoleKey.Enter);
        Console.WriteLine();

        try
        {
            MessageBox.Show(securePwd);
        }
        catch (Win32Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

در کدهای بالا رمز عبور از کاربر دریافت شده و در متغیر securepwd که شئی از کلاس SecureString می‌باشد ذخیره می‌شود.پس از آن شئی SecureString عبارت مربوطه را به صورت رمز گذاری شده در حافظه ذخیره میکند.
در این روش ابتدا مقدار کلید فشرده شده در متغیر Key که از نوع ConsoleKeyInfo تعریف شده ذخیره می‌شود. بعد از آن مقدار آن بررسی شده و اگر جزو حروف الفبای انگلیسی بود به انتهای متغیر securepwd افزوده می‌شود. این کار با متد AppendChar انجام می‌شود. این عملیات تا فشرده شدن کلید Enter ادامه پیدا میکند.

نظرات خوانندگان

نویسنده: فرید
تاریخ: ۱۳۹۱/۰۳/۳۱ ۴:۵۳

سلام
مرسی از مطلب خوبتون
ولی یه سوال :

آیا کرکرها و کیلاگرها به سادگی می‌توانند به آنها پی ببرند؟

نویسنده: پرهام
تاریخ: ۱۳۹۱/۰۳/۳۱ ۸:۲۱

به کدوم قسمت پی ببرن، بخش تایپ شاید؟! یعنی قبل از اینکه به securePwd اضافه بشه. ولی وقتی اضافه شد و در حافظه قرا گرفت احتمالا یا نه یا خیلی سخت. بستگی به الگریتم رمزگذاری داره.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۳/۳۱ ۸:۲۵

بحث key logger متفاوت است. این متد سبب نمی‌شود که کلید فشرده شده کاربر از دید یک لاگر مخصوص آن مخفی باقی بماند. این روش فقط اطلاعات رشته‌ای را حین استفاده در برنامه به صورت یک رشته ساده در حافظه قرار نمی‌دهد. بلافاصله پس از استفاده از آن رمزنگاری اطلاعات آن خودکار است.
جهت اطلاع password box در wpf به صورت پیش فرض از همین کلاس استفاده می‌کند.

نویسنده: محمد
تاریخ: ۱۳۹۱/۰۳/۳۱ ۱۰:۵

مرسی و خسته نباشید، مفید بود
مثلا اگه پسورد رو از دیتابیسی واکشی و با این مقدار مقایسه کنیم؛ برا مقایسه این رشته امن تابعی وجود داره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۳/۳۱ ۱۰:۲۶

- شما نباید کلمه عبور رو هش نشده در بانک اطلاعاتی ذخیره کنید. (یعنی واکشی کلمه عبور به صورت clear text کار اشتباهی است)
- برای مقایسه در اینجا بهتر است از یک loop و بررسی کاراکترها به نحوی که در اینجا بحث شده استفاده کنید. بعد هم حافظه رو تخریب کنید. بحث اصلی اینجا است که قرار است ردی در حافظه باقی نماند؛ آن هم به صورت رمزنگاری نشده.

نویسنده: محسن
تاریخ: ۱۳۹۱/۰۳/۳۱ ۲۰:۵۳

البته این رو هم باید اضافه می‌کردید استفاده از این متد در حالت UnSafe(با سوئیچ UnSafe) قابل استفاده است به این دلیل که رشته‌ی رمزنگاری شده در حافظه‌ی مدیریت نشده(خارج از کنترل CLR)قرار می‌گیرد.بنابراین دسترسی به این بخش تقریبا غیرممکن است.
موفق باشید.

نویسنده: پوریا عالی نژاد

تاریخ: ۱۳۹۱/۰۴/۰۱ ۹:۵۵

بنده متوجه نشدم. وقتی هکر یا کرکر می‌تواند با Marshal.PtrToStringBSTR و چند تابع کمکی دیگر محتویات درون SecureString را کشف کند، آیا هدف ما فقط از رده خارج کردن تعدادی از کرکرهای آماتور و ابزارمحور هستند؟
و یا اصلا چه دلیلی دارد که پسورد را در حافظه نگه داریم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۱ ۱۰:۳۵

مشکلی که با string معمولی وجود دارد این است که به صورت معمولی و غیر رمزنگاری شده در حافظه قرار می‌گیرند و همچنین توسط برنامه هم قابل حذف از حافظه نیستند. اگر محتویات آن‌را تغییر دهید، یک وهله دیگر ایجاد شده و وهله قبلی هنوز در حافظه قرار داد و البته ... هر زمان که GC صلاح دید آن‌را پس از مدتی حذف می‌کند اما نه بلافاصله. اصطلاحا به این نوع اشیاء immutable هم گفته می‌شود.
SecureString اینبار mutable است و می‌توان مقدار آن‌را توسط برنامه واقعا تخریب کرد و منتظر GC نشد و تغییرات در آن، چندین کپی از آن‌را ایجاد نمی‌کند. SecureString توسط کدهای unmanaged تهیه شده و دسترسی به محتوای آن به این سادگی‌ها نیست. برای دسترسی به آن خارج از برنامه، یک شخص باید اشاره‌گر به (IntPtr unmanaged) این رشته رمزنگاری شده را بیابد. Marshal.PtrToStringBSTR فقط یک string pointer را دراختیار شما قرار می‌دهد.

ضمن اینکه امنیت بحثی نسبی است. اگر از خانه خارج می‌شوید بهتر است درب آن‌را قفل کنید. اما این به معنای نفوذناپذیری 100 درصد خانه شما نیست ولی این امر «نسبت به» خانه‌ای که درب آن کاملا باز است، خیلی بهتر است .

نویسنده: ابراهیم
تاریخ: ۱۳۹۱/۰۴/۰۲ ۱۵:۱۴

خیلی ممنون از مطلب خوبتون
سوالی داشتم : چطور میشه از رشته ای که در این نوع داده ذخیره شده استفاده کرد؟ منظورم اینه که چطور میشه در داخل برنامه فهمید که چه داده ای در این متغیر وجود داره

نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۱/۰۴/۰۴ ۲۳:۴۵

همانند یک متغیر از نوع String با آن رفتار می‌شود. برای مثال برای نمایش مقدار موجود در این متغیر می‌توان از دستور زیر استفاده کرد:

```
MessageBox.Show(securePwd);
```

نویسنده: ابراهیم
تاریخ: ۱۳۹۱/۰۴/۰۵ ۱۵:۶

وقتی این دستور رو اجرا میکنم پیغام زیر نمایش داده میشه

```
System.Security.SecureString
```

و محتوای رشته نمایش داده نمیشه

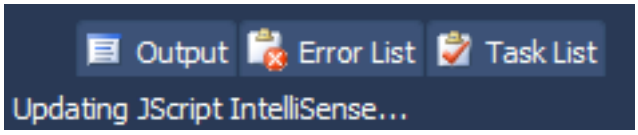
عنوان: **نکته ای برای استفاده راحت تر ECMA Scripts در توسعه تحت شیرپوینت**
نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۱/۰۴/۱۴ ۲۱:۳
آدرس: www.dotnettips.info
برچسب‌ها: SharePoint, Tips

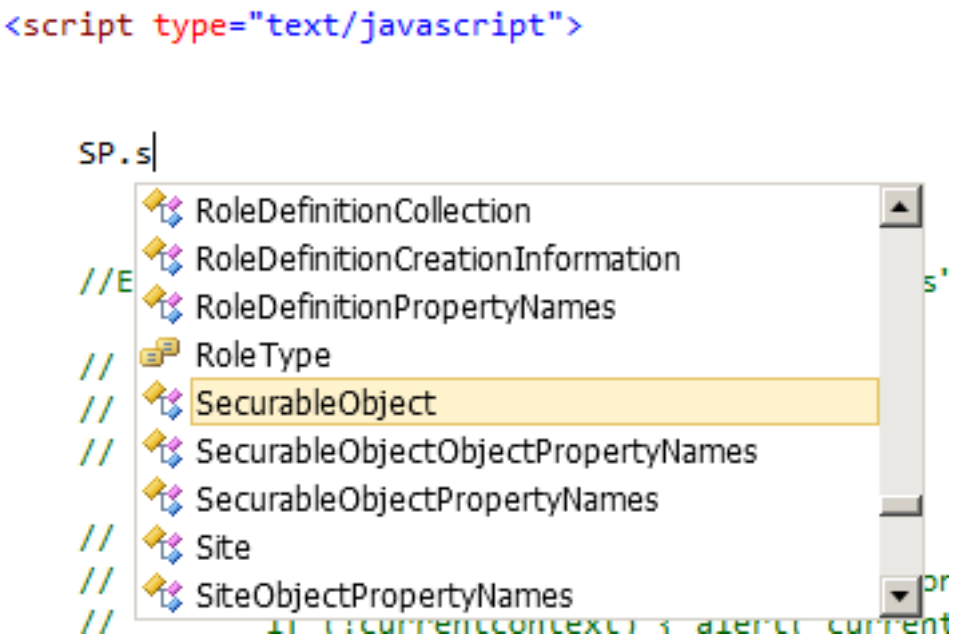
هنگام کدنویسی زبان‌های سمت کاربر (Client Side) اگر به عنوان برنامه نویس و توسعه دهنده ، با کتابخانه‌های آن آشنا باشید و آنها را حفظ باشید خیلی خوب است ولی اگر همین کتابخانه‌ها و توابع به ابعاد شیرپوینت گسترش یابد چه؟ مسلما حفظ کردن تمام آنها کار دشواری است ؛ راه حلی که در اینجا ارائه می‌شود ،بار گذاری این توابع در Intellisense ویژوال استودیو است . برای این کار در صورتی که در حال نوشتن Visual WebPart هستید ، می‌توانیدبه روش زیر عمل کنید :

ابتدا بالای صفحه مورد نظر کد زیر را وارد کنید :

```
<% #if SOME_UNDEFINED_CONSTANT %>
 <script type="text/javascript" src="/_layouts/MicrosoftAjax.js" ></script>
 <script type="text/javascript" src="/_layouts/SP.debug.js"></script>
<% #endif %>
```

در این لحظه ویژوال استودیو آغاز به روز رسانی متادیتای Intellisense خود می‌کند که می‌توانید این پیغام را در زیر نوار وضعیت این برنامه مشاهده کنید .





حال می‌توانید به راحتی از این امکانات لذت ببرید و به توسعه خود سرعت دهید.

نظرات خوانندگان

نویسنده: امیر هاشم زاده
تاریخ: ۱۳۹۱/۰۴/۱۵ ۰:۴۰

این ابزار برای jQuery هم وجود داره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۵ ۸:۲۳

برای jQuery یک سری فایل vsdoc وجود دارد . اگر از nuget برای دریافت jQuery استفاده کنید این فایل را هم به صورت خودکار به پروژه اضافه می‌کند (آخرین نگارش آن‌را).

عنوان: **استفاده از "," و ";" اضافه در C#**
نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۰۹ ۹:۱
آدرس: www.dotnettips.info
برچسب‌ها: C#, Tips, Tricks

در C# می‌تونید در انتهای تعریف آخرین آیتم یک Enum یا هنگام استفاده از سینتکس Object Initializer یا Collection Initializer، یک کامای اضافی قرار بدید.

اون طور که گفته شده ، این رفتار بدین دلیل است که Code Generatorها راحت‌تر بتوانند کد تولید کنند. مطمئناً اگر در یک حلقه‌ی تکرار برای ایجاد آیتم های یک Enum، در انتهای آیتم‌های اون، کاراکتر "," قرار می‌دید، حذف نکردن آخرین کاما از حذف کردن اون کار راحت‌تری است! همچنین Comment کردن آخرین آیتم نیز راحت‌تر صورت می‌پذیرد.

```
public enum MyEnum
{
    Item1 = 1,
    Item2 = 2,
    Item3 = 4,
    // Item4
}

MyViewModel viewModel = new MyViewModel()
{
    Property1 = "Value1",
    Property2 = "Value2",
    Property3 = "Value3",
};
```

و البته، در هنگام فراخوانی یک متد میشه به تعداد دلخواه در انتهای اون، کاراکتر ";" قرار داد.

```
myBusiness business = new myBusiness();
business.DoWork(); ; ; ; ; ; ;
```

نظرات خوانندگان

نویسنده: پژمان
تاریخ: ۱۳۹۱/۰۵/۱۲ ۱۸:۲۲

این رو میشه اسمش رو گذاشت forgiveness in language design

عنوان: **نکاتی در مورد نوشتن یک مطلب خوب و گیرا در یک سایت**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۴ ۱۴:۱۰
آدرس: www.dotnettips.info
گروه‌ها: Tips, Writing

تفاوتی نمی‌کند در چه رشته‌ای یا حرفه‌ای مشغول به کار هستید؛ تفاوتی نمی‌کند در چه زمینه‌ای قصد دارید مطلبی را منتشر کنید. برای تهیه یک مطلب خوب و ماندگار، باید یک سری اصول کلی را در نوشتن رعایت کرد که در ادامه به مرور آن‌ها خواهیم پرداخت.

1) مطلب شما نیاز به مقدمه دارد

نیاز به مقدمه داشتن به معنای نوشتن کلمه «مقدمه» در ابتدای یک متن نیست. به این معنا است که به خواننده بگوئید مشکل کجا بوده یا به چه دلیلی قصد دارید مطلب جاری را منتشر کنید. بنابراین جهت تهیه یک مطلب خوب، یک راست اصل مطلب را شروع نکنید. لازم است چند سطری در مورد علت انتشار آن توضیح دهید.

2) پیش از انتشار مطلب چندبار آن‌را مطالعه کنید

یک مطلب خوب نیاز به جمله بندی مناسب، استفاده از نقطه، ویرگول و امثال آن دارد و بدون استفاده از آن‌ها متن شما هرچقدر هم حرفه‌ای باشد، خواندنش مشکل خواهد بود و جلوه مناسبی نخواهد داشت.

3) سعی کنید در عنوان مطلب خود از کلمات کلیدی استفاده کنید

استفاده از کلمات کلیدی در عنوان مطالب، جستجوی آن‌ها را برای خواننده ساده‌تر کرده و همچنین کمک بزرگی است به موتورهای جستجو در یافتن نتایجی بهتر.

4) تکرار کلمات و جملات یکسان را در متن خود به حداقل برسانید

برای مثال مدام در متن خود جمله «همانطور که ملاحظه می‌کنید» را تکرار نکنید. استفاده از افعال تکراری و جملات تکراری در یک متن باید حداقل باشند. برای نمونه اگر جمله جاری به «می‌شود» ختم خواهد شد، جمله بعدی را به «می‌گردد» ختم کنید. اگر جمله‌ای دارای کلمات «برای مثال» است، جمله بعدی بهتر است به همراه کلمات «برای نمونه» باشد.

5) از جملات طولانی استفاده نکنید

یک جمله باید حداکثر یک سطر یا یک سطر و نصفی طول داشته باشد و گرنه خواننده را به شدت در دنبال کردن آن به زحمت خواهید انداخت. جملات طولانی را به جملاتی کوتاه‌تر خرد کنید.

6) استفاده از علامت تعجب را به حداقل برسانید

اشخاصی که مدام از چندین علامت تعجب پشت سرهم استفاده می‌کنند یا مدام از علامت سؤال به همراه چندین علامت تعجب بهره می‌گیرند، حس مسخره کردن شخص مقابل و همچنین عدم تعادل روانی خود را القاء می‌کنند.

7) در متن خود از تصاویر استفاده کنید

انسان موجودی است بصری. قدرت یادگیری ما از طریق دیدن چند برابر زمانی است که از طریق شنیدن یا خواندن نسبت به فراگیری مطلبی اقدام می‌کنیم.

« ما ...

10 درصد چیزهایی را که می‌خوانیم

20 درصد چیزهایی را که می‌شنویم

30 درصد چیزهایی را که می‌بینیم

50 درصد چیزهایی را که می‌بینیم و می‌شنویم

70 درصد چیزهایی را که در موردشان بحث می‌کنیم

80 درصد چیزهایی را که تجربه می‌کنیم

95 درصد چیزهایی را که به دیگران می‌آموزیم

... یاد می‌گیریم
«

8) اگر در سایتی مطلب می‌نویسید، اهداف کلی آن‌را حفظ کنید
اگر نام سایت شما برنامه نویسی است، خواننده به دنبال شعر، داستان و یا مطالب خیلی شخصی و خصوصی شما نمی‌گردد. سایت‌های زیادی هستند که به این مقوله‌ها می‌پردازند و خیلی زود سطح کاری خود را به این ترتیب به حداقل تنزل خواهید داد.

9) به صفحات داخلی سایت خود لینک دهید
در مطلب تهیه شده سعی کنید به مطالب مشابه داخلی سایت خود لینک دهید. احتمال کپی شدن مطالب شما بسیار زیاد است. به این ترتیب می‌توانید خواننده‌ها را در لابلای متن خود به مرجع اصلی هدایت کنید.

10) دست به اختراع برچسب‌های جدید، پراکنده و بیهوده نزنید
اگر گروه بندی مطالب یک سایت بر اساس برچسب‌ها است و تاکنون برچسب‌های متعددی تعریف شده است، بهتر است از همان‌ها استفاده کنید تا اینکه دست به اختراع زده و یک برچسب کاملا جدید را ثبت کنید. برای مثال اگر مطلب شما در مورد `Entity framework` است و تا کنون 20 مطلب ذیل این گروه ثبت شده، اختراع برچسب جدید `EF Code first` نه تنها کمکی نخواهد کرد، بلکه خواننده‌ای را که به دنبال یافتن مطالب یک گروه خاص است، سر در گم می‌کند. یا اگر قصد دارید یک برچسب کاملا جدید را اضافه کنید، حتما از یک برچسب کلی موجود نیز استفاده کنید تا روابط بین مطالب حفظ شوند.

11) مطالب شما بهتر است یک قسمت نتیجه گیری نیز داشته باشد
بهتر است در پایان یک مطلب، خلاصه بحث، پیشنهادها یا حتی سؤالاتی را مطرح کنید تا بتوانید خواننده را تا حدودی وادار به عکس العمل نمائید.

12) تا حد امکان از منابع سایت خود استفاده کنید
اگر قرار است تصویری در متن قرار گیرد، اگر نیاز است فایلی در سایت مطرح شود، بهتر است این موارد در سایت جاری هاست شوند؛ تا اینکه تصویر یک سایت دیگر را مستقیما در سایت خود لینک کنیم.

13) معرفی منابع
نیازی نیست در یک سایت، همانند مقالات علمی، در انتهای مطلب لیست منابع را ذکر نمود. در اینجا می‌شود قسمتی از جملات را به منبع استفاده شده لینک کرد. به این ترتیب دقیقا مشخص می‌شود هر قسمت و هر جمله‌ای از چه ماخذی استفاده کرده و پیگیری آن ساده‌تر می‌شود.

14) تصاویر ارائه شده را فشرده کنید
فرمت مناسب ارائه تصویر در یک سایت `bmp` نیست. بهترین فرمت برای سایت‌ها `png` است؛ از لحاظ حفظ تعداد رنگ و همچنین کاهش حجم. به علاوه ابزارهای زیادی برای کاهش حجم فایل‌های `png` با حداقل افت کیفیت وجود دارند.

15) در مورد کدهای خود توضیح دهید
این مورد خصوصا به سایت‌های برنامه نویسی مرتبط می‌شود. اینکه چند سطر کد بدون توضیح را در یک مطلب قرار داده‌اید، نه لطفی است و نه اهمیتی دارد! هزاران هزار سطر از این دست کدها در `GitHub`، `CodPlex` و `sourceforge` وجود دارند. فرق کار شما با آن‌ها در چیست؟
باید یک قسمت «توضیحات» ذیل کدهای ارائه شده وجود داشته باشد. همان نکاتی را که حین تهیه کدها در ذهن داشتید باید بتوانید توضیح دهید و گرنه ... کار شما ارزشی نخواهد داشت.

16) مطالب تجربی شما باید قابلیت تولید مجدد داشته باشند
برای مثال امروز در حین کار به یک مطلب جدید برخورده‌اید که قصد دارید آن‌را به اشتراک بگذارید. ذکر این نکته جدید به تنهایی کافی نیست. باید ابتدا بتوانید ذهن خواننده را جهت رسیدن به وضعیتی که قرار داشتید، آماده کنید. اگر قصد دارید قطعه کدی را به اشتراک بگذارید، خواننده باید بتواند این قطعه از پازل را در کنار قطعات دیگری که برای او ترسیم خواهید کرد، جای دهد. یا

حداقل بتواند قطعه کد شما را بدون خطا کامپایل کند.

17) یک راست به سراغ کدنویسی و اصل مطلب نروید

اگر قرار است در مورد یک فناوری جدید در طی چند جلسه بحث کنید، لازم است یک جلسه کامل در مورد «چرا به این فناوری نیاز داریم» توضیح دهید. بنابراین ذکر اینکه بدون مقدمه به سراغ کدنویسی می‌رویم، سؤالات بسیاری را به جا خواهند گذاشت مانند ... «مشکل روش‌های قبلی چی هست؟» «مزیت این روش جدید در کجاست؟» و تا نتوانید این مسایل را شرح دهید، کار شما خریدار نخواهد داشت.

18) در زبان فارسی نیم فاصله‌ها را رعایت کنید

در نگارش زبان فارسی فرق است بین «آمده ام» و «آمده‌ام» و یا «می شود» را باید «می‌شود» نوشت. می‌شود اندکی وقت گذاشت و با مبحث نیم فاصله آشنا شد .

در کل کار کردن در یک محیط گروهی و چند نویسنده‌ای، به مرور زمان تجربه‌هایی را به همراه خواهند داشت که با به اشتراک گذاشتن آن‌ها و یا طرح صریح آن‌ها، می‌توان از تکرار اشتباهات متداول جلوگیری کرد.

نظرات خوانندگان

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۵/۱۴ ۱۷:۲۰

خیلی خیلی ممنون.
بسیار مفید بود.

نویسنده: ایمان
تاریخ: ۱۳۹۱/۰۵/۱۴ ۱۸:۹

سلام، بسیار عالی بود جناب نصیری، امیدوارم هم من و هم بقیه این نکات رو رعایت کنند تا شاهد محتوای بهتری در وب فارسی باشیم

نویسنده: مهدی غیاثی
تاریخ: ۱۳۹۱/۰۵/۱۵ ۵:۳۸

بسیار عالی بود آقای نصیری.
نکات مهمی در اون یاد شده بود که شاید من و بسیاری دیگر از افراد از اون‌ها غافل هستیم.

سپاسگزارم!

نویسنده: فربد صالحی
تاریخ: ۱۳۹۱/۰۵/۱۵ ۹:۱۲

در مورد نکته هشتم...
من قبلا هم این نظر خودم رو عنوان کردم. ولی به دلیل رعایت همین نکته هشتم، شما حتی منتشرش هم نکردید!
کاملا قبول دارم که اینجا یک پایگاه کاملا فنی هستش و امتیاز مثبتش هم همینه. اما قبول کنید برای یک برنامه نویس (یا هر حرفه دیگه ای) علاوه بر نکات فنی، یک سری نکات هم هست که شاید بشه بهش گفت اخلاقیات حرفه، یا ترفندهای حرفه یا ...هر اسم دیگه ای. مثل تجربیاتی که مثلا شما از کار در محیط‌های مختلف به دست آوردید، روش‌های به روز نگه داشتن خود، چگونگی طی کردن مراحل پیشرفت و ..
به نظر من شاید بشه نکات فنی رو از کتاب‌ها دریافت کرد، ولی این نکات که من عنوان کردم حتی اگه نمونه خارج از کشوری هم داشته باشن، شاید کارایی لازم رو در داخل کشور نداشته باشن، یعنی بومی سازی نشدن! و البته این مطالب با مطالب بی فایده ای که بعضی وبلاگ‌ها منتشر می‌کنند و خاطرات و مسایل شخصی خودشون رو در محیط کار مطرح می‌کنند متفاوته.
امیدوارم تونسته باشم منظورم رو بیان کنم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۰:۱۷

قسمت اعظم این سایت تشکیل شده از تجربیات کاری و این ربطی به مورد هشتم نداره.
کسانی که سعی می‌کنند با رندی موضوع بحث را عوض کنند یا اینجا را با انجمن خصوصی یا مشاور خصوصی رایگان پروژه‌های تجاری خودشون اشتباه می‌گیرند، مطالبشان یک ضرب حذف خواهد شد.

نویسنده: ایمان محمدی
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۰:۳۲

در مورد بند دوم:
اگه بتونید قابلیت ذخیره بصورت پیش نویس رو برای این سایت تعریف کنید خیلی خوب میشه ، اینجوری می‌تونیم بند دوم رو بیشتر رعایت کنیم. + نویسنده باید مطلب در ورد یا برنامه ای دیگه ای تنظیم کنه و بعد از آماده شدن دوباره سبک و فرمت نوشته

رو پاک کنه و دوباره داخل ویرایشگر سایت تنظیم کنه.
مهندس جان تجربیات خودتون رو هم در سرعت انتشار مطالب به ما منتقل کنید. خدایش نوشتن یک یا چند مطلب فنی در روز کار هرکسی نیست.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۰:۴۶

نیازی به پیش نویس در سایت نیست. من الان 4 سال هست که دارم مطلبم رو داخل onenote می‌نویسم. 4 بار قبل از انتشار آن‌را در onenote می‌خونم و بعد مرحله انتشار نهایی آن فقط سه دقیقه طول می‌کشه. اول متن رو داخل notepad کپی می‌کنم تا فرمتش از بین بره (کلیه ادیتورهای غیر وبی، انبوهی از تگ‌های نام فرم را به متن html ایی شما اضافه می‌کنند که نمایش آن‌ها عموما در مرورگرها مساله ساز است). بعد متن ساده رو از notepad کپی و به داخل ادیتور سایت paste می‌کنم و سریع فرمت نهایی و انتشار. استفاده از ادیتور سایت، باید مرحله آخر کار شما باشد نه مرحله اول.

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۱/۰۵/۱۶ ۱۴:۳۱

خیلی مفید بود ممنون.

نویسنده: احمد احمدی
تاریخ: ۱۳۹۱/۰۵/۲۵ ۲:۵۲

سلام
ادیتور فعلی سایت امکان ltr کردن متن را ندارد . لطفا بفرمایید شما چگونه متنی که دارای عبارات لاتین هست رو به ادیتور منتقل می‌کنید ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۲۵ ۹:۱

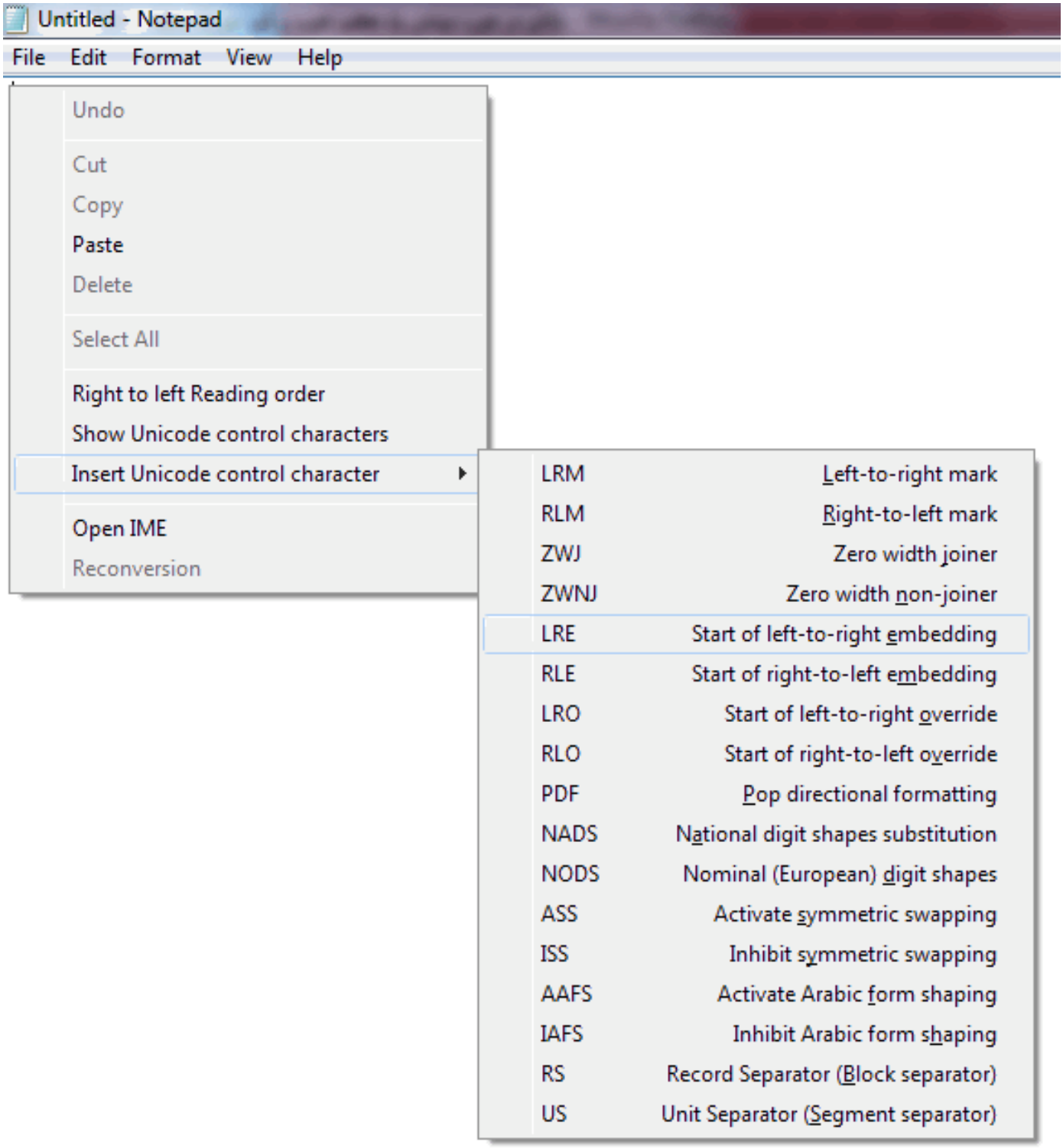
برای اینکار همیشه از امکانات ارسال code در سایت استفاده شده

نویسنده: احمد احمدی
تاریخ: ۱۳۹۱/۰۵/۲۵ ۱۸:۴۴

پس به نظر شما بنده در مقاله‌ی آموزش فایرباگ ، باید برای معرفی توابع فایرباگ ، هریک رو تو یک بلاک کد قرار می‌دادم ؟
و اگر در متن مقاله ، عباراتی شبیه به C# وجود داشت ، باید چکار کرد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۲۵ ۲۰:۲۷

یک سری کاراکتر یونیکد هستند که باید با اون‌ها آشنا بود:



برای مطالعه بیشتر

مشکل بهم ریختگی متون فارسی انگلیسی در کامپیوتر

سندرم شماره نامه به هم ریخته

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۰۶/۰۴ ۰:۵۷

خیلی ممنون مطلب مفیدی بود.

عنوان: **Entity Framework و InnerException**
نویسنده: ایمان محمدی
تاریخ: ۱۳۹۱/۰۵/۲۰ ۱۳:۵۴
آدرس: www.dotnettips.info
برچسب‌ها: Entity framework, Exception, Tips

در Entity Framework بیشتر استثناها تودرتو هستند و ما باید تمام استثناها رو بررسی کنیم تا به پیغام اصلی خطا برسیم. با استفاده از تکه کد زیر به راحتی می‌تونیم استثناها رو پیمایش کنیم و متن خطا را مشخص کنیم.

```
catch (Exception ex)
                {
                    StringBuilder errorMsg = new StringBuilder();
                    for (Exception current = ex; current != null; current = current.InnerException)
                    {
                        if (errorMsg.Length > 0)
                            errorMsg.Append("\n");

                        errorMsg.Append(current.Message.Replace("See the inner exception for details.",
string.Empty));
                    }
                    // log
                    errorMsg.ToString();
                }
```

برای استفاده در قسمت‌های مختلف برنامه یک متد الحاقی مانند زیر تعریف می‌کنیم:

```
public static string ExceptionToString(this Exception ex)
        {
            StringBuilder errorMsg = new StringBuilder();
            for (Exception current = ex; current != null; current = current.InnerException)
            {
                if (errorMsg.Length > 0)
                    errorMsg.Append("\n");
                errorMsg.Append(current.Message.
                    Replace("See the inner exception for details.", string.Empty));
            }
            return errorMsg.ToString();
        }
```

```
catch (Exception ex)
                {
                    // log
                    ex.ExceptionToString();
                }
```

نظرات خوانندگان

نویسنده: f.beigirad
تاریخ: ۱۳۹۲/۰۵/۰۴ ۱۵:۱۳

من که نتونستم از کد بالا استفاده کنم.

لینک رو ببینید : http://barnamenevis.org/showthread.php?410767

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۰۴ ۱۸:۳۹

نتونستید یعنی چکار کردید دقیقا؟ در خطای شما هم InnerException -> UpdateException بوده مثل توضیحات بالا. یعنی اول UpdateException صادر میشه، بعد باید محتویات InnerException اون رو بررسی کرد تا به خطای اصلی رسید مثل کدهای فوق.

عنوان: **نمایش خروجی RSS سایت‌های دیگر به کمک jQuery**
نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۶/۰۴ ۲۱:۵۱
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Tips

شاید خیلی از دوستان (مثل گذشته نه چندان دور خودم) خیلی بیش از اندازه به برنامه نویسی‌های سمت سرور اهمیت می دهند که این کار باعث از دست دادن ، سرعت و سادگی برنامه نویسی سمت کلاینت می‌شود.

معمولا ما برای کار با خروجی‌های XML از کدهای سمت سرور استفاده می‌کنیم ، بدون اینکه از قدرت جی کوئری در این زمینه اطلاعی داشته باشیم. البته در این مقاله خیلی به پردازش XML توسط جی کوئری نمی‌پردازیم و کار اصلی را گوگل برای ما انجام می‌دهد.
برای انجام این کار ما ابتدا توسط یکی از کتابخانه‌های گوگل ، خروجی RSS یک وب سایت رامی خوانیم و بعد به کمک jQuery آن‌ها را نمایش می‌دهیم.

```
<script src="jquery-1.8.0.min.js" type="text/javascript"></script>
    <script type="text/javascript" src="https://www.google.com/jsapi"></script>
    <script type="text/javascript">

        google.load("feeds", "1");

        function initializeda() {
            var feed = new google.feeds.Feed("http://www.drupaleasy.ir/rss.xml");
            feed.setNumEntries(5);
            feed.setResultFormat(google.feeds.Feed.JSON_FORMAT);
            feed.load(function (result) {
                if (!result.error) {
                    for (var i = 0; i < result.feed.entries.length; i++) {
                        var entry = result.feed.entries[i];
                        $('#drupaleasy ul').append('<li><a href="' + entry.link + '">' + entry.title +
'</a></li>');
                    }
                }
            });
        }

google.setOnLoadCallback(initializeda);
    </script>
```

توضیحات کد :
ابتدا نیاز داریم کتابخانه گوگل را به صفحه اضافه کنیم. شما می‌توانید مستندات کامل این کتابخانه در این لینک بخوانید.

```
<script type="text/javascript" src="https://www.google.com/jsapi"></script>
```

در ابتدا معرفی می‌کنیم که به کدام امکان این کتابخانه نیاز داریم ، در این جا ما از ورژن "1" تابع feed استفاده می‌کنیم.بد نیست برای آشنایی بیشتر سری هم به این لینک بزنید.

```
google.load("feeds", "1");
```

ابتدا لینک مورد نظر که باید خروجی از آن گرفته شود را معرفی می‌کنیم و آن را در متغییری قرار می‌دهیم.

```
var feed = new google.feeds.Feed("http://www.drupaleasy.ir/rss.xml");
```

تنظیمات دلخواه خودمان را نظیر تعداد پست واکشی شده و نوع خروجی مشخص می‌کنیم. تجربه (شخصی)نشان داده که بهترین نوع خروجی JSON است.

```
feed.setNumEntries(5);
feed.setResultFormat(google.feeds.Feed.JSON_FORMAT);
```

و بعد هم به کمک jQuery ساختار مورد نظر خودمان را تنظیم می‌کنیم. شما می‌توانید به کمک مستندات ، و سلیقه شخصی خودتان این کار را انجام بدهید ، و در آخر هم تنظیم می‌کنید ، اجرای تابع شما در چه زمانی اتفاق بیافتید.

```
feed.load(function (result) {
                if (!result.error) {
                    for (var i = 0; i < result.feed.entries.length; i++) {
                        var entry = result.feed.entries[i];
                        $('#drupaleasy ul').append('<li><a href="' + entry.link + '">' + entry.title +
'</a></li>');
                    }
                }
            });
```

در این جا من خروجی فید را به صورت یک لیست (ul) تنظیم کردم ، به این صورت که به ازای هر سطر یک li همراه با تگ a تولید کرده ام و به ul مورد نظر append کردم.

فایل HTML :

```
<div id="drupaleasy" class="feeds">
            <span>DrupalEasy.ir</span>
                <ul>
                </ul>
                <a href="http://drupaleasy.ir">more</a>
            </div>
```

همچنین خیلی راحت می‌توانید به کمک CSS استایل‌های دلخواه خودتون رو اعمال کنید.

```
.feeds
        {
            float: right;
            background-color: rgba(234, 242, 243, 0.73);
            margin: 5px;
            border-radius: 20px;
            padding: 8px;
            width: 293px;
            height: 217px;
            border: 1px solid #293883;
        }

        #drupaleasy ul
        {
            list-style-image: url("img/drupal.png");
        }
```

این هم شد خروجی کار من

DrupalEasy.ir
طراحی کاملا استاندارد برای اوبونتو
آموزش طراحی قالب دروپال - قسمت دوم
استفاده از گرادیانت سی اس اس 3 حتی در IE6 !
نگاهی به پایگاه داده دروپال
سی اس اس 3 (CSS3) در اختیار شماست
more

نظرات خوانندگان

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۶/۰۶ ۹:۳۶

جالبه که گوگل امکان cross site ajax request رو می‌ده.

نویسنده: امیرحسین
تاریخ: ۱۳۹۱/۰۶/۰۸ ۱۳:۱۹

بله این امکان رو یاهو هم می‌دهد.
ولی استفاده از گوگل خیلی خوش دست‌تر و راحت‌تر هست.

نویسنده: علیرضا صبوری
تاریخ: ۱۳۹۲/۱۲/۲۶ ۱:۱۵

سلام مطلب مفیدی بود ولی فکر میکنم بیشتر این قابلیت مربوط با کتابخانه گوگل هستش .. نه jquery . چون با جاواسکریپت هم به سادگی همین کار رو میشه کرد. موفق باشید

عنوان: Best Practice هایی برای ASP.NET MVC - قسمت اول
نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۸/۱۲ ۱۸:۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC, Tips

در سایت جاری مطالب زیادی درباره ASP.NET MVC نوشته شده است. این مطلب و قسمت بعدی آن مروری خواهد داشت بر Best Practiceها در ASP.NET MVC.

استفاده از NuGet Package Manager برای مدیریت وابستگی‌ها

درباره اهمیت NuGet برای مصرف کنندگان قبلا این مطلب نوشته شده است.
بجای صرف وقت برای اینکه بررسی کنیم آیا این نسخه‌ی جدید کتابخانه‌ی X یا اسکریپت jQuery آمده است یا خیر، می‌توان این وظیفه را به NuGet سپرد. علاوه بر این NuGet مزیت‌های دیگری هم دارد؛ مثلا تیم‌های برنامه نویسی می‌توانند کتاب خانه‌های مشترک خودشان را در مخزن‌های سفارشی NuGet قرار دهند و توزیع و Versioning آن‌را به NuGet بسپارند.

تکیه بر Abstraction (انتزاع)

Abstraction در طراحی سیستم‌ها منجر به تولید نرم افزار هایی Loosely coupled با قابلیت نگهداری بالا و همچنین فراهم شدن زمینه برای نوشتن Unit Test می‌شود.
اگر به مطالب قبلی وب سایت برگردیم در مطلب چرا ASP.NET MVC گفته شد که :
2) دستیابی به کنترل بیشتر بر روی اجزای فریم ورک :
در طراحی ASP.NET MVC همه‌جا interfaceها قابل مشاهد هستند. همین مساله به معنای افزونه پذیری اکثر قطعات تشکیل دهنده ASP.NET MVC است؛ برخلاف ASP.NET web forms. برای مثال تابحال چندین routing engine، view engine و غیره توسط برنامه نویس‌های مستقل برای ASP.NET MVC طراحی شده‌اند که هیچکدام با ASP.NET web forms میسر نیست. برای مثال از view engine پیش فرض آن خوشتان نمی‌آید؟ عوضش کنید! سیستم اعتبار سنجی توکار آن‌را دوست ندارید؟ آن‌را با یک نمونه بهتر تعویض کنید و الی آخر ...
به علاوه طراحی بر اساس interfaceها یک مزیت دیگر را هم به همراه دارد و آن هم ساده سازی mocking (تقلید) آن‌ها است جهت ساده سازی نوشتن آزمون‌های واحد.

از کلمه‌ی کلیدی New استفاده نکنید

هر جا ممکن است کار وهله سازی اشیاء را به لایه و حتی Framework دیگری بسپارید. هر زمان اشیاء نرم افزار خودتان را با کلمه‌ی new وهله سازی می‌کنید اصل Abstraction را فراموش کرده اید. هر زمان اشیاء نرم افزار را مستقیم وهله سازی می‌کنید در نظر داشته باشید می‌توانید آنها را به صورت وابستگی تزریق کنید.
در همین رابطه مطالب زیر را از دست ندهید :
تزریق وابستگی (dependency injection) به زبان ساده
تزریق وابستگی (Dependency Injection) و توسعه پذیری

از HttpContext مستقیما استفاده نکنید (از HttpContextBase استفاده کنید)

از .NET 4 به بعد فضای نامی تعریف شده که در بر دارنده‌ی کلاس‌های انتزاعی (Abstraction) خیلی از قسمت‌های اصلی ASP.NET است. یکی از مواردی که در توسعه‌ی ASP.NET معمولا زیاد استفاده می‌شود، شیء HttpContext است . استفاده از HttpContextBase را به استفاده از HttpContext ترجیح دهید. اهمیت این موضوع در راستای اهمیت انتزاع (Abstraction) می‌باشد.

از "رشته‌های جادویی" اجتناب کنید

استفاده از رشته‌های جادویی در خیلی از جاها کار را ساده می‌کند؛ بعضی وقت‌ها هم به آنها نیاز است اما مشکلات زیادی دارند :

رشته‌ها معنای باطنی ندارند (مثلا : دشوار است که از روی نام یک ID مشخص کنم این ID چطور به ID دیگری مرتبط است و یا اصلا ربط دارد یا خیر)
با اشتباهات املایی یا عدم رعایت حروف بزرگ و کوچک ایجاد مشکل می‌کنند.
به Refactoring واکنش خوبی نشان نمی‌دهند. (برای درک بهتر این مطلب را بخوانید.)
برای درک بهتر 2، یک مثال بررسی می‌شود؛ اولی از رشته‌های جادویی برای دسترسی به داده در ViewData استفاده می‌کند و دومی refactor شده‌ی مثال اول است که از strongly type مدل برای دسترسی به همان داده استفاده می‌کند.

```
<p>
    <label for="FirstName">First Name:</label>
    <span id="FirstName">@ViewData["FirstName"]</span>
</p>
```

مثال دوم :

```
<p>
    <label for="FirstName">First Name:</label>
    <span id="FirstName">@Model.FirstName</span>
</p>
```

مثلا مثال دوم مشکلات رشته‌های جادویی را ندارد.
در رابطه با Magic strings این مطلب را مطالعه بفرمایید.

از نوشتن HTML در کدهای "Backend" خودداری کنید

با توجه به اصل جداسازی وابستگی‌ها (Separation of Concerns) وظیفه‌ی کنترلرها و دیگر کدهای backend رندر کردن HTML نیست. (ساده سازی کنترلر ها) البته در نظر داشته باشید که قطعا تولید HTML در متدهای کمکی کلاس هایی که "تنها" وظیفه‌ی آنها کمک به Viewها جهت تولید کد هست ایرادی ندارد. این کلاس‌ها بخشی از View در نظر گرفته می‌شوند نه کدهای "backend".

در Viewها "Business logic" انجام ندهید

معکوس بند قبلی هم کاملا صدق می‌کند ، منظور این است که Viewها تا جایی که ممکن است باید حاوی کمترین Business logic ممکن باشند. در واقع تمرکز Viewها باید استفاده و نحوه‌ی نمایش داده ای که برای آن‌ها فراهم شده باشد نه انجام عملیات روی آن.

استفاده از Presentation Model را به استفاده مستقیم از Business Objectها ترجیح دهید

در مطالب مختلف وب سایت اشاره به اهمین ViewModelها شده است. برای اطلاعات بیشتر بند ج آموزش 11 از سری آموزش‌های ASP.NET MVC را مطالعه بفرمایید.

Ifهای شرطی را در Viewها را در متدهای کمکی کپسوله کنید

استفاده از شرطها در View کار توسعه دهنده را برای یک سری اعمال ساده می‌کند اما می‌تواند باعث کمی کثیف کاری هم شود. مثلا:

```
@if(Model.IsAnonymousUser) {
    <img src="@Url.Content("~/content/images/anonymous.jpg")" />
} else if(Model.IsAdministrator) {
    <img src="@Url.Content("~/content/images/administrator.jpg")" />
```

```
} else if(Model.Membership == Membership.Standard) {
    <img src="@Url.Content("~/content/images/member.jpg")" />
} else if(Model.Membership == Membership.Preferred) {
    <img src="@Url.Content("~/content/images/preferred_member.jpg")" />
}
```

می‌توان این کد که تا حدودی شامل منطق تجاری هم هست را در یک متد کمکی کپسوله کرد :

```
public static string UserAvatar(this HtmlHelper<User> helper)
{
    var user = helper.ViewData.Model;
    string avatarFilename = "anonymous.jpg";
    if (user.IsAnonymousUser)
    {
        avatarFilename = "anonymous.jpg";
    }
    else if (user.IsAdministrator)
    {
        avatarFilename = "administrator.jpg";
    }
    else if (user.Membership == Membership.Standard)
    {
        avatarFilename = "member.jpg";
    }
    else if (user.Membership == Membership.Preferred)
    {
        avatarFilename = "preferred_member.jpg";
    }
    var urlHelper = new UrlHelper(helper.ViewContext.RequestContext);
    var contentPath = string.Format("~/content/images/{0}", avatarFilename)
    string imageUrl = urlHelper.Content(contentPath);
    return string.Format("<img src='{0}' />", imageUrl);
}
```

اکنون برای نمایش آواتار کاربر به سادگی می‌توان نوشت :

```
@Html.UserAvatar()
```

به این ترتیب کد ما تمیزتر شده ، قابلیت نگهداری آن بالاتر رفته ، منطق تجاری یک بار و در یک قسمت نوشته شده از این کد در جاهای مختلف سایت می‌توان استفاده کرد و اگر لازم به تغییر باشد با تغییر در یک قسمت همه جا اعمال می‌شود.

منتظر قسمت بعدی باشید.

نظرات خوانندگان

نویسنده: محسن.د
تاریخ: ۱۳۹۱/۰۸/۱۲ ۱۹:۴۵

نکات جالبی بود .
یک نکته که به شخصه اون رو تجربه کرده ام (بخصوص در مورد استفاده از best practiceها) وضعیتی است که اصطلاحاً بهش overkilling میگن (اگر اشتباه نکنم) ، یعنی بعضی وقتا دیگه زیاده روی میشه . مثلا در به کاربردن اینترفیس‌ها و یا loose coupling .
برای مثال در مورد تشخیص زمان استفاده از اینترفیس در این کتاب ، یکی از بهترین راهکارها استفاده از آزمون واحد معرفی شده .
بسیار عالی میشه در صورت امکان در این مورد هم نکاتی را ذکر کنید .

نویسنده: افشار محبی
تاریخ: ۱۳۹۱/۰۸/۲۴ ۸:۲۶

خیلی مفید و عملی بود. ممنون.

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۱/۰۸/۲۴ ۱۴:۱۹

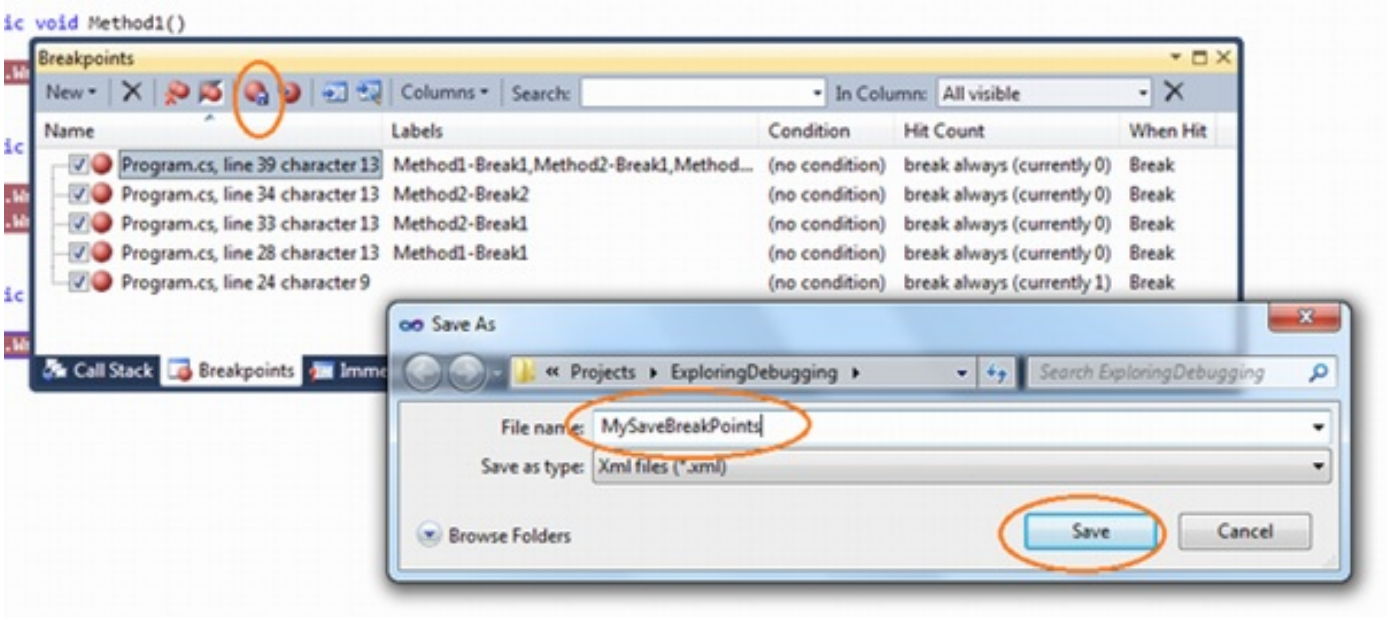
عالی بود ممنون،
بهتر نیست به جای این if و elseها از Dictionary استفاده کنیم؟

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۸/۲۴ ۱۴:۲۰

خواهش می‌کنم.
چرا همینطوره.
کدها همه مثال هستند و هدف این مطلب چیز دیگری بوده.

عنوان: Import و Export کردن Breakpointها در Visual Studio
نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۱/۰۹/۲۲ ۸:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips, Tricks, Visual Studio

ویژوال استدیو Breakpointها را در یک فایل XML ذخیره میکند.برای ذخیره Breakpointها فقط کافی است بر روی دکمه Export در پنجره Breakpoint که در شکل زیر نمایش داده شده است کلیک کنید.



شما می‌توانید فایل XML ذخیره شده را بعدا استفاده کنید و یا می‌توانید آن را به برنامه نویسان دیگر هم بدهید.
اجازه دهید نگاهی داشته باشیم بر محتویات داخل فایل XML . فایل XML کلکسیونی از تگ BreakPoints داخل BreakpointCollection است.هر تگ Breakpoint حاوی اطلاعاتی در مورد یک Breakpoint خاص است.

```
MySaveBreakPoints.xml*
<?xml version="1.0" encoding="utf-8"?>
<BreakpointCollection>
  <Breakpoints>
    <Breakpoint>
      <Version>15</Version>
      <IsEnabled>1</IsEnabled>
      <IsVisible>1</IsVisible>
      <IsEmulated>0</IsEmulated>
      <IsCondition>0</IsCondition>
      <ConditionType>WhenTrue</ConditionType>
      <LocationType>SourceLocation</LocationType>
      <TextPosition>
        <Version>4</Version>
        <FileName>.\ExploringDebugging\Program.cs</FileName>
        <startLine>38</startLine>
        <StartColumn>12</StartColumn>
        <EndLine>38</EndLine>
        <EndColumn>56</EndColumn>
        <MarkerId>0</MarkerId>
        <IsLineBased>0</IsLineBased>
        <IsDocumentPathNotFound>0</IsDocumentPathNotFound>
        <ShouldUpdateTextSpan>1</ShouldUpdateTextSpan>
        <Checksum>...</Checksum>
      </TextPosition>
    </Breakpoint>
    <Breakpoint>...</Breakpoint>
    <Breakpoint>...</Breakpoint>
    <Breakpoint>...</Breakpoint>
    <Breakpoint>...</Breakpoint>
  </Breakpoints>
</BreakpointCollection>
```

اگر شما هر زمانی همه Breakpointها را از کدتان حذف کردید به راحتی می‌توانید آن را تنها با کلیک بر روی Import وارد کدتان بکنید و تمام Breakpointهای ذخیره شده را بازآوری کنید.

نکته: Import کردن Breakpoint براساس شماره خط کد شما می‌باشد یعنی همان خطی که شما Breakpoint را گذاشته اید پس اگر شماره خط کد شما تغییر کند Breakpoint بروی خط قبلی گذاشته میشود.

عنوان: **دریافت خروجی سایت**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۰۲ ۱۴:۱۸
آدرس: www.dotnettips.info
برچسب‌ها: Tips, others

یکی از مشکلاتی که من هر روز در لاگ‌های سایت مشاهده می‌کنم، علاقه وافر بسیاری برای دریافت محتوای کل سایت با استفاده از برنامه‌های مخصوص آن است. البته این افراد هیچ وقت به طور کامل موفق نخواهند شد چون سایت پس از مدتی IP آن‌ها را خواهد بست. اما ... این بازی از روز اول تمامی نداشته ....

Error Log for Root

RSS FEED | RSS DIGEST | DOWNLOAD LOG | HELP | ABOUT

Errors 1 to 15 of total 5,523 (page 1 of 369). Start with 10, 15, 20, 25, 30, 50 or 100 errors per page.

Code	Type	Error	User	Date	Time
0	**Exception**	Banned IP: 188.158.19.217 Details...		12/22/2012	01:55 ب.ظ
0	**Exception**	Banned IP: 188.158.19.217 Details...		12/22/2012	01:54 ب.ظ
0	**Exception**	Banned IP: 188.158.19.217 Details...		12/22/2012	01:42 ب.ظ

بنابراین ضمن اعلام آتش بس (!) اعلام می‌دارد کل محتوای سایت به صورت یک فایل PDF در قسمت ویژه کاربران سایت، قابل دریافت است.
این محتوا به صورت خودکار هر روز صبح ساعت 5، بر اساس آخرین تغییرات سایت، به روز خواهد شد.

دریافت خروجی کامل سایت

فایل‌های به اشتراک گذاشته شده در این قسمت تنها به مدت هفت روز با لینک مستقیم در دسترس خواهند بود.
همچنین هر بامداد آخرین نسخه خروجی کامل سایت در این مکان قرار خواهد گرفت.

dot-net-tips-full-1391-10-02.pdf (65.54 MB)

نظرات خوانندگان

نویسنده: علی صداقت
تاریخ: ۱۳۹۱/۱۰/۰۲ ۱۶:۳۳

تلاش بی منت شما با این کار تکمیل شد. درود بر شما

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۱۰/۰۲ ۱۸:۴۴

آقای نصیری تشکر می‌کنم بابت زحمات شما.
ولی یه تجربه شخصی می‌گه اون هایی که به دنبال به دست اوردن تمام اطلاعات هستند ، هیچ وقت از اون اطلاعات استفاده نمی‌کنند!
هممون تعداد زیادی پی دی اف آموزشی توی کامپیوتر داریم، ولی واقعا تا حالا چندتاش رو خوندیم!؟
بهتره واقع بین باشیم

نویسنده: محسن.د
تاریخ: ۱۳۹۱/۱۰/۰۲ ۲۰:۱

در حالت کلی حرف شما درسته و اونرو تأیید می‌کنم .

اما یک مسئله ای که درباره این وب سایت بخصوص وجود داره سری‌های آموزشی بسیار عالی اون هستش مثل Entity Framework و یا MVC که واقعا خلاصه و کاربردی و به زبان فارسی هستند . و داشتنشون به صورت آفلاین باعث میشه خیلی سریعتر به نکاتی که در اونها ذکر شده دسترسی داشته باشیم . به شخصه خیلی از داشتن مطالب سایت چه به شکل قدیمی CHM چه الان که pdf هستش استفاده کردم و همیشه اونارو با خودم دارم .

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۱۰/۰۲ ۲۰:۵

با شما در مورد مفید بودن این سری مطلب‌ها موافقم

نویسنده: سعید شیرزادیان
تاریخ: ۱۳۹۱/۱۰/۰۲ ۲۲:۲۶

سلام؛ از زحماتی که برای همه در طول این مدت کشیده اید بسیار متشکرم.

نویسنده: میثم هوشمند
تاریخ: ۱۳۹۱/۱۰/۰۳ ۰:۵

به نظرم فونت مطالب بی نازنین باشید بهتر هست!و اینکه خروجی به صورت دسته بندی شده در فایل قرار بگیرد مثل همان فایل سی اچ ام!

نویسنده: علیرضا نوری
تاریخ: ۱۳۹۱/۱۰/۰۳ ۱:۵۳

به نظرم بهتر بود اگر می‌شد، یک پیش نمایش از این فایل می‌گذاشتید.
در ضمن، (با توجه به اینکه این نسخه به صورت نرم افزاری تولید می‌شه) اگر امکانش هست، یک نسخه با پس زمینه تاریک و متن روشن هم قرار بدید تا خوندن اون در مانیتور راحت‌تر باشه.

ممنون.

نویسنده: علیرضا صالحی
تاریخ: ۱۳۹۱/۱۰/۰۳ ۸:۴۴

واقعا جای تقدیر داره، نسخه CHM هم واقعا مفید بود ،بعضا مجموعه‌های اموزشی این بلاگ رو به دانشجوها معرفی می‌کنم. اگر امکان تولید این فایل PDF بر اساس برچسب‌ها بود حجم کمتری پیدا میکرد و دانلود اون برای بعضی راحت‌تر میشد.

ممنون از زحمات بی منت شما در این بلاگ

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۰۳ ۸:۵۵

- پیش نمایش آن به این صورت است: نسخه چاپی ذیل هر مطلب را تهیه کنید. همانند آن در فایل نهایی قرار گرفته.
- رنگ زمینه صفحات فایل‌های PDF مرسوم نیست مشکی باشد. من ندیدم تا بحال. فایل تولیدی برای چاپ بهینه سازی شده است. قلم آن هم از نوعی انتخاب شده تا در اندازه کوچک خوانا باشد.

نویسنده: ناصر طاهری
تاریخ: ۱۳۹۱/۱۰/۰۳ ۱۲:۱۳

تشکر بابت تمامی زحمت هایی که میکشید.
این اطلاعات اگه داخل یک فایل txt هم باشه، باز هم بسیار ارزشمنده. چه برسه به CHM و PDF.

نویسنده: بهمن خلفی
تاریخ: ۱۳۹۱/۱۰/۰۳ ۱۴:۴۰

سلام بابت این کار خیلی از شما سپاسگذارم.

بنده فکر کنم همان 1 درصد مرورگر safari باشم که سایت شما را با ipad مرور میکنم

از طرفی بدلیل اینکه مدیریت فایل PDF در دستگاههای مانند ipad خیلی خسته کننده است به شخصه از CHM بیشتر استفاده میکنم و کاربر پسندتر است

در هر صورت چه فایل CHM و چه فایل PDF باشد خیلی از شما متشکریم

به گفته دوستان داشتن حتی فایل txt برخی مطالب این سایت بسیار ارزشمندتر است

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۰۳ ۱۴:۴۷

امکان تولید فایل‌های CHM را در سرور IIS ندارم. CHM Compiler مایکروسافت یک برنامه native است که نیاز به دسترسی بالایی برای اجرا دارد.

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۱۰/۰۵ ۱۰:۱۶

می‌خواهید مثل سایت stack دیتابیس رو بدون اطلاعات کاربران با یه لایسنس عمومی کنید؟ :)

نویسنده: نام مستعار
تاریخ: ۱۳۹۱/۱۰/۱۳ ۲۰:۵۰

سلام و تشکر از زحمتتون ...

امکانش هست که یه فهرست به ابتداش اضافه کنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۳ ۲۱:۳۱

فهرست مطالب، نویسندگان، فایل‌های الصاقی و همچنین گروه‌ها به تفکیک در فایل موجود است:

نویسنده: مسعود مشهدی
تاریخ: ۱۳۹۱/۱۰/۱۴ ۱۱:۳۳

با سلام
در صورت امکان گرفتن خروجی از هر گروه در علاقه مندی‌های شخصی هر نفر رو هم بذارید.
با تشکر

نویسنده: omidpoyan
تاریخ: ۱۳۹۲/۰۴/۳۱ ۸:۵۶

آقا من بسیار شرمندم ولی مدتی است آخرین نسخه خروجی کامل سایت را نمیتوانم پیدا کنم لطفا راهنمایی بفرمایید

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۳۱ ۹:۸

حجم مطالب و فایل‌ها و تصاویر بالا رفته، رم سرور جاری اجازه تولید این فایل حجیم PDF رو نمی‌ده. اما خروجی کلی تمام گروه‌ها به تفکیک موجود است .
این خروجی هم صرفا برای کاربران فعال سایت درنظر گرفته شده است. کاربری در این سایت فعال نامیده می‌شود که حداقل یک مطلب جدید یا حداقل یک اشتراک جدید را ارسال کرده باشد.

پ.ن.
لطفا سعی نکنید برای رسیدن به این حد نصاب هر مطلب پیش پا افتاده یا هر لینک به سایت‌های آشپزی که مطالب به نظر علمی

را هم منتشر می‌کنند، ارسال کنید چون منجر به حذف اکانت شما خواهد شد.

عنوان: **intern pool جدول نگهداری رشته‌ها در دات‌نت**
نویسنده: رحمت اله رضایی
تاریخ: ۱۳۹۲/۰۲/۲۹ ۲۲:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: C#, Tips, CLR, string, intern, pool

کد زیر را در نظر بگیرید :

```
object text1 = "test";
object text2 = "test";

object num1 = 1;
object num2 = 1;

Console.WriteLine("text1 == text2 : " + (text1 == text2));
Console.WriteLine("num1 == num2 : " + (num1 == num2));
```

به نظر شما چه چیزی در خروجی نمایش داده میشود؟

هر چهار متغییر text1 و text2 و num1 و num2 از نوع object هستند. با اینکه مقدار text1 و text2 یکی و مقدار num1 و num2 هم یکی است، نتیجه text1==text2 برابر true است اما num1==num2 برابر false.

خطی که text2 تعریف شده است را تغییر میدهیم :

```
object text2 = "test".ToLower();
```

اینبار با این که باز مقدار text1 و text2 یکی و هر دو "test" است، اما نتیجه text1==text2 برابر false است. انتظار ما هم همین است. دو object ایجاد شده است و یکی نیستند. تنها در صورتی باید نتیجه == آنها true باشد که هر دو به یک object اشاره کنند.

اما چرا در کد اولی اینگونه نبود؟

دلیل این کار برمیگردد به رفتار دات‌نت نسبت به رشتههایی که به صورت صریح در برنامه تعریف میشوند. CLR یک جدول برای ذخیره رشتهها به نام **intern pool** برای برنامه میسازد. هر رشتهای تعریف میشود، اگر در intern pool رشتهای با همان مقدار وجود نداشته باشد، یک رشته جدید ایجاد و به جدول اضافه میشود، و اگر موجود باشد متغییر جدید فقط به آن اشاره میکند. در واقع اگر 100 جای برنامه حتی در کلاسهای مختلف، رشتههایی با مقادیر یکسان وجود داشته باشند، برای همه آنها یک نمونه وجود دارد.

بنابراین text1 و text2 در کد اولی واقعا یکی هستند و یک نمونه برای آنها ایجاد شده است.

البته چند نکته در اینجا هست :
اگر text1 و text2 به صورت string تعریف شوند، نتیجه text1==text2 در هر دو حالت فوق برابر true است. چون عملگر == در کلاس string یکبار دیگر overload شده است:

```
public sealed class String : ...
{
    ...
    public static bool operator ==(string a, string b)
    {
```

```
        return string.Equals(a, b);
    }
   ...
}
```

این که کدام یک از overloadها اجرا شوند (کلاس پایه، کلاس اصلی، ...) به نوع دو متغییر اطراف == بستگی دارد. مثلا در کد زیر :

```
string text1 = "test";
string text2 = "test".ToLower();

Console.WriteLine("text1 == text2 (string) : " + (text1 == text2));
Console.WriteLine("text1 == text2 (object) : " + ((object)text1 == (object)text2));
```

اولین نتیجه true و دومی false است. چون در اولی عملگر == تعریف شده در کلاس string مورد استفاده قرار می‌گیرد اما در دومی عملگر == تعریف شده در کلاس object.

اگر دقت نشود این رفتار مشکلزا میشود. مثلا حالتی را در نظر بگیرید که text1 ورودی کاربر است و text2 از بانک اطلاعاتی خوانده شده است و با اینکه مقادیر یکسان دارند نتیجه == آنها false است. اگر تعریف عملگرها در کلاس object به صورت virtual بود و در کلاس‌های دیگر override می‌شد، این تغییر نوع‌ها تاثیری نداشت. اما عملگرها به صورت static تعریف می‌شوند و امکان override شدن ندارند. به همین خاطر کلاس object متدی به اسم Equals در اختیار گذاشته که کلاس‌ها آنرا override می‌کنند و معمولا از این متد برای سنجش برابری دو کلاس استفاده می‌شود :

```
object text1 = "test";
object text2 = "test".ToLower();

Console.WriteLine("text1 Equals text2 : " + text1.Equals(text2));
Console.WriteLine("text1 Equals text2 : " + object.Equals(text1, text2));
```

البته یادآور می‌شوم که **فقط رشته‌هایی که به صورت صریح در برنامه تعریف شده‌اند، در intern pool قرار می‌گیرند و این فهرست شامل رشته‌هایی که از فایل یا بانک اطلاعاتی خوانده می‌شوند یا در برنامه تولید می‌شوند، نیست.** این کار منطقی است وگرنه حافظه زیادی مصرف خواهد شد.

با استفاده از متد string.Intern می‌توان یک رشته را که در intern pool وجود ندارد، به فهرست آن افزود. اگر رشته در intern pool وجود داشته باشد، reference آنرا بر می‌گرداند در غیر اینصورت یک reference به رشته جدید به intern pool اضافه می‌کند و آنرا برمی‌گرداند.

یک مورد استفاده آن هنگام lock روی رشته‌هاست. برای مثال در کد زیر DeviceId یک رشته است که از بانک اطلاعاتی خوانده می‌شود و باعث می‌شود که چند job همزمان به یک دستگاه وصل نشوند :

```
lock (job.DeviceId)
{
    job.Execute();
}
```

اگر یک job با DeviceId برابر COM1 در حال اجرا باشد، این lock جلوی اجرای همزمان job دیگری با همین DeviceId را

نمی‌گیرد. زیرا هر چند مقدار DeviceId دو job یکی است ولی به یک نمونه اشاره نمی‌کنند.

می‌توان lock را اینگونه اصلاح کرد :

```
lock (string.Intern(job.DeviceId))
{
    job.Execute();
}
```

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۳۰ ۰:۳۵

ممنون. البته شرایط کد خودتون رو کامل اینجا قرار ندادید ولی در حالت کلی توصیه میشه که برای استفاده از lock یک شیء private object در سطح کلاس تعریف بشه و از اون استفاده بشه تا حالت‌های دیگر .

نویسنده: رحمت اله رضایی
تاریخ: ۱۳۹۲/۰۲/۳۱ ۹:۵۶

- البته این فقط یک مثال بود برای درک متد string.Intern .
- چگونگی شی معرفی شده به lock هم بسته به شرایط ممکن است متفاوت باشد. ممکن است یک private object در سطح همان کلاسی که lock در آن استفاده می‌شود، جوابگو باشد. اما در شرایط دیگری ممکن است اینگونه نباشد. مانند مثال فوق.

عنوان: رفع مشکل postBack در Master Page و دریافت خطای Operation is not valid due to the current state of the object

نویسنده: حسین صفدری

تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۸:۳۵

آدرس: www.dotnettips.info

برچسب‌ها: ASP.Net, Tips, Handle Errors

با این خطا هنگامی مواجه شدم که در Master Page یک TextBox و یک دکمه قرار داده بودم و در رویداد دکمه به یک صفحه دیگر انتقال پیدا می‌کردم. کد زیر:

```
protected void imgBtnSearch_Click(object sender, ImageClickEventArgs e)
        {
            if (TxtbSearch.Text != string.Empty)
            {
                Response.Redirect("/Exhibition/ShowSearch.aspx?title=" +
                      Sanitizer.GetSafeHtmlFragment(TxtbSearch.Text) + "&b=true", false);
            }
        }
```

این کد در ماشین خودم کار می‌کرد و مشکلی نداشت. وقتی پروژه را به سرور انتقال می‌دادم و بر روی IIS 6 تست می‌کردم خطای 404 دریافت می‌کردم. ضمن اینکه به صفحه خواسته شده انتقال پیدا نمی‌کردم، و همه چیز کاملا گیج کننده بود. وقتی صفحه را از خود سرور باز و رویداد فوق را اجرا می‌کردم خطای Operation is not valid due to the current state of the object دریافت می‌کردم به همراه کد زیر در Track trace به شرح زیر:

```
at System.Web.HttpValueCollection.ThrowIfMaxHttpCollectionKeysExceeded()
at System.Web.HttpValueCollection.FillFromEncodedBytes(Byte[] bytes, Encoding encoding)
at System.Web.HttpRequest.FillInFormCollection()
```

با افزودن کد زیر در Web.config مشکل برطرف گردید.

```
<appSettings>
    <add key="aspnet:MaxHttpCollectionKeys" value="2001" />
  </appSettings>
```

منابع:

Code Project

stack overflow

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۸:۴۷

این مشکل زمانی به وجود میاد که تعداد انبوهی کنترل روی صفحه داشته باشید. ساختار صفحه شما به چه صورتی است؟ (کابین کنترل خلبان؟!)

نویسنده: حسین صفدری
تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۹:۲

D: نه بابا!

گفتم در master Page و صفحه اول سایت.شامل منو، 3 تا repatear (تبلیغات، اخبار) و چندین باکس دیگه...

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۲/۰۵/۰۸ ۲۰:۳۸

خیلی ممنون بابت این نکته.

ولی دیگه سعی کنید که کوچ کنید به MVC. تو وب فرم آدم نمیدونه تو وب هست یا ویندوز فرم. یک وب مصنوعی.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۰۸ ۲۰:۴۶

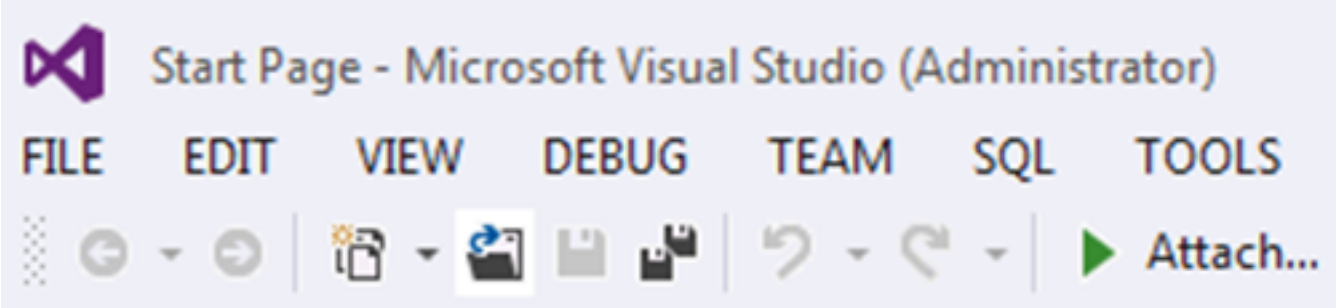
مشکل وب فرم‌ها اینه که کل صفحه رو یک فرم درنظر می‌گیره. بنابراین موقع postback به سرور تمام صفحه رو ارسال می‌کنه که ممکنه سنگین باشه. در MVC می‌تونید در یک صفحه چندتا فرم داشته باشید با محدوده‌های عملکرد کوچک‌تر.

نویسنده: حسین صفدری
تاریخ: ۱۳۹۲/۰۵/۰۸ ۲۲:۷

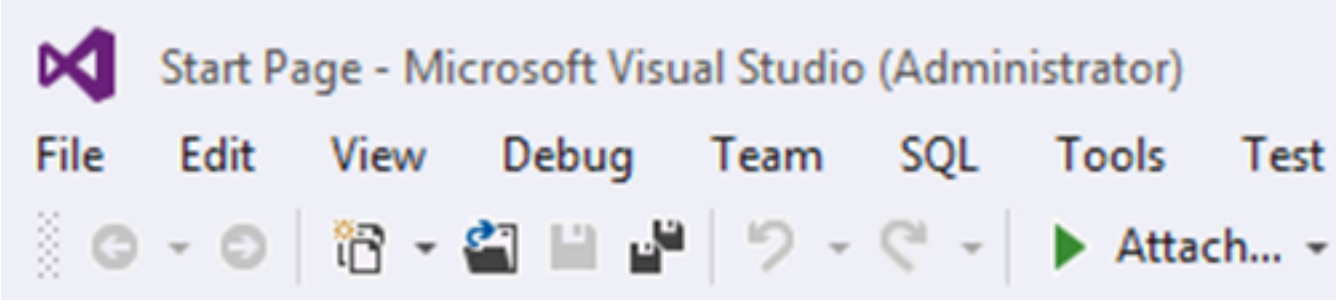
چشم، مرسی نکات زیبایی دوستان گفتند، ولی راستش این پروژه برای 2 سال پیش بود، امروز به دلایلی یه گزینه جستجو اضافه کردم و باقی ماجرا ;-)

عنوان: **نحوه نمایش منوهای Visual studio 2012 با حروف کوچک**
نویسنده: بهمن خلفی
تاریخ: ۱۳۹۲/۰۵/۱۵ ۱۶:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: Tips, Visual Studio

چند روز پیش بصورت اتفاقی به این فکر افتادم که چرا منوهای visual studio 2012 برخلاف ظاهر زیبای خود محیط، اینقدر زمخت و با حروف بزرگ نوشته است.



و اینکه به چه صورت میتوانم آنها را بصورت حروف کوچک نمایش دهم و حس کنجکاوی اونم از نوع مخصوص گل کرد.



برای اینکار دو روش وجود دارد :

روش 1 - تغییر مقدار در رجیستری سیستم عامل ویندوز

بدین صورت که شما باید به این مسیر مراجعه نموده

در ویندوز 7 :

```
HKEY_CURRENT_USER\Software\Microsoft\VisualStudio\11.0\General\
```

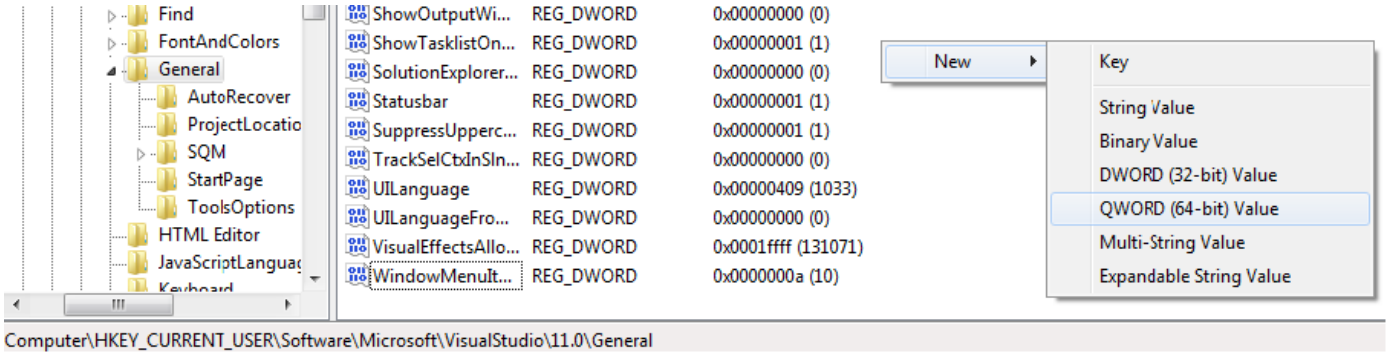
در ویندوز 8 :

```
HKEY_CURRENT_USER\Software\Microsoft\VSWinExpress\11.0\General\
```

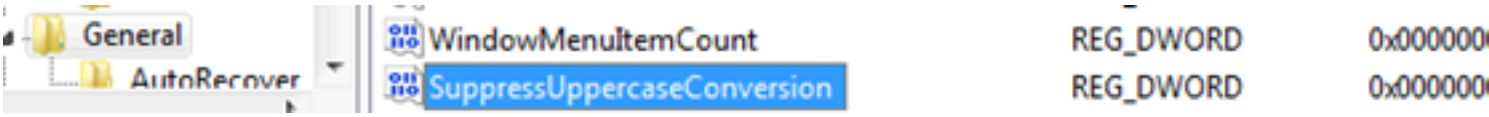
در نسخه web express:

```
HKEY_CURRENT_USER\Software\Microsoft\VSWDExpress\11.0\General\
```

ایجاد یک کلید از نوع DWORD :



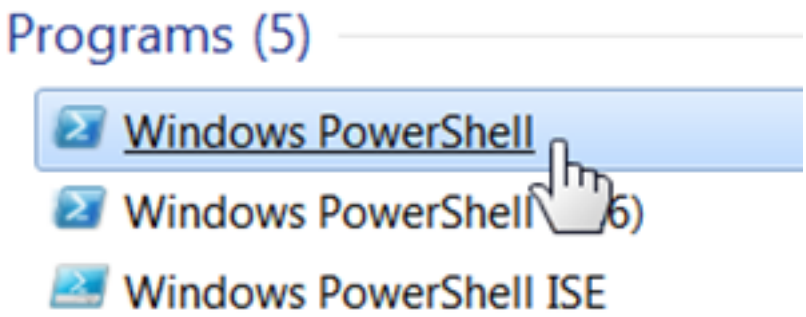
و با نام SuppressUppercaseConversion و با مقدار 1 در مسیر یاد شده تنظیم نمائید.



سپس راه اندازی مجدد visual studio و مشاهده منوهای تغییر یافته .

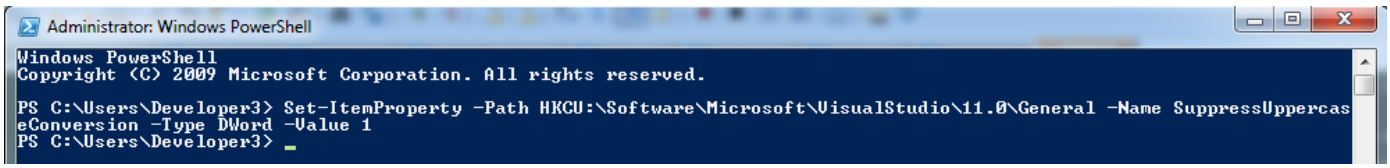
روش 2 - کسانی مثل من کمی تنبل هستند و از این کارهای فوق دوست ندارند راه آسانتر را میتوانند تجربه کنند بصورت ذیل:

در منوی start ویندوز با تایپ کلمه powershell و انتخاب Windows PowerShell به صفحه‌ای آبی رنگ (در ویندوز 7) وارد میشود .



دستور ذیل را کپی و به پنجره powershell انتقال دهید :

```
Set-ItemProperty -Path HKCU:\Software\Microsoft\VisualStudio\11.0\General -Name SuppressUppercas
eConversion -Type DWord -Value 1
```



سپس راه اندازی مجدد visual studio و مشاهده منوهای تغییر یافته

نظرات خوانندگان

نویسنده: مرتضی
تاریخ: ۱۳۹۲/۰۵/۱۵ ۱۸:۵۸

با استفاده از VSCommands هم میتونید

نویسنده: sysman
تاریخ: ۱۳۹۲/۰۵/۱۶ ۱۰:۳۲

می تونید از All Caps Menu Option هم استفاده کنید.

نویسنده: بهمن خلفی
تاریخ: ۱۳۹۲/۰۵/۱۶ ۱۱:۲۱

با تشکر از شما دوستان عزیز - مفید بود.
تنها مزیت کار فوق این است که نیازی به دانلود و یا نصب برنامه و یا پلاگین ندارد . البته هدف انجام این کار بود . از شما عزیزان هم تشکر و قدردانی میکنم .

عنوان: **آناتومی یک گزارش خطای خوب**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۲۴ ۲۲:۰
آدرس: www.dotnettips.info
گروه‌ها: Tips

به مشکلی در برنامه‌ای برخورده‌اید؟ کتابخانه‌ای کار نمی‌کند؟ خطایی را دریافت کرده‌اید؟ برنامه کامپایل نمی‌شود؟ برنامه آنطور که مدنظر شما است رفتار نمی‌کند؟ برای طرح این مسایل، صرف عنوان کردن «برنامه کار نمی‌کنه» یا «خطا می‌ده» منزلت خودتان را تا حد یک کاربر عادی تازه کار تنزل داده‌اید. در ادامه ساختار یک گزارش خطای خوب را بررسی خواهیم کرد، تا شما را سریعتر به مقصودتان برساند و همچنین کار پیگیری برنامه نویس یا برنامه نویس‌های مسئول را نیز مقداری ساده‌تر کند.

کارهای لازم پیش از طرح سؤال

- سعی کنید انجمن‌های مرتبط را یکبار بررسی و جستجو کنید.
- عین خطای دریافتی را در گوگل جستجو کنید. اگر از برنامه‌ها یا کتابخانه‌های معروف و متداول استفاده می‌کنید، یکی از مزیت‌های مهم کار با آن‌ها، «تنها نبودن» است! یقین داشته باشید خطایی را که دریافت کرده‌اید پیشتر توسط ده‌ها نفر دیگر در سایت‌های مختلف مطرح شده‌اند و بالاخره با بررسی آن‌ها می‌توان به پاسخ رسید.
- شاید راهنمای برنامه در این مورد خاص مطلبی را عنوان کرده است.

و ... به صورت خلاصه باید بتوانید به این سؤال پاسخ دهید: «خودت چکار کردی؟». حداقل نشان دهید که فرد حاضر و آماده طلبی نیستید و پیشتر یک حداقل تقلایی را انجام داده‌اید.

کجا باید سؤال پرسید؟

- اگر به انجمنی برای طرح سؤال خود مراجعه کرده‌اید، حتما زیر شاخه صحیحی را انتخاب کنید تا سؤال شما بسته نشود یا کلا حذف نگردد. برای مثال سؤال ASP.NET را در بخش سی‌شارپ نپرسید یا برعکس یا اگر سایتی مقاله‌ای را منتشر کرده، ذیل آن در مورد نحوه بک آپ گرفتن از اکانت توئیتر خود سؤال نپرسید!
- اگر پاسخی را دریافت کردید، ادامه بحث را ذیل همان مطلب پیگیری کنید و مجددا مطلب جدیدی را ایجاد نکنید.
- اگر تا نیم ساعت بعد جوابی را دریافت نکردید، کل بخش‌های یک سایت را با ارسال پیام خود اسپم نکنید. یکبار ارسال یک سؤال کافی است. اکثر این سایت‌ها حالت یک «چت آفلاین» را دارند. به این معنا که ابتدا پیغام خود را می‌گذارید، اگر مدتی بعد (ممکن است چند ساعت بعد) شخصی آن‌را مشاهده کرد و قادر به پاسخ دهی بود، به شما کمک خواهد کرد. بنابراین اگر سریعا به جواب نرسیدید، نه کل سایت را اسپم کنید و نه ... شروع به رفتارهای ناشایست کنید. اینکار با فریاد کشیدن وسط یک جمع تفاوتی ندارد. اشخاص مرتبط همواره آنلاین نیستند؛ ضمنا ممکن است واقعا پاسخی برای یک سؤال نداشته باشند. منصف باشید.
- از ایمیل‌های خصوصی افراد یا قسمت پیام‌های خصوصی سایت‌ها برای ارسال سؤالات شخصی استفاده نکنید. ایمیل خصوصی، مخصوص کارهای شخصی است. قسمت پیام‌های خصوصی یک سایت عموما مخصوص رسیدگی به مشکلات کاربری است. این تصور را نداشته باشید که اشخاص مشاور شخصی رایگان پروژه‌های تجاری شما هستند.
- بهترین محل برای پرسیدن سؤالات مرتبط با یک پروژه خاص، mailing list یا انجمن گفتگو و یا issue tracker آن پروژه است. وقت خودتان را با ارسال خطاهای یک پروژه خاص، در یک انجمن عمومی و همه منظوره تلف نکنید. کمی جستجو کنید که سایت اصلی پروژه کجا است. بعد دقت کنید آیا جایی برای پرسش و پاسخ دارد یا خیر. اکثر پروژه‌های خوب، مکانی را جهت جمع آوری بازخوردهای پروژه خود، اختصاص می‌دهند.

چطور باید سؤال پرسید؟

سؤال فنی خوب پرسیدن هم یک هنر است؛ که تعدادی از مشخصه‌های مهم آن‌را در ذیل مرور خواهیم کرد:
- عنوان مناسبی را برای سؤال خود انتخاب کنید. «لطفا کمک کنید» یا «من مشکل دارم» یا «مشکل در پروژه»، عموما واکنش‌های تندی را به همراه دارند؛ و تا حد ارسال اسپم در یک سایت بی‌کیفیت تلقی می‌شوند. ضمن اینکه انتخاب عنوان‌های مناسب، جستجوهای بعدی را در سایت ساده می‌کنند و کمک بزرگی خواهند بود به افراد بعدی.
- محیطی را که خطا در آن رخ داده است، توضیح دهید. ذکر IIS تنها کافی نیست. کدام نگارش آن؟ در کدام ویندوز؟ برای مثال شماره نگارش کتابخانه یا نرم افزار مورد استفاده را ذکر کنید. شاید خطایی که گرفته‌اید در نگارش بعدی آن برطرف شده است.

ذکر شماره نگارش VS.NET یا شماره نگارش دات نت مورد استفاده، سیستم عامل و کلا توصیف محیط بروز خطا، عموما بسیار مفید هستند.

- حتما کل خطای دریافت شده را ارسال کنید. اگر در یک برنامه C خطایی حاصل شود، احتمالا شکلی مانند Error 0xABCD را دارد. اما استثناءهای دات نت به همراه stack trace و حتی شماره سطر خطای حاصل نیز هستند. همین مساله می‌تواند به خطایابی نهایی بسیار کمک کند.

- سؤال خود را طوری مطرح کنید که شخص مقابل بتواند آن‌را در کمترین زمان ممکن «باز تولید» کند. برای مثال ذکر خطای دریافتی بسیار خوب است. اگر داده‌ای که سبب بروز این خطا شده است را هم ارسال کنید، مفیدتر خواهد بود؛ یا اگر دستور پاور شل خاصی در کنسول نیوگت خطا می‌دهد، صرفا عنوان نکنید که جواب نگرفته‌اید. چه دستوری را اجرا کرده‌اید؟ چه خطایی را دریافت کرده‌اید؟ ساختار پروژه شما چیست؟ آیا شخص مقابل می‌تواند بر اساس اطلاعاتی که ارائه دادید یک آزمایش شخصی را تدارک ببیند؟ آیا می‌تواند آن‌را با توضیحات شما مجددا تولید کند؟

زمان باز تولید خطا را هم مدنظر داشته باشید. برای مثال اگر بتوانید قطعه کدی را ارائه دهید که در کمترین زمان ممکن، صرفا با کپی و پیست آن در VS.NET قابل کامپایل باشد، بسیاری علاقمند به پاسخگویی به شما خواهند شد. در غیراینصورت آنچنان انتظار نداشته باشید که شخص پاسخ دهنده وقت زیادی را برای رسیدگی به جزئیات سؤال شما صرف کند؛ یا مدتی مشغول به تهیه یک مثال جدید بر مبنای توضیحات شما شود.

حجم کدهای ارسالی شما نیز در اینجا مهم هستند. کل پروژه خود را ارسال نکنید! سعی کنید یک مثال کوچک را که بتواند سریعا خطای مدنظر شما را بازتولید کند، ارسال کنید و نه بیشتر. همچنین کدهایی که برای اجرا نیاز به GUI نداشته باشند نیز در این حالت اولویت دارند.

و به صورت خلاصه، خودتان را بجای پاسخ دهنده قرار دهید. آیا با چند جمله‌ای که ارائه داده‌اید، می‌توان انتظار پاسخی را داشت یا خیر.

- ایمیل شخصی خود را در انتهای پیام ارسال نکنید. کسی اهمیتی نمی‌دهد! اگر سؤال شما پاسخی داشته باشد، همانجا دریافت خواهید کرد و نه در میل باکس شخصی.

- املاء و انشای متنی را که ارسال می‌کنید، یکبار بررسی کنید. اگر برای شما اهمیتی ندارد که چه کلمات و جمله بندی را باید بکار برد، برای شخص مقابل هم آنچنان اهمیتی نخواهد داشت که زیاد وقت صرف کند.

- از بکار بردن smileyهای بیش از حد یا قرار دادن تعداد علامت تعجب‌های بیش از حد خودداری کنید. این موارد عموما به مسخره کردن شخص مقابل تفسیر می‌شوند.

- در بدو امر فریاد نکشید که «باگ» پیدا کرده‌اید؛ خصوصا اگر به mailing list اختصاصی یک پروژه پیامی را ارسال می‌کنید. چون اگر مشکل شما واقعا باگ نباشد، بیشتر یک توهین تلقی خواهد شد و در دفعات بعدی پاسخ دادن به شما به صورت ضمنی مؤثر خواهند بود؛ یا جواب نمی‌گیرید و یا جدی گرفته نخواهید شد.

- هدف از کاری را که مشغول به انجام آن بوده‌اید را نیز ذکر کنید. ذکر خطای دریافتی بسیار مفید است اما اگر بتوانید یک دید کلی را نسبت به کاری که مشغول به آن بوده‌اید، ایجاد کنید، شاید پاسخ بهتری را دریافت کنید. برای مثال جهت رسیدن به هدف و مقصود شما بهتر است از روش دیگری استفاده کنید.

- پس از اینکه پیامی را دریافت کردید، یک حداقل واکنشی را ارسال کنید. مثلا خوب بود؛ کمک کرد و یا مفید نبود. همین واکنش‌ها در آینده به کمک نتایج جستجوهای انجام شده خواهند آمد و اشخاص بعدی حداقل خواهند دانست که پاسخ داده شده صحیح بوده است یا خیر.

و همیشه بخاطر داشته باشید: تمام خدماتی که سایت‌های عمومی به شما ارائه می‌دهند «یک لطف» است و حقی را برای شما ایجاد نمی‌کنند. این اشخاص از شما پول نمی‌گیرند تا به سؤالات شما پاسخ دهند یا تبدیل به مشاور خصوصی رایگان شما شوند. می‌توانید محیط را برای این اشخاص، با اندکی احترام، ملایمت و انصاف، دلپذیرتر کنید.

نظرات خوانندگان

نویسنده: محسن جمشیدی
تاریخ: ۱۳۹۲/۰۶/۲۵ ۸:۳۰

درود بر جناب نصیری عزیز
با تک تک جملاتتون موافقم و نمونه بارزی از عدم رعایت نکاتی که اشاره کردین سایت برنامه نویس هست با یک نگاه به بخش سی شارپ می‌تونید تمام این مواردی رو که اشاره کردین مشاهده کنید. بعد از چند سال نتونستیم جا بندازیم که دوستان عنوان مناسب برای تاپیک خودتون انتخاب کنید در شرح اون به بطور کامل به تشریح مسئله بپردازید اما دریغ از یک اثر مثبت.
یکی از مواردی هم که میشه بهش اشاره کرد نوع ادبیات و بیان احساسات هست
- استادمون گیر داده می‌گه...
- آقا من یه مشکل دارم
- داداش مثله اینکه نمی‌فهمی چی می‌گم

اما اینجا جای تشکر داره از افرادی که "مثلا در سایت StackOverFlow" با مدیریت خودشون (حذف یا ویرایش سوالات) باعث می‌شن که سوالات بی مورد و تکراری کمتر بشن و ما با یک جستجوی ساده بتونیم به جواب برسیم

نویسنده: بهزاد
تاریخ: ۱۳۹۲/۰۶/۲۵ ۱۱:۱۴

خیلی خوب بود، ازش پرینت کردم و به دیوار شرکت چسبوندم، کاش فقط توش می‌نوشتید اول به مدیر شرکت مراجعه نکنند!

نویسنده: میثم ثوامری
تاریخ: ۱۳۹۲/۰۶/۲۸ ۹:۱

با سلام خدمت آقای نصیری
معمولا زمانی که مشکلی برام پیش میاد میرم سراغ StackOverFlow .
من معمولا با پرسیدن سوال تو این وبسایت مشکل دارم البته اونم به لحاظ ضعیف بودن زبان انگلیسی بنده است.یک استاندارد برای پرسیدن سوال وجود داره و به تازگی قبل از پرسیدن سوال کلی توضیح و راهنمایی نمایش داده میشه
?#How to resolve errors with unsafe pointer in C
سوال بالا آخرین موضوعی بود که چندی پیش پرسیدم.کلی امتیاز منفی بهمراه داشت و در انتها سوالم Close شد.

نویسنده: ح مراداف
تاریخ: ۱۳۹۲/۱۱/۲۴ ۱۷:۱۴

سلام،
برای دوستانی که انگلیسیشون زیاد خوب نیست (مثل خودم) و کلا با انگلیسی مشکل دارن، جهت سرچ در گوگل روش زیر پیشنهاد می‌گردد.
معمولا بهتره توی گوگل مشکلتون رو اینطوری سرچ کنین :

```
how to <doing problem>   in <your programming language>
```

مثال:

```
how to convert int to string in c#
```

این روش هم خوبه :

```
<programming language> how <problem>
```

دو روش فوق معمولا در 90% موارد جواب میده.

نکته دوم اینه که اگر توی نتیجه سرچ لینکی از سایت StackOverFlow دیدین ، اول بهتره اون لینک رو باز کنین.
چون معمولا این سایت بصورت سوال و جوابه و سریع با دیدن کدهای نمونه ای که در پاسخ‌ها موجوده مشکلتون حل میشه و معمولا اصلا نیاز نیست متون انگلیسی رو بخونین.
مخصوصا که برای هر پاسخ یک امتیاز وجود داره که با بررسی امتیاز هر پاسخ می‌توان تا حدودی به درست بودن و بهینه بودن راه حل پی برد (این مورد حدودی است)

البته سایت codeProject هم خوب است ولی تا جایی که من می‌دونم بصورت مقاله ای است و باید کلی توضیح و مثال و ... رو بررسی کنین تا به جواب سوالتون برسین که یکم وقت گیر تره ...

نکته بعدی: بد نیست برخی مشکلات رو بصورت فارسی هم سرچ بگیرین ، در برخی موارد مطلب فارسی هم در مورد برخی خطاها پیدا میشه (معمولا فارسی زبونا زیاد مقاله درباره خطاها نمی‌نویسن و بیشتر تکنولوژی رو آموزش میدن ...)

{البته همه دوستان حرفه ای هستند ، به جهت دوستانی که مثل خودم زیاد با انگلیسی میونه خوبی ندارند عرض کردم}

ضمنا در تکمیل مقاله آقای نصیری باید ذکر کنم که :
*همیشه به خاطر داشته باشید که هیچکس علمشو مجانی بدست نیاورده که مجانی در اختیار شما قرار بده !
پس اگر کسی داره مقاله منتشر می‌کنه یا جواب سوالتونو میده ، داره به شما لطف می‌کنه...
پس سعی کنین همیشه ازش تشکر کنین و بهش احترام بذارین تا بیشتر تشویق بشه که مطالب بیشتری بذاره ... (به نظر من دست کسی که مجانی داره بهت چیز یاد میده ، باید بوسید؛ چون در حقیقت وقت و انرژی و علمشو داره به شما هدیه می‌کنه که خیلی ارزشمنده)

*نوشتن کامنت هایی که دانش نفر رو زیر سوال ببره ، مثلا "این روش که دیگه خدا بیامرز شده ، الان هر برنامه نویس حرفه ای از فلان روش استفاده می‌کنه" خیلی بده و نفر رو نا امید می‌کنه.

*لحن گفتن یک سوال یا بیان یک کامنت خیلی خیلی مهمه.
مثلا همین مقال بالا رو میشه بدین صورت هم گفت : "در تکمیل مقاله جناب X باید ذکر نمایم که روش Y نیز بسیار جالب می‌باشد و دارای کد نویسی بهتری است" و حتی یک لینک آموزشی خوب هم درباره روش جدید ارائه دهیم.
(اینطوری هر دو طرف سود میکنن)

همین!
موفق باشید.

نویسنده: ح مراداف
تاریخ: ۱۳۹۲/۱۱/۲۴ ۱۷:۱۷

خداییش اینو خوب اومدی:
"- ایمیل شخصی خود را در انتهای پیام ارسال نکنید. کسی اهمیتی نمی‌دهد! اگر سؤال شما پاسخی داشته باشد، همانجا دریافت خواهید کرد و نه در میل باکس شخصی. "

در بسیاری از سایت‌ها نفر سوال می‌کنه و میگه : لطفا جواب را به ایمیل بنده ارسال فرمایید : X@Y.Com
یا
فلان فایل رو نتونستم دانلود کنم ، لطفا به ایمیلم ارسالش کنین !

آدم می‌مونه بهش چی بگه !
یا حق

عنوان: **چطور باید یک پروژه سورس باز را خوب مدیریت کرد؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۲۸ ۱۲:۰
آدرس: www.dotnettips.info
برچسب‌ها: Tips, Open Source, Project Management

اگر مایل هستید که پروژه خود را به صورت سورس باز ارائه دهید، نیاز است یک سری شرایط را رعایت کنید تا کاربران این پروژه بتوانند به سادگی از آن استفاده نمایند.

- فایل ReadMe را فراموش نکنید

حتی اگر پروژه شما از یک سایت اختصاصی استفاده می‌کند، اولین محلی که عموم کاربران برای دریافت اطلاعات کار با پروژه، به آن مراجعه می‌کنند، فایل ReadMe برنامه است. این فایل می‌تواند حاوی مشخصات ذیل باشد:

الف) وابستگی‌های پروژه را مشخص کنید

واقعیت این است که برخلاف شمای برنامه نویس، عموم استفاده کنندگان، آشنایی چندانی با جزئیات محیط و شرایط تهیه برنامه شما ندارند. به این ترتیب بسیاری از مسایلی که برای شما بدیهی هستند، برای عموم اینگونه نخواهند بود. بنابراین مساله‌ای که به سرعت می‌تواند سبب خشم کاربران و صرفنظر از کار شما گردد، مشخص نبودن نحوه نصب و وابستگی‌های لازم برای اجرای برنامه است.

ب) وضعیت بلوغ پروژه خود را مشخص کنید

آیا از این برنامه، مدتی است که در محیط کاری استفاده می‌کنید؟ آیا به نظر شما هنوز ناتمام است؟ آیا API کتابخانه شما در نگارش بعدی کاملا دگرگون خواهد شد؟ تمام این مسایل و سؤالات را به نحو واضحی توضیح دهید و مشخص کنید. همین توضیحات کوتاه می‌توانند ساعت‌های بسیاری از زندگی دیگران را صرفه جویی کند.

ج) اگر پروژه شما یک کتابخانه است، نوع زبان و Runtimeهای پشتیبانی شده را مشخص کنید

برای مثال اگر یک کتابخانه دات نتی را ارائه می‌دهید، مشخص کنید که از کدام نگارش دات نت به بعد را پشتیبانی می‌کنید.

د) مجوز استفاده از پروژه را مشخص کنید

مطلب مقایسه مجوزهای سورس باز را یکبار مطالعه نمائید و سپس مجوز صحیحی را برای کار خود انتخاب کنید. همچنین آن‌را به نحو واضحی در مستندات پروژه خود قید نمائید.
به علاوه به‌خاطر داشته باشید که امکان ارائه مجوزهای دوگانه مانند AGPL نیز وجود دارند. در این حالت کاربر یا باید سورس محصول خودش را ارائه دهد، یا مجوز کتابخانه شما را خریداری کند. مانند RavenDB که از این نوع مجوز استفاده می‌کند.

- یک پروژه نیاز به مستندات دارد

مستند سازی کار، سخت و زمانبر است؛ اما بهترین لطفی است که می‌توانید به کاربران خود نمائید. مستندات نه تنها زمان جستجوی بسیاری را صرفه جویی خواهند کرد، همچنین حس اطمینان خاطر را به کاربر القاء می‌کنند. از این جهت که احساس می‌کنند شما برای کارتان ارزش قائل بوده‌اید و احتمال اینکه این برنامه در آینده نزدیک به یک abandonware تبدیل شود، کم است (منظور یک برنامه فراموش شده و خاتمه یافته).

- به روز رسانی را ساده کنید

بالاخره زمانی نیاز خواهد بود تا نگارش جدیدی از کار خود را ارائه دهید. در این حالت نیاز است یک سری از شرایط را مدنظر داشته باشید:

الف) سازگاری قبلی را مدنظر داشته باشید

یکی از بدترین حالات به روز رسانی یک کتابخانه زمانی است که کاربر آن با ده‌ها خطای کامپایل حاصل از به روز رسانی مواجه شود. اگر نیاز است قسمتی از کد خود را حذف کنید یا تغییر دهید، استفاده از ویژگی Obsolete را فراموش نکنید و اینکار باید مرحله به مرحله انجام شود. در یک نگارش، ویژگی Obsolete را معرفی کنید. در دو نگارش بعد، API را تغییر دهید.

ب) حتما یک Change log را تکمیل کنید

پس از ارائه یک نگارش جدید، حداقل در چند سطر مشخص کنید که چه مواردی تغییر کرده‌اند، چه مواردی اضافه شده‌اند و چه مواردی را حذف کرده‌اید.
همچنین اگر مواردی تغییر کرده‌اند، نحوه ارتقاء کدهای قدیمی را به نگارش جدید، شرح دهید. اگر مورد جدیدی اضافه شده‌است، لینکی را به مثالی درباره‌ی آن ارائه دهید.

- نگارش‌های جدید را اعلام کنید

برای مثال در طی ارائه یک مطلب جدید در وبلاگ خود، ارائه نگارش جدیدی از کتابخانه یا برنامه خود را به عموم اعلام کنید. در این حالت، حتما لینکی را به change log، ارائه داده و مشخص کنید که وضعیت سازگاری آن با قبل چگونه است.

- محلی را برای دریافت بازخوردهای پروژه خود مشخص کنید

نیاز است بتوانید پروژه خود را پشتیبانی کنید یا به سؤالات مربوطه پاسخ دهید. اگر سورس کنترل یا برنامه مدیریت پروژه شما، امکان پرسش و پاسخ را دارد، که بسیار خوب. اگر خیر، می‌توانید مثلا یک گروه گوگل جدید و امثال آن‌را برای دریافت بازخوردهای پروژه ایجاد کنید.
همچنین نیاز است لینک به این محل را در فایل ReadME پروژه به صراحت مشخص کنید.

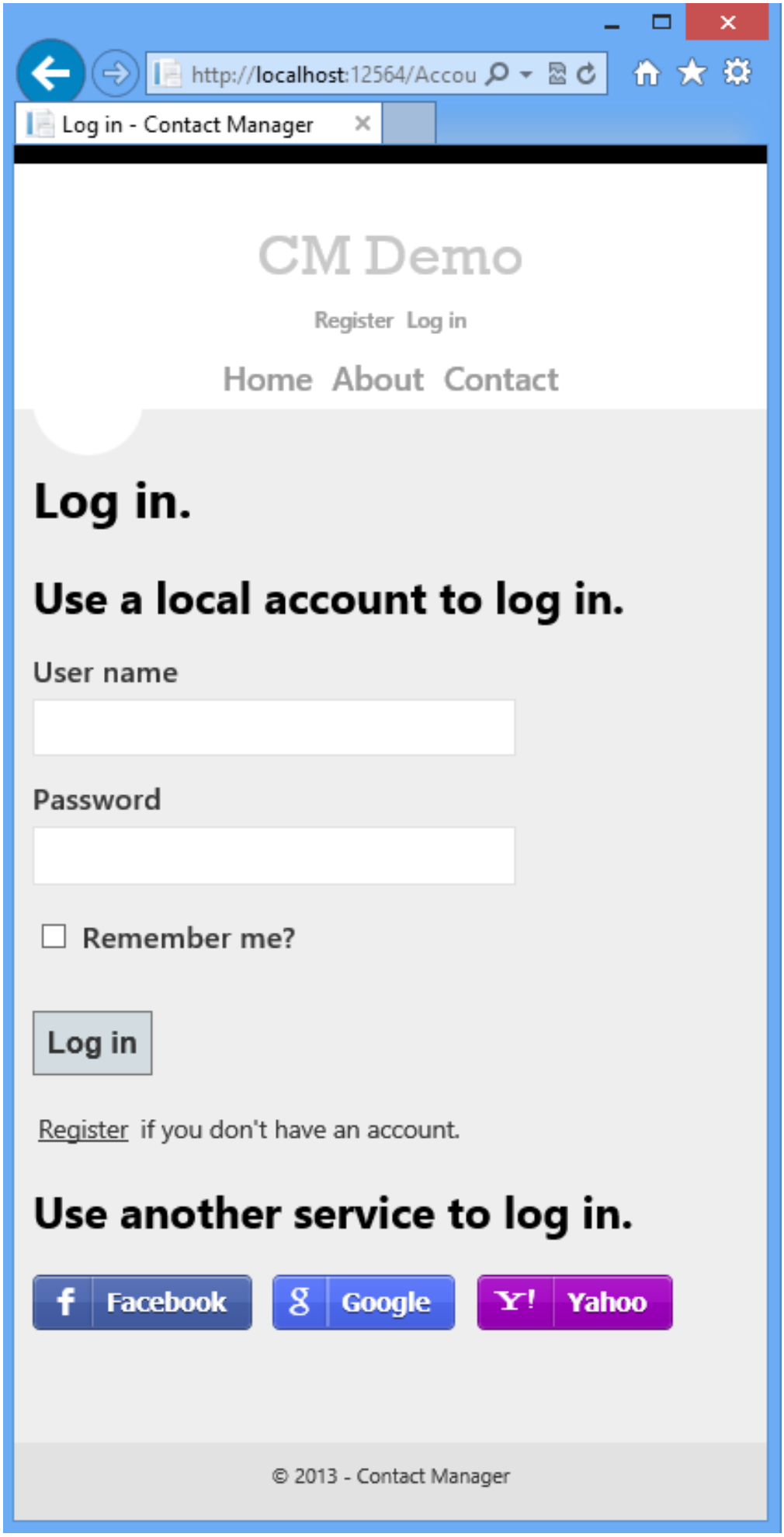
- گذر از پروژه

بالاخره روزی فراخواهد رسید که دیگر علاقه‌ای به نگهداری پروژه نداشته باشید. این مساله را در مکان جمع آوری بازخوردهای خود اعلام کنید یا شخص دیگری را به نگهداری پروژه دعوت نمائید. اگر این کار را انجام ندهید، سبب خواهید شد forkهای متعددی از این پروژه بی‌جهت ایجاد شده و در نهایت مشخص نباشد که کدامیک بهتر است و کدامیک مشکلات کمتری دارند.

عنوان: **توزیع یک اپلیکیشن ASP.NET MVC 5 روی Windows Azure**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۹ ۱۰:۵۰
آدرس: www.dotnettips.info
گروه‌ها: Tips, OpenID, ASP.NET MVC 5, ASP.NET Identity, Windows Azure, OAuth, MVC, Security

این مقاله به شما نشان می‌دهد چگونه یک اپلیکیشن وب ASP.NET MVC 5 بسازید که کاربران را قادر می‌سازد با اطلاعات Facebook یا Google احراز هویت شده و به سایت وارد شوند. همچنین این اپلیکیشن را روی Windows Azure توزیع (Deploy) خواهید کرد.
می توانید بصورت رایگان یک حساب کاربری Windows Azure بسازید. اگر هم Visual Studio 2013 را ندارید، بسته SDK بصورت خودکار Visual Studio 2013 for Web را نصب می‌کند. پس از آن می‌توانید به توسعه رایگان اپلیکیشن‌های Azure بپردازید، اگر می‌خواهید از Visual Studio 2012 استفاده کنید به این مقاله مراجعه کنید. این مقاله نسبت به لینک مذکور بسیار ساده‌تر است.
این مقاله فرض را بر این می‌گذارد که شما هیچ تجربه ای در کار با Windows Azure ندارید. در انتهای این مقاله شما یک اپلیکیشن مبتنی بر داده (data-driven) و امن خواهید داشت که در فضای رایانش ابری اجرا می‌شود.
چیزی که شما یاد می‌گیرید:
چطور یک اپلیکیشن وب ASP.NET MVC 5 بسازید و آن را روی یک وب سایت Windows Azure منتشر کنید.
چگونه از OpenID ، OAuth و سیستم عضویت ASP.NET برای ایمن سازی اپلیکیشن خود استفاده کنید.
چگونه از API جدید سیستم عضویت برای مدیریت اعضا و نقش‌ها استفاده کنید.
چگونه از یک دیتابیس SQL برای ذخیره داده‌ها در Windows Azure استفاده کنید.

شما یک اپلیکیشن مدیریت تماس (Contact Manager) ساده خواهید نوشت که بر پایه ASP.NET MVC 5 بوده و از Entity Framework برای دسترسی داده استفاده می‌کند. تصویر زیر صفحه ورود نهایی اپلیکیشن را نشان می‌دهد.


http://localhost:12564/Accou
Log in - Contact Manager
CM Demo
Register Log in
Home About Contact
Log in.
Use a local account to log in.
User name
Password
Remember me?
Log in
Register if you don't have an account.
Use another service to log in.
Facebook
Google
Yahoo
© 2013 - Contact Manager

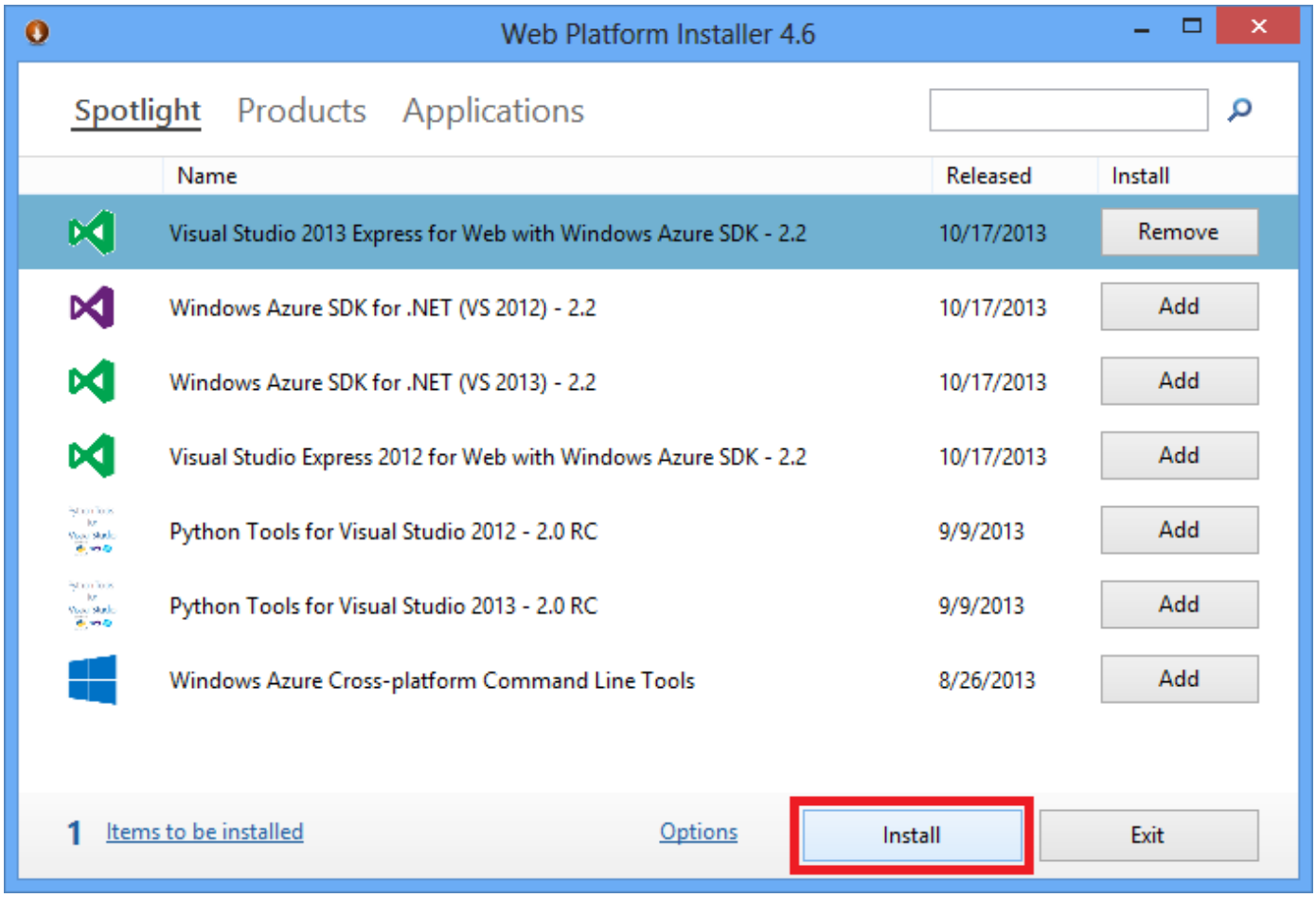
توجه: برای تمام کردن این مقاله به یک حساب کاربری Windows Azure نیاز دارید، که بصورت رایگان می‌توانید آن را بسازید. برای اطلاعات بیشتر به Windows Azure Free Trial مراجعه کنید.

در این مقاله:

برپایی محیط توسعه (development environment)
برپایی محیط Windows Azure
ایجاد یک اپلیکیشن ASP.NET MVC 5
توزیع اپلیکیشن روی Windows Azure
افزودن یک دیتابیس به اپلیکیشن
افزودن یک OAuth Provider
استفاده از Membership API
توزیع اپلیکیشن روی Windows Azure
قدم‌های بعدی

برپایی محیط توسعه

برای شروع Windows Azure SDK for .NET را نصب کنید. برای اطلاعات بیشتر به Windows Azure SDK for Visual Studio 2013 مراجعه کنید. بسته به اینکه کدام یک از وابستگی‌ها را روی سیستم خود دارید، پروسه نصب می‌تواند از چند دقیقه تا نزدیک دو ساعت طول بکشد. توسط Web Platform می‌توانید تمام نیازمندی‌های خود را نصب کنید.



هنگامی که این مرحله با موفقیت به اتمام رسید، تمام ابزار لازم برای شروع به کار را در اختیار دارید.

برپایی محیط Windows Azure

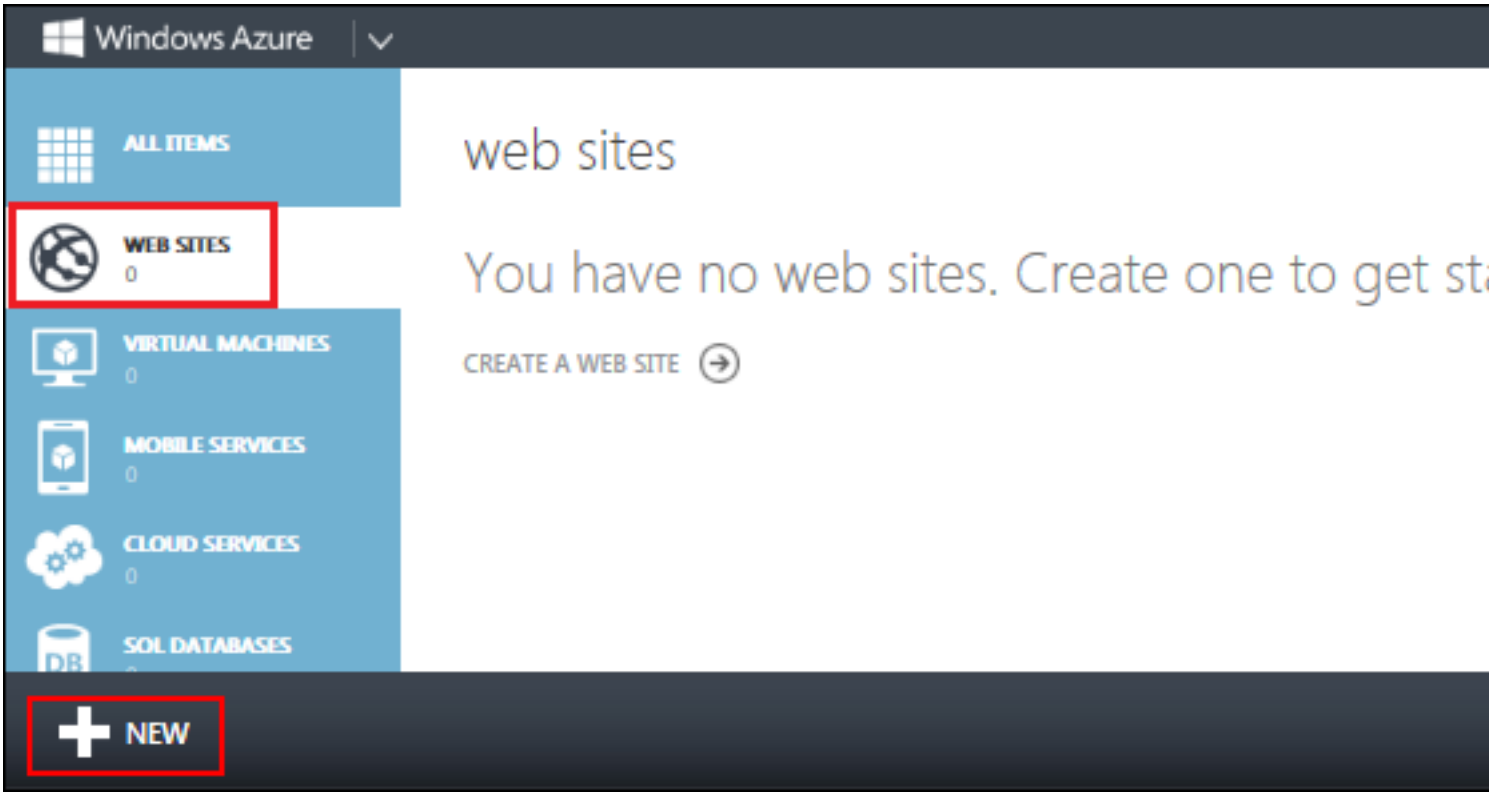
در قدم بعدی باید یک وب سایت Windows Azure و یک دیتابیس بسازیم.

ایجاد یک وب سایت و دیتابیس در Windows Azure

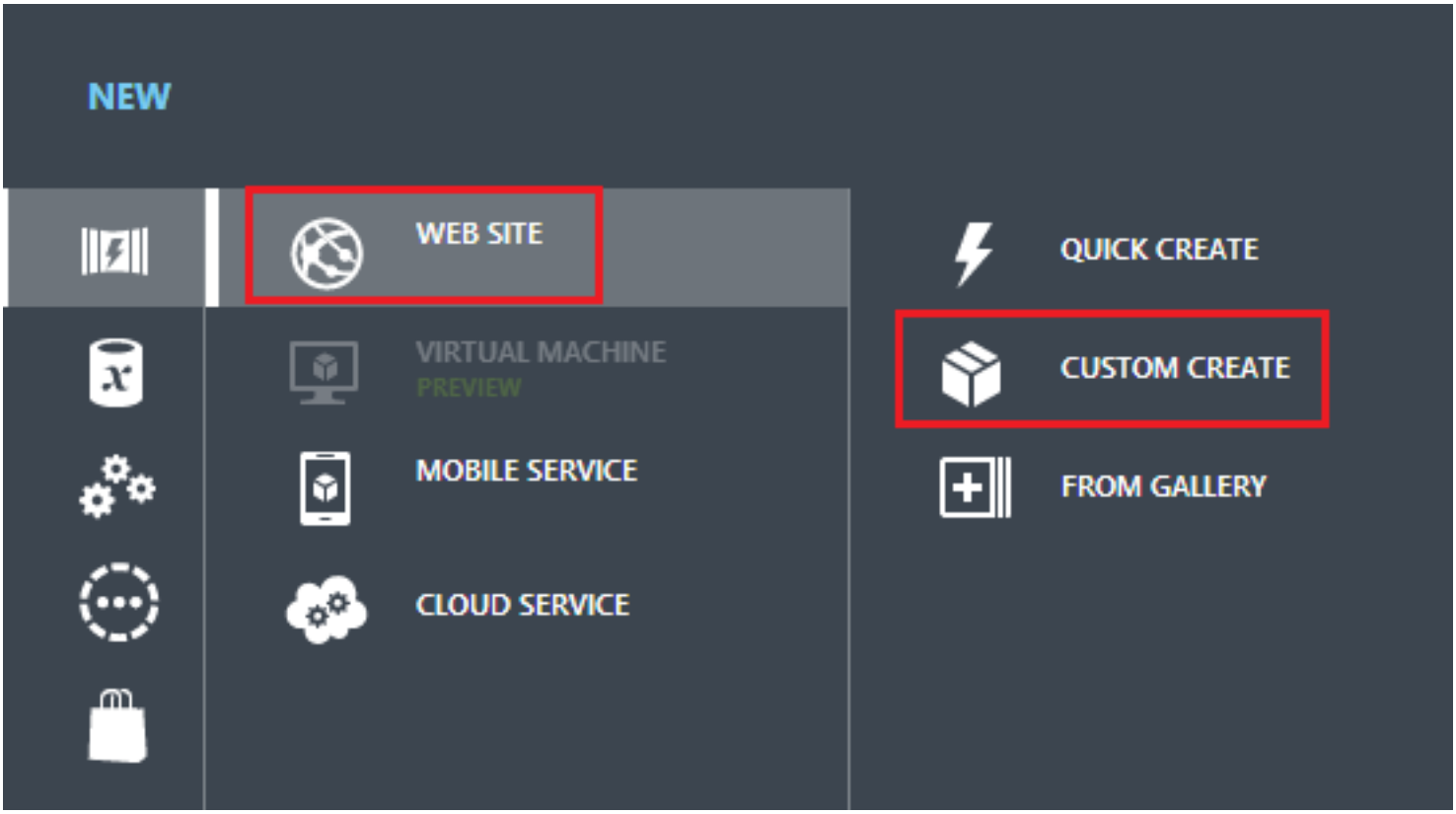
وب سایت Windows Azure شما در یک محیط اشتراکی (shared) میزبانی می‌شود، و این بدین معنا است که وب سایت‌های شما روی ماشین‌های مجازی (virtual machines) اجرا می‌شوند که با مشتریان دیگر Windows Azure به اشتراک گذاشته شده اند. یک محیط میزبانی اشتراکی گزینه ای کم هزینه برای شروع کار با رایانش‌های ابری است. اگر در آینده ترافیک وب سایت شما رشد چشم گیری داشته باشد، می‌توانید اپلیکیشن خود را طوری توسعه دهید که به نیازهای جدید پاسخگو باشد و آن را روی یک ماشین مجازی اختصاصی (dedicated VMs) میزبانی کنید. اگر معماری پیچیده‌تری نیاز دارید، می‌توانید به یک سرویس Windows Azure Cloud مهاجرت کنید. سرویس‌های ابری روی ماشین‌های مجازی اختصاصی اجرا می‌شوند که شما می‌توانید تنظیمات آنها را بر اساس نیازهای خود پیکربندی کنید.

Windows Azure SQL Database یک سرویس دیتابیس رابطه ای (relational) و مبتنی بر Cloud است که بر اساس تکنولوژی‌های SQL Server ساخته شده. ابزار و اپلیکیشن هایی که با SQL Server کار می‌کنند با SQL Database نیز می‌توانند کار کنند.

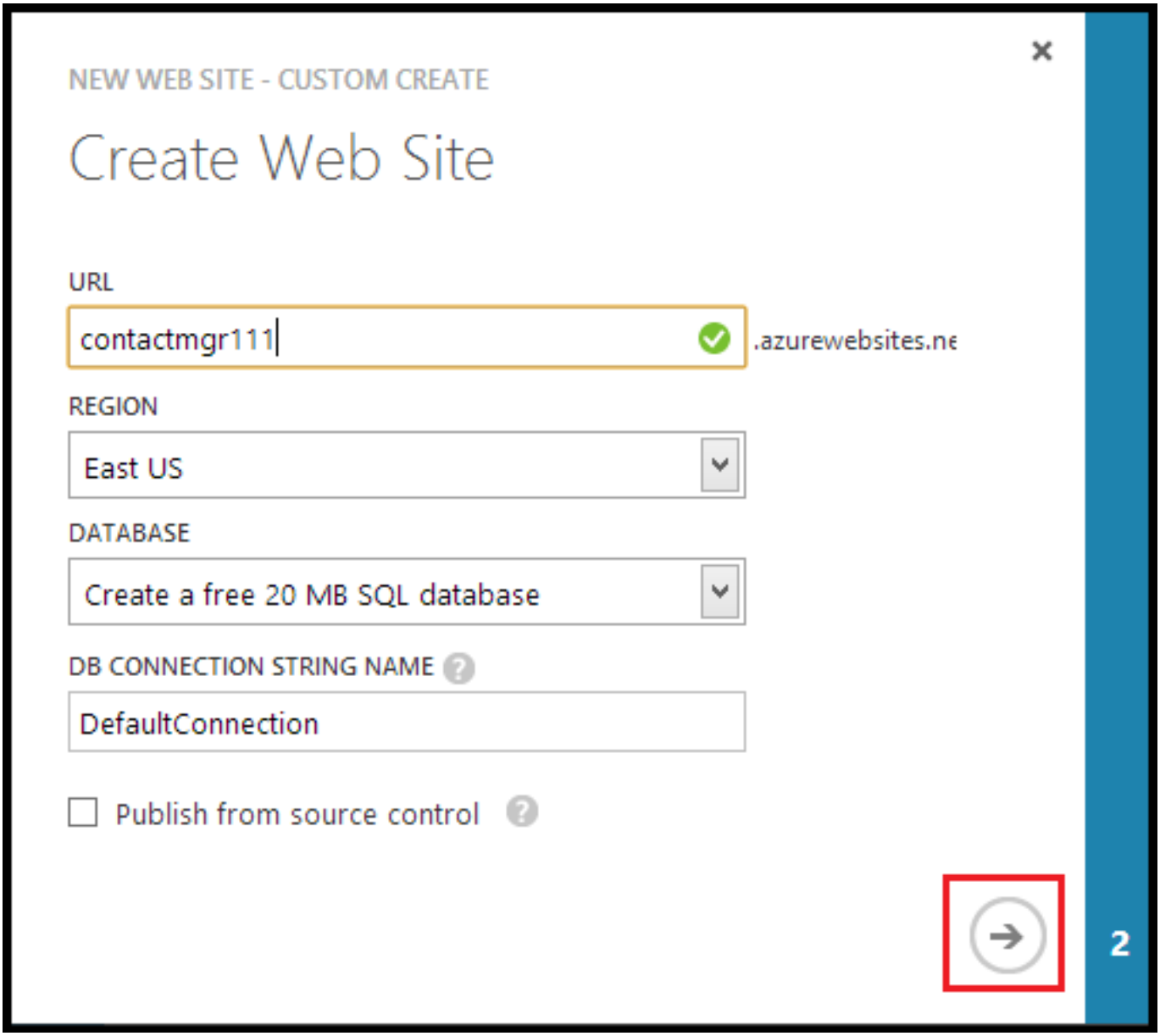
در پرتال مدیریتی Windows Azure روی **Web Sites** در قسمت چپ صفحه کلیک کنید، و گزینه **New** را برگزینید.



روی **Web Site** و سپس **Custom Create** کلیک کنید.



در مرحله **Create Web Site** در قسمت **URL** یک رشته وارد کنید که آدرسی منحصر بفرد برای اپلیکیشن شما خواهد بود. آدرس کامل وب سایت شما، ترکیبی از مقدار این فیلد و مقدار روبروی آن است.



در لیست **Database** گزینه **Create a free 20 MB SQL Database** را انتخاب کنید.

در لیست **Region** همان مقداری را انتخاب کنید که برای وب سایت تان انتخاب کرده اید. تنظیمات این قسمت مشخص می‌کند که ماشین مجازی (VM) شما در کدام مرکز داده (data center) خواهد بود.

در قسمت **DB Connection String Name** مقدار پیش فرض *DefaultConnection* را بپذیرید.

دکمه فلش پایین صفحه را کلیک کنید تا به مرحله بعد، یعنی مرحله **Specify Database Settings** بروید.

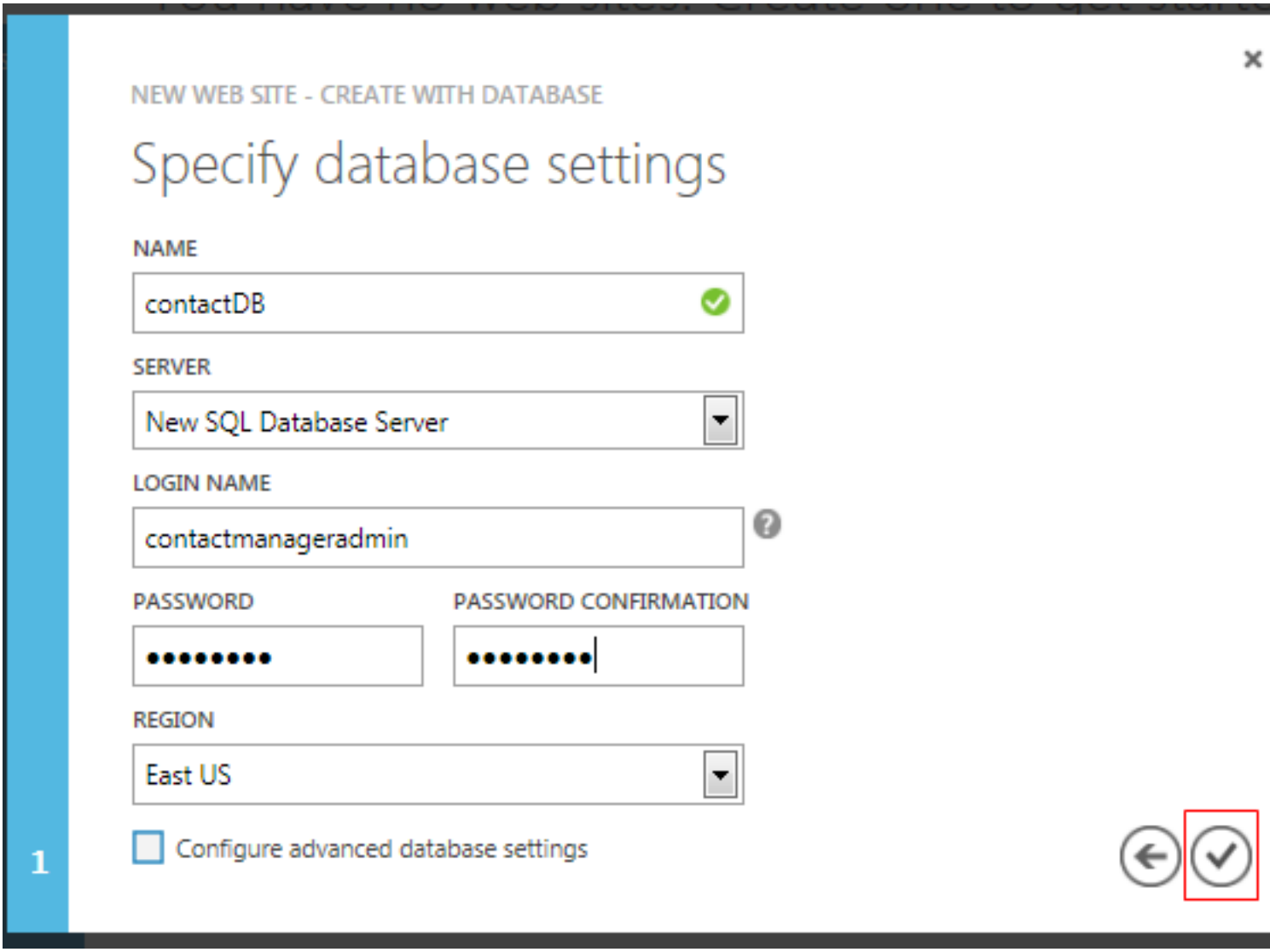
در قسمت **Name** مقدار *ContactDB* را وارد کنید (تصویر زیر).

در قسمت **Server** گزینه **New SQL Database Server** را انتخاب کنید. اگر قبلا دیتابیس ساخته اید می‌توانید آن را از کنترل dropdown انتخاب کنید.

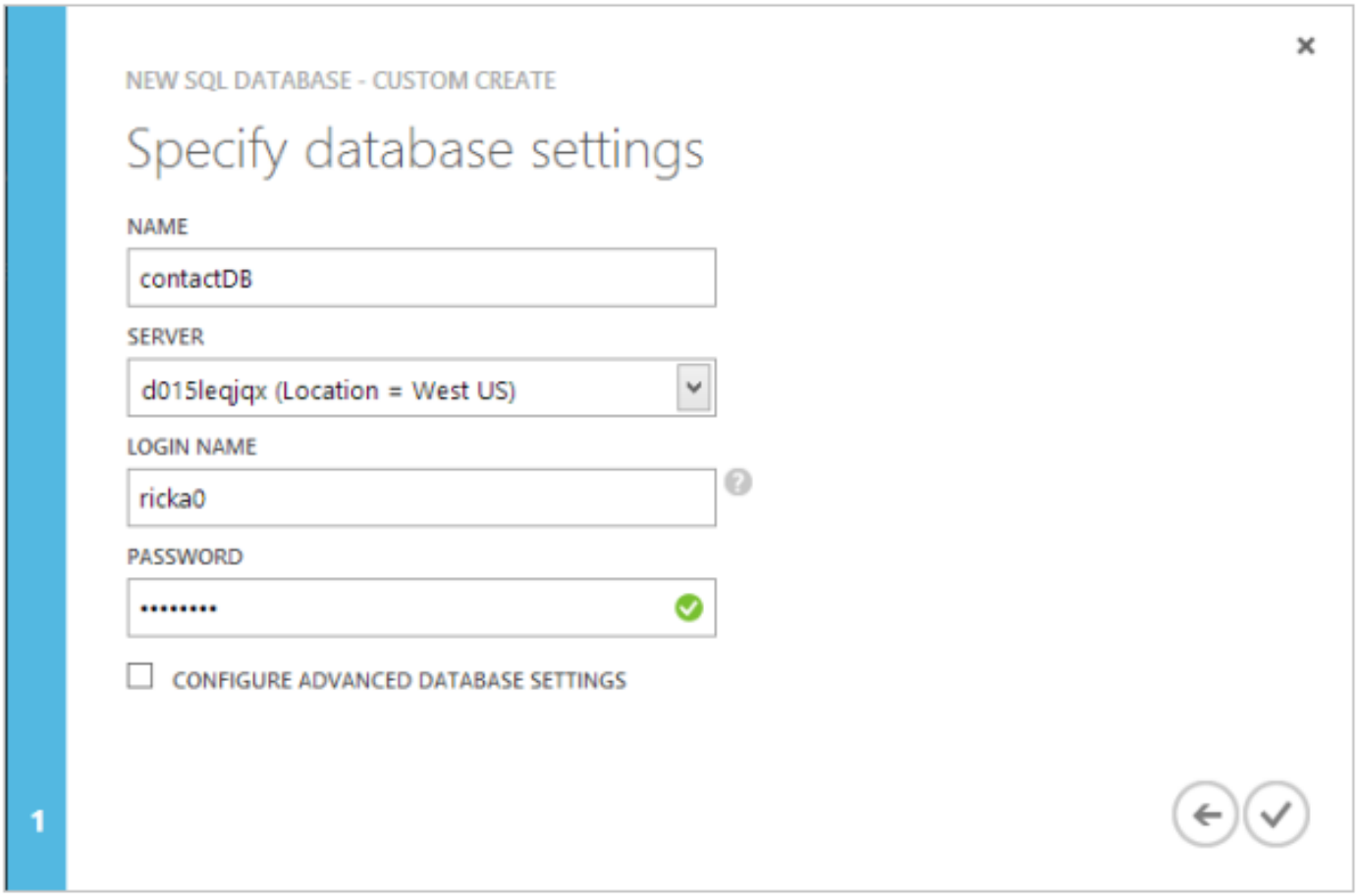
مقدار قسمت **Region** را به همان مقداری که برای ایجاد وب سایت تان تنظیم کرده اید تغییر دهید.

یک **Login Name** و **Password** مدیر (administrator) وارد کنید. اگر گزینه **New SQL Database server** را انتخاب کرده اید، چنین کاربری وجود ندارد و در واقع اطلاعات یک حساب کاربری جدید را وارد می‌کنید تا بعدا هنگام دسترسی به دیتابیس از آن استفاده کنید. اگر دیتابیس دیگری را از لیست انتخاب کرده باشید، اطلاعات یک حساب کاربری موجود از شما دریافت خواهد شد. در مثال این مقاله ما گزینه **Advanced** را رها می‌کنیم. همچنین در نظر داشته باشید که برای دیتابیس‌های رایگان تنها از یک Collation می‌توانید استفاده کنید.

دکمه تایید پایین صفحه را کلیک کنید تا مراحل تمام شود.



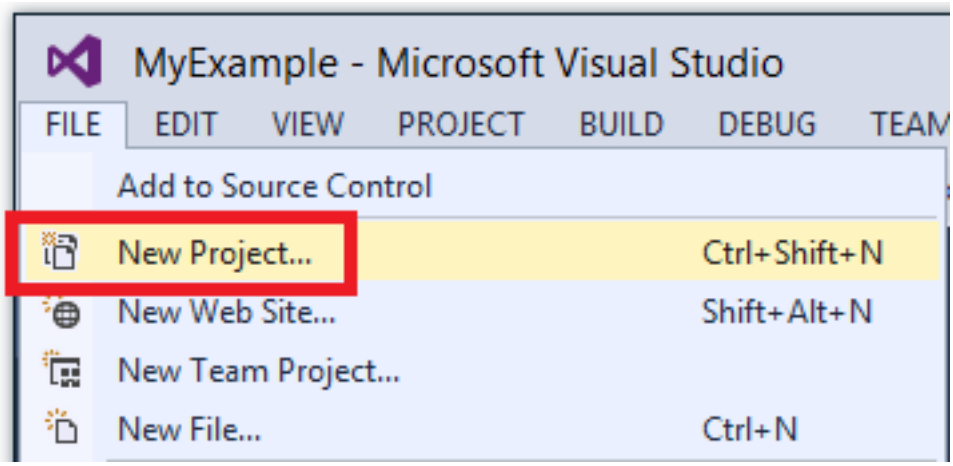
تصویر زیر استفاده از یک SQL Server و حساب کاربری موجود (existing) را نشان می‌دهد.



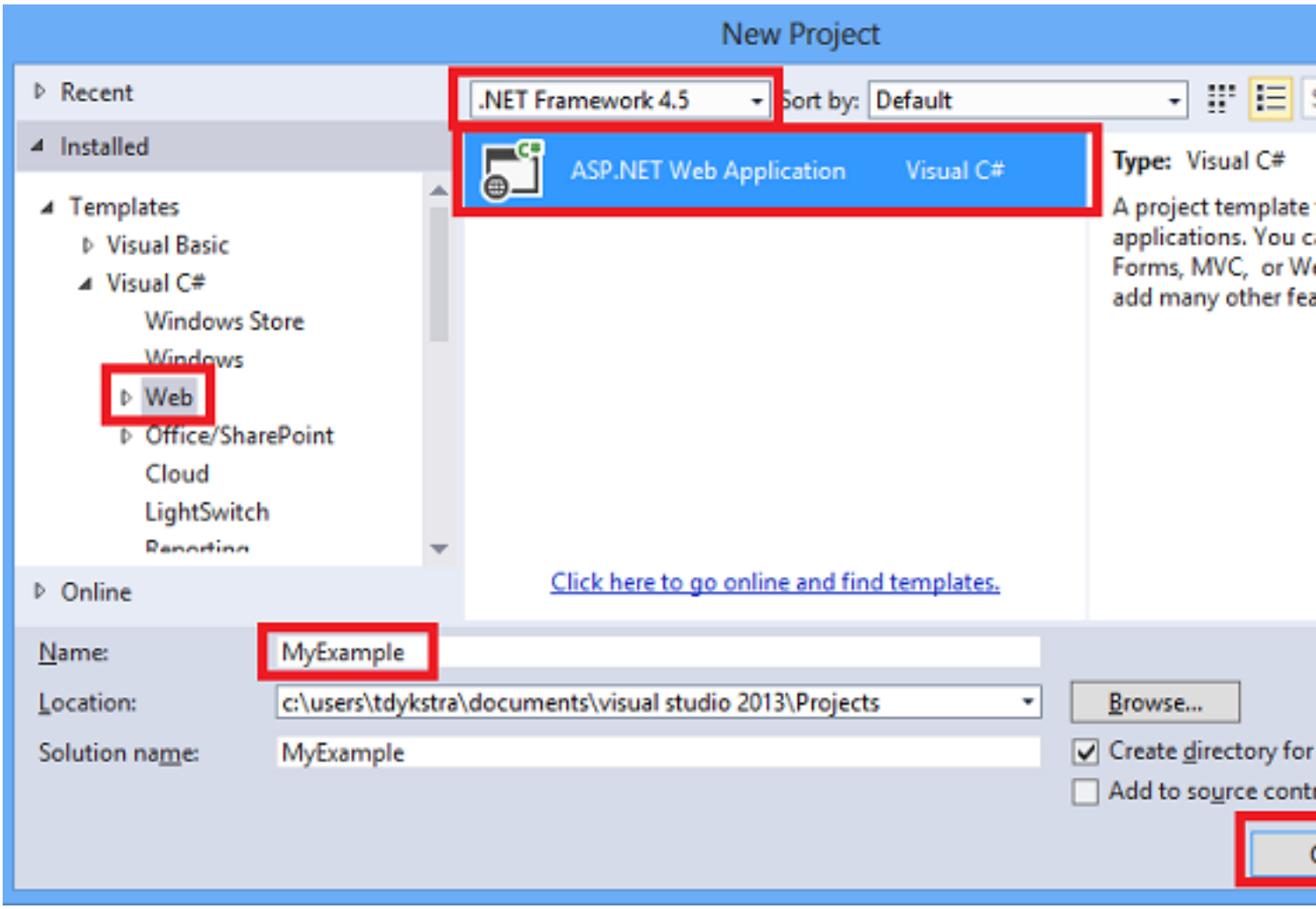
پرتال مدیریتی پس از اتمام مراحل، به صفحه وب سایت‌ها باز می‌گردد. ستون **Status** نشان می‌دهد که سایت شما در حال ساخته شدن است. پس از مدتی (معمولا کمتر از یک دقیقه) این ستون نشان می‌دهد که سایت شما با موفقیت ایجاد شده. در منوی پیمایش سمت چپ، تعداد سایت هایی که در اکانت خود دارید در کنار آیکون **Web Sites** نمایش داده شده است، تعداد دیتابیس‌ها نیز در کنار آیکون **SQL Databases** نمایش داده می‌شود.

یک اپلیکیشن ASP.NET MVC 5 بسازید

شما یک وب سایت Windows Azure ساختید، اما هنوز هیچ محتوایی در آن وجود ندارد. قدم بعدی ایجاد یک اپلیکیشن وب در ویژوال استودیو و انتشار آن است. ابتدا یک پروژه جدید بسازید.

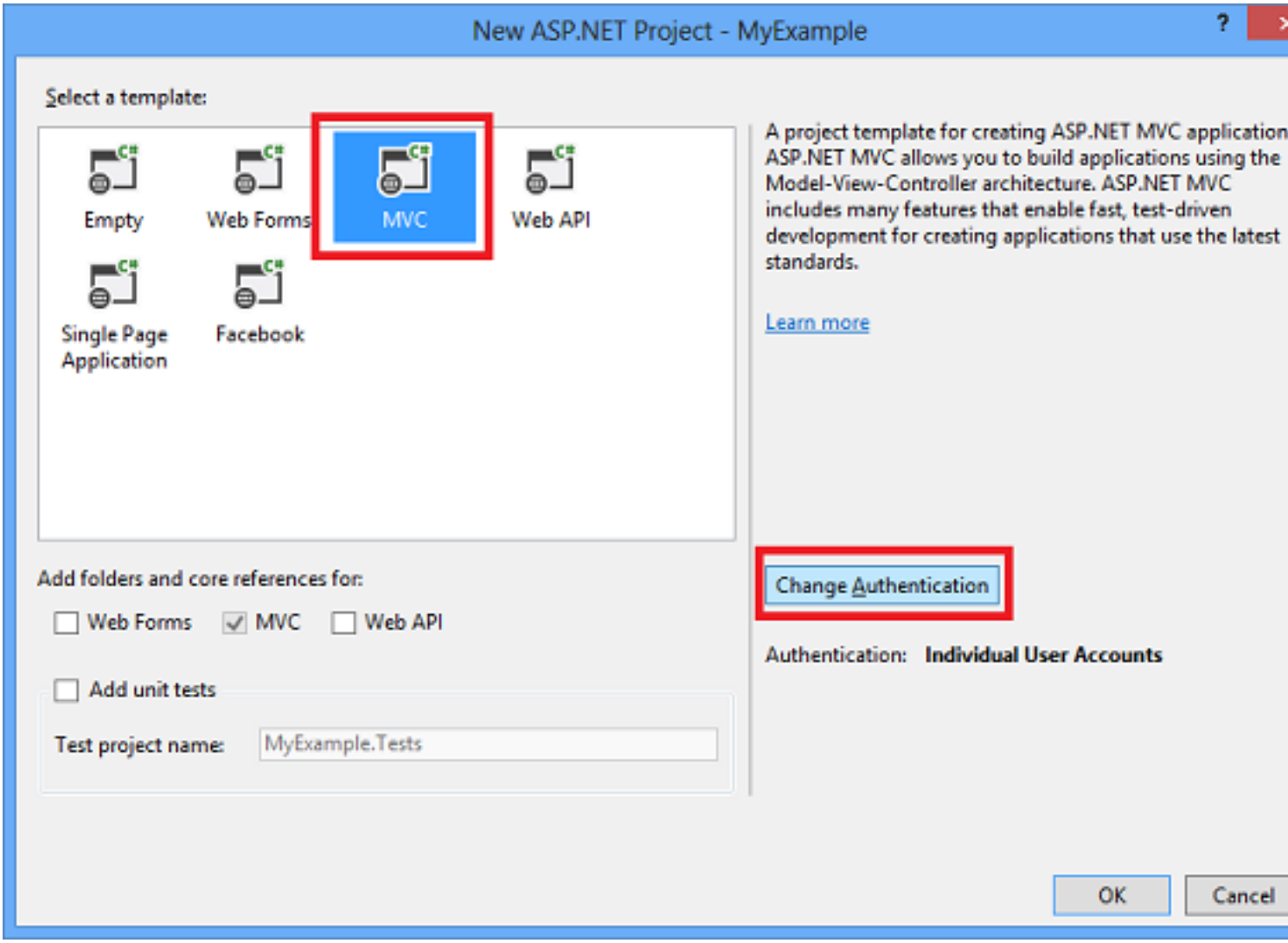


نوع پروژه را **ASP.NET Web Application** انتخاب کنید.



نکته: در تصویر بالا نام پروژه "MyExample" است اما حتما نام پروژه خود را به "ContactManager" تغییر دهید. قطعه کدهایی که در ادامه مقاله خواهید دید نام پروژه را ContactManager فرض می‌کنند.

در دیالوگ جدید ASP.NET نوع اپلیکیشن را **MVC** انتخاب کنید و دکمه **Change Authentication** را کلیک کنید.



گزینه پیش فرض **Individual User Accounts** را بپذیرید. برای اطلاعات بیشتر درباره متدهای دیگر احراز هویت به این لینک مراجعه کنید. دکمه‌های OK را کلیک کنید تا تمام مراحل تمام شوند.

تنظیم تیتر و پاورقی سایت

فایل *_Layout.cshtml* را باز کنید. دو نمونه از متن "My ASP.NET MVC Application" را با عبارت "Contact Manager" جایگزین کنید.

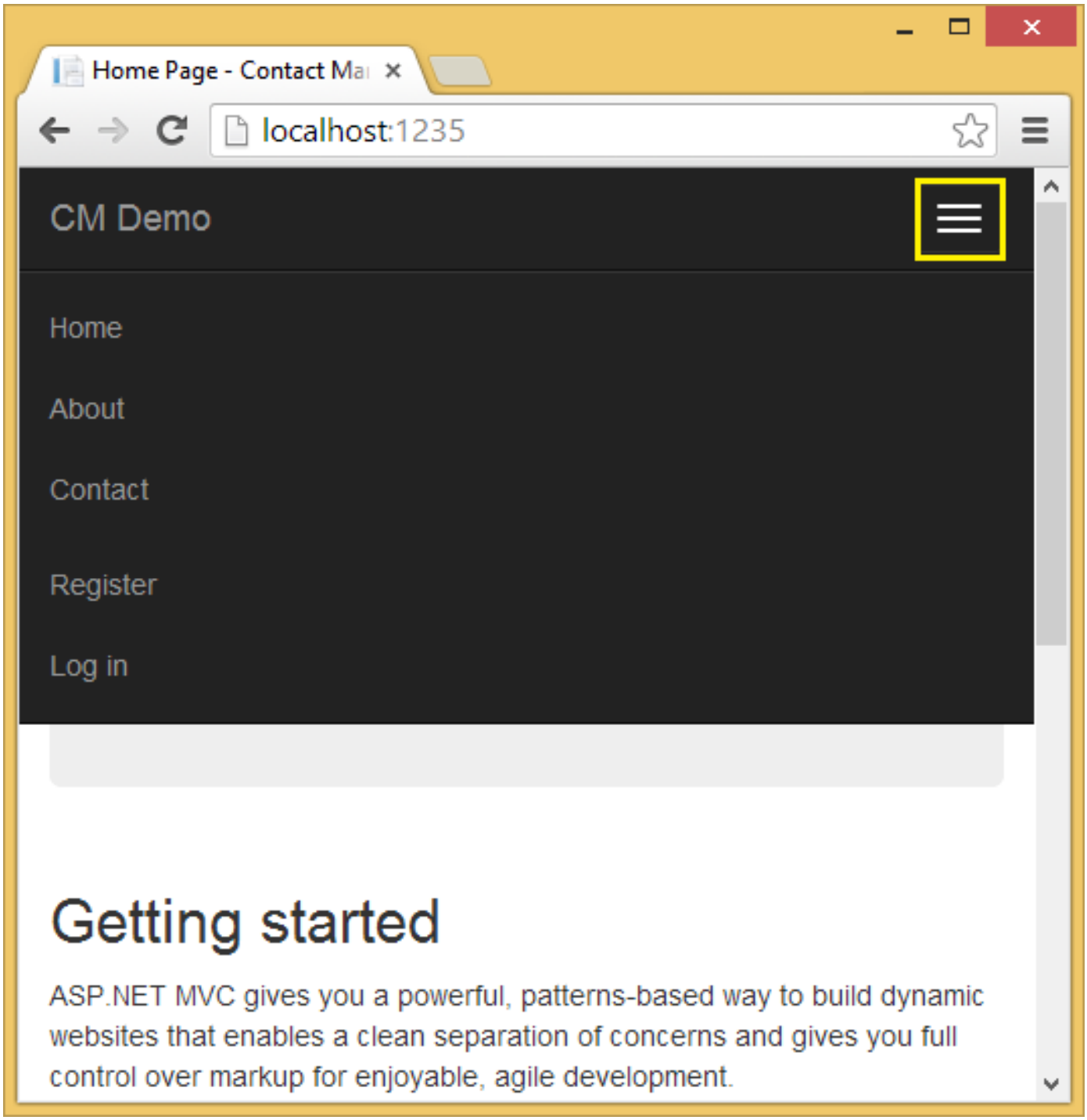
عبارت "Application name" را هم با "CM Demo" جایگزین کنید.

اولین Action Link را ویرایش کنید و مقدار *Home* را با *Cm* جایگزین کنید تا از *CmController* استفاده کند.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title - Contact Manager</title>
    @Styles.Render("~/Content/css")
    @Scripts.Render("~/bundles/modernizr")

</head>
<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="colla
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                @Html.ActionLink("CM Demo", "Index", "Cm", null, new { @class
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li>@Html.ActionLink("Home", "Index", "Home")</li>
                    <li>@Html.ActionLink("About", "About", "Home")</li>
                    <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
                </ul>
                @Html.Partial("_LoginPartial")
            </div>
        </div>
    </div>
    <div class="container body-content">
        @RenderBody()
        <hr />
        <footer>
            <p>&copy; @DateTime.Now.Year - Contact Manager</p>
        </footer>
    </div>

    @Scripts.Render("~/bundles/jquery")
    @Scripts.Render("~/bundles/bootstrap")
    @RenderSection("scripts", required: false)
</body>
</html>
```

اپلیکیشن را بصورت محلی اجرا کنید

اپلیکیشن را با Ctrl + F5 اجرا کنید. صفحه اصلی باید در مرورگر پیش فرض باز شود.

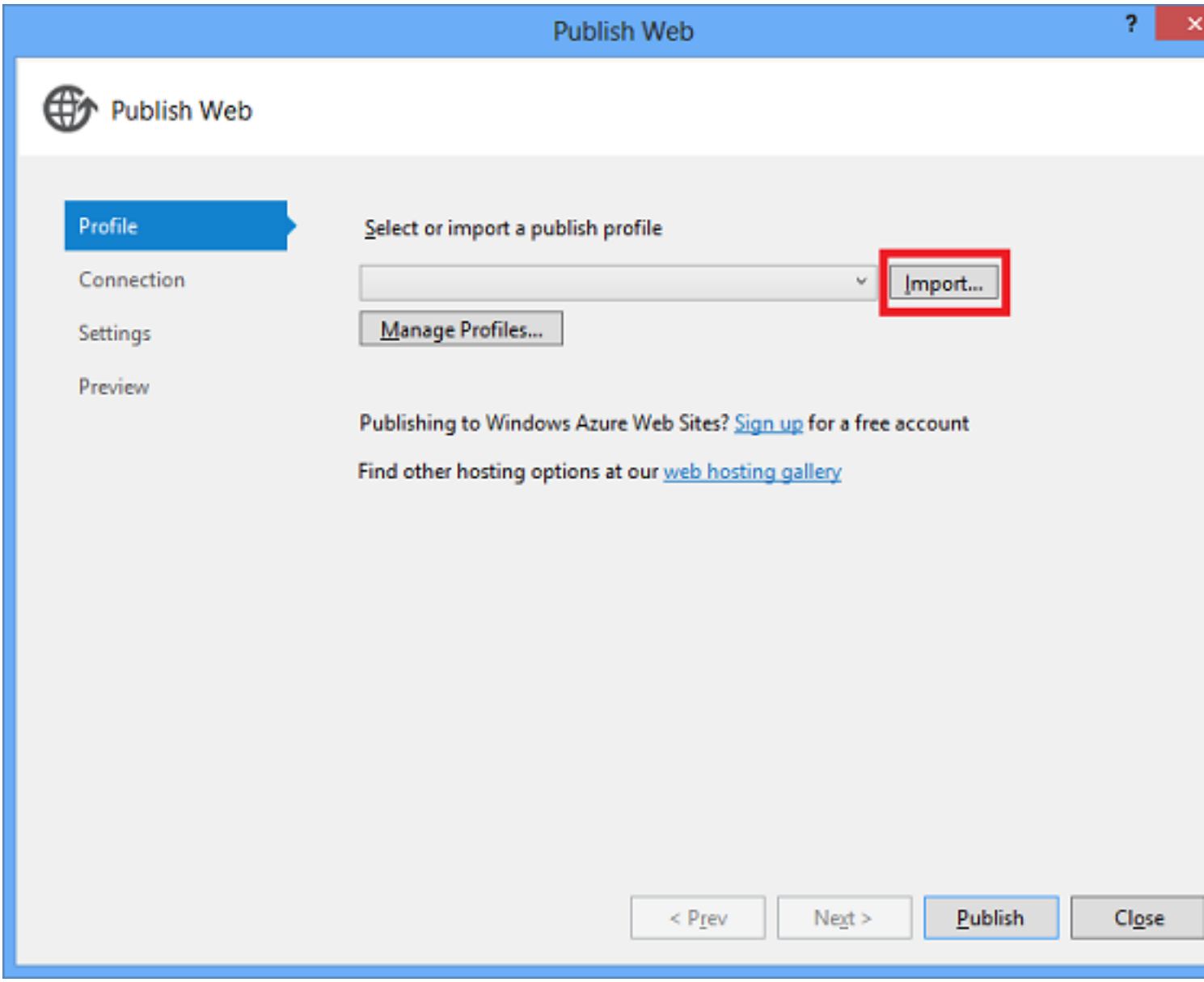


اپلیکیشن شما فعلا آماده است و می‌توانید آن را روی Windows Azure توزیع کنید. بعدا دیتابیس و دسترسی داده نیز اضافه خواهد شد.

اپلیکیشن را روی Windows Azure منتشر کنید

در ویژوال استودیو روی نام پروژه کلیک راست کنید و گزینه **Publish** را انتخاب کنید. ویزارد **Publish Web** باز می‌شود.

در قسمت **Profile** روی **Import** کلیک کنید.



حال دیالوگ **Import Publish Profile** نمایش داده می‌شود.

یکی از متدهای زیر را استفاده کنید تا ویژوال استودیو بتواند به اکانت Windows Azure شما متصل شود.
روی **Sign In** کلیک کنید تا با وارد کردن اطلاعات حساب کاربری وارد Windows Azure شوید.

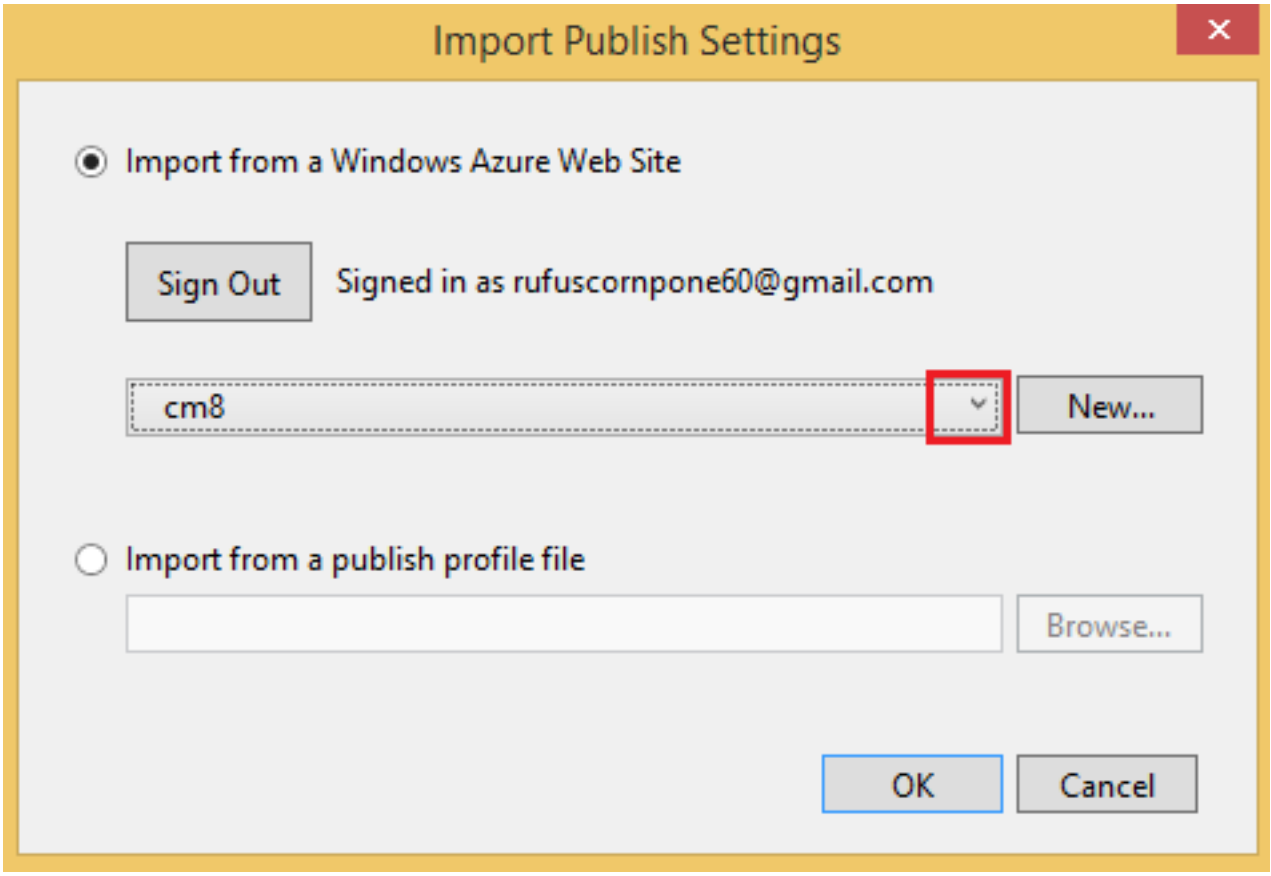
این روش ساده‌تر و سریع‌تر است، اما اگر از آن استفاده کنید دیگر قادر به مشاهده Windows Azure SQL Database یا Mobile Services در پنجره **Server Explorer** نخواهید بود.
روی **Manage subscriptions** کلیک کنید تا یک management certificate نصب کنید، که دسترسی به حساب کاربری شما را ممکن می‌سازد.

در دیالوگ باکس **Manage Windows Azure Subscriptions** به قسمت **Certificates** بروید. سپس **Import** را کلیک کنید. مراحل را دنبال کنید تا یک فایل subscription را بصورت دانلود دریافت کنید (فایل‌های *publishsettings*.) که اطلاعات اکانت Windows Azure شما را دارد.

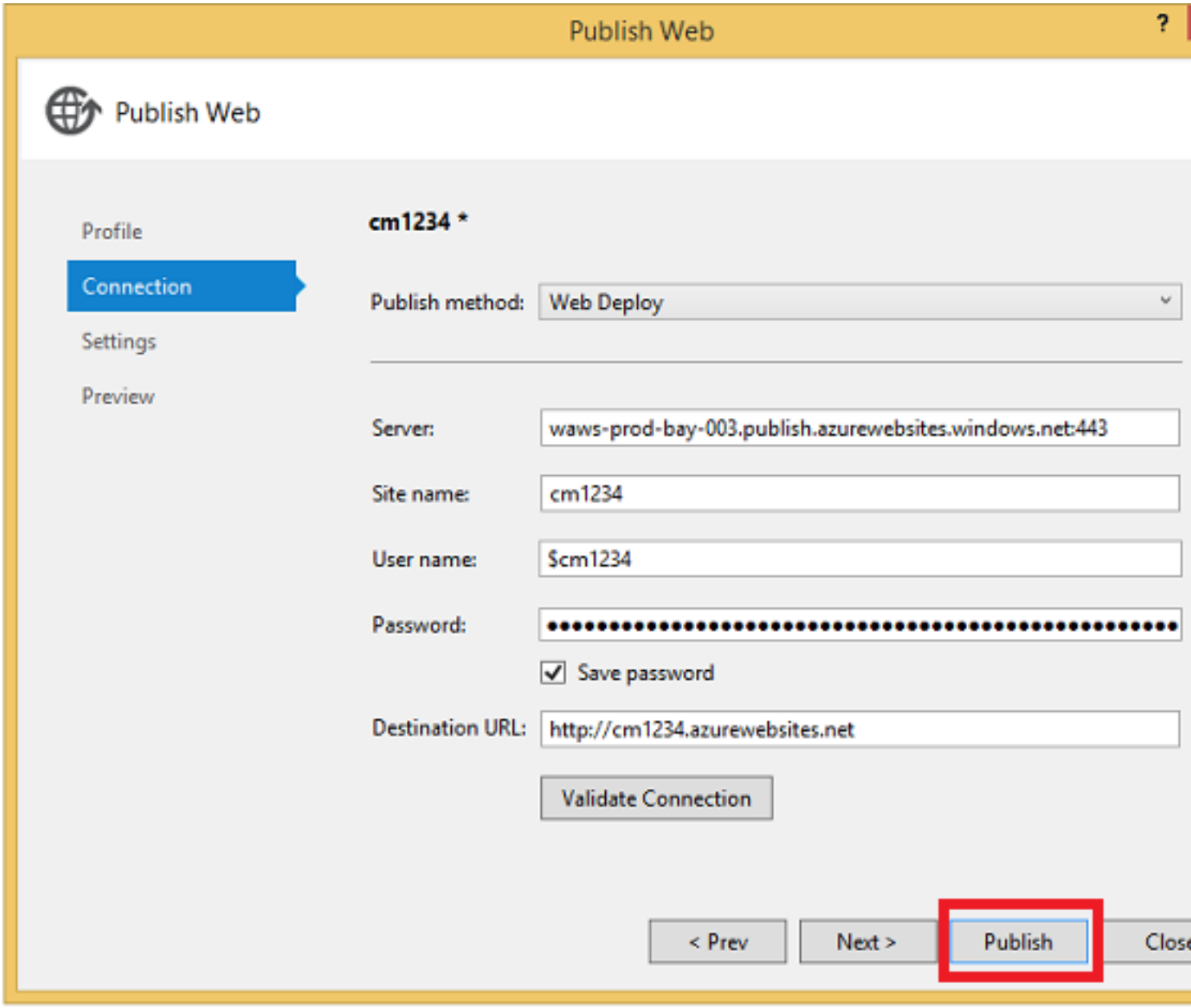
نکته امنیتی: این فایل تنظیمات را بیرون از پوشه‌های سورس کد خود دانلود کنید، مثلا پوشه Downloads. پس از اتمام عملیات Import هم این فایل را حذف کنید. کاربر مخربی که به این فایل دسترسی پیدا کند قادر خواهد بود تا سرویس‌های Windows Azure شما را کاملا کنترل کند.

برای اطلاعات بیشتر به How to Connect to Windows Azure from Visual Studio مراجعه کنید.

در دیالوگ باکس **Import Publish Profile** وب سایت خود را از لیست انتخاب کنید و OK را کلیک کنید.



در دیالوگ باکس **Publish Web** روی **Publish** کلیک کنید.



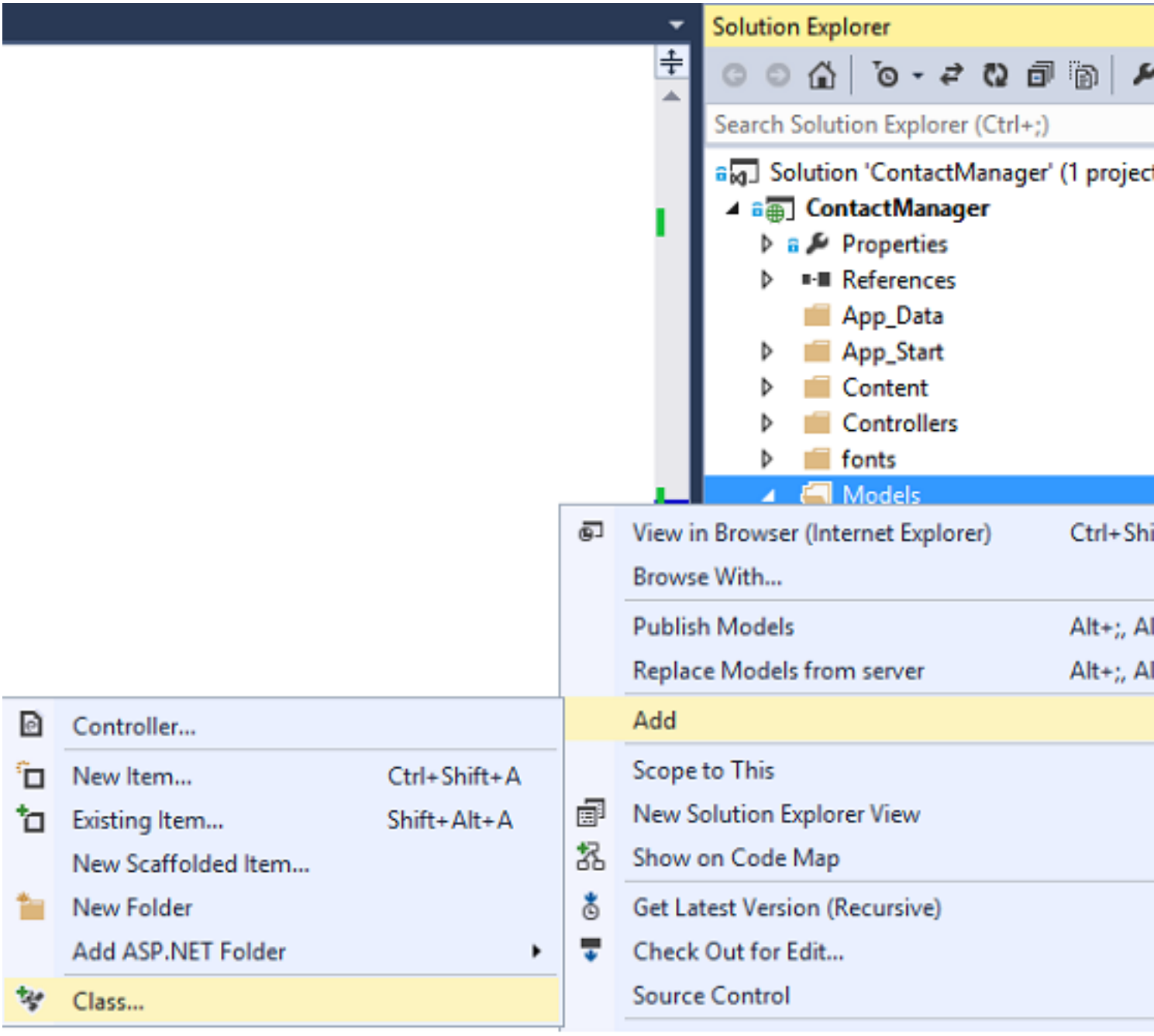
اپلیکیشن شما حالا در فضای ابری اجرا می‌شود. دفعه بعد که اپلیکیشن را منتشر کنید تنها فایل‌های تغییر کرده (یا جدید) آپلود خواهند شد.

یک دیتابیس به اپلیکیشن اضافه کنید

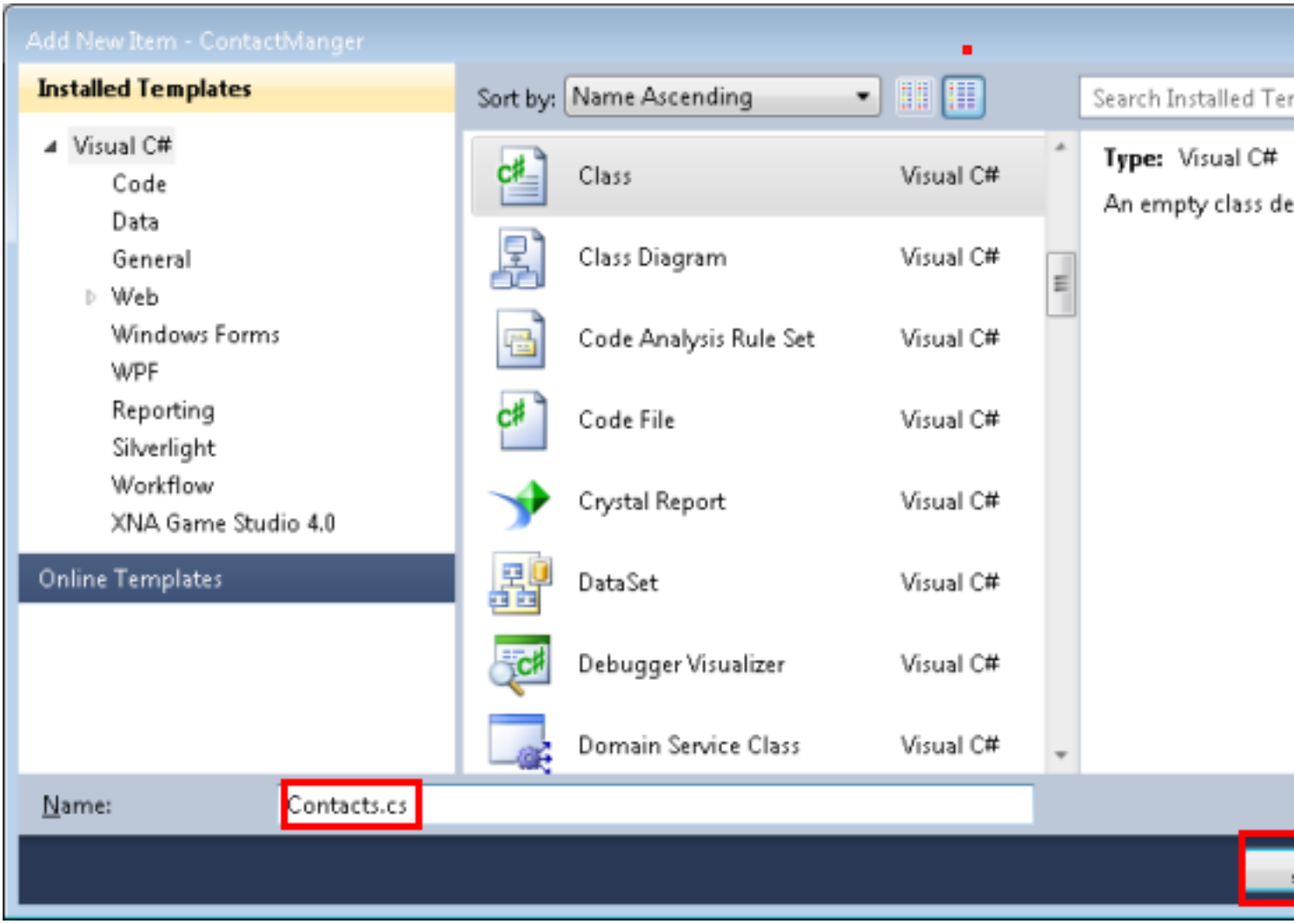
در مرحله بعد یک دیتابیس خواهیم ساخت تا اپلیکیشن ما بتواند اطلاعات را نمایش دهد و ویرایش کند. برای ایجاد دیتابیس و دسترسی به داده‌ها از Entity Framework استفاده خواهیم کرد.

کلاس‌های مدل Contacts را اضافه کنید

در پوشه Models پروژه یک کلاس جدید ایجاد کنید.



نام کلاس را به *Contact.cs* تغییر دهید و دکمه Add را کلیک کنید.



کد فایل Contact.cs را با قطعه کد زیر مطابقت دهید.

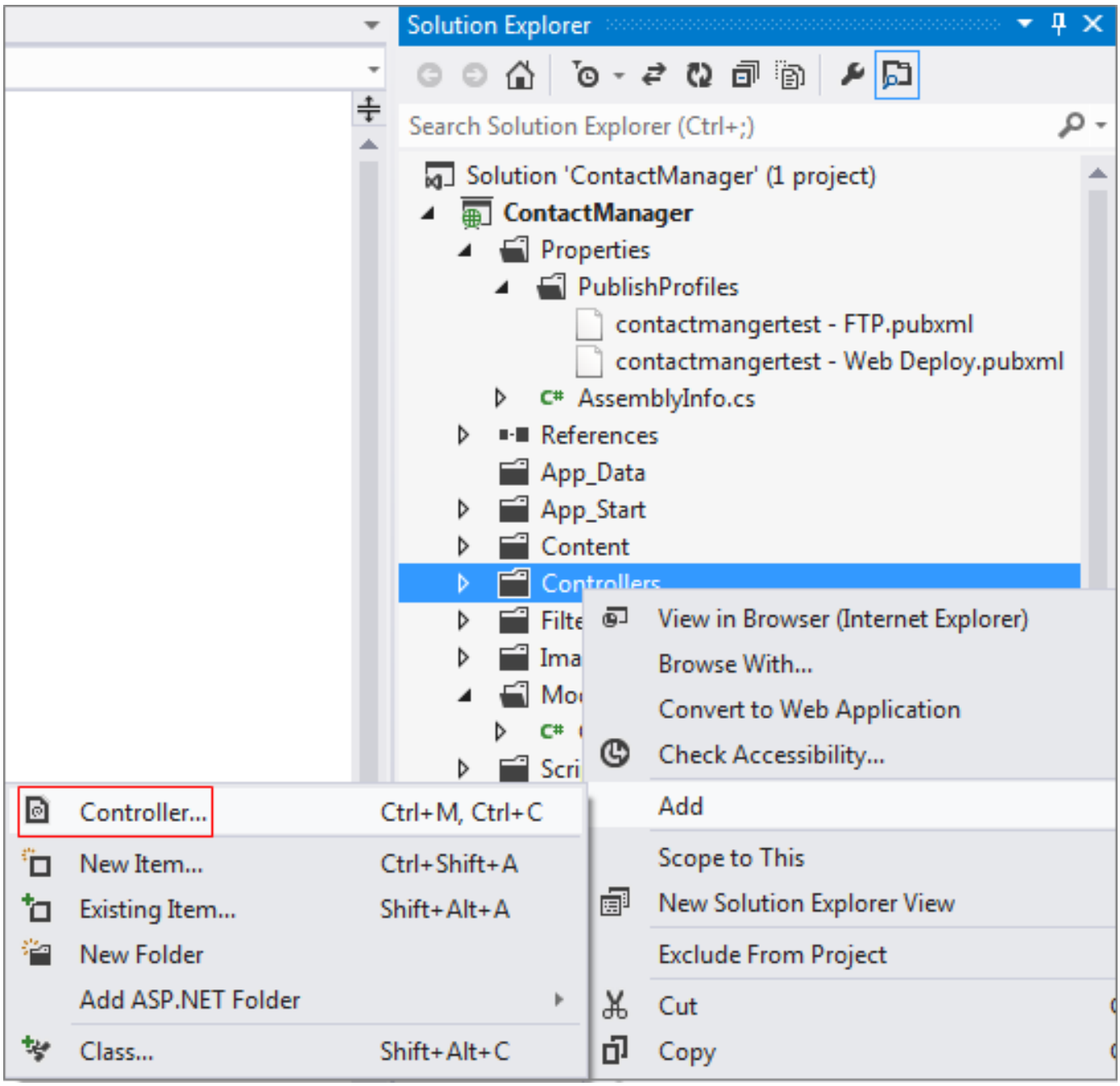
```
using System.ComponentModel.DataAnnotations;
using System.Globalization;
namespace ContactManager.Models
{
    public class Contact
    {
        public int ContactId { get; set; }
        public string Name { get; set; }
        public string Address { get; set; }
        public string City { get; set; }
        public string State { get; set; }
        public string Zip { get; set; }
        [DataType(DataType.EmailAddress)]
        public string Email { get; set; }
    }
}
```

این کلاس موجودیت Contact را در دیتابیس معرفی می‌کند. داده هایی که می‌خواهیم برای هر رکورد ذخیره کنیم تعریف شده اند، بعلاوه یک فیلد Primary Key که دیتابیس به آن نیاز دارد.

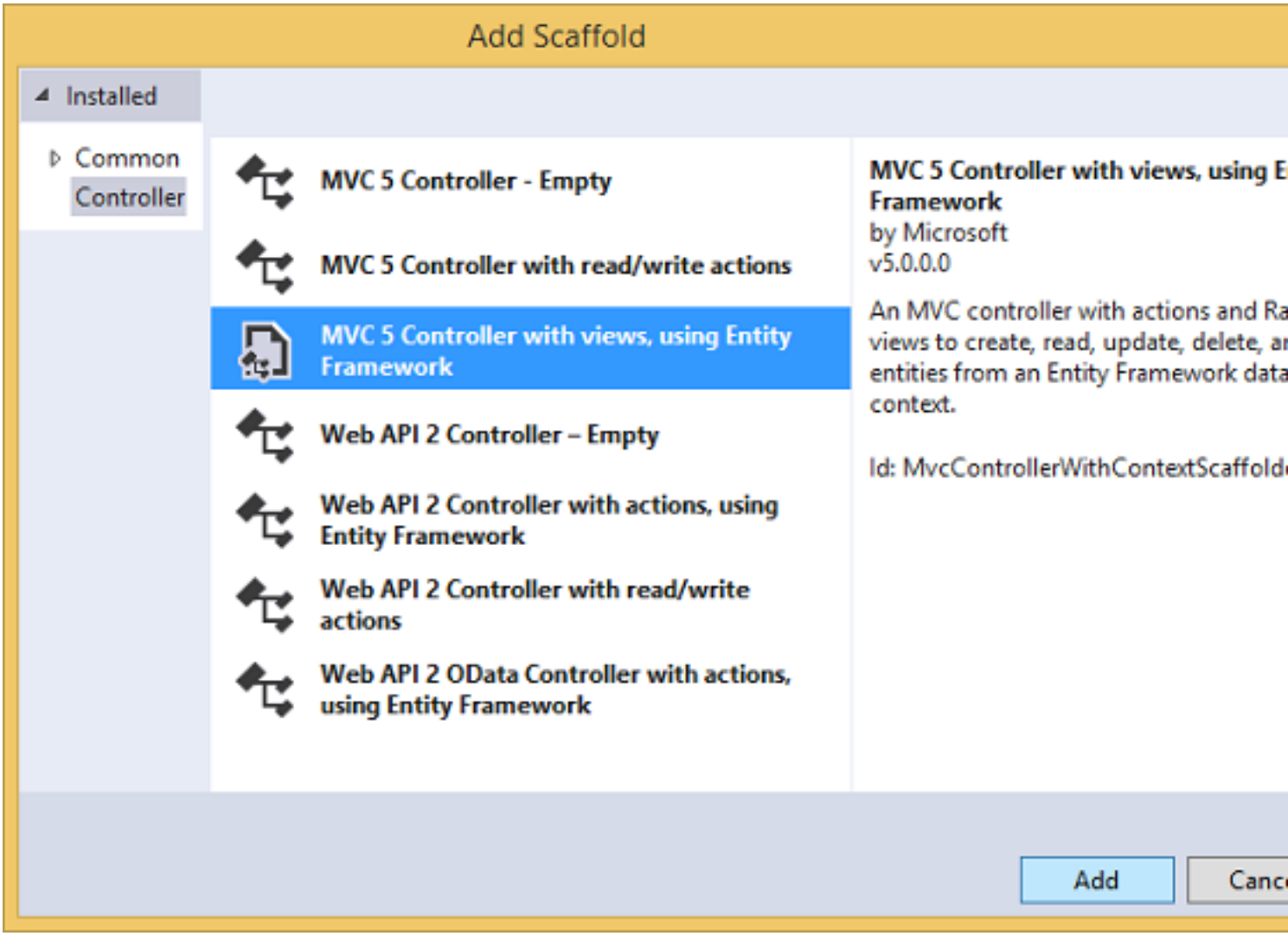
یک کنترلر و نما برای داده‌ها اضافه کنید

ابتدا پروژه را Build کنید (Ctrl + Shift+ B). این کار را باید پیش از استفاده از مکانیزم Scaffolding انجام دهید.

یک کنترلر جدید به پوشه Controllers اضافه کنید.



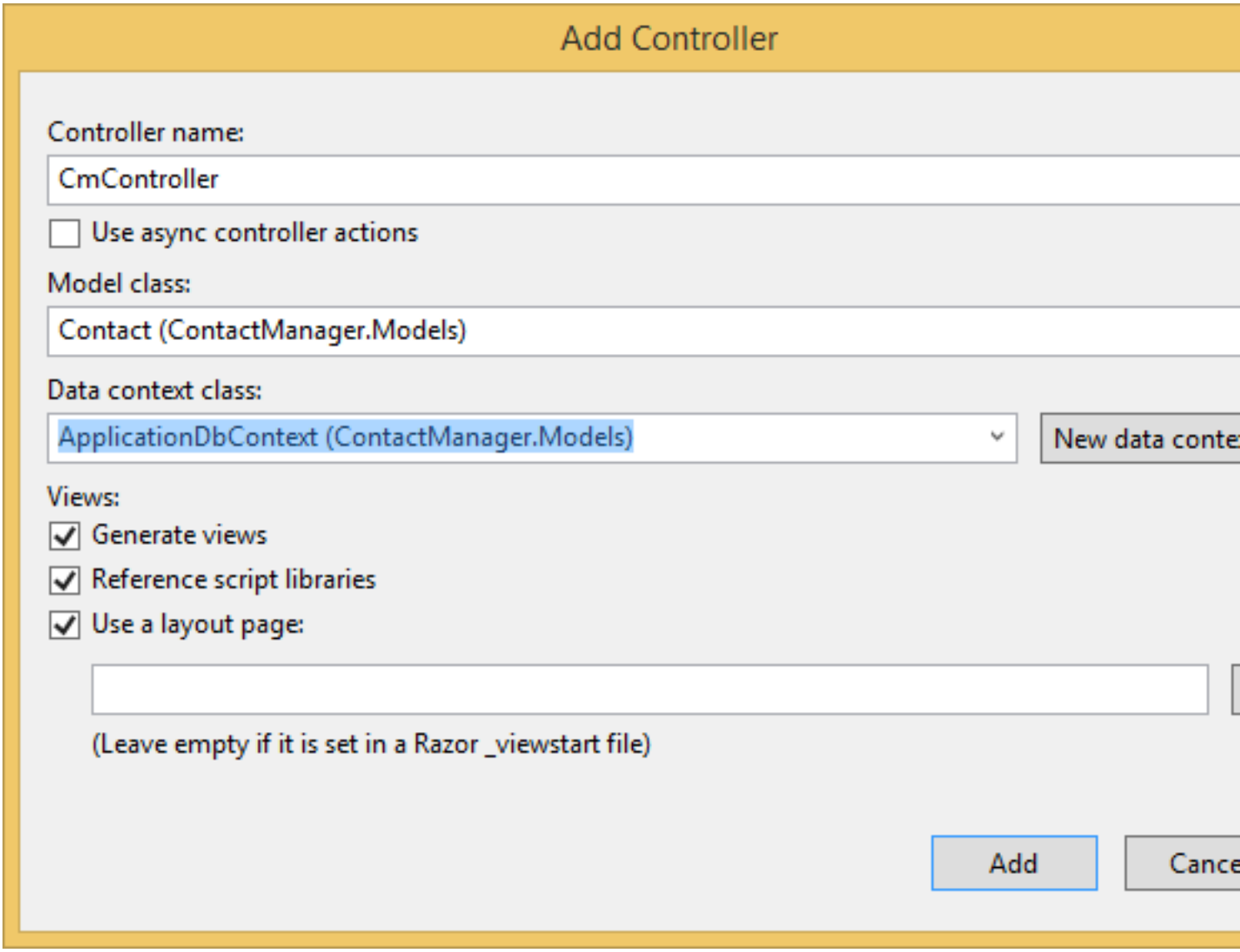
در دیالوگ باکس **Add Scaffold** گزینه **MVC 5 Controller with views, using EF** را انتخاب کنید.



در دیالوگ **Add Controller** نام "CmController" را برای کنترلر وارد کنید. (تصویر زیر.)

در لیست **Model** گزینه **Contact (ContactManager.Models)** را انتخاب کنید.

در قسمت **Data context class** گزینه **ApplicationDbContext (ContactManager.Models)** را انتخاب کنید. این **ApplicationDbContext** هم برای اطلاعات سیستم عضویت و هم برای داده‌های Contacts استفاده خواهد شد.

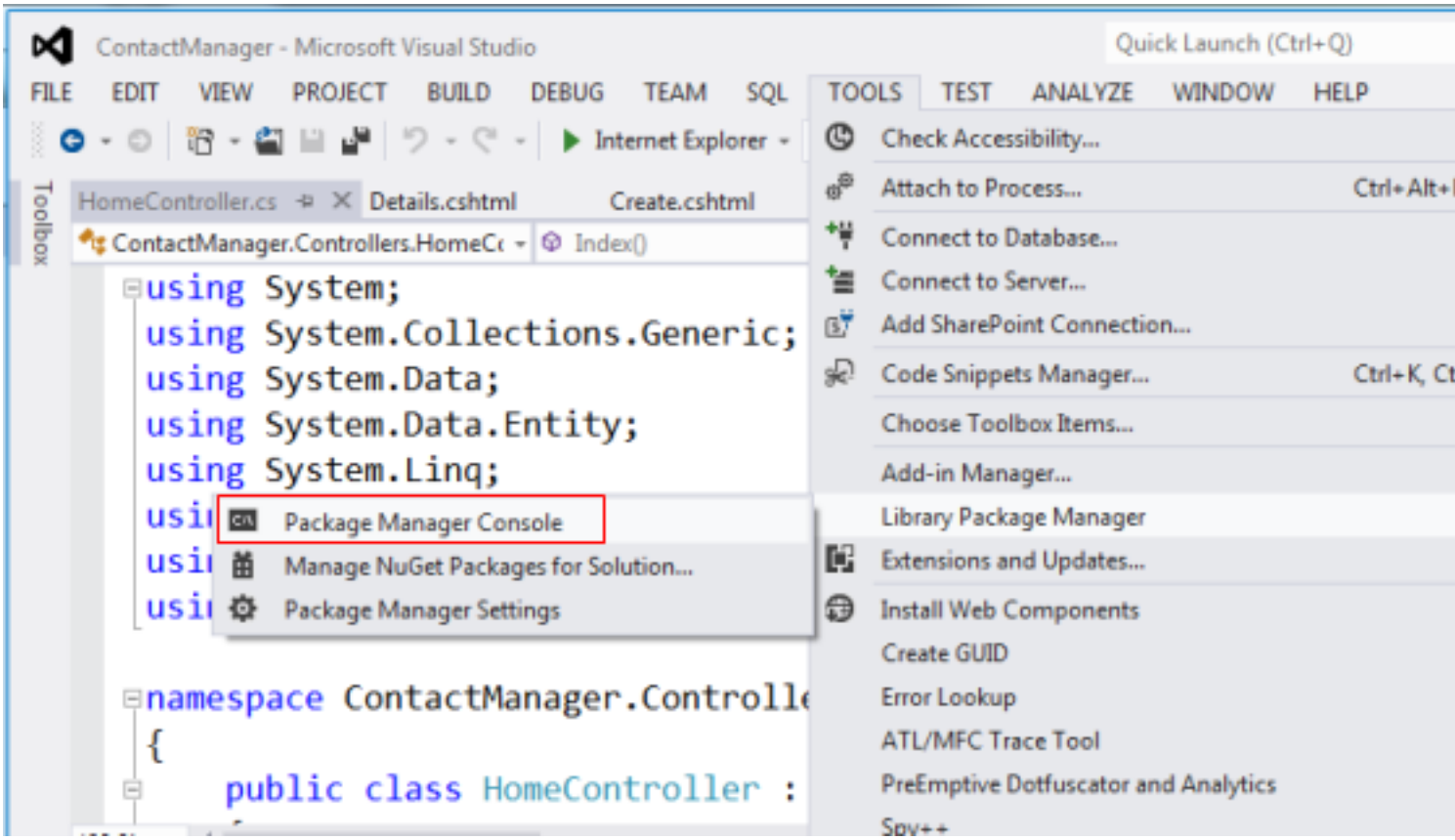


روی Add کلیک کنید. ویژوال استودیو بصورت خودکار با استفاده از Scaffolding متدها و Viewهای لازم برای عملیات CRUD را فراهم می‌کند، که همگی از مدل Contact استفاده می‌کنند.

فعالسازی مهاجرت ها، ایجاد دیتابیس، افزودن داده نمونه و یک راه انداز

مرحله بعدی فعال کردن قابلیت Code First Migrations است تا دیتابیس را بر اساس الگویی که تعریف کرده اید بسازد.

از منوی **Tools** گزینه **Library Package Manager** و سپس **Package Manager Console** را انتخاب کنید.



در پنجره باز شده فرمان زیر را وارد کنید.

```
enable-migrations
```

فرمان **enable-migrations** یک پوشه با نام *Migrations* می سازد و فایلی با نام *Configuration.cs* را به آن اضافه می‌کند. با استفاده از این کلاس می‌توانید داده‌های اولیه دیتابیس را وارد کنید و مهاجرت‌ها را نیز پیکربندی کنید.

در پنجره **Package Manager Console** فرمان زیر را وارد کنید.

```
add-migration Initial
```

فرمان **add-migration initial** فایلی با نام **initial <data_stamp>** ساخته و آن را در پوشه *Migrations* ذخیره می‌کند. در این مرحله دیتابیس شما ایجاد می‌شود. در این فرمان، مقدار **initial** اختیاری است و صرفا برای نامگذاری فایل مهاجرت استفاده شده. فایل‌های جدید را می‌توانید در **Solution Explorer** مشاهده کنید.

در کلاس **Initial** متد **Up** جدول Contacts را می‌سازد. و متد **Down** (هنگامی که می‌خواهید به وضعیت قبلی بازگردید) آن را drop می‌کند.

حال فایل *Migrations/Configuration.cs* را باز کنید. فضای نام زیر را اضافه کنید.

```
using ContactManager.Models;
```

حال متد *Seed* را با قطعه کد زیر جایگزین کنید.

```
protected override void Seed(ContactManager.Models.ApplicationDbContext context)
{
    context.Contacts.AddOrUpdate(p => p.Name,
       new Contact
       {
           Name = "Debra Garcia",
           Address = "1234 Main St",
           City = "Redmond",
           State = "WA",
           Zip = "10999",
           Email = "debra@example.com",
       },
        new Contact
        {
            Name = "Thorsten Weinrich",
            Address = "5678 1st Ave W",
            City = "Redmond",
            State = "WA",
            Zip = "10999",
            Email = "thorsten@example.com",
        },
        new Contact
        {
            Name = "Yuhong Li",
            Address = "9012 State st",
            City = "Redmond",
            State = "WA",
            Zip = "10999",
            Email = "yuhong@example.com",
        },
        new Contact
        {
            Name = "Jon Orton",
            Address = "3456 Maple St",
            City = "Redmond",
            State = "WA",
            Zip = "10999",
            Email = "jon@example.com",
        },
        new Contact
        {
            Name = "Diliana Alexieva-Bosseva",
            Address = "7890 2nd Ave E",
            City = "Redmond",
            State = "WA",
            Zip = "10999",
            Email = "diliana@example.com",
        }
        );
}
```

این متد دیتابیس را Seed می‌کند، یعنی داده‌های پیش فرض و اولیه دیتابیس را تعریف می‌کند. برای اطلاعات بیشتر به Seeding and Debugging Entity Framework (EF) DBs مراجعه کنید.

در پنجره **Package Manager Console** فرمان زیر را وارد کنید.

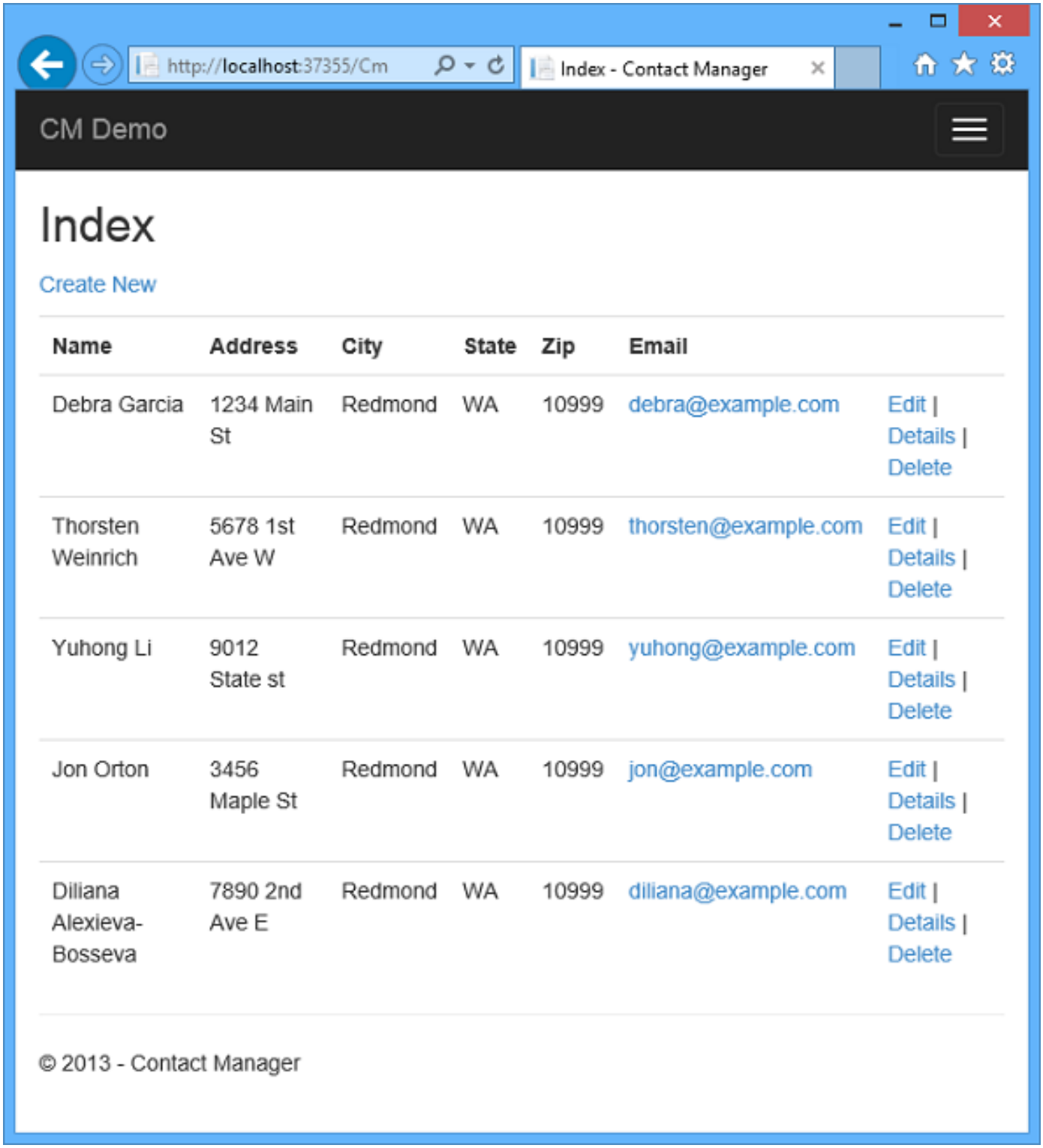
```
update-database
```

```
Package Manager Console
Package source: NuGet official package source
PM> enable-migrations
Checking if the context targets an existing database...
Code First Migrations enabled for project ContactManager.
PM> add-migration Initial
Scaffolding migration 'Initial'.
The Designer Code for this migration file includes a snapshot of your current Code Fi
 model. This snapshot is used to calculate the changes to your model when you scaffol
the next migration. If you make additional changes to your model that you want to
include in this migration, then you can re-scaffold it by running 'Add-Migration
201209182148203_Initial' again.
PM> update-database
Specify the '-Verbose' flag to view the SQL statements being applied to the target
database.
Applying code-based migrations: [201209182148203_Initial].
Applying code-based migration: 201209182148203_Initial.
Running Seed method.
PM>
100 %
```

فرمان **update-database** مهاجرت نخست را اجرا می‌کند، که دیتابیس را می‌سازد. بصورت پیش فرض این یک دیتابیس SQL Server Express LocalDB است.

حال پروژه را با CTRL + F5 اجرا کنید.

همانطور که مشاهده می‌کنید، اپلیکیشن داده‌های اولیه (Seed) را نمایش می‌دهد، و لینک هایی هم برای ویرایش، حذف و مشاهده جزئیات رکوردها فراهم می‌کند. می‌توانید داده‌ها را مشاهده کنید، رکورد جدید ثبت کنید و یا داده‌های قبلی را ویرایش و حذف کنید.



یک تامین کننده OAuth2 و OpenID اضافه کنید OAuth یک پروتکل باز است که امکان authorization امن توسط یک متد استاندارد را فراهم می‌کند. این پروتکل می‌تواند در اپلیکیشن‌های وب، موبایل و دسکتاپ استفاده شود. قالب پروژه ASP.NET MVC internet از OAuth و OpenID استفاده می‌کند تا فیسبوک، توییتر، گوگل و حساب‌های کاربری مایکروسافت را بعنوان تامین کنندگان خارجی تعریف کند. به سادگی می‌توانید قطعه کدی را ویرایش کنید و از تامین کننده احراز هویت مورد نظرتان استفاده کنید. مراحلی که برای اضافه کردن این تامین کنندگان باید دنبال کنید، بسیار مشابه همین مراحلی است که در این مقاله دنبال خواهید کرد. برای اطلاعات بیشتر درباره نحوه استفاده از فیسبوک بعنوان یک تامین کننده احراز هویت به Create an ASP.NET MVC 5 App with Facebook and Google OAuth2 and OpenID Sign-on مراجعه کنید.

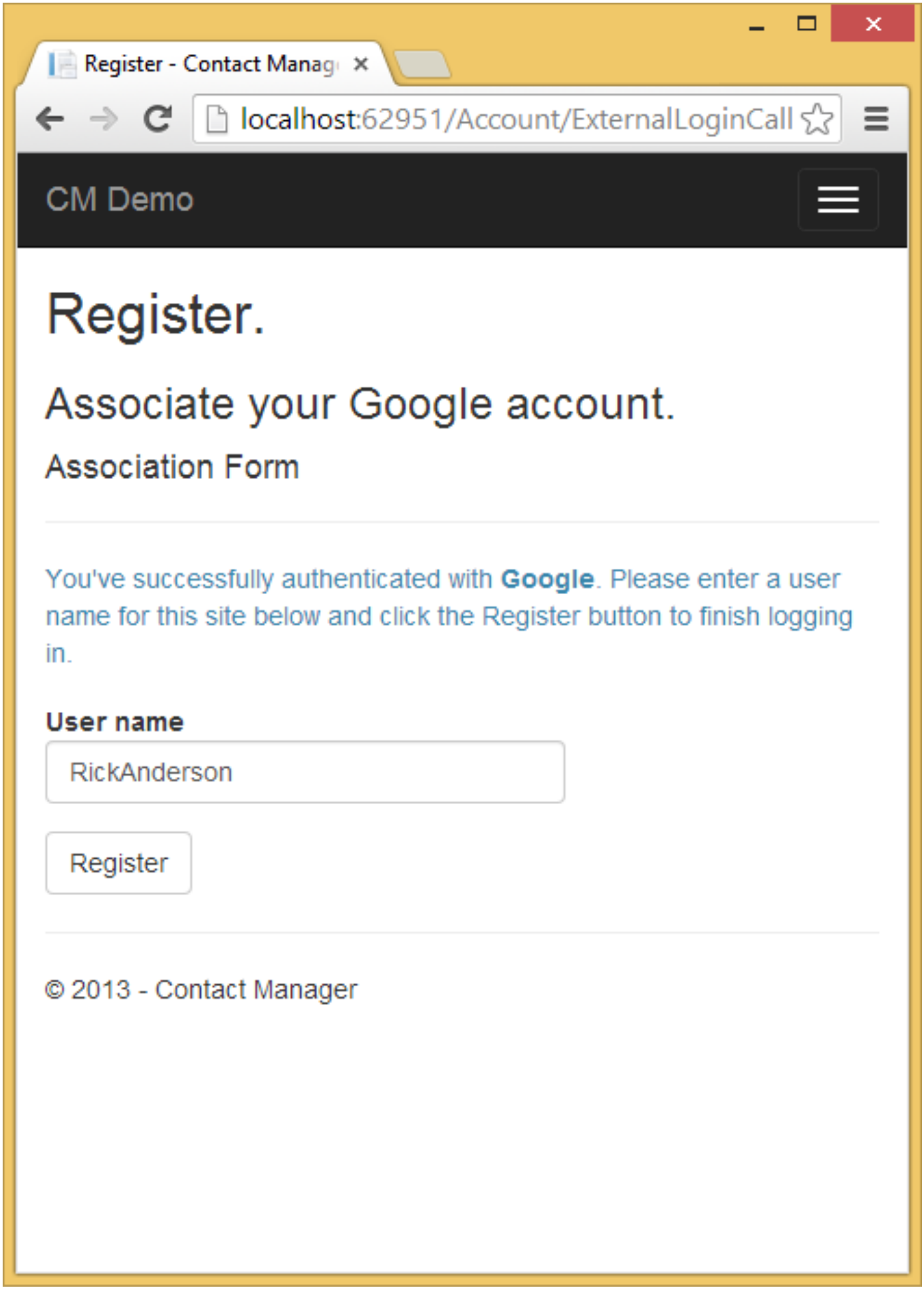
علاوه بر احراز هویت، اپلیکیشن ما از نقش‌ها (roles) نیز استفاده خواهد کرد تا از authorization پشتیبانی کند. تنها کاربرانی که به نقش *canEdit* تعلق داشته باشند قادر به ویرایش اطلاعات خواهند بود (یعنی ایجاد، ویرایش و حذف رکورد ها).

فایل *App_Start/Startup.Auth.cs* را باز کنید. توضیحات متد *app.UseGoogleAuthentication* را حذف کنید.

حال اپلیکیشن را اجرا کنید و روی لینک **Log In** کلیک کنید.

زیر قسمت **User another service to log in** روی دکمه **Google** کلیک کنید. اطلاعات کاربری خود را وارد کنید. سپس **Accept** را کلیک کنید تا به اپلیکیشن خود دسترسی کافی بدهید (برای آدرس ایمیل و اطلاعات پایه).

حال باید به صفحه ثبت نام (Register) هدایت شوید. در این مرحله می‌توانید در صورت لزوم نام کاربری خود را تغییر دهید. نهایتا روی **Register** کلیک کنید.


Register - Contact Manag
localhost:62951/Account/ExternalLoginCall
CM Demo
Register.
Associate your Google account.
Association Form
You've successfully authenticated with Google. Please enter a user name for this site below and click the Register button to finish logging in.
User name
RickAnderson
Register
© 2013 - Contact Manager

استفاده از Membership API

در این قسمت شما یک کاربر محلی و نقش *canEdit* را به دیتابیس عضویت اضافه می‌کنید. تنها کاربرانی که به این نقش تعلق دارند قادر به ویرایش داده‌ها خواهند بود. یکی از بهترین تمرین‌ها (best practice) نام گذاری نقش‌ها بر اساس عملیاتی است که می‌توانند اجرا کنند. بنابراین مثلا *canEdit* نسبت به نقشی با نام *admin* ترجیح داده می‌شود. هنگامی که اپلیکیشن شما رشد می‌کند و بزرگتر می‌شود، شما می‌توانید نقش‌های جدیدی مانند *canDeleteMembers* اضافه کنید، بجای آنکه از نام‌های گنگی مانند *superAdmin* استفاده کنید.

فایل *Migrations/Configuration.cs* را باز کنید و عبارات زیر را به آن اضافه کنید.

```
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;
```

متد **AddUserAndRole** را به این کلاس اضافه کنید.

```
bool AddUserAndRole(ContactManager.Models.ApplicationDbContext context)
 {
    IdentityResult ir;
    var rm = new RoleManager<IdentityRole>
        (new RoleStore<IdentityRole>(context));
    ir = rm.Create(new IdentityRole("canEdit"));
    var um = new UserManager<ApplicationUser>(
        new UserStore<ApplicationUser>(context));
    var user = new ApplicationUser()
    {
       UserName = "user1",
    };
    ir = um.Create(user, "Passw0rd1");
    if (ir.Succeeded == false)
       return ir.Succeeded;
    ir = um.AddToRole(user.Id, "canEdit");
    return ir.Succeeded;
 }
```

حالا از متد **Seed** این متد جدید را فراخوانی کنید.

```
protected override void Seed(ContactManager.Models.ApplicationDbContext context)
{
    AddUserAndRole(context);
    context.Contacts.AddOrUpdate(p => p.Name,
        // Code removed for brevity
}
```

این کدها نقش جدیدی با نام *canEdit* و کاربری با نام *user1* می سازد. سپس این کاربر به نقش مذکور اضافه می‌شود.

کدی موقتی برای تخصیص نقش *canEdit* به کاربران جدید Social Provider ها

در این قسمت شما متد **ExternalLoginConfirmation** در کنترلر Account را ویرایش خواهید کرد. یا این تغییرات، کاربران جدیدی که توسط OAuth یا OpenID ثبت نام می‌کنند به نقش *canEdit* اضافه می‌شوند. تا زمانی که ابزاری برای افزودن و مدیریت نقش‌ها بسازیم، از این کد موقتی استفاده خواهیم کرد. تیم مایکروسافت امیدوار است ابزاری مانند WSAT برای مدیریت کاربران و نقش‌ها در آینده عرضه کند. بعدا در این مقاله با اضافه کردن کاربران به نقش‌ها بصورت دستی از طریق **Server Explorer** نیز آشنا خواهید شد.

فایل **Controllers/AccountController.cs** را باز کنید و متد **ExternalLoginConfirmation** را پیدا کنید.

درست قبل از فراخوانی **SignInAsync** متد **AddToRoleAsync** را فراخوانی کنید.

```
await UserManager.AddToRoleAsync(user.Id, "CanEdit");
```

کد بالا کاربر ایجاد شده جدید را به نقش canEdit اضافه می‌کند، که به آنها دسترسی به متدهای ویرایش داده را می‌دهد. تصویری

از تغییرات کد در زیر آمده است.

```
//
// POST: /Account/ExternalLoginConfirmation
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> ExternalLoginConfirmation(ExternalLoginConfirmati
{
    if (User.Identity.IsAuthenticated)
    {
        return RedirectToAction("Manage");
    }

    if (ModelState.IsValid)
    {
        // Get the information about the user from the external login provider
        var info = await AuthenticationManager.GetExternalLoginInfoAsync();
        if (info == null)
        {
            return View("ExternalLoginFailure");
        }
        var user = new ApplicationUser() { UserName = model.UserName };
        var result = await UserManager.CreateAsync(user);
        if (result.Succeeded)
        {
            result = await UserManager.AddLoginAsync(user.Id, info.Login);
            if (result.Succeeded)
            {
                await UserManager.AddToRoleAsync(user.Id, "CanEdit");
                await SignInAsync(user, isPersistent: false);
                return RedirectToLocal(returnUrl);
            }
        }
        AddErrors(result);
    }

    ViewBag.ReturnUrl = returnUrl;
    return View(model);
}
```

در ادامه مقاله اپلیکیشن خود را روی Windows Azure منتشر خواهید کرد و با استفاده از Google و تامین کنندگان دیگر وارد سایت می‌شوید. هر فردی که به آدرس سایت شما دسترسی داشته باشد، و یک حساب کاربری Google هم در اختیار داشته باشد

می‌تواند در سایت شما ثبت نام کند و سپس دیتابیس را ویرایش کند. برای جلوگیری از دسترسی دیگران، می‌توانید وب سایت خود را متوقف (stop) کنید.

در پنجره **Package Manager Console** فرمان زیر را وارد کنید.

```
Update-Database
```

فرمان را اجرا کنید تا متد **Seed** را فراخوانی کند. حال **AddUserAndRole** شما نیز اجرا می‌شود. تا این مرحله نقش *canEdit* ساخته شده و کاربر جدیدی با نام *user1* ایجاد و به آن افزوده شده است.

محافظت از اپلیکیشن توسط SSL و خاصیت Authorize

در این قسمت شما با استفاده از خاصیت Authorize دسترسی به اکشن متدها را محدود می‌کنید. کاربران ناشناس (Anonymous) تنها قادر به مشاهده متد **Index** در کنترلر **home** خواهند بود. کاربرانی که ثبت نام کرده اند به متدهای **Index** و **Details** در کنترلر **Cm** و صفحات About و Contact نیز دسترسی خواهند داشت. همچنین دسترسی به متدهایی که داده‌ها را تغییر می‌دهند تنها برای کاربرانی وجود دارد که در نقش *canEdit* هستند.

خاصیت Authorize و RequireHttps را به اپلیکیشن اضافه کنید. یک راه دیگر افزودن این خاصیت‌ها به تمام کنترلرها است، اما تجارب امنیتی توصیه می‌کند که این خاصیت‌ها روی کل اپلیکیشن اعمال شوند. با افزودن این خاصیت‌ها بصورت global تمام کنترلرها و اکشن متدهایی که می‌سازید بصورت خودکار محافظت خواهند شد، و دیگر لازم نیست بیاد داشته باشید کدام کنترلرها و متدها را باید ایمن کنید.

برای اطلاعات بیشتر به Securing your ASP.NET MVC App and the new AllowAnonymous Attribute مراجعه کنید.

فایل *App_Start/FilterConfig.cs* را باز کنید و متد *RegisterGlobalFilters* را با کد زیر مطابقت دهید.

```
public static void
RegisterGlobalFilters(GlobalFilterCollection filters)
{
    filters.Add(new HandleErrorAttribute());
    filters.Add(new System.Web.Mvc.AuthorizeAttribute());
    filters.Add(new RequireHttpsAttribute());
}
```

خاصیت Authorize در کد بالا از دسترسی کاربران ناشناس به تمام متدهای اپلیکیشن جلوگیری می‌کند. شما برای اعطای دسترسی به متدهایی خاص از خاصیت AllowAnonymous استفاده خواهید کرد. در آخر خاصیت RequireHTTPS باعث می‌شود تا تمام دسترسی‌ها به اپلیکیشن وب شما از طریق HTTPS صورت گیرد.

حالا خاصیت AllowAnonymous را به متد **Index** در کنترلر Home اضافه کنید. از این خاصیت برای اعطای دسترسی به تمامی کاربران سایت استفاده کنید. قسمتی از کد کنترلر Home را در زیر می‌بینید.

```
namespace ContactManager.Controllers
 {
    public class HomeController : Controller
    {
       [AllowAnonymous]
       public ActionResult Index()
       {
          return View();
       }
```

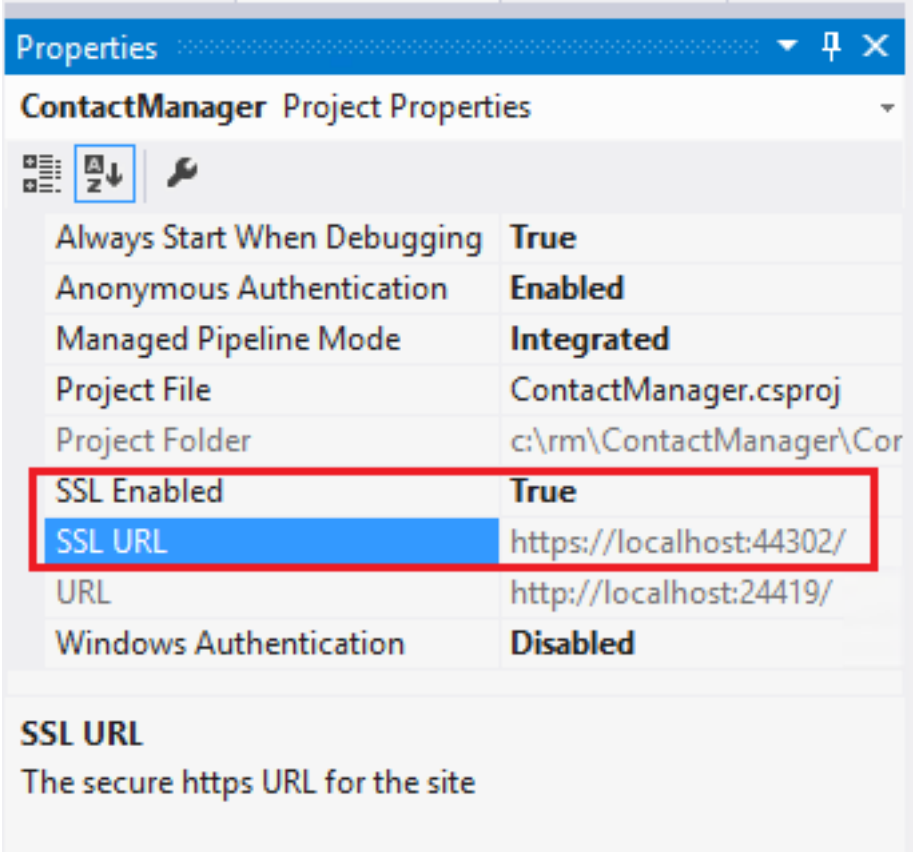
یک جستجوی عمومی برای عبارت *AllowAnonymous* انجام دهید. همانطور که مشاهده می‌کنید این خاصیت توسط متدهای ورود و ثبت نام در کنترلر Account نیز استفاده شده است.

در کنترلر *CmController* خاصیت *[Authorize(Roles="canEdit")]* را به تمام متدهایی که با داده سر و کار دارند اضافه کنید، به غیر از متدهای Index و Details. قسمتی از کد کامل شده در زیر آمده است.

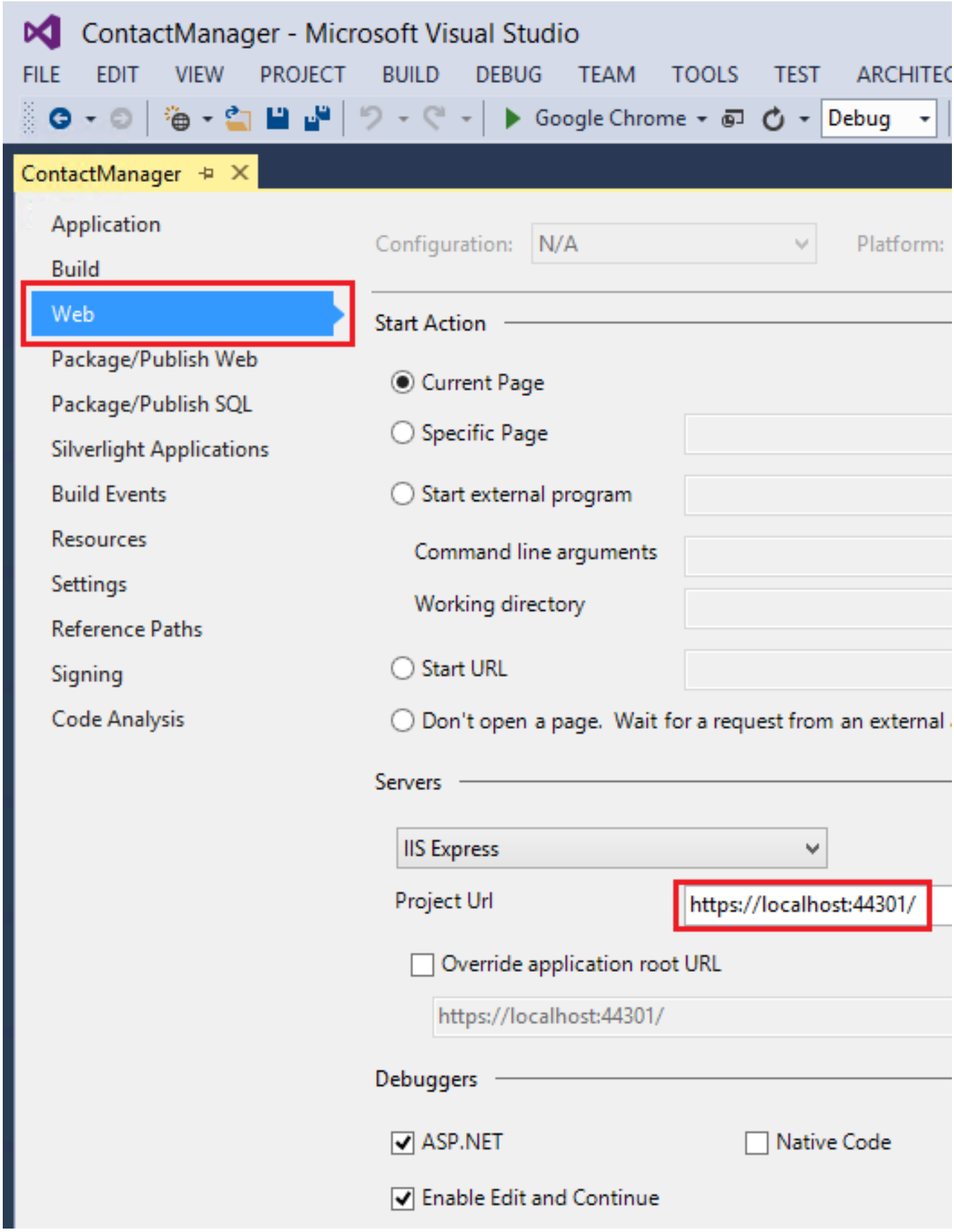
```
public class CmController : Controller
{
    private ContactManagerContext db = new ContactManagerContext();

    // GET: /Cm/Create

    [Authorize(Roles = "canEdit")]
    public ActionResult Create()
    {
        return View();
    }

    // POST: /Cm/Create
    [HttpPost]
    [ValidateAntiForgeryToken]
    [Authorize(Roles = "canEdit")]
    public ActionResult Create([Bind(Include = "ContactId,Name,Address,City,
    {
        if (ModelState.IsValid)
        {
            db.Contacts.Add(contact);
            db.SaveChanges();
            return RedirectToAction("Index");
        }

        return View(contact);
    }

    // GET: /Cm/Edit/5
    [Authorize(Roles = "canEdit")]
    public ActionResult Edit(int? id)
    {
        if (id == null)
        {
            return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
        }
        Contact contact = db.Contacts.Find(id);
        if (contact == null)
        {
            return HttpNotFound();
        }
        return View(contact);
    }
```

فعال سازی SSL برای پروژه

در Solution Explorer پروژه خود را انتخاب کنید. سپس کلید F4 را فشار دهید تا دیالوگ خواص (Properties) باز شود. حال مقدار خاصیت **SSL Enabled** را به true تنظیم کنید. آدرس **SSL URL** را کپی کنید. این آدرس چیزی شبیه به https://localhost:44300/ خواهد بود.

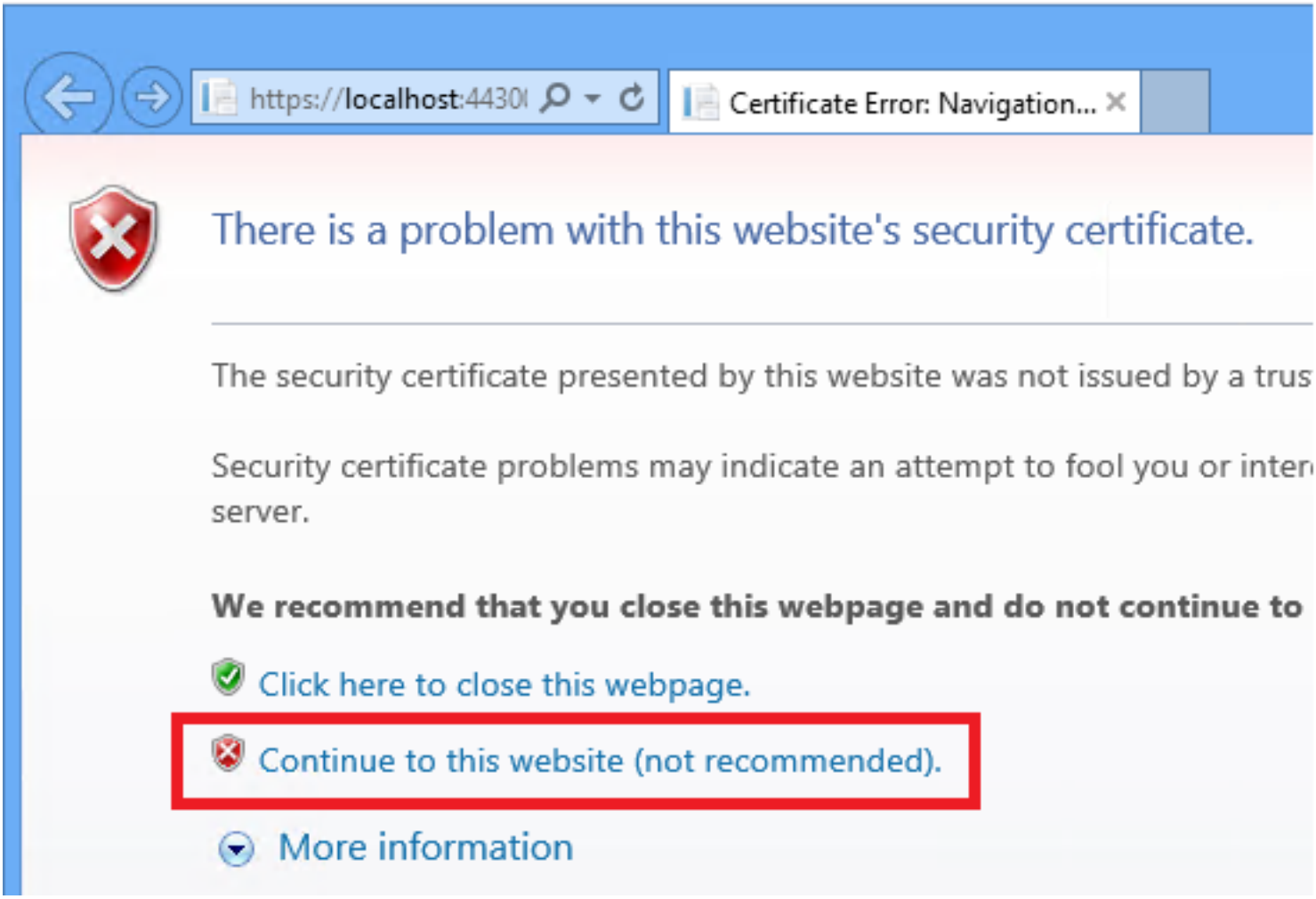


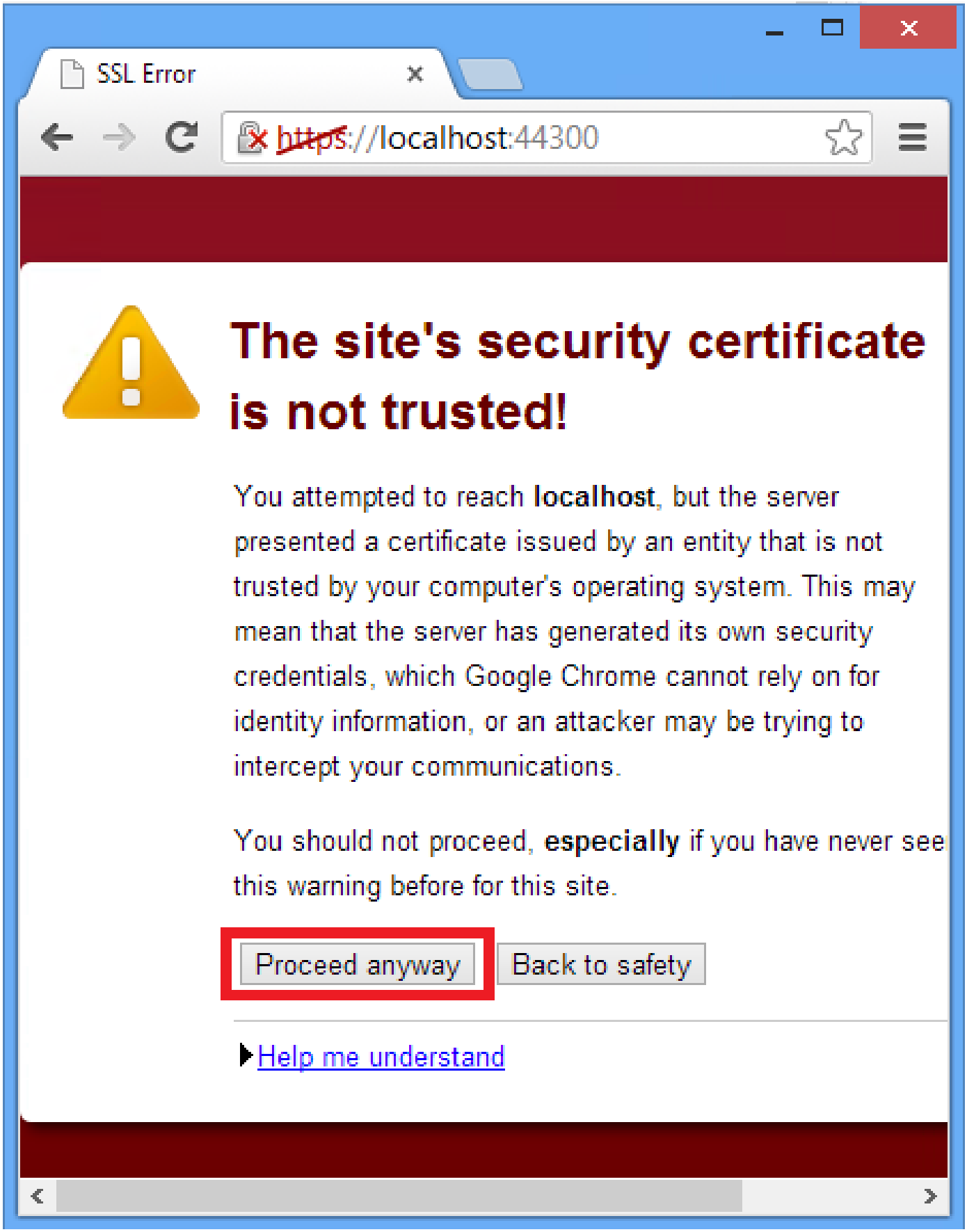
روی نام پروژه کلیک راست کنید و **Properties** را انتخاب کنید. در قسمت چپ گزینه **Web** را انتخاب کنید. حالا مقدار **Project Url** را به آدرسی که کپی کرده اید تغییر دهید. نهایتا تغییرات را ذخیره کنید و پنجره را ببندید.



حال پروژه را اجرا کنید. مرورگر شما باید یک پیام خطای اعتبارسنجی به شما بدهد. دلیلش این است که اپلیکیشن شما از یک

Valid Certificate استفاده نمی‌کند. هنگامی که پروژه را روی Windows Azure منتشر کنید دیگر این پیغام را نخواهید دید. چرا که سرورهای مایکروسافت همگی لایسنس‌های معتبری دارند. برای اپلیکیشن ما می‌توانید روی **Continue to this website** را انتخاب کنید.





حال مرورگر پیش فرض شما باید صفحه **Index** از کنترلر home را به شما نمایش دهد.

اگر از یک نشست قبلی هنوز در سایت هستید (logged-in) روی لینک **Log out** کلیک کنید و از سایت خارج شوید.

روی لینک‌های **About** و **Contact** کلیک کنید. باید به صفحه ورود به سایت هدایت شوید چرا که کاربران ناشناس اجازه دسترسی به این صفحات را ندارند.

روی لینک **Register** کلیک کنید و یک کاربر محلی با نام *Joe* بسازید. حال مطمئن شوید که این کاربر به صفحات Home, About و Contact دسترسی دارد.

روی لینک *CM Demo* کلیک کنید و مطمئن شوید که داده‌ها را مشاهده می‌کنید.

حال روی یکی از لینک‌های ویرایش (Edit) کلیک کنید. این درخواست باید شما را به صفحه ورود به سایت هدایت کند، چرا که کاربران محلی جدید به نقش *canEdit* تعلق ندارند.

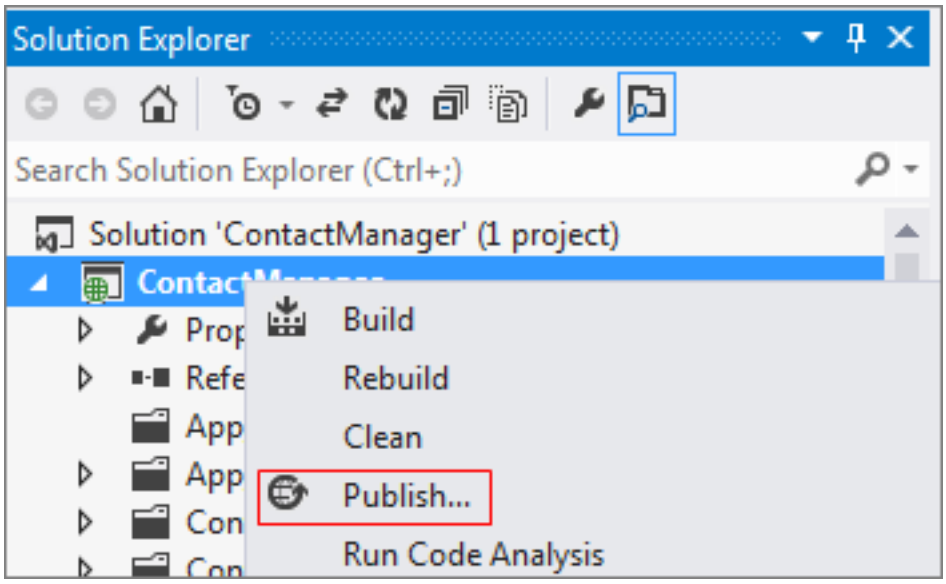
با کاربر *user1* که قبلا ساختید وارد سایت شوید. حال به صفحه ویرایشی که قبلا درخواست کرده بودید هدایت می‌شوید.

اگر نتوانستید با این کاربر به سایت وارد شوید، کلمه عبور را از سورس کد کپی کنید و مجددا امتحان کنید. اگر همچنان نتوانستید به سایت وارد شوید، جدول **AspNetUsers** را بررسی کنید تا مطمئن شوید کاربر *user1* ساخته شده است. این مراحل را در ادامه مقاله خواهید دید.

در آخر اطمینان حاصل کنید که می‌توانید داده‌ها را تغییر دهید.

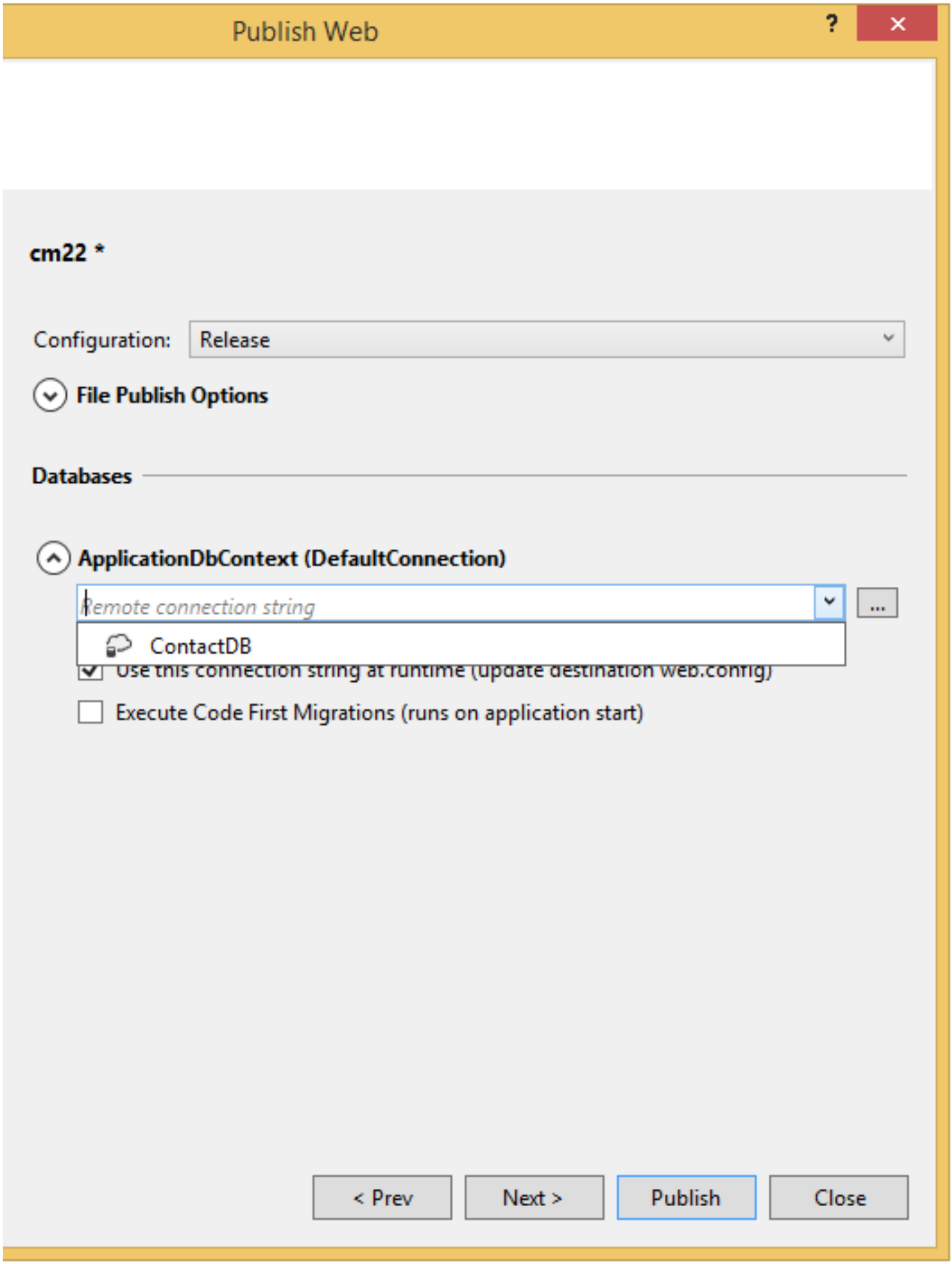
اپلیکیشن را روی Windows Azure منتشر کنید

ابتدا پروژه را Build کنید. سپس روی نام پروژه کلیک راست کرده و گزینه **Publish** را انتخاب کنید.

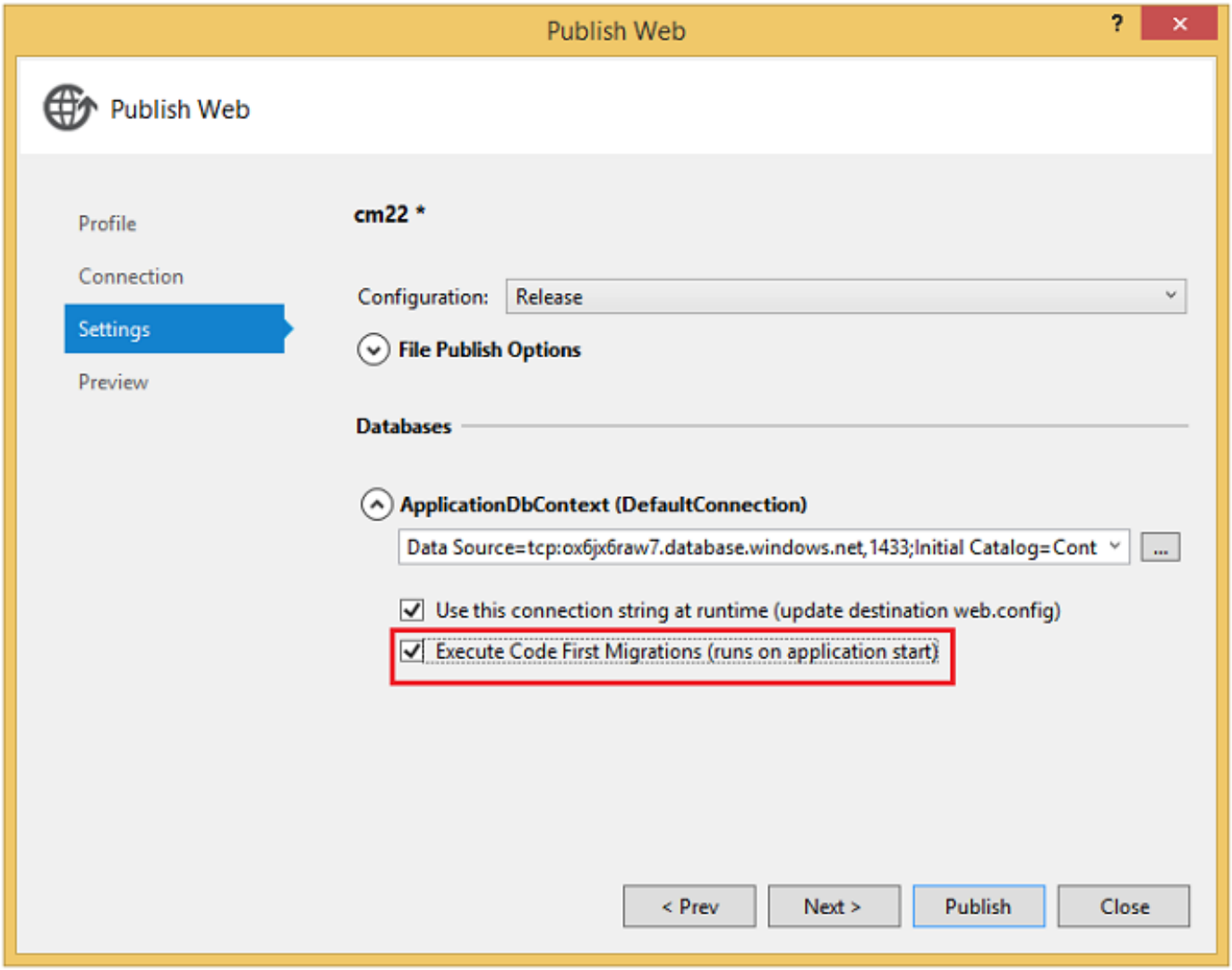


در دیالوگ باز شده روی قسمت **Settings** کلیک کنید. روی **File Publish Options** کلیک کنید تا بتوانید **Remote connection string** را برای **ApplicationDbContext** و دیتابیس ContactDB انتخاب کنید.

اگر ویژوال استودیو را پس از ساخت Publish profile بسته و دوباره باز کرده اید، ممکن است رشته اتصال را در لیست موجود نبینید. در چنین صورتی، بجای ویرایش پروفایل انتشار، یک پروفایل جدید بسازید. درست مانند مراحلی که پیشتر دنبال کردید.


Publish Web
cm22 *
Configuration:
Release
File Publish Options
Databases
ApplicationDbContext (DefaultConnection)
Remote connection string
...
ContactDB
Use this connection string at runtime (update destination web.config)
Execute Code First Migrations (runs on application start)
< Prev
Next >
Publish
Close

زیر قسمت **ContactManagerContext** گزینه **Execute Code First Migrations** را انتخاب کنید.

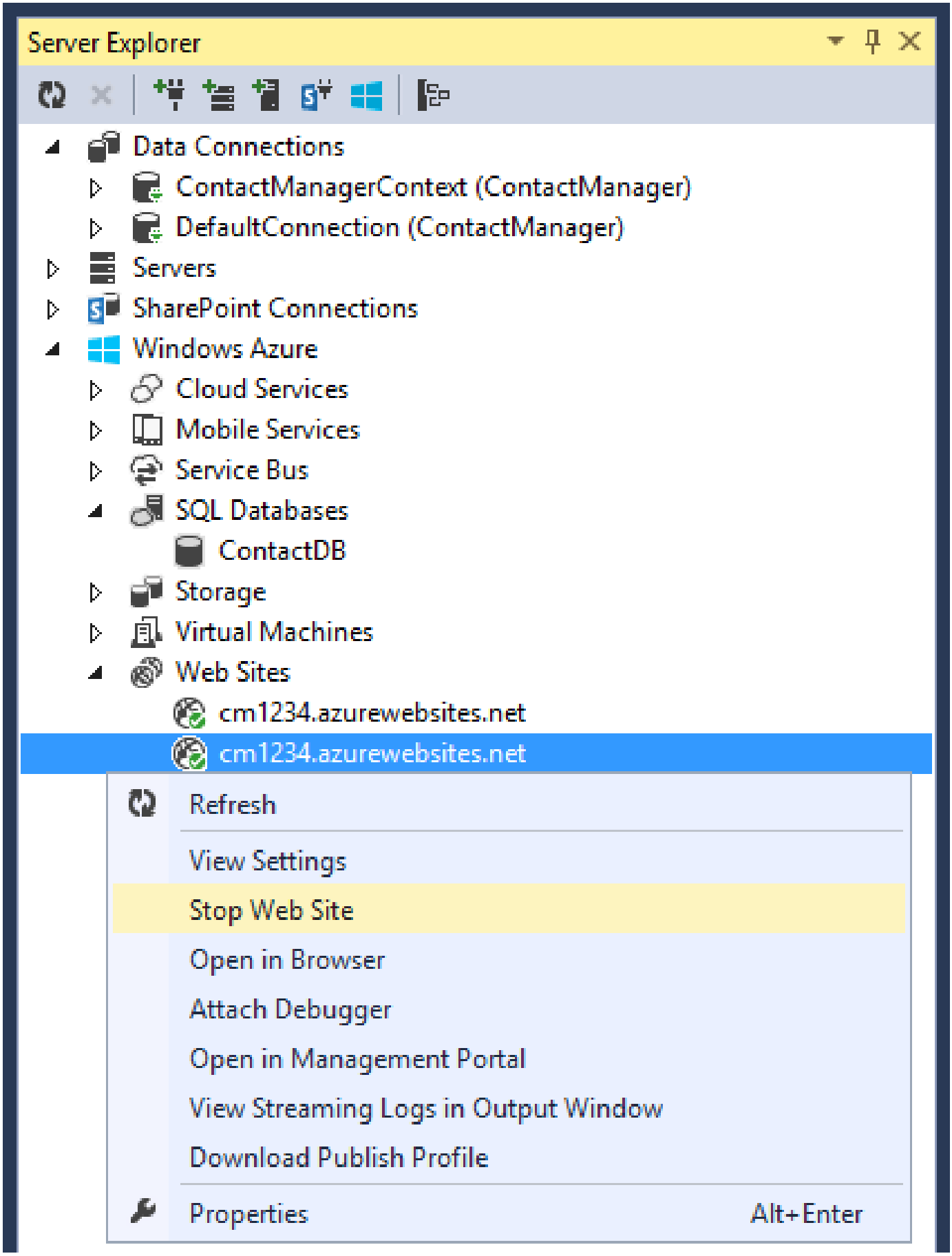


حال **Publish** را کلیک کنید تا اپلیکیشن شما منتشر شود. با کاربر *user1* وارد سایت شوید و بررسی کنید که می‌توانید داده‌ها را ویرایش کنید یا خیر.

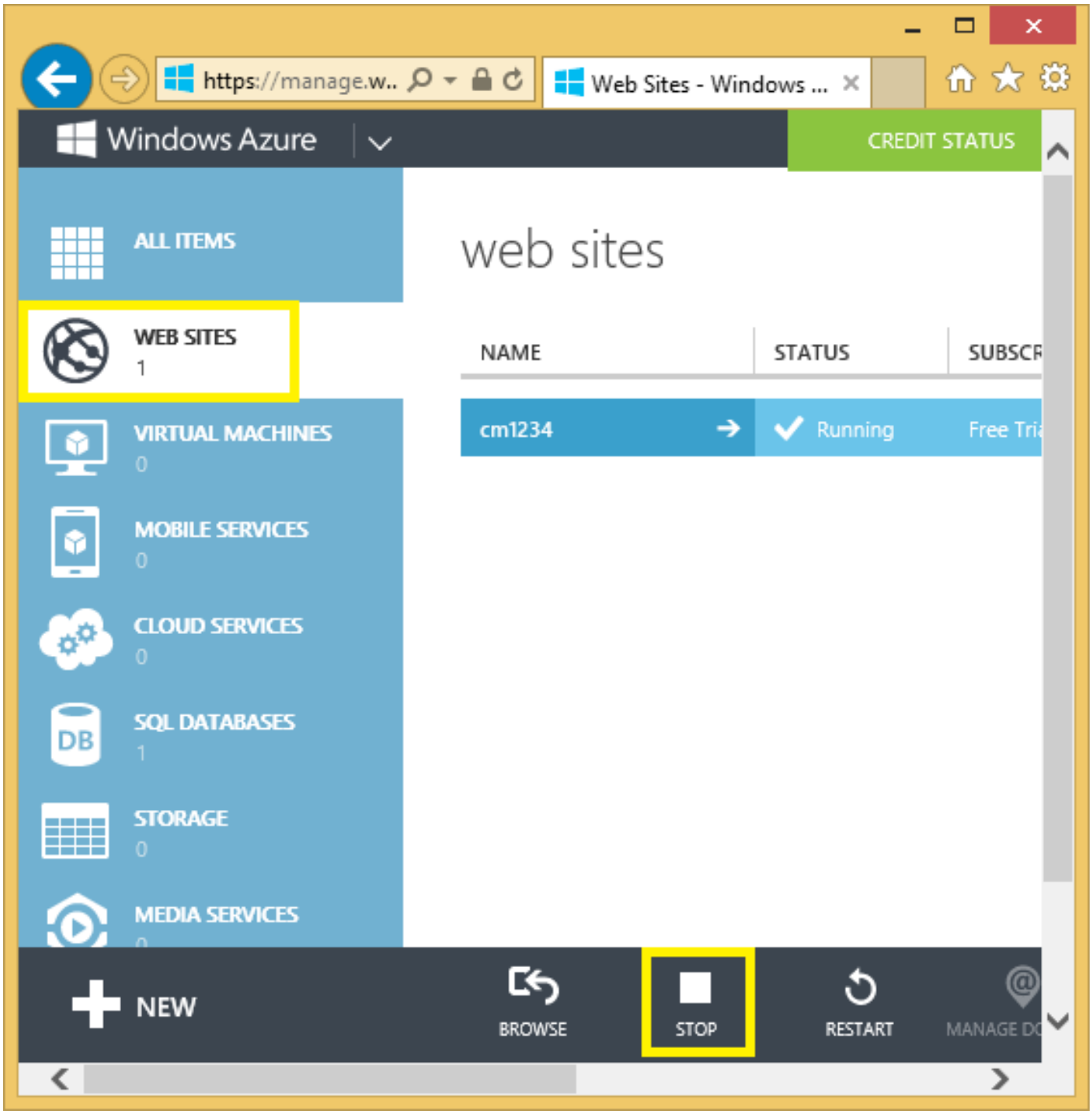
حال از سایت خارج شوید و توسط یک اکانت Google یا Facebook وارد سایت شوید، که در این صورت نقش canEdit نیز به شما تعلق می‌گیرد.

برای جلوگیری از دسترسی دیگران، وب سایت را متوقف کنید

در **Server Explorer** به قسمت **Web Sites** بروید. حال روی هر نمونه از وب سایت‌ها کلیک راست کنید و گزینه **Stop Web Site** را انتخاب کنید.


Server Explorer
Data Connections
ContactManagerContext (ContactManager)
DefaultConnection (ContactManager)
Servers
SharePoint Connections
Windows Azure
Cloud Services
Mobile Services
Service Bus
SQL Databases
ContactDB
Storage
Virtual Machines
Web Sites
cm1234.azurewebsites.net
cm1234.azurewebsites.net
Refresh
View Settings
Stop Web Site
Open in Browser
Attach Debugger
Open in Management Portal
View Streaming Logs in Output Window
Download Publish Profile
Properties
Alt+Enter

یک راه دیگر متوقف کردن وب سایت از طریق پرتال مدیریت Windows Azure است.

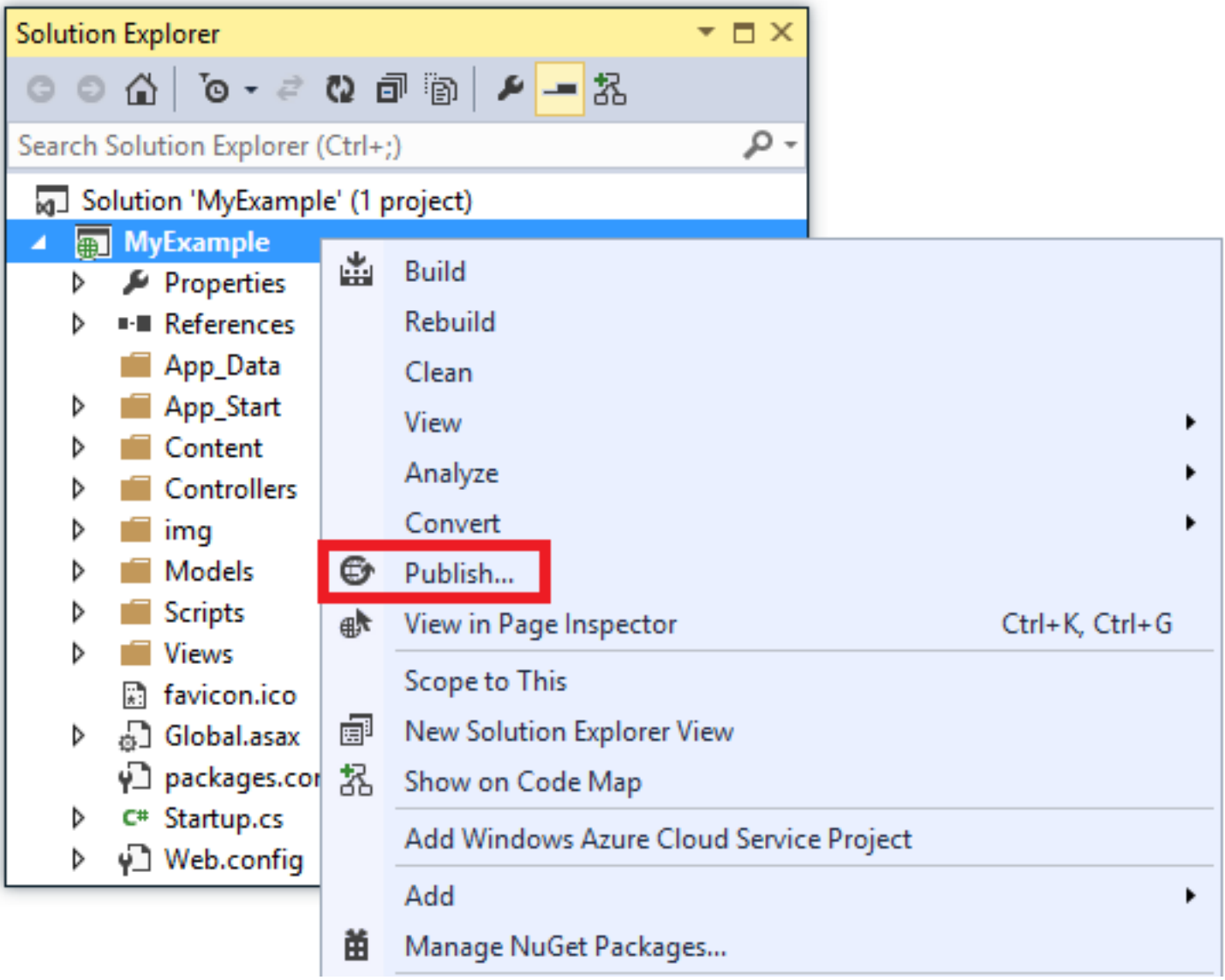


فراخوانی AddToRoleAsync را حذف و اپلیکیشن را منتشر و تست کنید

کنترلر Account را باز کنید و کد زیر را از متد **ExternalLoginConfirmation** حذف کنید.

```
await UserManager.AddToRoleAsync(user.Id, "CanEdit");
```

پروژه را ذخیره و Build کنید. حال روی نام پروژه کلیک راست کرده و **Publish** را انتخاب کنید.



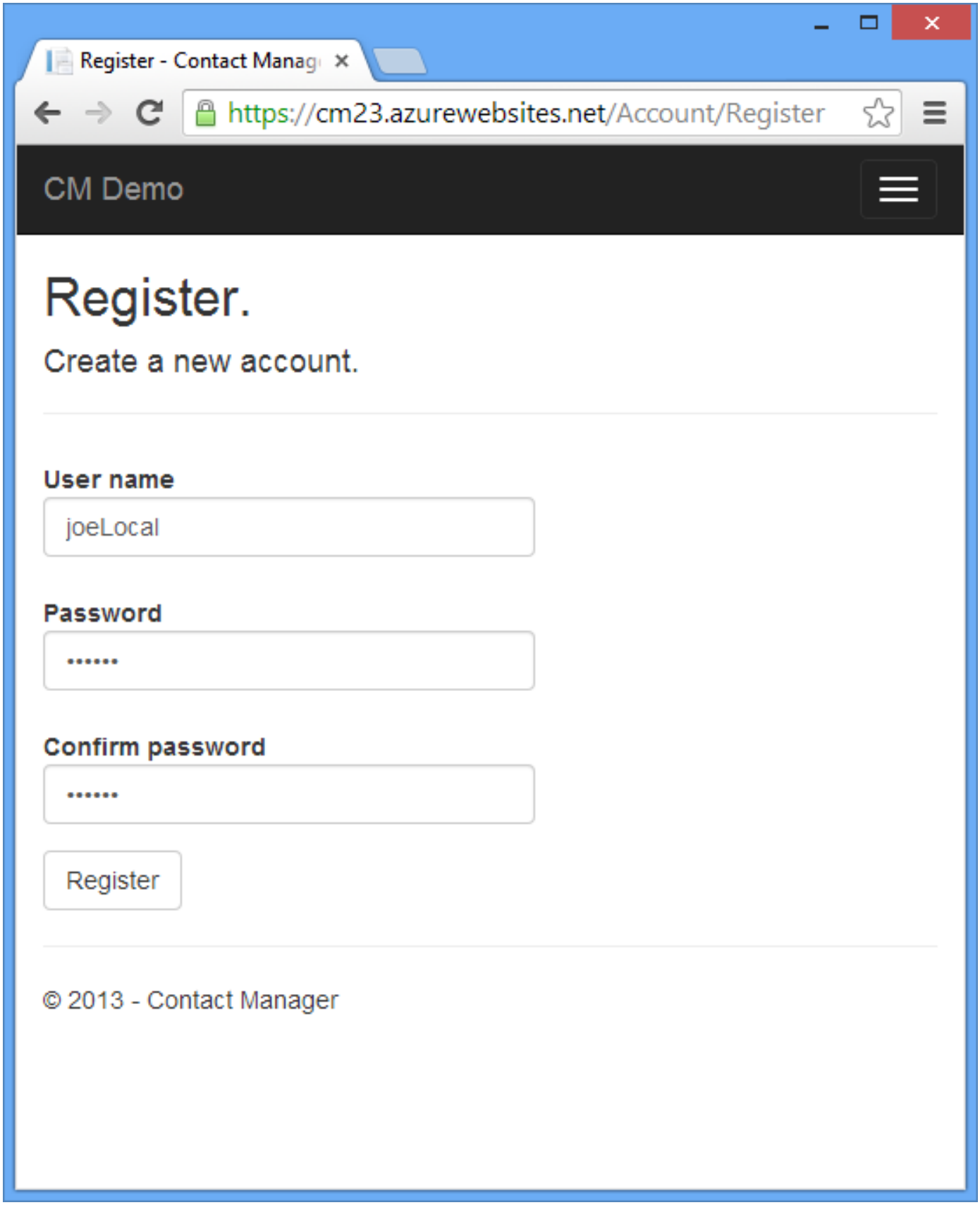
دکمه **Start Preview** را فشار دهید. در این مرحله تنها فایل هایی که نیاز به بروز رسانی دارند آپلود خواهند شد.

وب سایت را راه اندازی کنید. ساده‌ترین راه از طریق پرتال مدیریت Windows Azure است. توجه داشته باشید که تا هنگامی که وب سایت شما متوقف شده، نمی‌توانید اپلیکیشن خود را منتشر کنید.

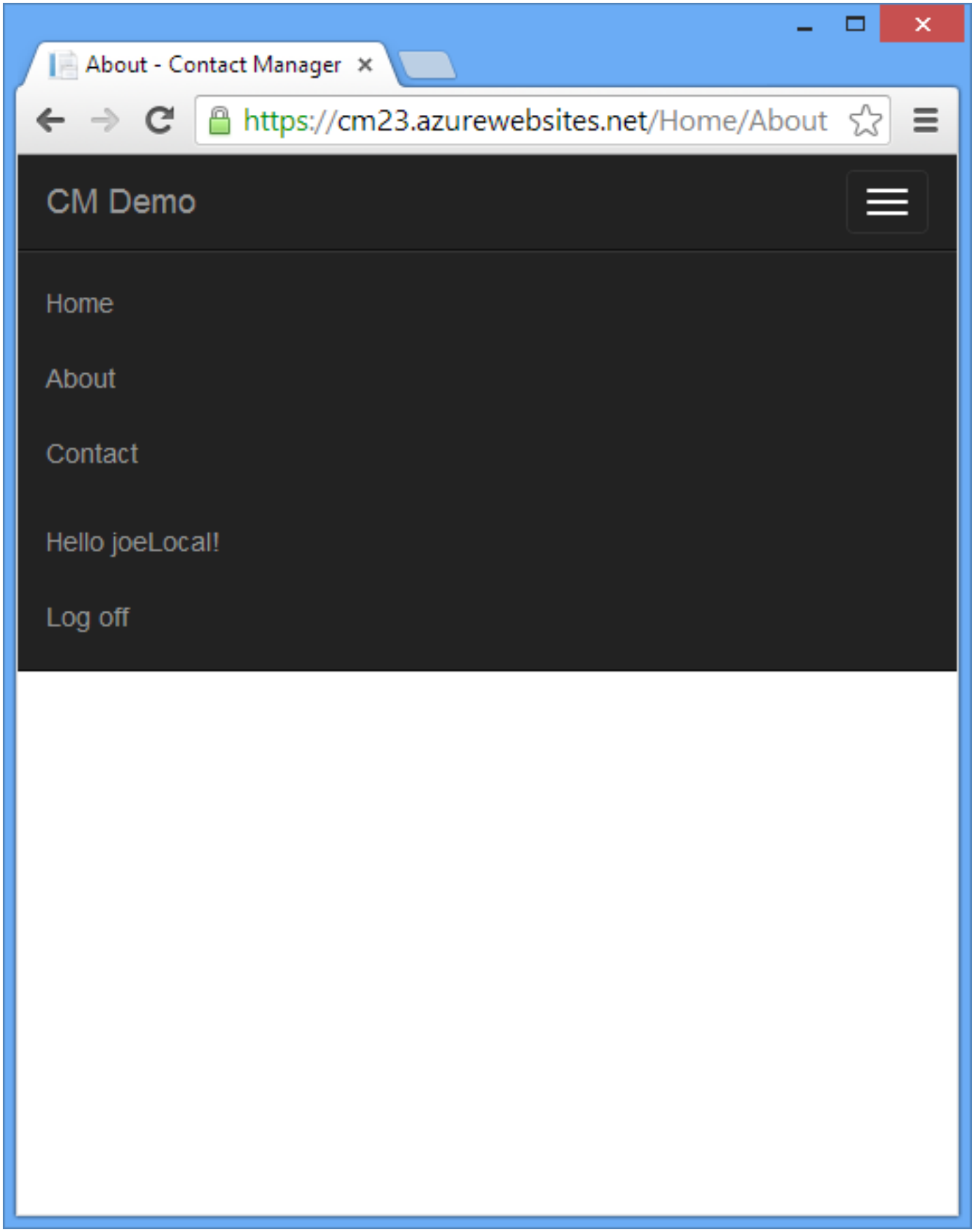
حال به ویژوال استودیو بازگردید و اپلیکیشن را منتشر کنید. اپلیکیشن Windows Azure شما باید در مرورگر پیش فرض تان باز شود. حال شما در حال مشاهده صفحه اصلی سایت بعنوان یک کاربر ناشناس هستید.

روی لینک **About** کلیک کنید، که شما را به صفحه ورود هدایت می‌کند.

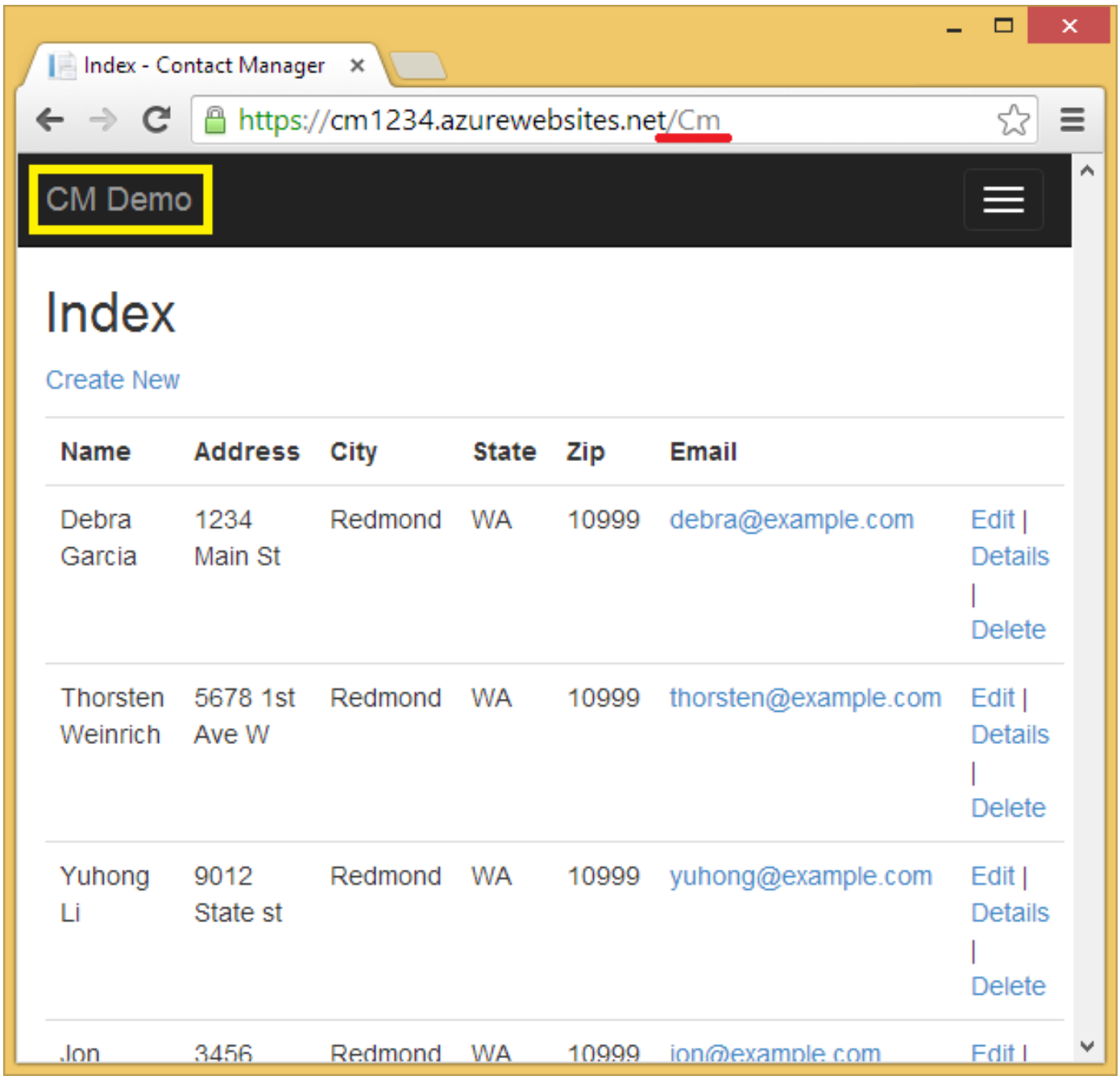
روی لینک **Register** در صفحه ورود کلیک کنید و یک حساب کاربری محلی بسازید. از این حساب کاربری برای این استفاده می‌کنیم که ببینیم شما به صفحات فقط خواندنی (read-only) و نه صفحاتی که داده‌ها را تغییر می‌دهند دسترسی دارید یا خیر. بعدا در ادامه مقاله، دسترسی حساب‌های کاربری محلی (local) را حذف می‌کنیم.



مطمئن شوید که به صفحات About و Contact دسترسی دارید.



لینک **CM Demo** را کلیک کنید تا به کنترلر *CmController* هدایت شوید.



روی یکی از لینک‌های Edit کلیک کنید. این کار شما را به صفحه ورود به سایت هدایت می‌کند. در زیر قسمت **User another service to log in** یکی از گزینه‌های Google یا Facebook را انتخاب کنید و توسط حساب کاربری ای که قبلا ساختید وارد شوید.

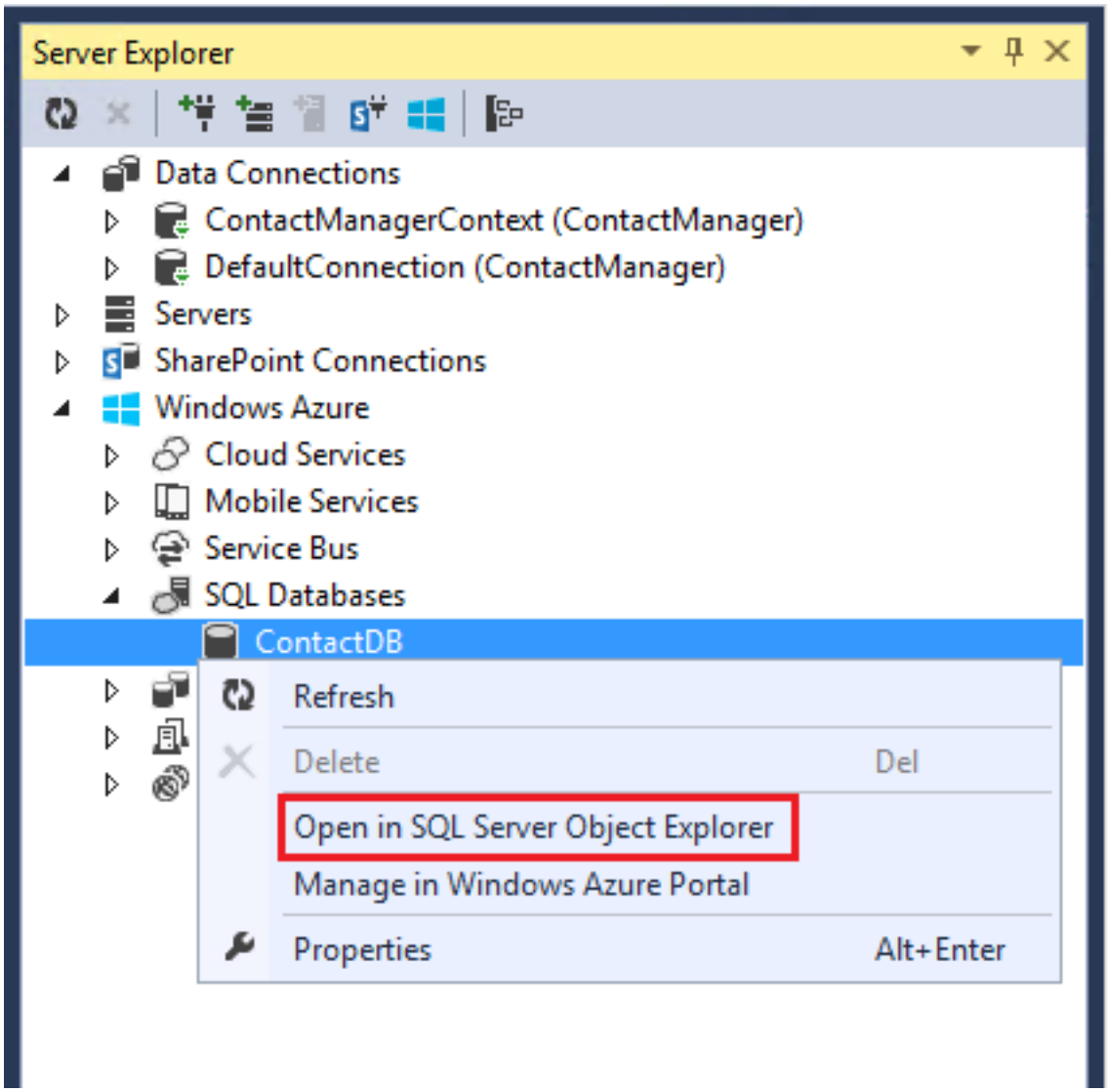
حال بررسی کنید که امکان ویرایش اطلاعات را دارید یا خیر.

نکته: شما نمی‌توانید در این اپلیکیشن از اکانت گوگل خود خارج شده، و با همان مرورگر با اکانت گوگل دیگری وارد اپلیکیشن شوید. اگر دارید از یک مرورگر استفاده می‌کنید، باید به سایت گوگل رفته و از آنجا خارج شوید. برای وارد شدن به اپلیکیشن توسط یک اکانت دیگر می‌توانید از یک مرورگر دیگر استفاده کنید.

دیتابیس SQL Azure را بررسی کنید

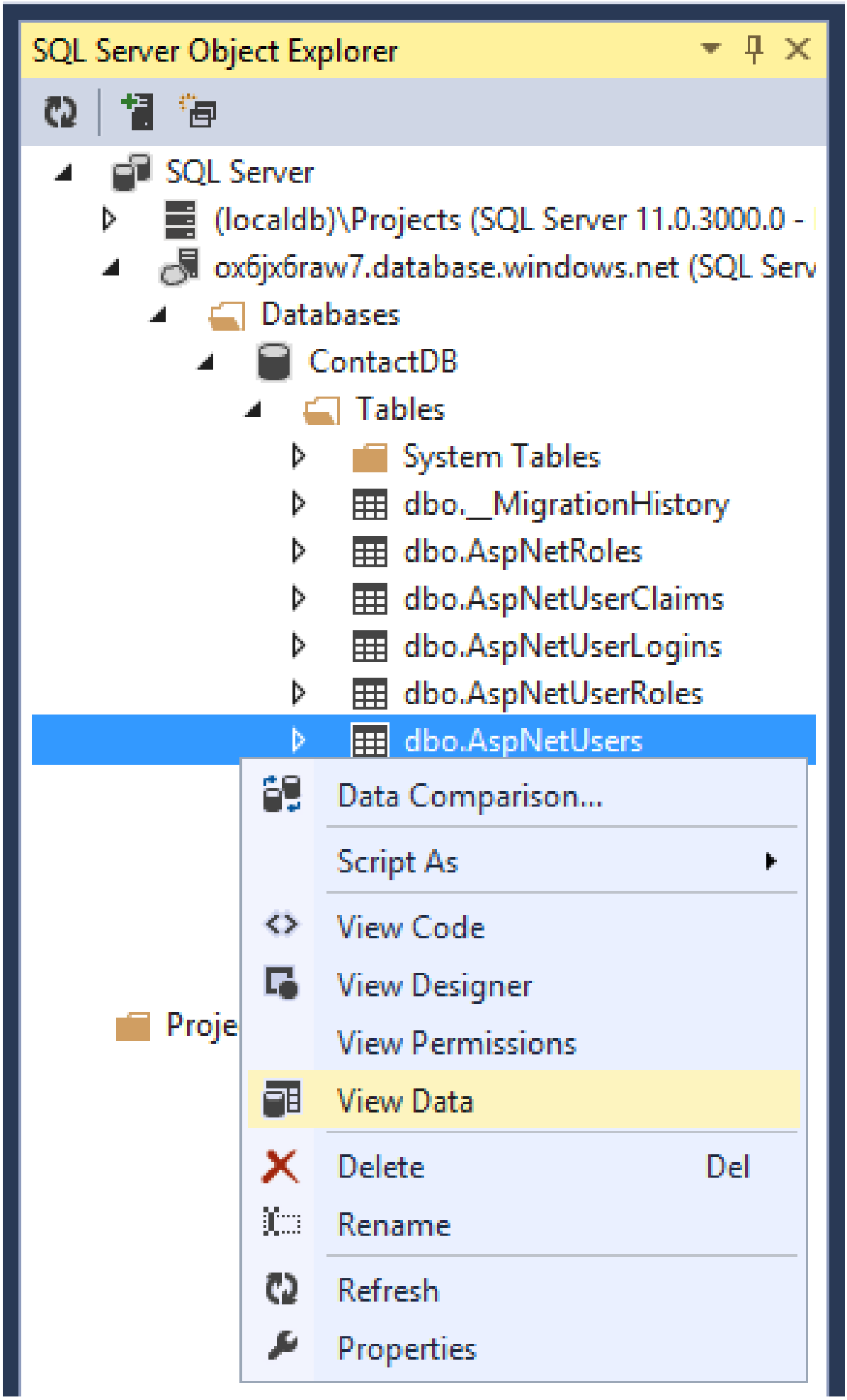
در **Server Explorer** دیتابیس **ContactDB** را پیدا کنید. روی آن کلیک راست کرده و **Open in SQL Server Object Explorer** را

انتخاب کنید.



توجه: اگر نمی‌توانید گره **SQL Databases** را باز کنید و یا **ContactDB** را در ویژوال استودیو نمی‌بینید، باید مراحلی را طی کنید تا یک پورت یا یکسری پورت را به فایروال خود اضافه کنید. دقت داشته باشید که در صورت اضافه کردن Port Rangeها ممکن است چند دقیقه زمان نیاز باشد تا بتوانید به دیتابیس دسترسی پیدا کنید.

روی جدول **AspNetUsers** کلیک راست کرده و **View Data** را انتخاب کنید.


SQL Server Object Explorer
SQL Server
(localdb)\Projects (SQL Server 11.0.3000.0 -
ox6jx6raw7.database.windows.net (SQL Serv
Databases
ContactDB
Tables
System Tables
dbo.__MigrationHistory
dbo.AspNetRoles
dbo.AspNetUserClaims
dbo.AspNetUserLogins
dbo.AspNetUserRoles
dbo.AspNetUsers
Data Comparison...
Script As
View Code
View Designer
View Permissions
View Data
Delete
Del
Rename
Refresh
Properties

dbo.AspNetUsers [Data]

Max Rows: 1000

Id	UserName	PasswordHash	Secu...	Discriminator
3b7fd83f-fc68-4f4d-ae27-b9922d17602d	JoeLocalUser	AGgn9Gfmxw...	c3337...	ApplicationUser
8a5687c2-86ed-40c9-853b-26ef40b7a2bb	RickGM000	NULL	5489a...	ApplicationUser
94715c84-9919-4146-ad2b-0cf692c7c70d	JoeLocalUser2	AOIMz9t+/+z...	d9f47...	ApplicationUser
9af370af-8055-4cef-965f-990214c24ccf	user1	AJjTVn2u86D...	7f3ab...	ApplicationUser
a18b44da-a9ac-4153-89a0-d9c808f22df8	joeLocal	ACDwfL01Klf...	2977f...	ApplicationUser
NULL	NULL	NULL	NULL	NULL

حالا روی **AspNetUserRoles** کلیک راست کنید و **View Data** را انتخاب کنید.

dbo.AspNetUserRoles [Data] | dbo.AspNetUsers [Data]

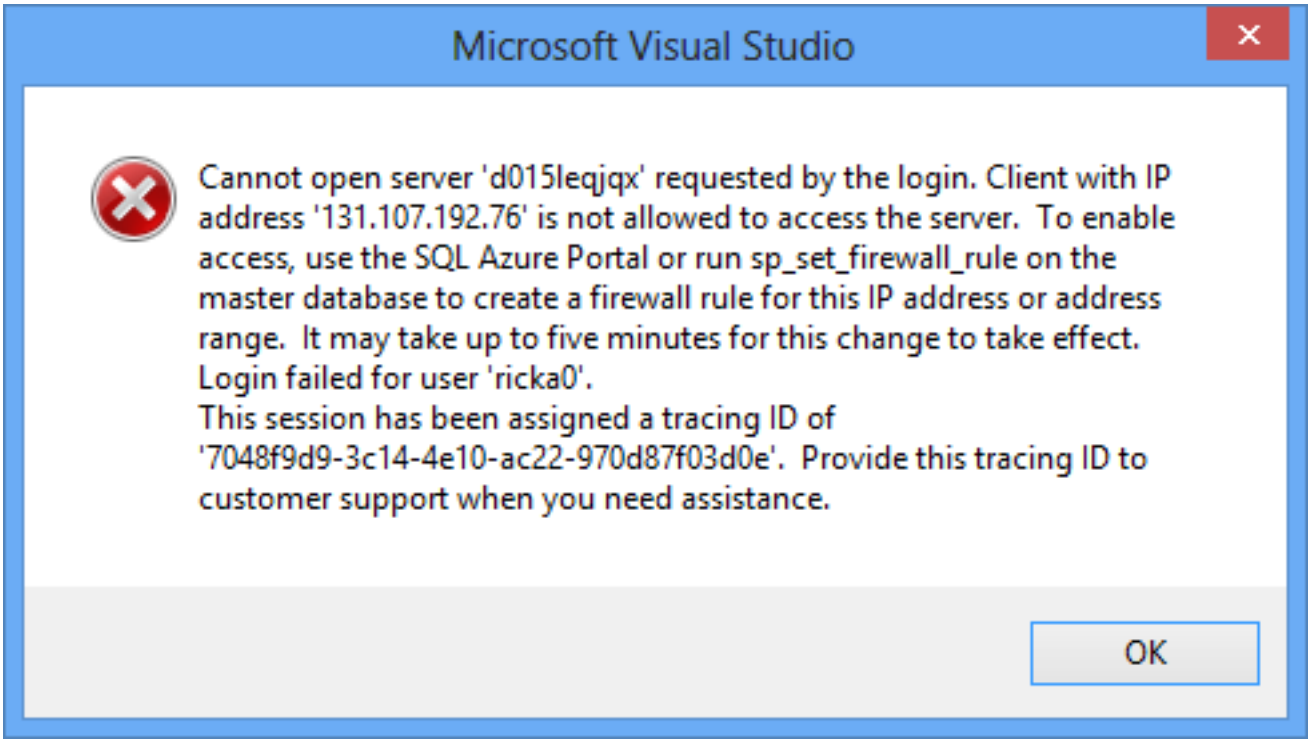
Max Rows: 1000

UserId	RoleId
8a5687c2-86ed-40c9-853b-26ef40b7a2bb	e43a4145-7089-4292-9057-af56e5d8e940
9af370af-8055-4cef-965f-990214c24ccf	e43a4145-7089-4292-9057-af56e5d8e940
NULL	NULL

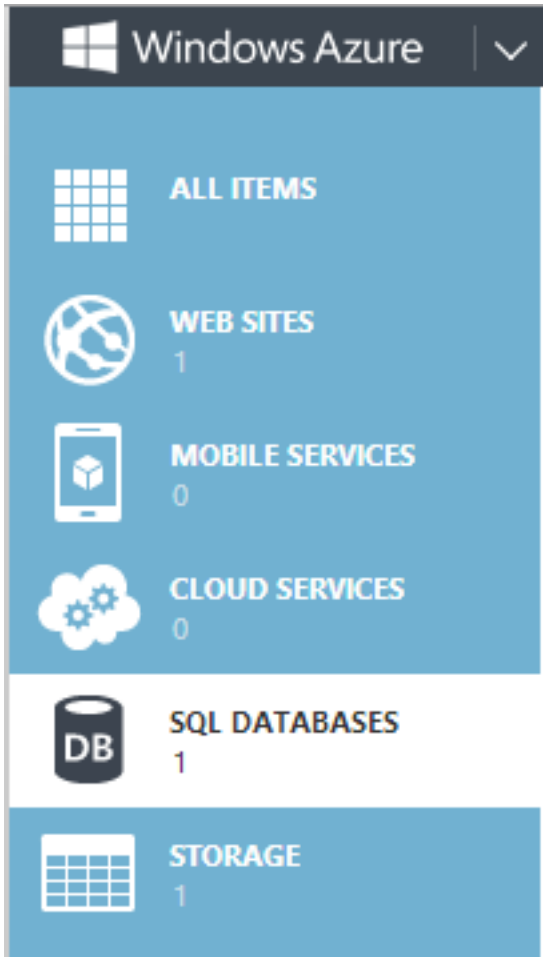
اگر شناسه کاربران (User ID) را بررسی کنید، مشاهده می‌کنید که تنها دو کاربر *user1* و اکانت گوگل شما به نقش *canEdit* تعلق دارند.

Cannot open server login error

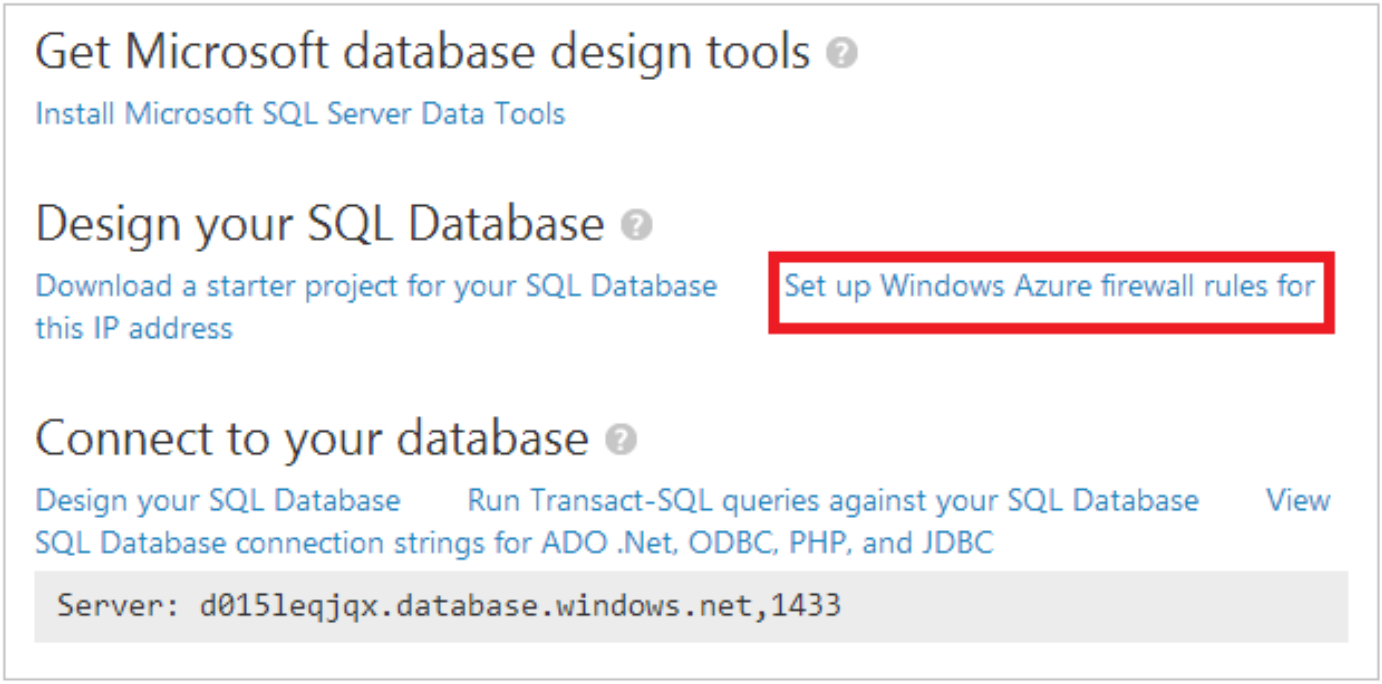
اگر خطایی مبنی بر "Cannot open server" دریافت می‌کنید، مراحل زیر را دنبال کنید.



شما باید آدرس IP خود را به لیست آدرس‌های مجاز (Allowed IPs) اضافه کنید. در پرتال مدیریتی Windows Azure در قسمت چپ صفحه، گزینه **SQL Databases** را انتخاب کنید.



دیتابیس مورد نظر را انتخاب کنید. حالا روی لینک **Set up Windows Azure firewall rules for this IP address** کلیک کنید.



هنگامی که با پیغام "The current IP address xxx.xxx.xxx.xxx is not included in existing firewall rules. Do you want? to update the firewall rules" مواجه شدید **Yes** را کلیک کنید. افزودن یک آدرس IP بدین روش معمولا کافی نیست و در فایروال‌های سازمانی و بزرگ باید Range بیشتری را تعریف کنید.

مرحله بعد اضافه کردن محدوده آدرس‌های مجاز است.

مجددا در پرتال مدیریتی Windows Azure روی **SQL Databases** کلیک کنید. سروری که دیتابیس شما را میزبانی می‌کند انتخاب کنید.

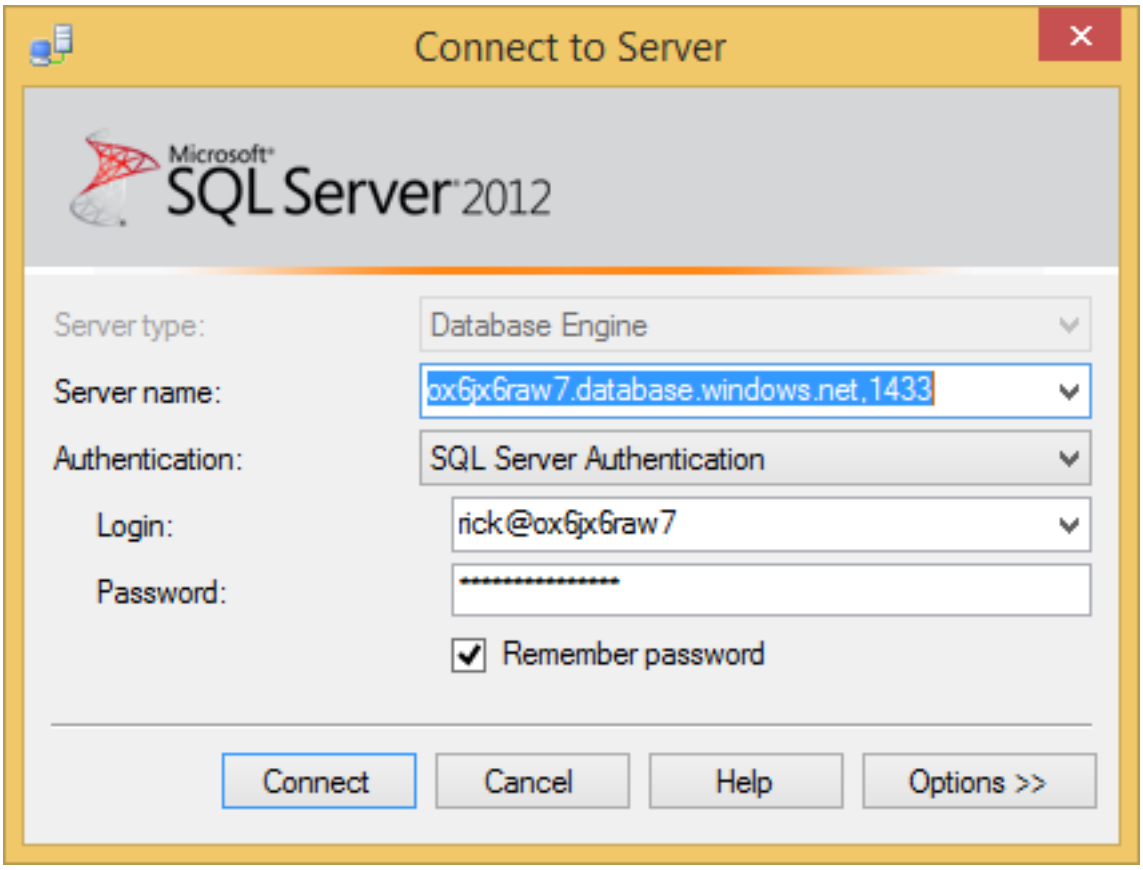
در بالای صفحه لینک **Configure** را کلیک کنید. حالا نام rule جدید، آدرس شروع و پایان را وارد کنید.

در پایین صفحه **Save** را کلیک کنید.

در آخر می‌توانید توسط SSOX به دیتابیس خود متصل شوید. از منوی **View** گزینه **SQL Server Object Explorer** را انتخاب کنید. روی **SQL Server** کلیک راست کرده و **Add SQL Server** را انتخاب کنید.

در دیالوگ **Connect to Server** متد احراز هویت را به **SQL Server Authentication** تغییر دهید. این کار نام سرور و اطلاعات ورود پرتال Windows Azure را به شما می‌دهد.

در مرورگر خود به پرتال مدیریتی بروید و **SQL Databases** را انتخاب کنید. دیتابیس **ContactDB** را انتخاب کرده و روی **View SQL Database connection strings** کلیک کنید. در صفحه **Connection Strings** مقادیر **Server** و **User ID** را کپی کنید. حالا مقادیر را در دیالوگ مذکور در ویژوال استودیو بچسبانید. مقدار فیلد **User ID** در قسمت **Login** وارد می‌شود. در آخر هم کلمه عبوری که هنگام ساختن دیتابیس تنظیم کردید را وارد کنید.



حالا می‌توانید با مراحلی که پیشتر توضیح داده شد به دیتابیس **Contact DB** مراجعه کنید.

افزودن کاربران به نقش canEdit با ویرایش جداول دیتابیس

پیشتر در این مقاله، برای اضافه کردن کاربران به نقش canEdit از یک قطعه کد استفاده کردیم. یک راه دیگر تغییر جداول دیتابیس بصورت مستقیم است. مراحلی که در زیر آمده اند اضافه کردن کاربران به یک نقش را نشان می‌دهند.

در **SQL Server Object Explorer** روی جدول **AspNetUserRoles** کلیک راست کنید و **View Data** را انتخاب کنید.

dbo.AspNetUserRoles [Data] | dbo.AspNetUsers [Data]

Max Rows: 1000

UserId	RoleId
8a5687c2-86ed-40c9-853b-26ef40b7a2bb	e43a4145-7089-4292-9057-af56e5d8e940
9af370af-8055-4cef-965f-990214c24ccf	e43a4145-7089-4292-9057-af56e5d8e940
NULL	NULL

حالا *RoleId* را کپی کنید و در ردیف جدید بچسبانید.

dbo.AspNetUserRoles [Data] | dbo.AspNetUsers [Data]

Max Rows: 1000

UserId	RoleId
8a5687c2-86ed-40c9-853b-26ef40b7a2bb	e43a4145-7089-4292-9057-af56e5d8e940
9af370af-8055-4cef-965f-990214c24ccf	e43a4145-7089-4292-9057-af56e5d8e940
	e43a4145-7089-4292-9057-af56e5d8e940
NULL	NULL

شناسه کاربر مورد نظر را از جدول **AspNetUsers** پیدا کنید و مقدار آن را در ردیف جدید کپی کنید. همین! کاربر جدید شما به نقش canEdit اضافه شد.

نکاتی درباره ثبت نام محلی (Local Registration)

ثبت نام فعلی ما از بازنشانی کلمه‌های عبور (password reset) پشتیبانی نمی‌کند. همچنین اطمینان حاصل نمی‌شود که کاربران سایت انسان هستند (مثلا با استفاده از یک CAPTCHA). پس از آنکه کاربران توسط تامین کنندگان خارجی (مانند گوگل) احراز هویت شدند، می‌توانند در سایت ثبت نام کنند. اگر می‌خواهید ثبت نام محلی را برای اپلیکیشن خود غیرفعال کنید این مراحل را دنبال کنید:

در کنترلر Account متدهای *Register* را ویرایش کنید و خاصیت **AllowAnonymous** را از آنها حذف کنید (هر دو متد GET و POST).

این کار ثبت نام کاربران ناشناس و بدافزارها (bots) را غیر ممکن می‌کند.

در پوشه *Views/Shared* فایل *_LoginPartial.cshtml* را باز کنید و لینک Register را از آن حذف کنید.

در فایل *Views/Account/Login.cshtml* نیز لینک Register را حذف کنید.

اپلیکیشن را دوباره منتشر کنید.

قدم‌های بعدی

برای اطلاعات بیشتر درباره نحوه استفاده از Facebook بعنوان یک تامین کننده احراز هویت، و اضافه کردن اطلاعات پروفایل به قسمت ثبت نام کاربران به لینک زیر مراجعه کنید. Create an ASP.NET MVC 5 App with Facebook and Google OAuth2 and

OpenID Sign-on

برای یادگیری بیشتر درباره ASP.NET MVC 5 هم به سری مقالات Getting Started with ASP.NET MVC 5 می توانید مراجعه کنید. همچنین سری مقالات Getting Started with EF and MVC مطالب خوبی درباره مفاهیم پیشرفته EF ارائه می‌کند.

نظرات خوانندگان

نویسنده: مهمان
تاریخ: ۱۳۹۲/۱۰/۱۹ ۱۴:۴

دوست عزیز

با صبر و حوصله و دقت فراوان یک مقاله خوب را منتشر کردید. ممنون(رای من 5)

عنوان: **ساخت یک اپلیکیشن ساده ToDo با ASP.NET Identity**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۹ ۱۳:۳۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, Tips, ASP.NET MVC 5, ASP.NET Identity, MVC

یک سناریوی فرضی را در نظر بگیرید. اگر بخواهیم IdentityDbContext و دیگر DbContextهای اپلیکیشن را ادغام کنیم چه باید کرد؟ مثلا یک سیستم وبلاگ که برخی کاربران می‌توانند پست جدید ثبت کنند، برخی تنها می‌توانند کامنت بگذارند و تمامی کاربران هم اختیارات مشخص دیگری دارند. در چنین سیستمی شناسه کاربران (User ID) در بسیاری از مدل‌ها (موجودیت‌ها و مدل‌های اپلیکیشن) وجود خواهد داشت تا مشخص شود هر رکورد به کدام کاربر متعلق است. در این مقاله چنین سناریو هایی را بررسی می‌کنیم و best practiceهای مربوطه را مرور می‌کنیم.
در این پست یک اپلیکیشن ساده ToDo خواهیم ساخت که امکان تخصیص to-doها به کاربران را نیز فراهم می‌کند. در این مثال خواهیم دید که چگونه می‌توان مدل‌های مختص به سیستم عضویت (IdentityDbContext) را با مدل‌های دیگر اپلیکیشن مخلوط و استفاده کنیم.

تعریف نیازمندی‌های اپلیکیشن

تنها کاربران احراز هویت شده قادر خواهند بود تا لیست ToDoهای خود را ببینند، آیتم‌های جدید ثبت کنند یا داده‌های قبلی را ویرایش و حذف کنند.
کاربران نباید آیتم‌های ایجاد شده توسط دیگر کاربران را ببینند.
تنها کاربرانی که به نقش Admin تعلق دارند باید بتوانند تمام ToDoهای ایجاد شده را ببینند.

پس بگذارید ببینیم چگونه می‌شود اپلیکیشنی با ASP.NET Identity ساخت که پاسخگوی این نیازمندی‌ها باشد.
ابتدا یک پروژه ASP.NET MVC جدید با مدل احراز هویت Individual User Accounts بسازید. در این اپلیکیشن کاربران قادر خواهند بود تا بصورت محلی در وب سایت ثبت نام کنند و یا با تامین کنندگان دیگری مانند گوگل و فیسبوک وارد سایت شوند.
برای ساده نگاه داشتن این پست ما از حساب‌های کاربری محلی استفاده می‌کنیم.
در مرحله بعد ASP.NET Identity را راه اندازی کنید تا بتوانیم نقش مدیر و یک کاربر جدید بسازیم. می‌توانید با اجرای اپلیکیشن راه اندازی اولیه را انجام دهید. از آنجا که سیستم ASP.NET Identity توسط Entity Framework مدیریت می‌شود می‌توانید از تنظیمات پیکربندی Code First برای راه اندازی دیتابیس خود استفاده کنید.
در قدم بعدی راه انداز دیتابیس را در Global.asax تعریف کنید.

```
Database.SetInitializer<MyDbContext>(new MyDbInitializer());
```

ایجاد نقش مدیر و کاربر جدیدی که به این نقش تعلق دارد

اگر به قطعه کد زیر دقت کنید، می‌بینید که در خط شماره 5 متغیری از نوع UserManager ساخته ایم که امکان اجرای عملیات گوناگونی روی کاربران را فراهم می‌کند. مانند ایجاد، ویرایش، حذف و اعتبارسنجی کاربران. این کلاس که متعلق به سیستم ASP.NET Identity است همتای SQLMembershipProvider در ASP.NET 2.0 است.
در خط 6 یک RoleManager می‌سازیم که امکان کار با نقش‌ها را فراهم می‌کند. این کلاس همتای SQLRoleMembershipProvider در ASP.NET 2.0 است.
در این مثال نام کلاس کاربران (موجودیت کاربر در IdentityDbContext) برابر با "MyUser" است، اما نام پیش فرض در قالب‌های پروژه VS 2013 برابر با "ApplicationUser" می‌باشد.

```
public class MyDbInitializer : DropCreateDatabaseAlways<MyDbContext>
     {
          protected override void Seed(MyDbContext context)
          {
              var UserManager = new UserManager<MyUser>(new

                                                  UserStore<MyUser>(context));

              var RoleManager = new RoleManager<IdentityRole>(new
                                              RoleStore<IdentityRole>(context));
```

```
            string name = "Admin";
            string password = "123456";


            //Create Role Admin if it does not exist
            if (!RoleManager.RoleExists(name))
            {
                var roleresult = RoleManager.Create(new IdentityRole(name));
            }

            //Create User=Admin with password=123456
            var user = new MyUser();
            user.UserName = name;
            var adminresult = UserManager.Create(user, password);

            //Add User Admin to Role Admin
            if (adminresult.Succeeded)
            {
                var result = UserManager.AddToRole(user.Id, name);
            }
            base.Seed(context);
        }
    }
```

حال فایلی با نام Models/AppModels.cs بسازید و مدل EF Code First اپلیکیشن را تعریف کنید. از آنجا که از EF استفاده می‌کنیم، روابط کلیدها بین کاربران و ToDoها بصورت خودکار برقرار می‌شود.

```
public class MyUser : IdentityUser
      {
          public string HomeTown { get; set; }
          public virtual ICollection<ToDo>
                              ToDoes { get; set; }
      }

      public class ToDo
      {
          public int Id { get; set; }
          public string Description { get; set; }
          public bool IsDone { get; set; }
          public virtual MyUser User { get; set; }
      }
```

در قدم بعدی با استفاده از مکانیزم Scaffolding کنترلر جدیدی بهمراه تمام Viewها و متدهای لازم برای عملیات CRUD بسازید.
برای اطلاعات بیشتر درباره نحوه استفاده از مکانیزم Scaffolding به این لینک مراجعه کنید.
لطفا دقت کنید که از DbContext فعلی استفاده کنید. این کار مدیریت داده‌های Identity و اپلیکیشن شما را یکپارچه‌تر می‌کند.
DbContext شما باید چیزی شبیه به کد زیر باشد.

```
public class MyDbContext : IdentityDbContext<MyUser>
      {
          public MyDbContext()
              : base("DefaultConnection")
          {
           }

           protected override void OnModelCreating(DbModelBuilder modelBuilder)
           {
          public System.Data.Entity.DbSet<AspnetIdentitySample.Models.ToDo>
                     ToDoes { get; set; }
      }
```

تنها کاربران احراز هویت شده باید قادر به اجرای عملیات CRUD باشند

برای این مورد از خاصیت Authorize استفاده خواهیم کرد که در MVC 4 هم وجود داشت. برای اطلاعات بیشتر لطفا به این لینک مراجعه کنید.

```
[Authorize]
public class ToDoController : Controller
```

کنترلر ایجاد شده را ویرایش کنید تا کاربران را به ToDoها اختصاص دهد. در این مثال تنها اکشن متدهای Create و List را بررسی خواهیم کرد. با دنبال کردن همین روش می‌توانید متدهای Edit و Delete را هم بسادگی تکمیل کنید.
یک متد constructor جدید بنویسید که آبجکتی از نوع UserManager می‌پذیرد. با استفاده از این کلاس می‌توانید کاربران را در ASP.NET Identity مدیریت کنید.

```
private MyDbContext db;
          private UserManager<MyUser> manager;
          public ToDoController()
          {
              db = new MyDbContext();
              manager = new UserManager<MyUser>(new UserStore<MyUser>(db));
          }
```

اکشن متد Create را بروز رسانی کنید

هنگامی که یک ToDo جدید ایجاد می‌کنید، کاربر جاری را در ASP.NET Identity پیدا می‌کنیم و او را به ToDoها اختصاص می‌دهیم.

```
public async Task<ActionResult> Create
          ([Bind(Include="Id,Description,IsDone")] ToDo todo)
          {
              var currentUser = await manager.FindByIdAsync
                                              (User.Identity.GetUserId());
              if (ModelState.IsValid)
              {
                  todo.User = currentUser;
                  db.ToDoes.Add(todo);
                  await db.SaveChangesAsync();
                  return RedirectToAction("Index");
              }

              return View(todo);
          }
```

اکشن متد List را بروز رسانی کنید

در این متد تنها ToDoهای کاربر جاری را باید بگیریم.

```
public ActionResult Index()
          {
              var currentUser = manager.FindById(User.Identity.GetUserId());

               return View(db.ToDoes.ToList().Where(
                                  todo => todo.User.Id == currentUser.Id));
          }
```

تنها مدیران سایت باید بتوانند تمام ToDoها را ببینند

بدین منظور ما یک اکشن متد جدید به کنترل مربوطه اضافه می‌کنیم که تمام ToDoها را لیست می‌کند. اما دسترسی به این متد را تنها برای کاربرانی که در نقش مدیر وجود دارند میسر می‌کنیم.

```
[Authorize(Roles="Admin")]
          public async Task<ActionResult> All()
          {
```

```
            return View(await db.ToDoes.ToListAsync());
        }
```

نمایش جزئیات کاربران از جدول ToDo ها

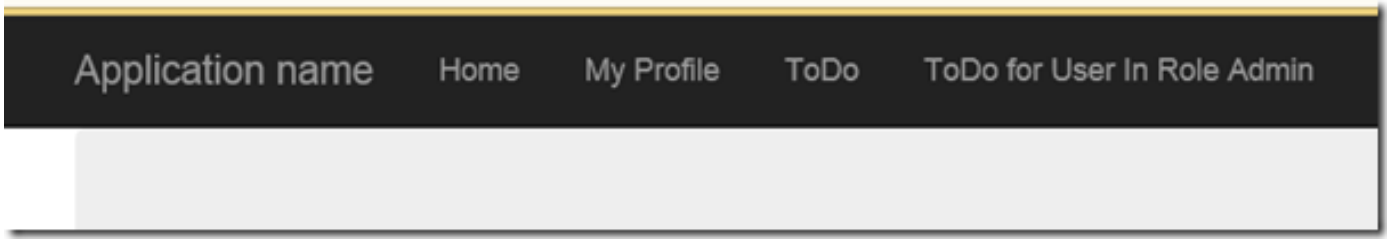
از آنجا که ما کاربران را به ToDo هایشان مرتبط می‌کنیم، دسترسی به داده‌های کاربر ساده است. مثلا در متدی که مدیر سایت تمام آیتم‌ها را لیست می‌کند می‌توانیم به اطلاعات پروفایل تک تک کاربران دسترسی داشته باشیم و آنها را در نمای خود بگنجانیم. در این مثال تنها یک فیلد بنام **HomeTown** اضافه شده است، که آن را در کنار اطلاعات ToDo نمایش می‌دهیم.

```
@model IEnumerable<AspnetIdentitySample.Models.ToDo>

  @{
    ViewBag.Title = "Index";
  }

  <h2>List of ToDoes for all Users</h2>
  <p>
      Notice that we can see the User info (UserName) and profile info such as HomeTown for the user as
well.
      This was possible because we associated the User object with a ToDo object and hence
      we can get this rich behavior.
  12:  </p>

  <table class="table">
      <tr>
          <th>
              @Html.DisplayNameFor(model => model.Description)
          </th>
          <th>
              @Html.DisplayNameFor(model => model.IsDone)
          </th>
          <th>@Html.DisplayNameFor(model => model.User.UserName)</th>
          <th>@Html.DisplayNameFor(model => model.User.HomeTown)</th>
      </tr>
  25:
  26:      @foreach (var item in Model)
  27:      {
  28:          <tr>
  29:              <td>
  30:                  @Html.DisplayFor(modelItem => item.Description)
  31:              </td>
  32:              <td>
                  @Html.DisplayFor(modelItem => item.IsDone)
              </td>
              <td>
                  @Html.DisplayFor(modelItem => item.User.UserName)
              </td>
              <td>
                  @Html.DisplayFor(modelItem => item.User.HomeTown)
              </td>
          </tr>
      }

  </table>
```

صفحه Layout را بروز رسانی کنید تا به ToDoها لینک شود

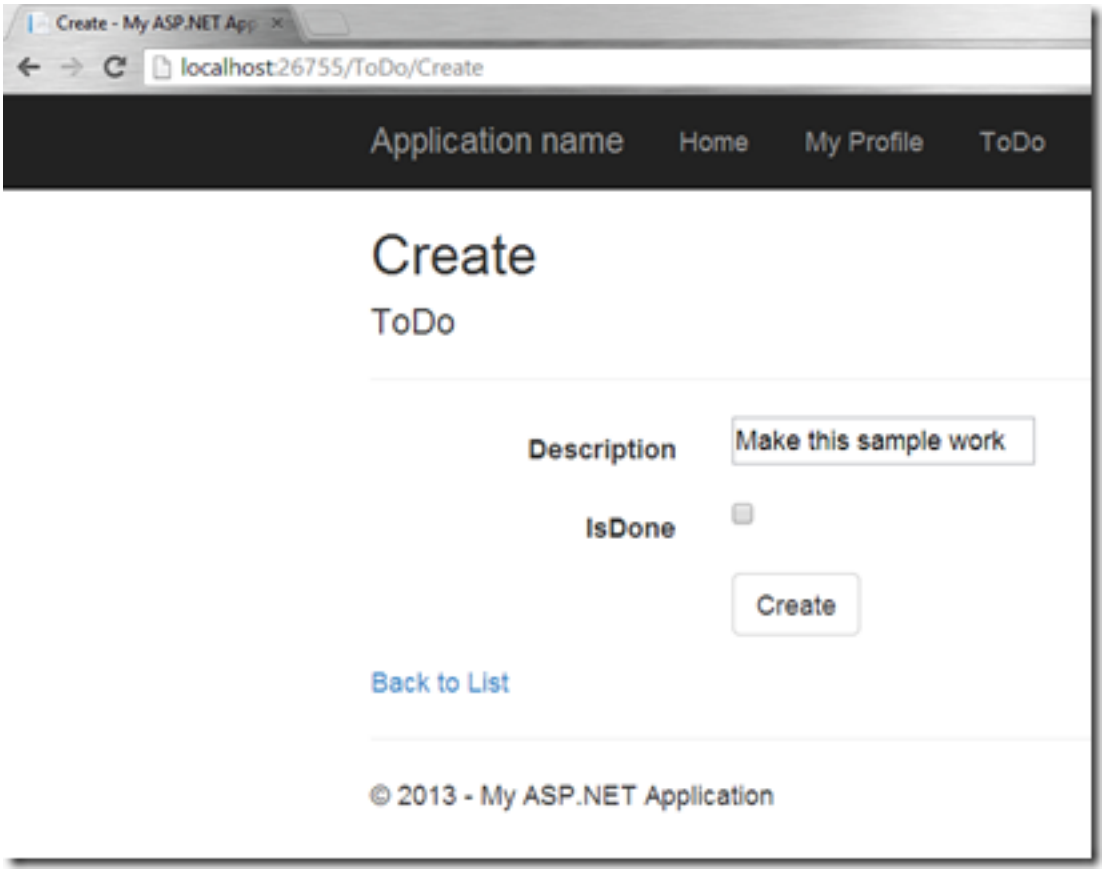
```
<li>@Html.ActionLink("ToDo", "Index", "ToDo")</li>
 <li>@Html.ActionLink("ToDo for User In Role Admin", "All", "ToDo")</li>
```

حال اپلیکیشن را اجرا کنید. همانطور که مشاهده می‌کنید دو لینک جدید به منوی سایت اضافه شده اند.

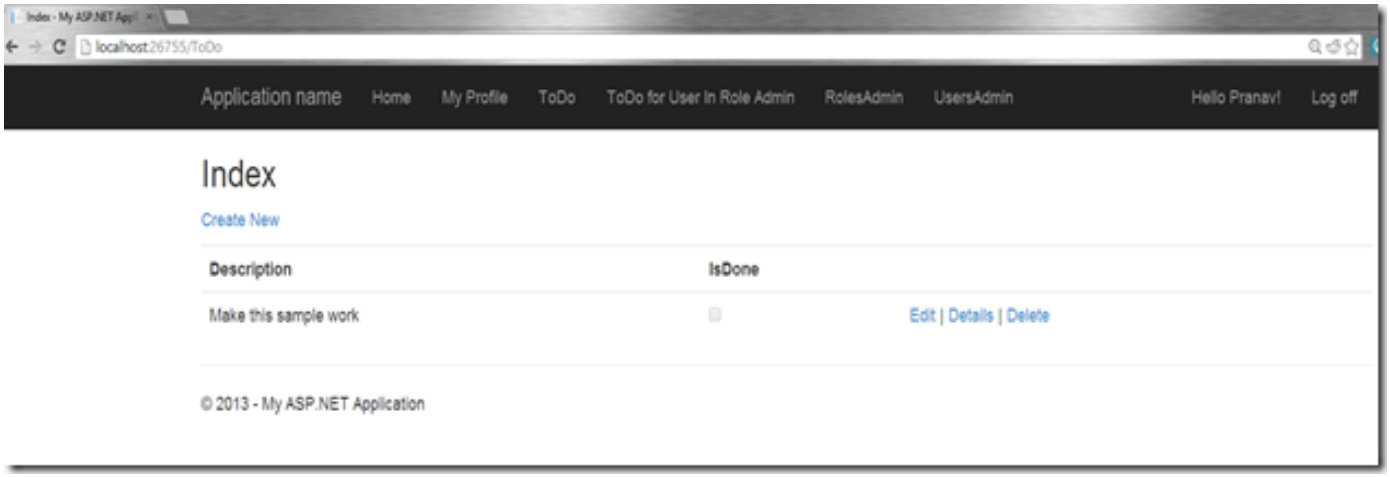


ساخت یک ToDo بعنوان کاربر عادی

روی لینک ToDo کلیک کنید، باید به صفحه ورود هدایت شوید چرا که دسترسی تنها برای کاربران احراز هویت شده تعریف وجود دارد. می‌توانید یک حساب کاربری محلی ساخته، با آن وارد سایت شوید و یک ToDo بسازید.



پس از ساختن یک ToDo می‌توانید لیست رکوردهای خود را مشاهده کنید. دقت داشته باشید که رکوردهایی که کاربران دیگر ثبت کرده اند برای شما نمایش داده نخواهند شد.

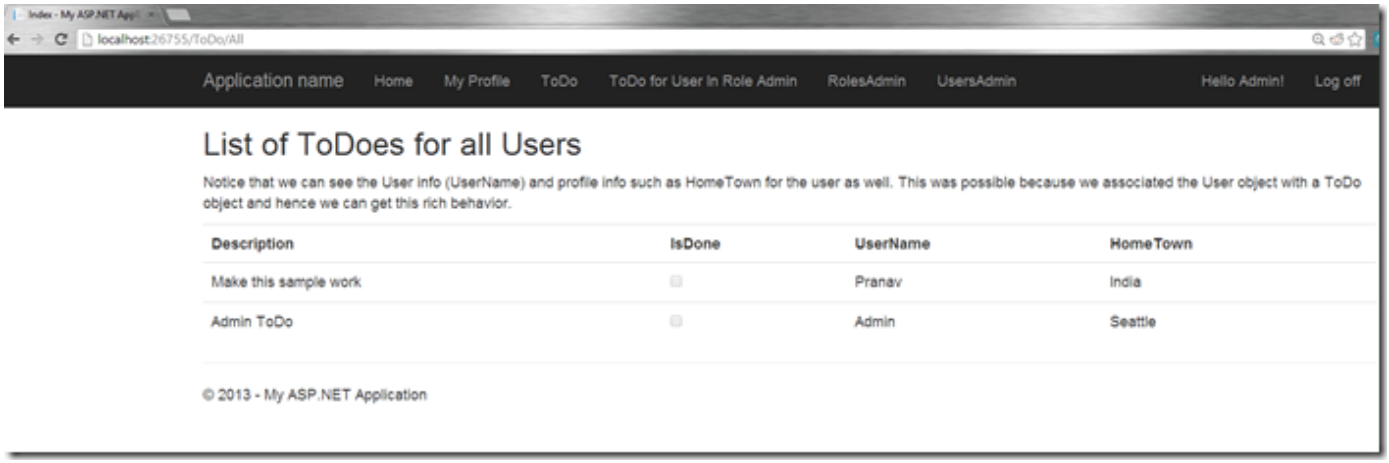


مشاهده تمام ToDoها بعنوان مدیر سایت

روی لینک **ToDoes for User in Role Admin** کلیک کنید. در این مرحله باید مجددا به صفحه ورود هدایت شوید چرا که شما در نقش مدیر نیستید و دسترسی کافی برای مشاهده صفحه مورد نظر را ندارید. از سایت خارج شوید و توسط حساب کاربری مدیری که هنگام راه اندازی اولیه دیتابیس ساخته اید وارد سایت شوید.

```
User = Admin
Password = 123456
```

پس از ورود به سایت بعنوان یک مدیر، می‌توانید ToDoهای ثبت شده توسط تمام کاربران را مشاهده کنید.



نظرات خوانندگان

نویسنده: اس ام
تاریخ: ۱۳۹۳/۰۱/۰۱ ۲۰:۲۴

میشه این پروژه رو برا دانلود بزارید؟ ممنون.

نویسنده: میثم سلیمانی
تاریخ: ۱۳۹۳/۰۴/۲۸ ۱۷:۴۰

```
var currentUser = await manager.FindByIdAsync(User.Identity.GetUserId());
```

این GetUserId() چرا وجود نداره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۲۸ ۱۷:۴۹

در فضای نام Microsoft.AspNet.Identity تعریف شده.

عنوان: **افزودن تصدیق ایمیل به ASP.NET Identity در MVC 5**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۹ ۲۱:۴۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, Tips, ASP.NET MVC 5, ASP.NET Identity, MVC, Security

در پست قبلی نحوه سفارشی کردن پروفایل کاربران در ASP.NET Identity را مرور کردیم. اگر بیاد داشته باشید یک فیلد آدرس ایمیل به کلاس کاربر اضافه کردیم. در این پست از این فیلد استفاده میکنیم تا در پروسه ثبت نام ایمیل‌ها را تصدیق کنیم. بدین منظور پس از ثبت نام کاربران یک ایمیل فعالسازی برای آنها ارسال می‌کنیم که حاوی یک لینک است. کاربران با کلیک کردن روی این لینک پروسه ثبت نام خود را تایید می‌کنند و می‌توانند به سایت وارد شوند. پیش از تایید پروسه ثبت نام، کاربران قادر به ورود نیستند.

در ابتدا باید اطلاعات کلاس کاربر را تغییر دهید تا دو فیلد جدید را در بر گیرد. یک فیلد شناسه تایید (confirmation token) را ذخیره می‌کند، و دیگری فیلدی منطقی است که مشخص می‌کند پروسه ثبت نام تایید شده است یا خیر. پس کلاس *ApplicationUser* حالا باید بدین شکل باشد.

```
public class ApplicationUser : IdentityUser
{
    public string Email { get; set; }
    public string ConfirmationToken { get; set; }
    public bool IsConfirmed { get; set; }
}
```

اگر پیش از این کلاس ApplicationUser را تغییر داده اید، باید مهاجرت‌ها را فعال کنید و دیتابیس را بروز رسانی کنید. حالا می‌توانیم از این اطلاعات جدید در پروسه ثبت نام استفاده کنیم و برای کاربران ایمیل‌های تاییدیه را بفرستیم.

```
private string CreateConfirmationToken()
{
    return ShortGuid.NewGuid();
}

private void SendEmailConfirmation(string to, string username, string confirmationToken)
{
    dynamic email = new Email("RegEmail");
    email.To = to;
    email.UserName = username;
    email.ConfirmationToken = confirmationToken;
    email.Send();
}

//
// POST: /Account/Register
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Register(RegisterViewModel model)
{
    if (ModelState.IsValid)
    {
        string confirmationToken = CreateConfirmationToken();
        var user = new ApplicationUser()
        {
            UserName = model.UserName,
            Email = model.Email,
            ConfirmationToken = confirmationToken,
                IsConfirmed = false };
        var result = await UserManager.CreateAsync(user, model.Password);
        if (result.Succeeded)
        {
            SendEmailConfirmation(model.Email, model.UserName, confirmationToken);
            return RedirectToAction("RegisterStepTwo", "Account");
        }
        else
        {
            AddErrors(result);
        }
    }

    // If we got this far, something failed, redisplay form
```

```
    return View(model);
}
```

برای تولید شناسه‌های تایید (tokens) از کلاسی بنام *ShortGuid* استفاده شده است. این کلاس یک مقدار GUID را encode می‌کند که در نتیجه آن مقدار خروجی کوتاه‌تر بوده و برای استفاده در URLها ایمن است. کد این کلاس را از این وبلاگ گرفته ام. پس از ایجاد حساب کاربری باید شناسه تولید شده را به آن اضافه کنیم و مقدار فیلد IsConfirmed را به false تنظیم کنیم. برای تولید ایمیل‌ها من از Postal استفاده می‌کنم. Postal برای ساختن ایمیل‌های دینامیک شما از موتور Razor استفاده می‌کند. می‌توانید ایمیل‌های ساده (plain text) یا HTML بسازید، عکس و فایل در آن درج و ضمیمه کنید و امکانات بسیار خوب دیگر. اکشن متد RegisterStepTwo تنها کاربر را به یک View هدایت می‌کند که پیامی به او نشان داده می‌شود.

بعد از اینکه کاربر ایمیل را دریافت کرد و روی لینک تایید کلیک کرد به اکشن متد *RegisterConfirmation* باز می‌گردیم.

```
private bool ConfirmAccount(string confirmationToken)
{
    ApplicationDbContext context = new ApplicationDbContext();
    ApplicationUser user =  context.Users.SingleOrDefault(u => u.ConfirmationToken ==
confirmationToken);
    if (user != null)
    {
        user.IsConfirmed = true;
        DbSet<ApplicationUser> dbSet = context.Set<ApplicationUser>();
        dbSet.Attach(user);
        context.Entry(user).State = EntityState.Modified;
        context.SaveChanges();

        return true;
    }
    return false;
}

[AllowAnonymous]
public ActionResult RegisterConfirmation(string Id)
{
    if (ConfirmAccount(Id))
    {
        return RedirectToAction("ConfirmationSuccess");
    }
    return RedirectToAction("ConfirmationFailure");
}
```

متد *ConfirmAccount* سعی می‌کند کاربری را در دیتابیس پیدا کند که شناسه تاییدش با مقدار دریافت شده از URL برابر است. اگر این کاربر پیدا شود، مقدار خاصیت IsConfirmed را به true تغییر می‌دهیم و همین مقدار را به تابع باز می‌گردانیم. در غیر اینصورت false بر می‌گردانیم. اگر کاربر تایید شده است، می‌تواند به سایت وارد شود. برای اینکه مطمئن شویم کاربران پیش از تایید ایمیل شان نمی‌توانند وارد سایت شوند، باید اکشن متد *Login* را کمی تغییر دهیم.

```
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)
{
    if (ModelState.IsValid)
    {
        var user = await UserManager.FindAsync(model.UserName, model.Password);
        if (user != null && user.IsConfirmed)
        {
            await SignInAsync(user, model.RememberMe);
            return RedirectToLocal(returnUrl);
        }
        else
        {
            ModelState.AddModelError("", "Invalid username or password.");
        }
    }

    // If we got this far, something failed, redisplay form
    return View(model);
}
```

تنها کاری که می‌کنیم این است که به دنبال کاربری می‌گردیم که فیلد IsConfirmed آن true باشد. اگر مقدار این فیلد false باشد کاربر را به سایت وارد نمی‌کنیم و پیغام خطایی نمایش می‌دهیم.
همین. این تمام چیزی بود که برای اضافه کردن تصدیق ایمیل به اپلیکیشن خود نیاز دارید. از آنجا که سیستم ASP.NET Identity با Entity Framework مدیریت می‌شود و با مدل Code First ساخته شده، سفارشی کردن اطلاعات کاربران و سیستم عضویت ساده‌تر از همیشه است.

توضیحاتی درباره کار با Postal

اگر به متد SendEmailConfirmation دقت کنید خواهید دید که آبجکتی از نوع Email می‌سازیم (که در اسمبلی‌های Postal وجود دارد) و از آن برای ارسال ایمیل استفاده می‌کنیم. عبارت "RegEmail" نام نمایی است که باید برای ساخت ایمیل استفاده شود. این متغیر از نوع dynamic است، مانند خاصیت ViewBag. بدین معنا که می‌توانید مقادیر مورد نظر خود را بصورت خواص دینامیک روی این آبجکت تعریف کنید. از آنجا که Postal از موتور Razor استفاده می‌کند، بعدا در View ایمیل خود می‌توانید به این مقادیر دسترسی داشته باشید.
در پوشه Views پوشه جدیدی بنام Emails بسازید. سپس یک فایل جدید با نام RegEmail.cshtml در آن ایجاد کنید. کد این فایل را با لیست زیر جایگزین کنید.

```
To: @ViewBag.To
From: YOURNAME@gmail.com
Subject: Confirm your registration

Hello @ViewBag.UserName,
Please confirm your registration by following the link bellow.

@Html.ActionLink(Url.Action("RegisterConfirmation", "Account", new { id = @ViewBag.ConfirmationToken
}), "RegisterConfirmation", "Account", new { id = @ViewBag.ConfirmationToken }, null)
```

این فایل، قالب ایمیل‌های شما خواهد بود. ایمیل‌ها در حال حاظر بصورت plain text ارسال می‌شوند. برای اطلاعات بیشتر درباره ایمیل‌های HTML و امکانات پیشرفته‌تر به سایت پروژه Postal مراجعه کنید.
همانطور که مشاهده می‌کنید در این نما همان خاصیت‌های دینامیک تعریف شده را فراخوانی می‌کنیم تا مقادیر لازم را بدست آوریم.

To. ViewBag آدرس ایمیل گیرنده را نشان می‌دهد.
UserName. ViewBag نام کاربر جاری را نمایش می‌دهد.
ConfirmationToken. ViewBag شناسه تولید شده برای تایید کاربر است.

در این قالب لینکی به متد RegisterConfirmation در کنترلر Account وجود دارد که شناسه تایید را نیز با پارامتری بنام id انتقال می‌دهد.
یک فایل _ViewStart.cshtml هم در این پوشه بسازید و کد آن را با لیست زیر جایگزین کنید.

```
@{ Layout = null; /* Overrides the Layout set for regular page views. */ }
```

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۱۹ ۲۲:۱۴

با تشکر از شما. لطفا View ایمیل ارسالی را (متن حاوی لینک) که توسط کتابخانه Postal پردازش می‌شود، نیز ارسال نمائید. چون الان به نظر متد SendEmailConfirmation مشخص نیست چه متنی را ارسال می‌کند و چطور آن متن را دریافت می‌کند.

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۹ ۲۳:۷

با تشکر از شما، پست بروز رسانی شد.

نویسنده: کاربر
تاریخ: ۱۳۹۲/۱۱/۰۵ ۱۶:۵۲

با تشکر لطفا تنظیمات smtp مربوطه رو هم قرار بدید
سایت سازنده تنظیمات رو در وب کانفیگ قرار داده بود، برای جیمیل من تنظیمات زیر رو مینویسم ولی خطای timed out میده

```
<system.net>
<mailSettings>
<smtp >
  <network host="smtp.gmail.com" userName="My Gmail Email"
  password="my Password" enableSsl="true" port="465" defaultCredentials="true"/>
</smtp>
</mailSettings>
  </system.net>
```

لطفا راهنمایی بفرمایید با تشکر

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۰۵ ۱۹:۵۲

```
<system.net>
  <mailSettings>
    <smtp deliveryMethod="Network" from="armin.zia@gmail.com">
      <network host="smtp.gmail.com" port="587" defaultCredentials="false" enableSsl="true"
userName="YOUR-EMAIL" password="YOUR-PASSWORD" />
    </smtp>
  </mailSettings>
</system.net>
```

نویسنده: داود
تاریخ: ۱۳۹۲/۱۲/۰۵ ۰:۲۷

اگر بخوایم کاربر در فرم لاگین هم با username و هم با email که تأیید شده وارد بشه باید چه کار کنیم؟

نویسنده: Mr.J
تاریخ: ۱۳۹۳/۰۲/۰۱ ۱۵:۱۳

سلام
من دقیقا طبق دستورات بالا کدهام رو نوشتم اما این خطارو میگیره...

```
The SMTP host was not specified.
```

و در قسمت web.config اصلی سایت هم این کدهارو اضافه کردم

```
<system.net>
<mailSettings>
<smtp from="my_gmail">
  <network host="smtp.gmail.com" port="587" defaultCredentials="false" enableSsl="true"
userName="my_gmail" password="mypassword" />
</smtp>
</mailSettings>
  </system.net>
```

مشکل از کجاست.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۰۱ ۱۷:۱۲

نباید مشکلی باشد. مگر اینکه محل قرارگیری تنظیمات system.net شما توسط برنامه قابل یافت شدن نباشد. مثلا آن‌را داخل system.web قرار داده باشید یا مکان دیگری. system.net یک مدخل مجزا و مستقل است. همچنین اگر سایت چندین وب کانفیگ دارد (مانند برنامه‌های ASP.NET MVC)، وب کانفیگ موجود در ریشه سایت باید تنظیم شود و نه مورد موجود در پوشه‌ی Views برنامه.

عنوان: **معرفی کتابخانه Postal برای ASP.NET MVC**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۰ ۰:۱۵
آدرس: www.dotnettips.info
گروه‌ها: Tips, Razor, ASP.Net MVC, MVC, Email, SMTP

Postal کتابخانه ای برای تولید و ارسال ایمیل توسط نماهای ASP.NET MVC است. برای شروع این کتابخانه را به پروژه خود اضافه کنید. پنجره **Package Manager Console** را باز کرده و فرمان زیر را اجرا کنید.

```
PM> Install-Package Postal
```

شروع به کار با Postal

نحوه استفاده از Postal در کنترلرهای خود را در کد زیر مشاهده می‌کنید.

```
using Postal;

public class HomeController : Controller
{
  public ActionResult Index()
  {
      dynamic email = new Email("Example");
      email.To = "webninja@example.com";
      email.FunnyLink = DB.GetRandomLolcatLink();
      email.Send();
      return View();
  }
}
```

Postal نمای ایمیل را در مسیر Views\Emails\Example.cshtml جستجو می‌کند.

```
To: @ViewBag.To
From: lolcats@website.com
Subject: Important Message

Hello,
You wanted important web links right?
Check out this: @ViewBag.FunnyLink

<3
```

پیکربندی SMTP

Postal ایمیل‌ها را توسط **SmtpClient** ارسال می‌کند که در فریم ورک دات نت موجود است. تنظیمات SMTP را می‌توانید در فایل web.config خود پیکربندی کنید. برای اطلاعات بیشتر به MSDN Documentation مراجعه کنید.

```
<configuration>
  ...
  <system.net>
    <mailSettings>
      <smtp deliveryMethod="network">
        <network host="example.org" port="25" defaultCredentials="true"/>
      </smtp>
    </mailSettings>
  </system.net>
  ...
</configuration>
```

ایمیل‌های Strongly-typed

همه خوششان نمی‌آید از آبجکت‌های دینامیک استفاده کنند. علاوه بر آن آبجکت‌های دینامیک مشکلاتی هم دارند. مثلا قابلیت

IntelliSense و یا Compile-time error را نخواهید داشت.

قدم اول - کلاسی تعریف کنید که از Email ارث بری می‌کند.

```
namespace App.Models
{
  public class ExampleEmail : Email
  {
    public string To { get; set; }
    public string Message { get; set; }
  }
}
```

قدم دوم - از این کلاس استفاده کنید!

```
public void Send()
{
  var email = new ExampleEmail
  {
    To = "hello@world.com",
    Message = "Strong typed message"
  };
  email.Send();
}
```

قدم سوم - نمایی ایجاد کنید که از مدل شما استفاده می‌کند. نام نما، بر اساس نام کلاس مدل انتخاب شده است. بنابراین مثلا *ExampleEmail* نمایی با نام *Example.cshtml* لازم دارد.

```
@model App.Models.ExampleEmail
To: @Model.To
From: postal@example.com
Subject: Example

Hello,
@Model.Message
Thanks!
```

آزمون‌های واحد (Unit Testing)

هنگام تست کردن کدهایی که با Postal کار می‌کنند، یکی از کارهایی که می‌خواهید انجام دهید حصول اطمینان از ارسال شدن ایمیل‌ها است. البته در بدنه تست‌ها نمی‌خواهیم هیچ ایمیلی ارسال شود.

Postal یک قرارداد بنام **IEmailService** و یک پیاده سازی پیش فرض از آن بنام **EmailService** ارائه می‌کند، که در واقع ایمیل‌ها را ارسال هم می‌کند. با در نظر گرفتن این پیش فرض که شما از یک IoC Container استفاده می‌کنید (مانند StructureMap, Ninject)، آن را طوری پیکربندی کنید تا یک نمونه از IEmailService به کنترلرها تزریق کند. سپس از این سرویس برای ارسال آبجکت‌های ایمیل‌ها استفاده کنید (بجای فراخوانی متد Email.Send()).

```
public class ExampleController : Controller
{
    public ExampleController(IEmailService emailService)
    {
        this.emailService = emailService;
    }

    readonly IEmailService emailService;

    public ActionResult Index()
    {
        dynamic email = new Email("Example");
        // ...
        emailService.Send(email);
        return View();
    }
}
```

این کنترلر را با ساختن یک Mock از اینترفیس IEmailService تست کنید. یک مثال با استفاده از FakeItEasy را در زیر مشاهده می‌کنید.

```
[Test]
public void ItSendsEmail()
{
    var emailService = A.Fake<IEmailService>();
    var controller = new ExampleController(emailService);
    controller.Index();
    A.CallTo(() => emailService.Send(A<Email>._))
     .MustHaveHappened();
}
```

ایمیل‌های ساده و HTML

Postal ارسال ایمیل‌های ساده (plain text) و HTML را بسیار ساده می‌کند.

قدم اول - نمای اصلی را بسازید. این نما headerها را خواهد داشت و نماهای مورد نیاز را هم رفرنس می‌کند. مسیر نما ~\Views\Emails\Example.cshtml است.

```
To: test@test.com
From: example@test.com
Subject: Fancy email
Views: Text, Html
```

قدوم دوم - نمای تکست را ایجاد کنید. به قوانین نامگذاری دقت کنید، Example.cshtml به Example.Text.cshtml تغییر یافته. مسیر فایل Views\Emails\Example.Text.cshtml است.

```
Content-Type: text/plain; charset=utf-8

Hello @ViewBag.PersonName,
This is a message
```

دقت داشته باشید که تنها یک Content-Type باید تعریف کنید.

قدم سوم - نمای HTML را ایجاد کنید (باز هم فقط با یک Content-Type). مسیر فایل ~\Views\Emails\Example.Html.cshtml است.

```
Content-Type: text/html; charset=utf-8

<html>
  <body>
    <p>Hello @ViewBag.PersonName,</p>
    <p>This is a message</p>
  </body>
</html>
```

ضمیمه ها

برای افزودن ضمائم خود به ایمیل ها، متد Attach را فراخوانی کنید.

```
dynamic email = new Email("Example");
email.Attach(new Attachment("c:\\attachment.txt"));
email.Send();
```

جاسازی تصاویر در ایمیل ها

Postal یک HTML Helper دارد که امکان جاسازی (embedding) تصاویر در ایمیل‌ها را فراهم می‌کند. دیگر نیازی نیست به یک URL خارجی اشاره کنید.

ابتدا مطمئن شوید که فایل web.config شما فضای نام Postal را اضافه کرده است. این کار دسترسی به HTML Helper مذکور در نمای‌های ایمیل را ممکن می‌سازد.

```
<configuration>
  <system.web.webPages.razor>
    <pages pageBaseType="System.Web.Mvc.WebViewPage">
      <namespaces>
        <add namespace="Postal" />
      </namespaces>
    </pages>
  </system.web.webPages.razor>
</configuration>
```

متد **EmbedImage** تصویر مورد نظر را در ایمیل شما جاسازی می‌کند و توسط یک تگ <img/> آن را رفرنس می‌کند.

```
To: john@example.org
From: app@example.org
Subject: Image

@Html.EmbedImage("~/content/postal.jpg")
```

Postal سعی می‌کند تا نام فایل تصویر را، بر اساس مسیر تقریبی ریشه اپلیکیشن شما تعیین کند.

Postal بیرون از ASP.NET

Postal می‌تواند نماهای ایمیل‌ها را بیرون از فضای ASP.NET رندر کند. مثلا در یک اپلیکیشن کنسول یا یک سرویس ویندوز. این امر توسط یک View Engine سفارشی میسر می‌شود. تنها نماهای Razor پشتیبانی می‌شوند. نمونه کدی را در زیر مشاهده می‌کنید.

```
using Postal;

class Program
{
    static void Main(string[] args)
    {
        // Get the path to the directory containing views
        var viewsPath = Path.GetFullPath(@"..\..\Views");

        var engines = new ViewEngineCollection();
        engines.Add(new FileSystemRazorViewEngine(viewsPath));

        var service = new EmailService(engines);

        dynamic email = new Email("Test");
        // Will look for Test.cshtml or Test.vbhtml in Views directory.
        email.Message = "Hello, non-asp.net world!";
        service.Send(email);
    }
}
```

محدودیت ها: نمی توانید برای نمای ایمیل هایتان از Layoutها استفاده کنید. همچنین در نماهای خود تنها از مدل‌ها (Models) می‌توانید استفاده کنید، و نه ViewBag.

Email Headers: برای در بر داشتن نام، در آدرس ایمیل از فرمت زیر استفاده کنید.

```
To: John Smith <john@example.org>
```

Multiple Values: برخی از headerها می‌توانند چند مقدار داشته باشند. مثلا Bcc و CC. اینگونه مقادیر را می‌توانید به دو روش در نمای خود تعریف کنید:

جدا کردن مقادیر با کاما:

```
Bcc: john@smith.com, harry@green.com
Subject: Example
```

```
etc
```

و یا تکرار header:

```
Bcc: john@smith.com
Bcc: harry@green.com
Subject: Example

etc
```

ساختن ایمیل بدون ارسال آن

لازم نیست برای ارسال ایمیل هایتان به Postal تکیه کنید. در عوض می‌توانید یک آبجکت از نوع System.Net.Mail.MailMessage تولید کنید و به هر نحوی که می‌خواهید آن را پردازش کنید. مثلا شاید بخواهید بجای ارسال ایمیل ها، آنها را به یک صف پیام مثل MSMQ انتقال دهید یا بعدا توسط سرویس دیگری ارسال شوند. این آبجکت MailMessage تمامی Header ها، محتوای اصلی ایمیل و ضمائم را در بر خواهد گرفت.

کلاس EmailService در Postal متدی با نام CreateMailMessage فراهم می‌کند.

```
public class ExampleController : Controller
{
    public ExampleController(IEmailService emailService)
    {
        this.emailService = emailService;
    }

    readonly IEmailService emailService;

    public ActionResult Index()
    {
        dynamic email = new Email("Example");
        // ...

        var message = emailService.CreateMailMessage(email);
        CustomProcessMailMessage(message);

        return View();
    }
}
```

در این پست با امکانات اصلی کتابخانه Postal آشنا شدید و دیدید که به سادگی می‌توانید ایمیل‌های Razor بسازید. برای اطلاعات بیشتر لطفا به سایت پروژه Postal مراجعه کنید.

عنوان: پیاده سازی Password Policy سفارشی توسط ASP.NET Identity
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۲ ۲۱:۱۰
آدرس: www.dotnettips.info
گروه‌ها: Tips, Identity, Visual Studio 2013, ASP.NET Identity

برای فراهم کردن یک تجربه کاربری ایمن‌تر و بهتر، ممکن است بخواهید پیچیدگی password policy را سفارشی سازی کنید. مثلا ممکن است بخواهید حداقل تعداد کاراکترها را تنظیم کنید، استفاده از چند حروف ویژه را اجباری کنید، جلوگیری از استفاده نام کاربر در کلمه عبور و غیره. برای اطلاعات بیشتر درباره سیاست‌های کلمه عبور به این لینک مراجعه کنید. بصورت پیش فرض ASP.NET Identity کاربران را وادار می‌کند تا کلمه‌های عبوری بطول حداقل 6 کاراکتر وارد نمایند. در ادامه نحوه افزودن چند خط مشی دیگر را هم بررسی می‌کنیم.

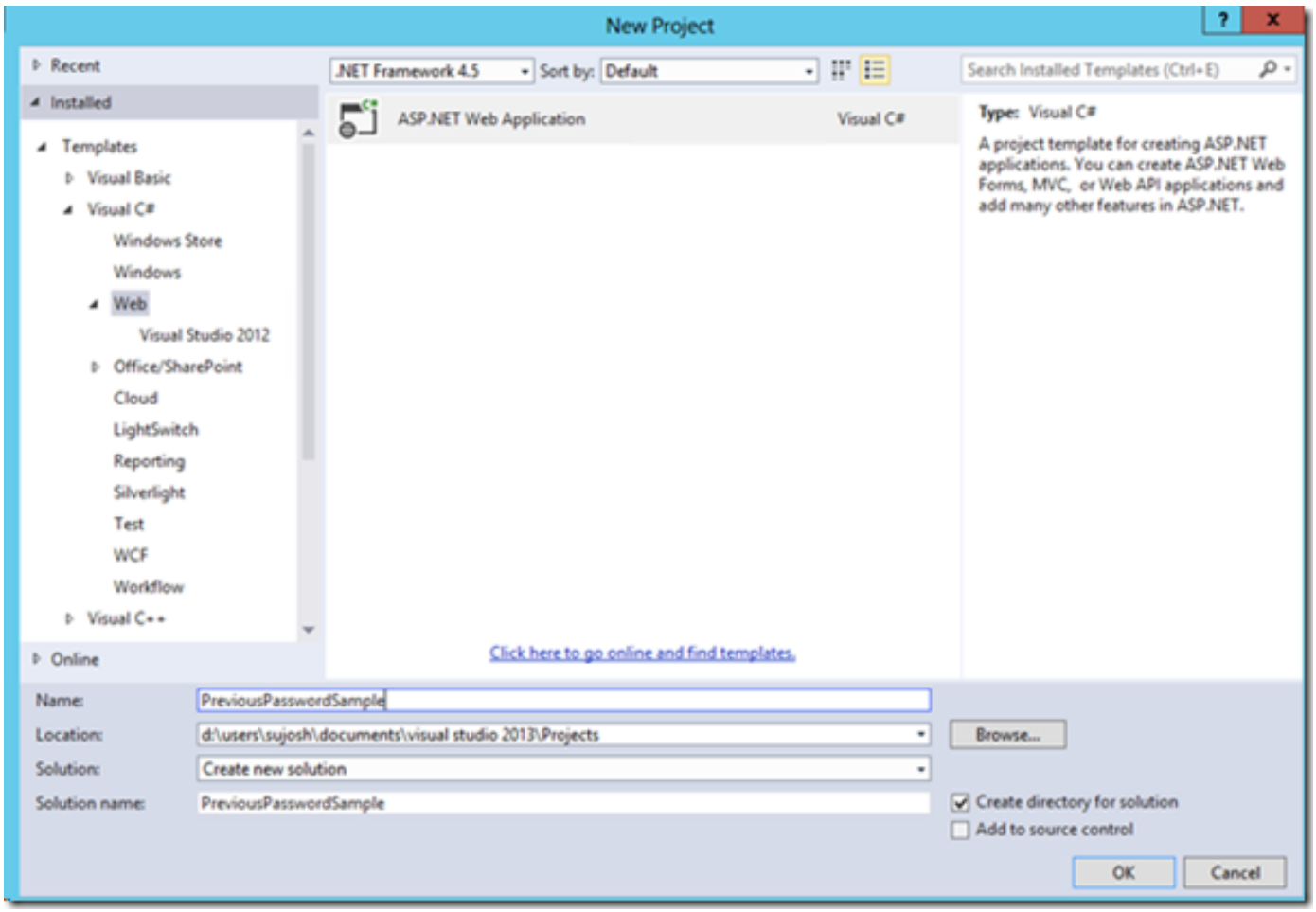
با استفاده از ویژوال استودیو 2013 پروژه جدیدی خواهیم ساخت تا از ASP.NET Identity استفاده کند. مواردی که درباره کلمه‌های عبور می‌خواهیم اعمال کنیم در زیر لیست شده اند.

تنظیمات پیش فرض باید تغییر کنند تا کلمات عبور حداقل 10 کاراکتر باشند
کلمه عبور حداقل یک عدد و یک کاراکتر ویژه باید داشته باشد
امکان استفاده از 5 کلمه عبور اخیری که ثبت شده وجود ندارد

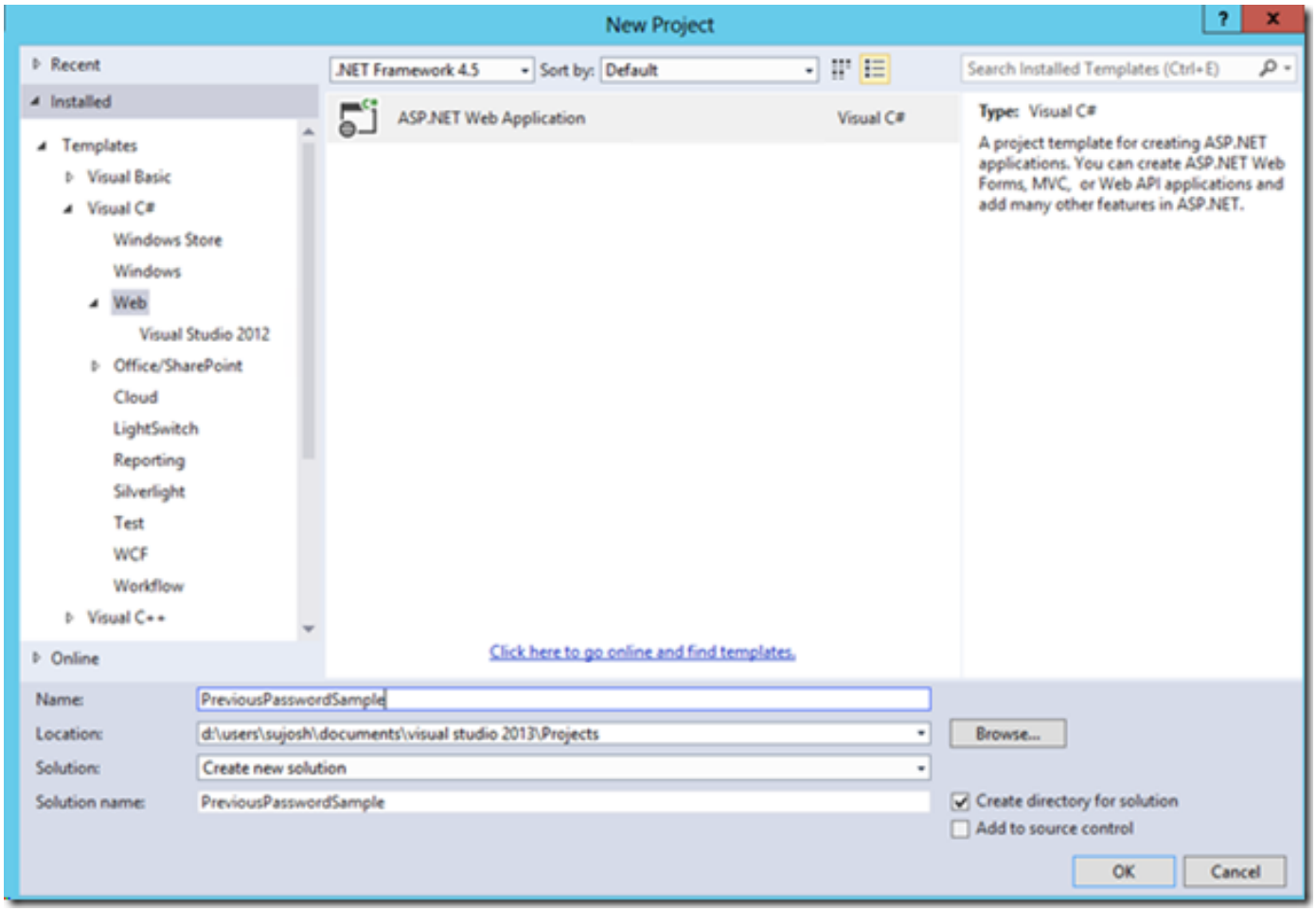
در آخر اپلیکیشن را اجرا می‌کنیم و عملکرد این قوانین جدید را بررسی خواهیم کرد.

ایجاد اپلیکیشن جدید

در Visual Studio 2013 اپلیکیشن جدیدی از نوع ASP.NET MVC 4.5 بسازید.



در پنجره Solution Explorer روی نام پروژه کلیک راست کنید و گزینه Manage NuGet Packages را انتخاب کنید. به قسمت **Update** بروید و تمام انتشارات جدید را در صورت وجود نصب کنید.



بگذارید تا به روند کلی ایجاد کاربران جدید در اپلیکیشن نگاهی بیاندازیم. این به ما در شناسایی نیازهای جدیدمان کمک می‌کند. در پوشه Controllers فایلی بنام AccountController.cs وجود دارد که حاوی متدهایی برای مدیریت کاربران است.

کنترلر Account از کلاس UserManager استفاده می‌کند که در فریم ورک Identity تعریف شده است. این کلاس به نوبه خود از کلاس دیگری بنام UserStore استفاده می‌کند که برای دسترسی و مدیریت داده‌های کاربران استفاده می‌شود. در مثال ما این کلاس از Entity Framework استفاده می‌کند که پیاده سازی پیش فرض است.

متد Register POST یک کاربر جدید می‌سازد. متد CreateAsync به طبع متد 'ValidateAsync' را روی خاصیت PasswordValidator فراخوانی می‌کند تا کلمه عبور دریافتی اعتبارسنجی شود.

```
var user = new ApplicationUser() { UserName = model.UserName };
var result = await UserManager.CreateAsync(user, model.Password);

if (result.Succeeded)
{
    await SignInAsync(user, isPersistent: false);
    return RedirectToAction("Index", "Home");
}
```

در صورت موفقیت آمیز بودن عملیات ایجاد حساب کاربری، کاربر به سایت وارد می‌شود.

قانون 1: کلمه‌های عبور باید حداقل 10 کاراکتر باشند

بصورت پیش فرض خاصیت PasswordValidator در کلاس UserManager به کلاس MinimumLengthValidator تنظیم شده است، که اطمینان حاصل می‌کند کلمه عبور حداقل 6 کاراکتر باشد. هنگام وهله سازی UserManager می‌توانید این مقدار را تغییر دهید.
مقدار حداقل کاراکترهای کلمه عبور به دو شکل می‌تواند تعریف شود. راه اول، تغییر کنترلر Account است. در متد سازنده این کنترلر کلاس UserManager وهله سازی می‌شود، همینجا می‌توانید این تغییر را اعمال کنید. راه دوم، ساختن کلاس جدیدی است که از UserManager ارث بری می‌کند. سپس می‌توان این کلاس را در سطح global تعریف کرد. در پوشه IdentityExtensions کلاس جدیدی با نام ApplicationUserManager بسازید.

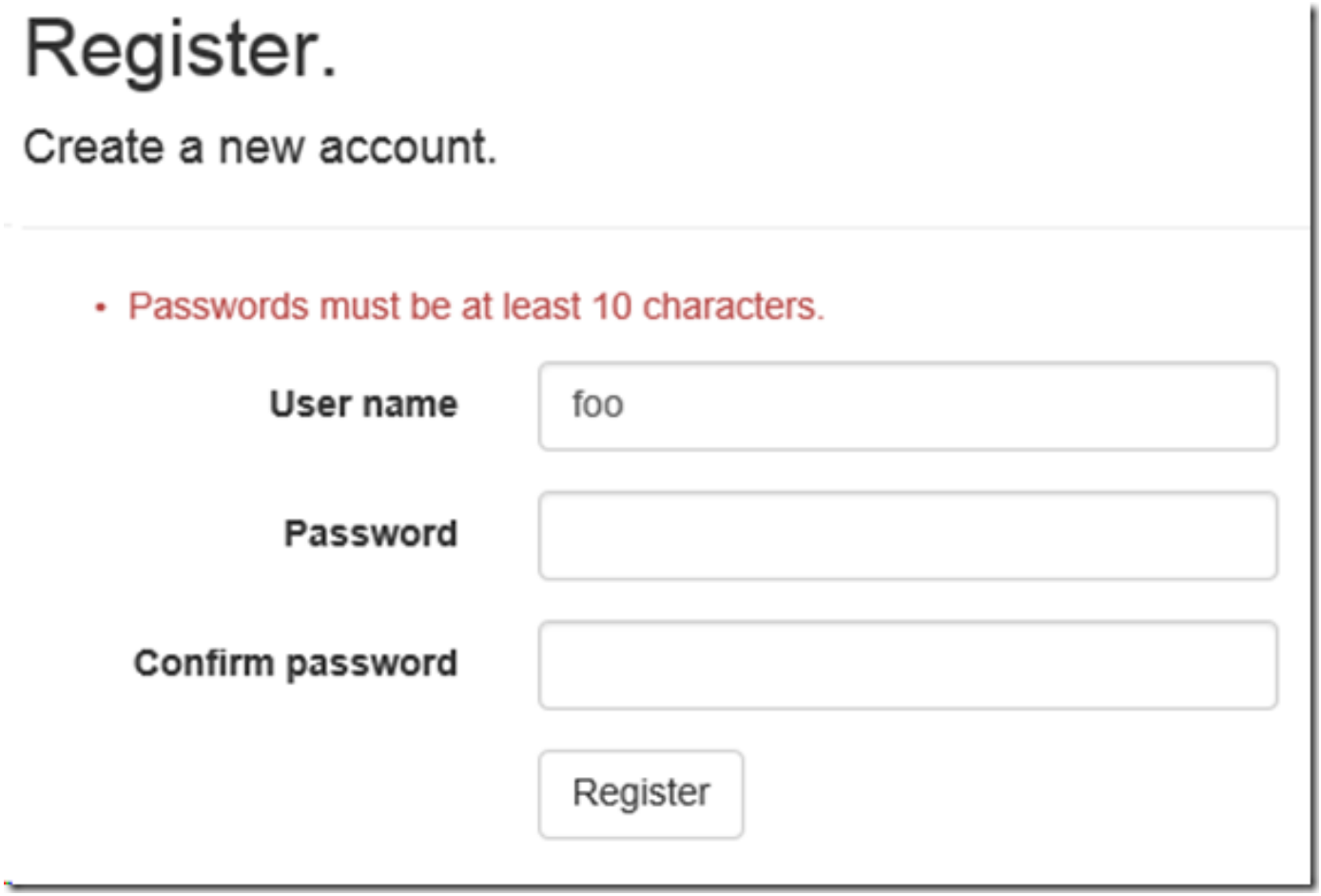
```
public class ApplicationUserManager : UserManager<ApplicationUser>
{
    public ApplicationUserManager(): base(new UserStore<ApplicationUser>(new ApplicationDbContext()))
    {
        PasswordValidator = new MinimumLengthValidator (10);
    }
}
```

کلاس UserManager یک نمونه از کلاس IUserStore را دریافت می‌کند که پیاده سازی APIهای مدیریت کاربران است. از آنجا که کلاس UserStore مبتنی بر Entity Framework است، باید آبجکت DbContext را هم پاس دهیم. این کد در واقع همان کدی است که در متد سازنده کنترلر Account وجود دارد.
یک مزیت دیگر این روش این است که می‌توانیم متدهای UserManager را بازنویسی (overwrite) کنیم. برای پیاده سازی نیازمندهای بعدی دقیقا همین کار را خواهیم کرد.

حال باید کلاس ApplicationUserManager را در کنترلر Account استفاده کنیم. متد سازنده و خاصیت UserManager را مانند زیر تغییر دهید.

```
public AccountController() : this(new ApplicationUserManager())
        {
        }

        public AccountController(ApplicationUserManager userManager)
        {
            UserManager = userManager;
        }
        public ApplicationUserManager UserManager { get; private set; }
```

حالا داریم از کلاس سفارشی جدیدمان استفاده می‌کنیم. این به ما اجازه می‌دهد مراحل بعدی سفارشی سازی را انجام دهیم، بدون آنکه کدهای موجود در کنترلر از کار بیافتند.
اپلیکیشن را اجرا کنید و سعی کنید کاربر محلی جدیدی ثبت نمایید. اگر کلمه عبور وارد شده کمتر از 10 کاراکتر باشد پیغام خطای زیر را دریافت می‌کنید.



قانون 2: کلمه‌های عبور باید حداقل یک عدد و یک کاراکتر ویژه داشته باشند

چیزی که در این مرحله نیاز داریم کلاس جدیدی است که اینترفیس IIdentityValidator را پیاده سازی می‌کند. چیزی که ما می‌خواهیم اعتبارسنجی کنیم، وجود اعداد و کاراکترهای ویژه در کلمه عبور است، همچنین طول مجاز هم بررسی می‌شود. نهایتا این قوانین اعتبارسنجی در متد 'ValidateAsync' بکار گرفته خواهند شد.

در پوشه IdentityExtensions کلاس جدیدی بنام CustomPasswordValidator بسازید و اینترفیس مذکور را پیاده سازی کنید. از آنجا که نوع کلمه عبور رشته (string) است از IIdentityValidator<string> استفاده می‌کنیم.

```
public class CustomPasswordValidator : IIdentityValidator<string>
{
    public int RequiredLength { get; set; }

    public CustomPasswordValidator(int length)
    {
        RequiredLength = length;
    }

    public Task<IdentityResult> ValidateAsync(string item)
    {
        if (String.IsNullOrEmpty(item) || item.Length < RequiredLength)
        {
            return Task.FromResult(IdentityResult.Failed(String.Format("Password should be of length
{0}",RequiredLength)));
        }

    string pattern = @"^(?=.*[0-9])(?=.*[!@#$%^&*])[0-9a-zA-Z!@#$%^&*0-9]{10,}$";

    if (!Regex.IsMatch(item, pattern))
    {
        return Task.FromResult(IdentityResult.Failed("Password should have one numeral and one special
character"));
    }
```

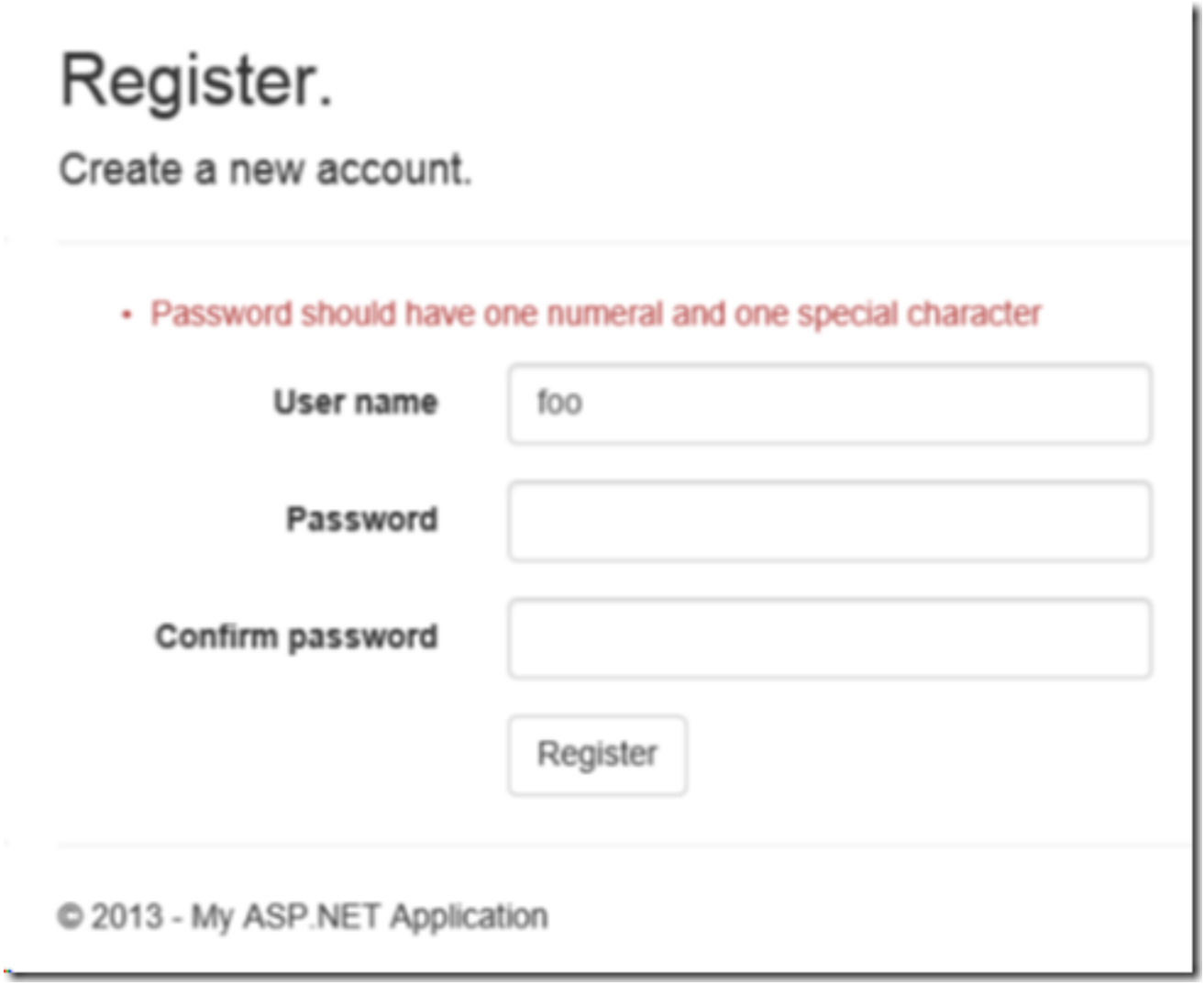
```
    return Task.FromResult(IdentityResult.Success);
 }
```

در متد **ValidateAsync** بررس می‌کنیم که طول کلمه عبور معتبر و مجاز است یا خیر. سپس با استفاده از یک RegEx وجود کاراکترهای ویژه و اعداد را بررسی می‌کنیم. دقت کنید که regex استفاده شده تست نشده و تنها بعنوان یک مثال باید در نظر گرفته شود.

قدم بعدی تعریف این اعتبارسنج سفارشی در کلاس UserManager است. باید مقدار خاصیت PasswordValidator را به این کلاس تنظیم کنیم. به کلاس ApplicationUserManager که پیشتر ساختید بروید و مقدار خاصیت PasswordValidator را به CustomPasswordValidator تغییر دهید.

```
public class ApplicationUserManager : UserManager<ApplicationUser>
{
    public ApplicationUserManager() : base(new UserStore<ApplicationUser(new ApplicationDbContext()))
    {
        PasswordValidator = new CustomPasswordValidator(10);
    }
 }
```

هیچ تغییر دیگری در کلاس AccountController لازم نیست. حال سعی کنید کاربر جدید دیگری بسازید، اما اینبار کلمه عبوری وارد کنید که خطای اعتبارسنجی تولید کند. پیغام خطایی مشابه تصویر زیر باید دریافت کنید.



قانون 3: امکان استفاده از 5 کلمه عبور اخیر ثبت شده وجود ندارد

هنگامی که کاربران سیستم، کلمه عبور خود را بازنشانی (reset) می‌کنند یا تغییر می‌دهند، می‌توانیم بررسی کنیم که آیا مجددا از یک کلمه عبور پیشین استفاده کرده اند یا خیر. این بررسی بصورت پیش فرض انجام نمی‌شود، چرا که سیستم Identity تاریخچه کلمه‌های عبور کاربران را ذخیره نمی‌کند. می‌توانیم در اپلیکیشن خود جدول جدیدی بسازیم و تاریخچه کلمات عبور کاربران را در آن ذخیره کنیم. هربار که کاربر سعی در بازنشانی یا تغییر کلمه عبور خود دارد، مقدار Hash شده را در جدول تاریخچه بررسی میکنیم.

فایل IdentityModels.cs را باز کنید. مانند لیست زیر، کلاس جدیدی بنام 'PreviousPassword' بسازید.

```
public class PreviousPassword
{

 public PreviousPassword()
 {
    CreateDate = DateTimeOffset.Now;
 }

 [Key, Column(Order = 0)]

 public string PasswordHash { get; set; }

 public DateTimeOffset CreateDate { get; set; }

 [Key, Column(Order = 1)]

 public string UserId { get; set; }

 public virtual ApplicationUser User { get; set; }

 }
```

در این کلاس، فیلد 'Password' مقدار Hash شده کلمه عبور را نگاه میدارد و توسط فیلد 'UserId' رفرنس می‌شود. فیلد 'CreateDate' یک مقدار timestamp ذخیره می‌کند که تاریخ ثبت کلمه عبور را مشخص می‌نماید. توسط این فیلد می‌توانیم تاریخچه کلمات عبور را فیلتر کنیم و مثلا 5 رکورد آخر را بگیریم.

Entity Framework Code First جدول 'PreviousPasswords' را می‌سازد و با استفاده از فیلدهای 'UserId' و 'Password' کلید اصلی (composite primary key) را ایجاد می‌کند. برای اطلاعات بیشتر درباره قرارداهای EF Code First به این لینک مراجعه کنید.

خاصیت جدیدی به کلاس ApplicationUser اضافه کنید تا لیست آخرین کلمات عبور استفاده شده را نگهداری کند.

```
public class ApplicationUser : IdentityUser
{
    public ApplicationUser() : base()
    {
        PreviousUserPasswords = new List<PreviousPassword>();
     }

     public virtual IList<PreviousPassword> PreviousUserPasswords { get; set; }
}
```

همانطور که پیشتر گفته شد، کلاس UserStore پیاده سازی APIهای لازم برای مدیریت کاربران را در بر می‌گیرد. هنگامی که کاربر برای نخستین بار در سایت ثبت می‌شود باید مقدار Hash کلمه عبورش را در جدول تاریخچه کلمات عبور ذخیره کنیم. از آنجا که UserStore بصورت پیش فرض متدی برای چنین عملیاتی معرفی نمی‌کند، باید یک override تعریف کنیم تا این مراحل را انجام دهیم. پس ابتدا باید کلاس سفارشی جدیدی بسازیم که از UserStore ارث بری کرده و آن را توسعه می‌دهد. سپس از این کلاس سفارشی در ApplicationUserManager بعنوان پیاده سازی پیش فرض UserStore استفاده می‌کنیم. پس کلاس جدیدی در پوشه IdentityExtensions ایجاد کنید.

```
public class ApplicationUserStore : UserStore<ApplicationUser>
{
    public ApplicationUserStore(DbContext context) : base(context)  { }
```

```
    public override async Task CreateAsync(ApplicationUser user)
    {
        await base.CreateAsync(user);
        await AddToPreviousPasswordsAsync(user, user.PasswordHash);
    }

     public Task AddToPreviousPasswordsAsync(ApplicationUser user, string password)
     {
        user.PreviousUserPasswords.Add(new PreviousPassword() { UserId = user.Id, PasswordHash =
password });

        return UpdateAsync(user);
     }
 }
```

متد 'AddToPreviousPasswordsAsync' کلمه عبور را در جدول 'PreviousPasswords' ذخیره می‌کند.

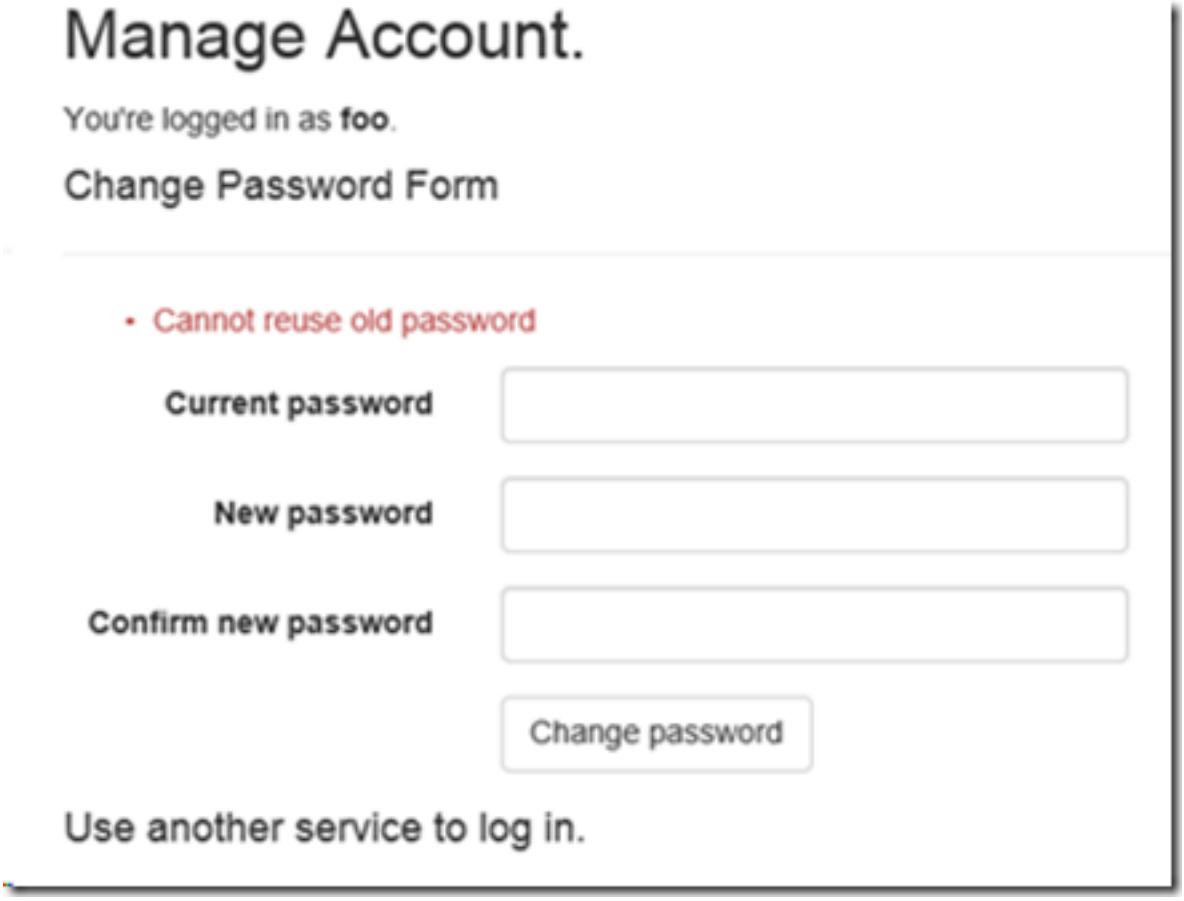
هرگاه کاربر سعی در بازنشانی یا تغییر کلمه عبورش دارد باید این متد را فراخوانی کنیم. APIهای لازم برای این کار در کلاس UserManager تعریف شده اند. باید این متدها را override کنیم و فراخوانی متد مذکور را پیاده کنیم. برای این کار کلاس ApplicationUserManager را باز کنید و متدهای ChangePassword و ResetPassword را بازنویسی کنید.

```
public class ApplicationUserManager : UserManager<ApplicationUser>
    {
        private const int PASSWORD_HISTORY_LIMIT = 5;

        public ApplicationUserManager() : base(new ApplicationUserStore(new ApplicationDbContext()))
        {
            PasswordValidator = new CustomPasswordValidator(10);
        }

        public override async Task<IdentityResult> ChangePasswordAsync(string userId, string
currentPassword, string newPassword)
        {
            if (await IsPreviousPassword(userId, newPassword))
            {
                return await Task.FromResult(IdentityResult.Failed("Cannot reuse old password"));
            }

            var result = await base.ChangePasswordAsync(userId, currentPassword, newPassword);

            if (result.Succeeded)
            {
                var store = Store as ApplicationUserStore;
                await store.AddToPreviousPasswordsAsync(await FindByIdAsync(userId),
PasswordHasher.HashPassword(newPassword));
            }

            return result;
        }

        public override async Task<IdentityResult> ResetPasswordAsync(string userId, string token,
string newPassword)
        {
            if (await IsPreviousPassword(userId, newPassword))
            {
                return await Task.FromResult(IdentityResult.Failed("Cannot reuse old password"));
            }

            var result = await base.ResetPasswordAsync(userId, token, newPassword);

            if (result.Succeeded)
            {
                var store = Store as ApplicationUserStore;
                await store.AddToPreviousPasswordsAsync(await FindByIdAsync(userId),
PasswordHasher.HashPassword(newPassword));
            }

            return result;
        }

        private async Task<bool> IsPreviousPassword(string userId, string newPassword)
        {
            var user = await FindByIdAsync(userId);
```

```
            if (user.PreviousUserPasswords.OrderByDescending(x => x.CreateDate).
                Select(x => x.PasswordHash).Take(PASSWORD_HISTORY_LIMIT)
                .Where(x => PasswordHasher.VerifyHashedPassword(x, newPassword) !=
PasswordVerificationResult.Failed).Any())
            {
                return true;
            }

            return false;
        }
    }
```

فیلد 'PASSWORD_HISTORY_LIMIT' برای دریافت X رکورد از جدول تاریخچه کلمه عبور استفاده می‌شود. همانطور که می‌بینید از متد سازنده کلاس ApplicationUserStore برای گرفتن متد جدیدمان استفاده کرده ایم. هرگاه کاربری سعی می‌کند کلمه عبورش را بازنشانی کند یا تغییر دهد، کلمه عبورش را با 5 کلمه عبور قبلی استفاده شده مقایسه می‌کنیم و بر این اساس مقدار true/false بر می‌گردانیم.

کاربر جدیدی بسازید و به صفحه **Manage** بروید. حال سعی کنید کلمه عبور را تغییر دهید و از کلمه عبور فعلی برای مقدار جدید استفاده کنید تا خطای اعتبارسنجی تولید شود. پیغامی مانند تصویر زیر باید دریافت کنید.



سورس کد این مثال را می‌توانید از این لینک دریافت کنید. نام پروژه **Identity-PasswordPolicy** است، و زیر قسمت Samples/Identity قرار دارد.

عنوان: **استفاده از Web Fonts در اپلیکیشن های ASP.NET MVC**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۴ ۲۰:۲۰
آدرس: www.dotnettips.info
گروه‌ها: MVC, Tips, Web fonts

Typography در طراحی وب

برای بسیاری از توسعه دهندگان، طراحی یک اپلیکیشن وب شاید مقوله زیاد جالبی نباشد. مگر آنکه با طراحان حرفه ای سر و کار داشته باشند، و یا خودشان در زمینه طراحی فعال و با استعداد باشند. توسعه دهندگان می‌توانند به راحتی از اهمیت یا نیاز به Typography بگذرند، و تنها روی المان‌های گرافیکی پایه ای وقت بگذارند که در اپلیکیشن وب نهایتا استفاده خواهند شد.

به همین دلیل است که فریم ورک‌های HTML/CSS هر روزه محبوب‌تر می‌شوند. فریم ورک هایی مانند Twitter Bootstrap, HTML5 Boilerplate, Foundation و غیره. این فریم ورک‌ها طراحی و ساخت اپلیکیشن‌های وب را ساده‌تر و سریع‌تر می‌کنند، چرا که بسیاری از نیازهای رایج در طراحی وب پیاده سازی شده اند و دیگر نیازی به اختراع مجدد آنها نیست. اما به غیر از طراحی وب سایتی با ظاهری شیک و پسندیده، مقوله Typography شاید آخرین چیزی باشد که توسعه دهندگان به آن فکر می‌کنند.

Typography می‌تواند ظاهر بصری و تجربه کاربری را بسیار بهتر کند. انتخاب font-typeهای مناسب، وزن و استایل آنها می‌تواند به القای ایده ها، اهمیت مطالب و احساسات به کاربر کمک کند. همچنین خوانایی و درک مطالب شما هم بهبود می‌یابد، که همگی به کیفیت کلی طراحی شما کمک خواهند کرد.

انتخاب و استفاده از فونت ها

چالش انتخاب فونت برای وب، این است که باید بدانید آنها دقیقا چگونه برای کاربران رندر می‌شوند. برای اینکه فونت‌های شما همانطور که قصد دارید رندر شوند، مرورگر کاربر باید به فایل فونت‌ها دسترسی داشته باشد. ممکن است این فونت‌ها در سیستم عاملی که مرورگر در آن اجرا می‌شود، وجود نداشته باشند یا دسترسی به آنها ممکن نباشد.

طراحان وب طی سالیان متمادی، از فونت هایی استفاده می‌کردند که احتمال وجود آنها روی بازه گسترده ای از کامپیوترها زیاد بود. اینگونه فونت‌ها با عنوان 'web-safe' یاد می‌شوند:

Arial
Courier New
Helvetica
Times New Roman
Trebuchet
Georgia
Verdana

این لیست کوتاه، تنوع بسیار محدودی را در اختیار طراحان قرار می‌داد. برای رفع این محدودیت تکنیک‌های مختلفی استفاده می‌شد. مثلا متون مورد نظر را بصورت تصویر درج می‌کردند، یا با استفاده از افزونه ای در مرورگر از فونت خاصی استفاده می‌شد. مشکل عمده این روش‌ها این بود که متون مورد نظر قابل انتخاب و جستجو نبودند.

Web Fonts

یکی از راه حل‌ها برای رفع این محدودیت، توسعه فونت‌های وب بود. فونت‌های وب روی هر پلفترمی قابل استفاده هستند و توسط یک درخواست HTTP بارگذاری می‌شوند. با استفاده از فونت‌های وب، متون ما قابل انتخاب، جستجو و ترجمه به زبان‌های دیگر هستند. ناگفته نماند که بازه type-faceهای بسیار گسترده‌تری هم در دست داریم. از طرف دیگر به دلیل اینکه اکثر مرورگرهای وب امروزی از Web Fonts پشتیبانی می‌کنند، می‌توان اطمینان داشت که خروجی مورد نظر تقریبا روی تمام پلتفرم‌ها یکسان خواهد بود.

استفاده از فونت‌های وب در اپلیکیشن‌های ASP.NET MVC

خوب، چگونه می‌توانیم از فونت‌های وب در اپلیکیشن‌های MVC استفاده کنیم؟ در ادامه یک نمونه را بررسی میکنیم.

1. یک سرویس فونت انتخاب کنید. سرویس دهندگان زیادی وجود دارند که فونت‌های وب رایگان و پولی متنوعی را فراهم می‌کنند. برخی از سرویس دهندگان محبوب:

Google Web Fonts

Fonts.com

Typekit

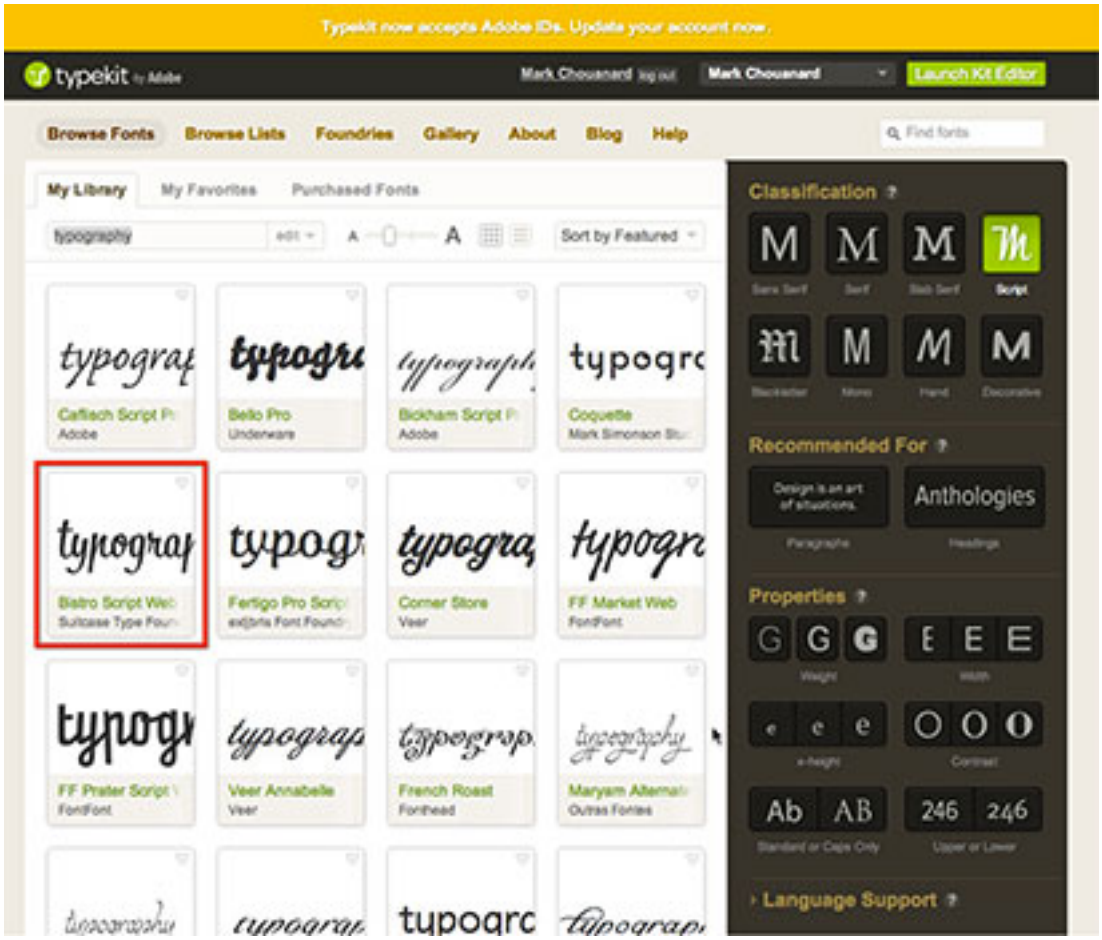
Font Squirrel

Webtype

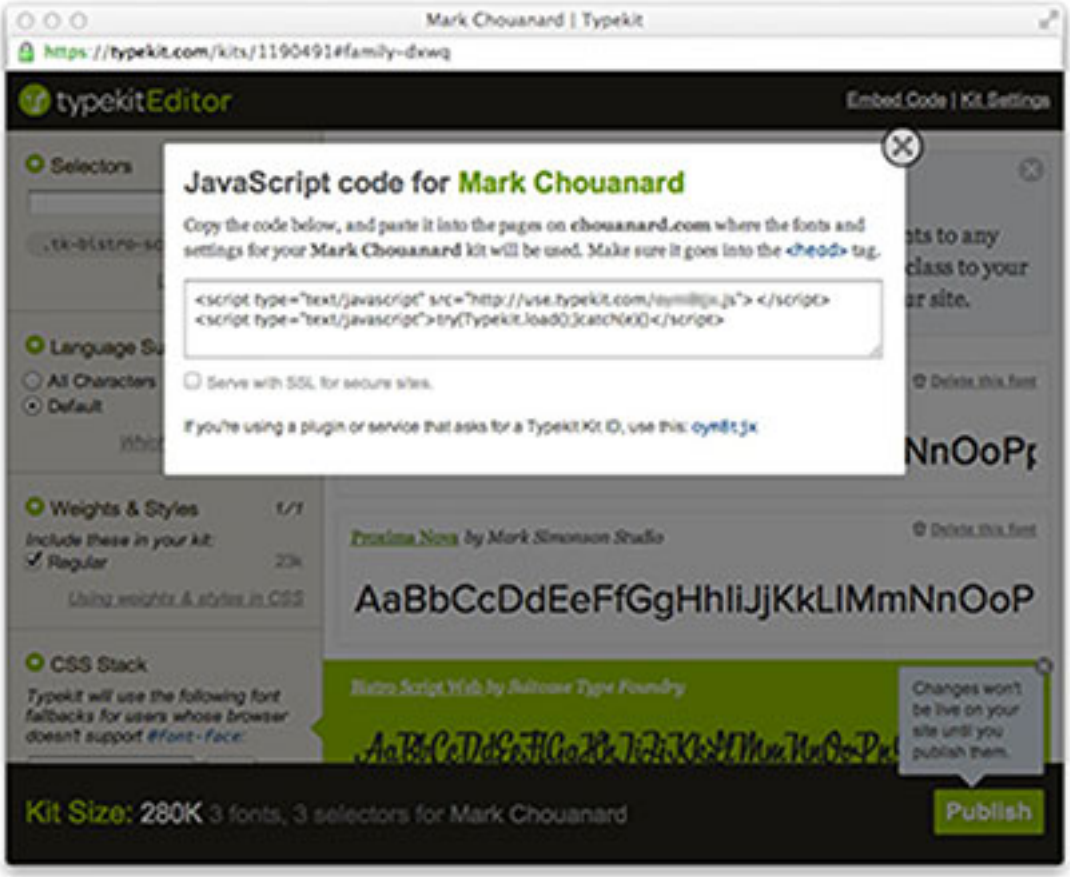
Fontdeck

WebINK

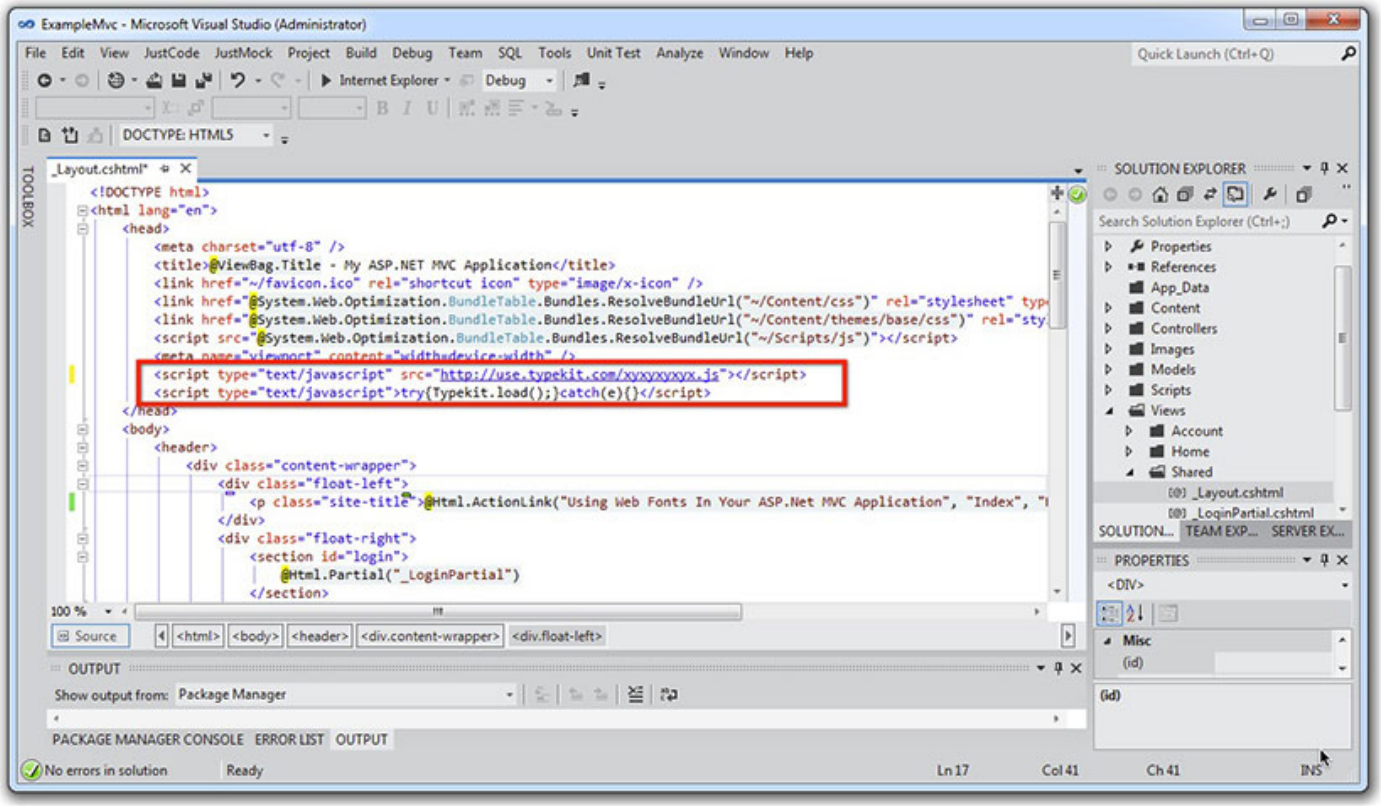
در مثال جاری از سرویس Typekit استفاده خواهیم کرد. نحوه استفاده از دیگر سرویس‌ها هم تقریبا یکسان است.
2. فونت مورد نظر را انتخاب کنید. سرویس‌های مذکور کتابخانه‌های بزرگی از فونت‌های وب دارند، که توسط رابط کاربری قوی آنها می‌توانید رندر نهایی فونت‌ها را مشاهده کنید. همچنین می‌توانید لیست فونت‌های موجود را بر اساس پارامترهای مختلفی مانند خواص، طبقه بندی، توصیه شده‌ها و غیره فیلتر کنید. برای مثال جاری فونت Bistro Script Web را انتخاب می‌کنیم.



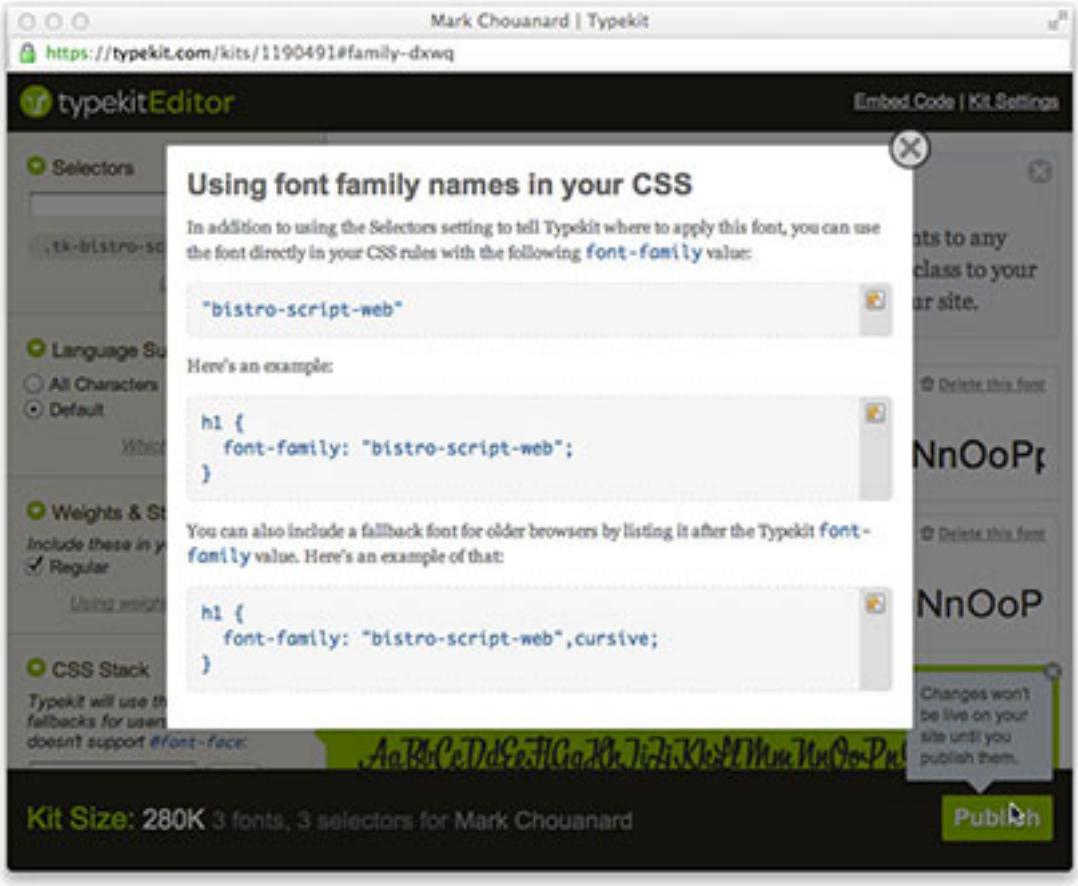
3. کد جاوا اسکریپت خود را تولید کنید (تمام سرویس دهندگان این امکان را پیاده سازی کرده اند).



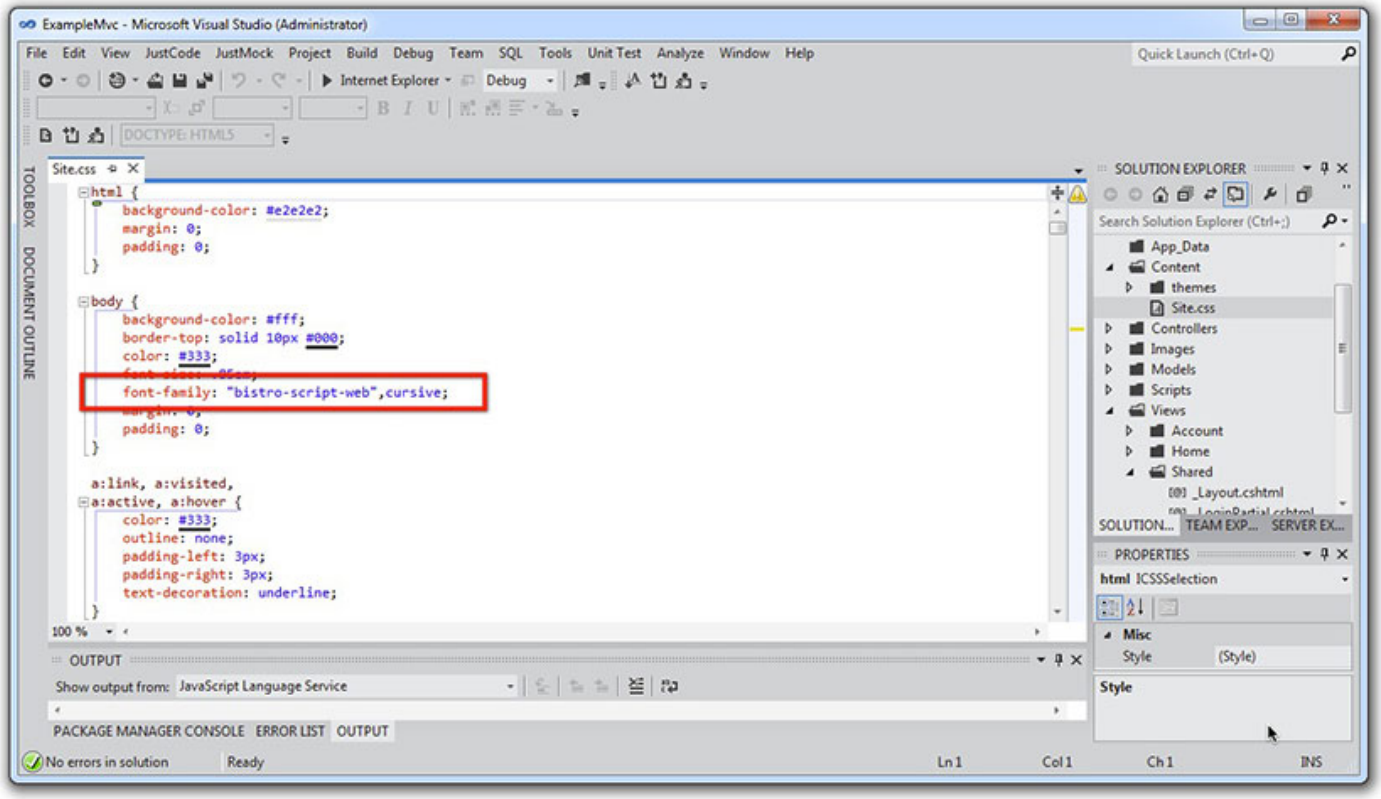
4. قطعه کد جاوا اسکریپت تولید شده را، در قسمت <head> فایل _Layout.cshtml (Razor) یا Site.Master (ASPX) کپی کنید.



5. CSS Selectorهای لازم برای فونت مورد نظر را تولید کنید.

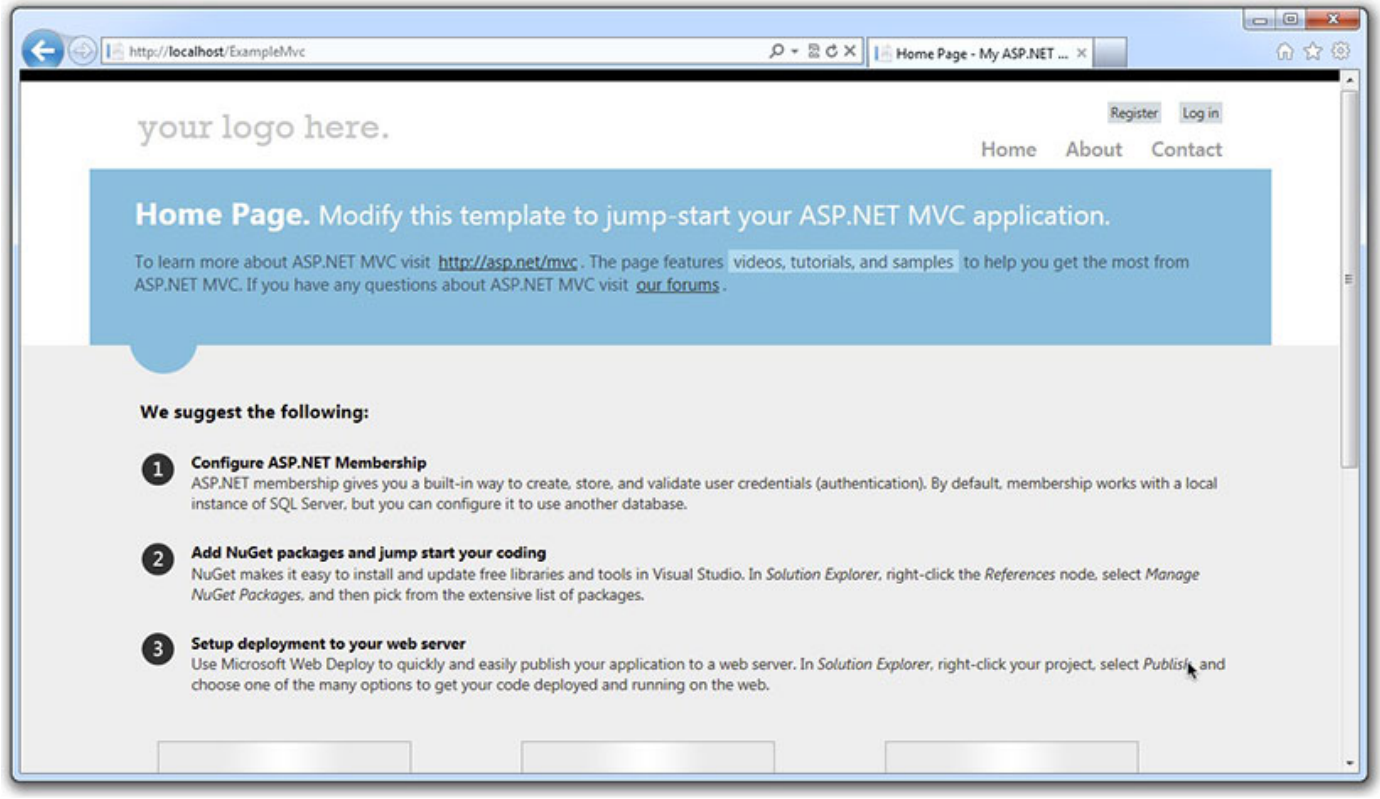


6. کد css تولید شده را در فایل Site.css کپی کنید. در مثال جاری فونت کل اپلیکیشن را تغییر می‌دهیم. برای اینکار، خانواده فونت "bistro-script-web" را به تگ body اضافه می‌کنیم.

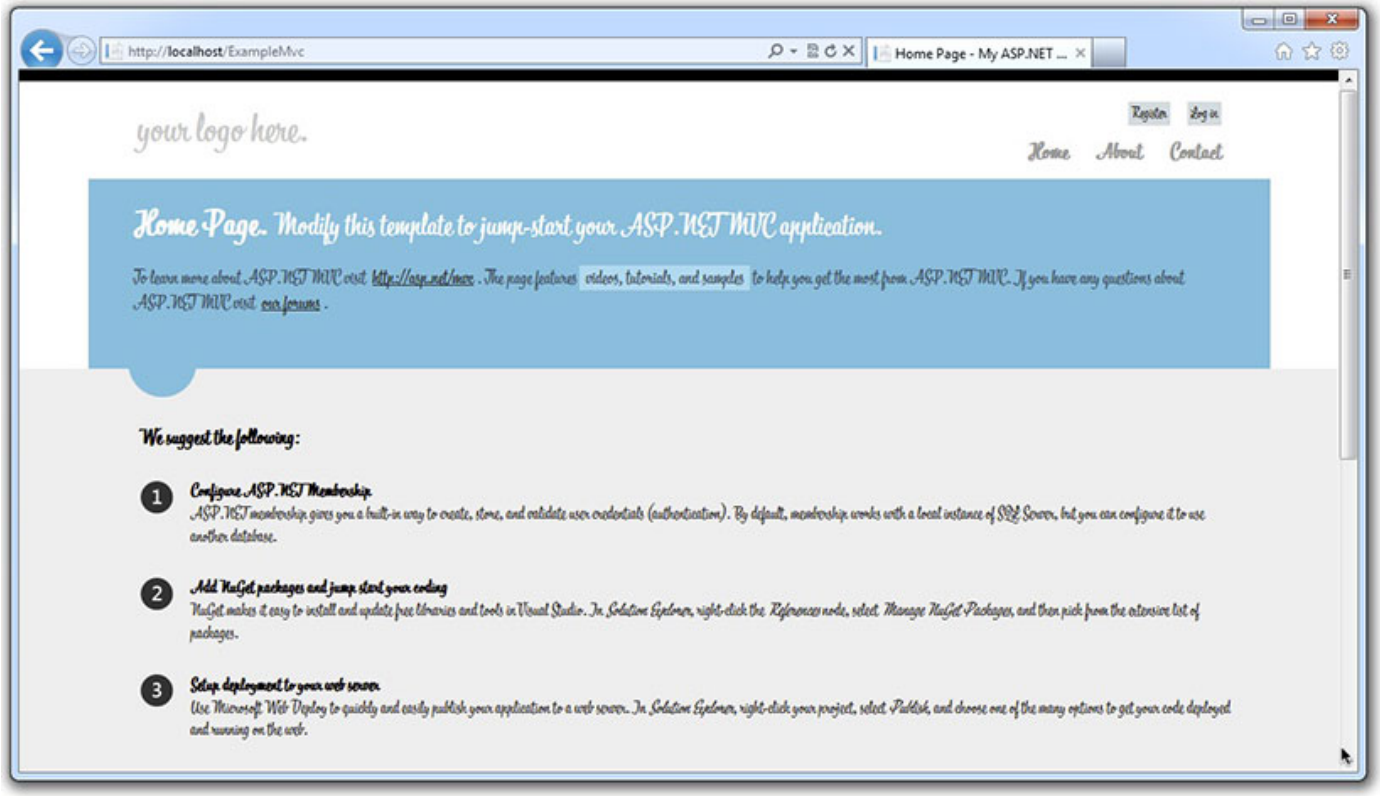


نکته: فونت cursive بعنوان fallback تعریف شده. یعنی در صورتی که بارگذاری و رندر فونت مورد نظر با خطا مواجه شد، از این فونت استفاده می‌شود. بهتر است فونت‌های fallback به مقادیری تنظیم کنید که در اکثر پلتفرم‌ها وجود دارند.

همین! حالا می‌توانیم تغییرات اعمال شده را مشاهده کنیم. بصورت پیش فرض قالب پروژه‌ها از فونت "Segoe UI" استفاده می‌کنند، که خروجی زیر را رندر می‌کند.



استفاده از فونت جدید "Bistro Script Web" وب سایت را مانند تصویر زیر رندر خواهد کرد.



همانطور که می‌بینید استفاده از فونت‌های وب بسیار ساده است. اما بهتر است از اینگونه فونت‌ها برای کل سایت استفاده نشود و تنها روی المنت‌های خاصی مانند سر تیترها (h1,h2,etc) اعمال شوند.

نظرات خوانندگان

نویسنده: رضا
تاریخ: ۱۳۹۲/۱۰/۲۴ ۲۰:۴۲

برای فونت فارسی از چی استفاده کنیم؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۴ ۲۲:۲۱

می‌تونی از فونت‌های فارسی استاندارد سازی شده جدید استفاده کنی.
دایرکتوری فونت‌های فارسی هم هست.

نویسنده: محمد رعیت پیشه
تاریخ: ۱۳۹۲/۱۰/۲۵ ۱۴:۲۰

این سایت هم برای دریافت فونت‌های فارسی یونیکد، جهت نمایش در مرورگرهای مختلف مفید هست.

نویسنده: محمد رعیت پیشه
تاریخ: ۱۳۹۲/۱۰/۲۵ ۱۴:۲۴

متاسفانه سایت Typekit که در مثال آورده اید برای ipهای ایران بسته است ! :(
آیا اگر سایت روی هاست داخلی باشد، این ارجاع‌ها هم بلاگ می‌شوند ؟!

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۵ ۱۶:۴۴

کاری به محل هاست نداره. فونت رو مرورگر کاربر هست که از سرور ارائه دهنده فونت درخواست می‌کنه. این درخواست سمت سرور نیست؛ سمت کلاینت است.

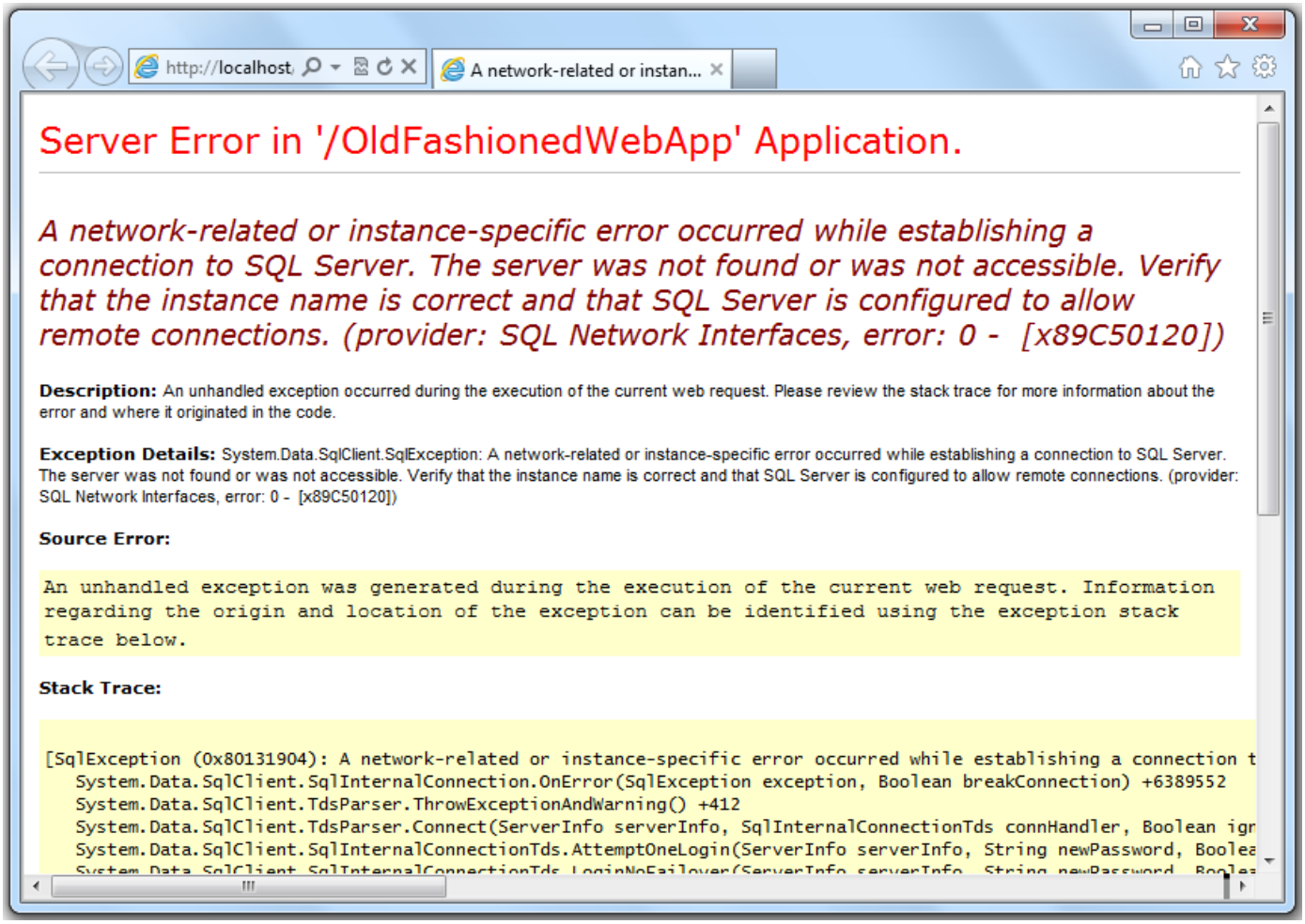
نویسنده: علی نوروزی
تاریخ: ۱۳۹۲/۱۰/۳۰ ۱۵:۲۲

از این روش http://css-tricks.com/snippets/css/using-font-face
و از این فونت‌ها http://awebfont.ir

عنوان: **استفاده از LocalDb در IIS، قسمت اول: پروفایل کاربر**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۹ ۱۶:۴۰
آدرس: www.dotnettips.info
گروه‌ها: IIS, Tips, LocalDB

این مقاله قسمت اول یک سری دو قسمتی است، که در آن به نحوه استفاده از LocalDB در IIS می‌پردازیم.

LocalDb دیتابیس توصیه شده برای ویژوال استودیو است و برای انواع پروژه‌ها مانند اپلیکیشن‌های وب می‌تواند استفاده شود. هنگام استفاده از این دیتابیس در IIS Express یا Cassini همه چیز طبق انتظار کار میکند. اما به محض آنکه بخواهید از آن در Full IIS استفاده کنید با خطاهایی مواجه می‌شوید. مقصود از Full IIS همان نسخه ای است که بهمراه ویندوز عرضه می‌شود و در قالب یک Windows Service اجرا می‌گردد.



هنگام استفاده از Full IIS دو خاصیت از LocalDb باعث بروز مشکل می‌شوند:
LocalDb نیاز دارد پروفایل کاربر بارگذاری شده باشد
بصورت پیش فرض، وهله LocalDb متعلق به یک کاربر بوده و خصوصی است

در ادامه این مقاله روی بارگذاری پروفایل کاربر تمرکز می‌کنیم، و در قسمت بعدی به مالکیت وهله LocalDb می‌پردازیم.

بارگذاری پروفایل کاربر

بگذارید دوباره به خطای موجود نگاهی بیاندازیم:

System.Data.SqlClient.SqlException: A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured to allow remote connections. (provider: SQL Network Interfaces, error: ([0 - [x89C50120

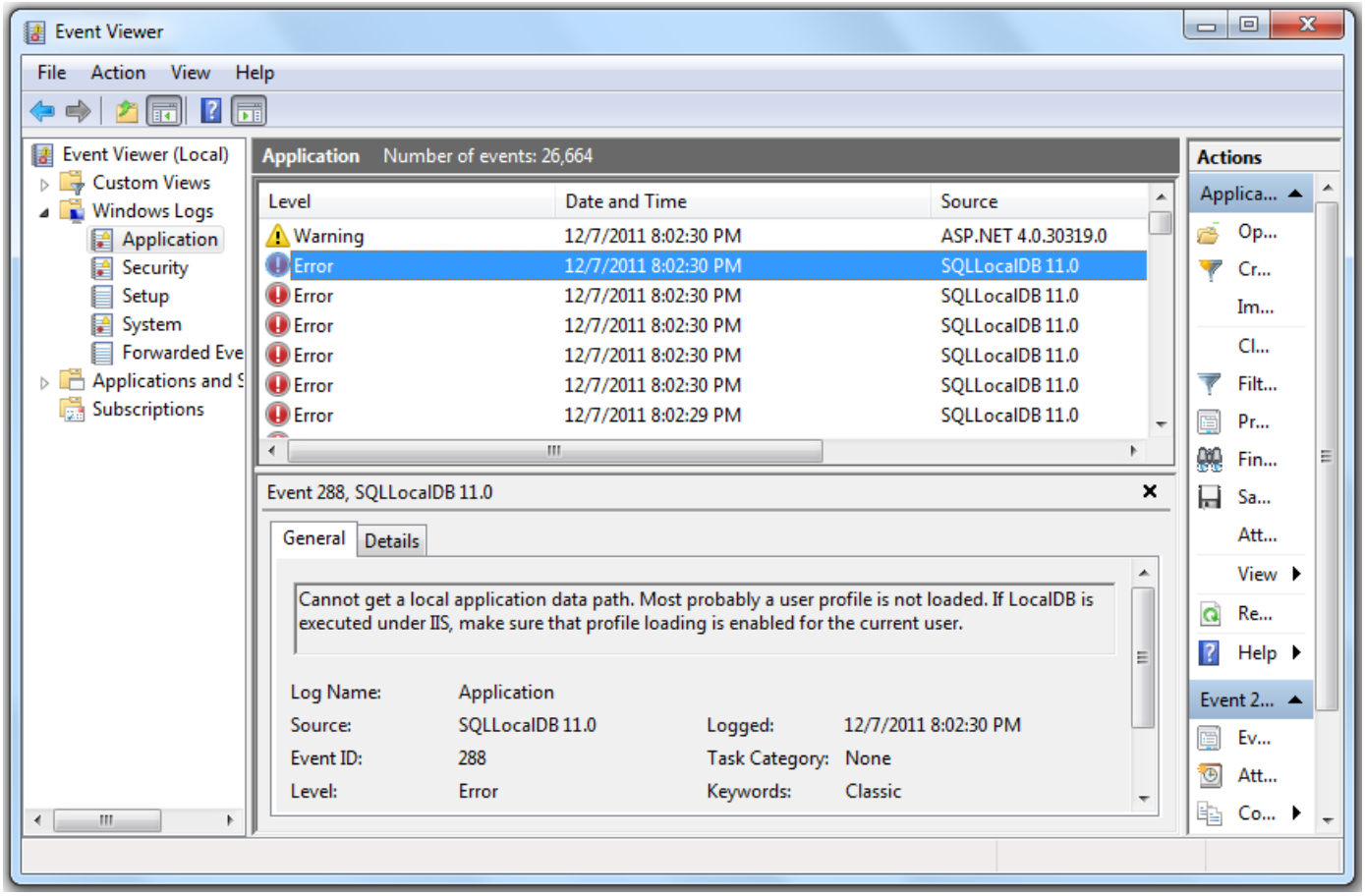
این پیغام خطا زیاد مفید نیست، اما LocalDb اطلاعات بیشتری در Event Log ویندوز ذخیره می‌کند. اگر **Windows Logs** را باز کنید و به قسمت **Application** بروید پیغام زیر را مشاهده خواهید کرد.

Windows API call SHGetKnownFolderPath returned error code: 5. Windows system error message is: Access is denied Reported at line: 400

بعلاوه این پیام خطا:

Cannot get a local application data path. Most probably a user profile is not loaded. If LocalDB is executed under IIS, make sure that profile loading is enabled for the current user.

به احتمال زیاد تعداد بیشتری از این دو خطا در تاریخچه وقایع وجود خواهد داشت، چرا که منطق کانکشن ADO.NET چند بار سعی می‌کند در بازه‌های مختلف به دیتابیس وصل شود.

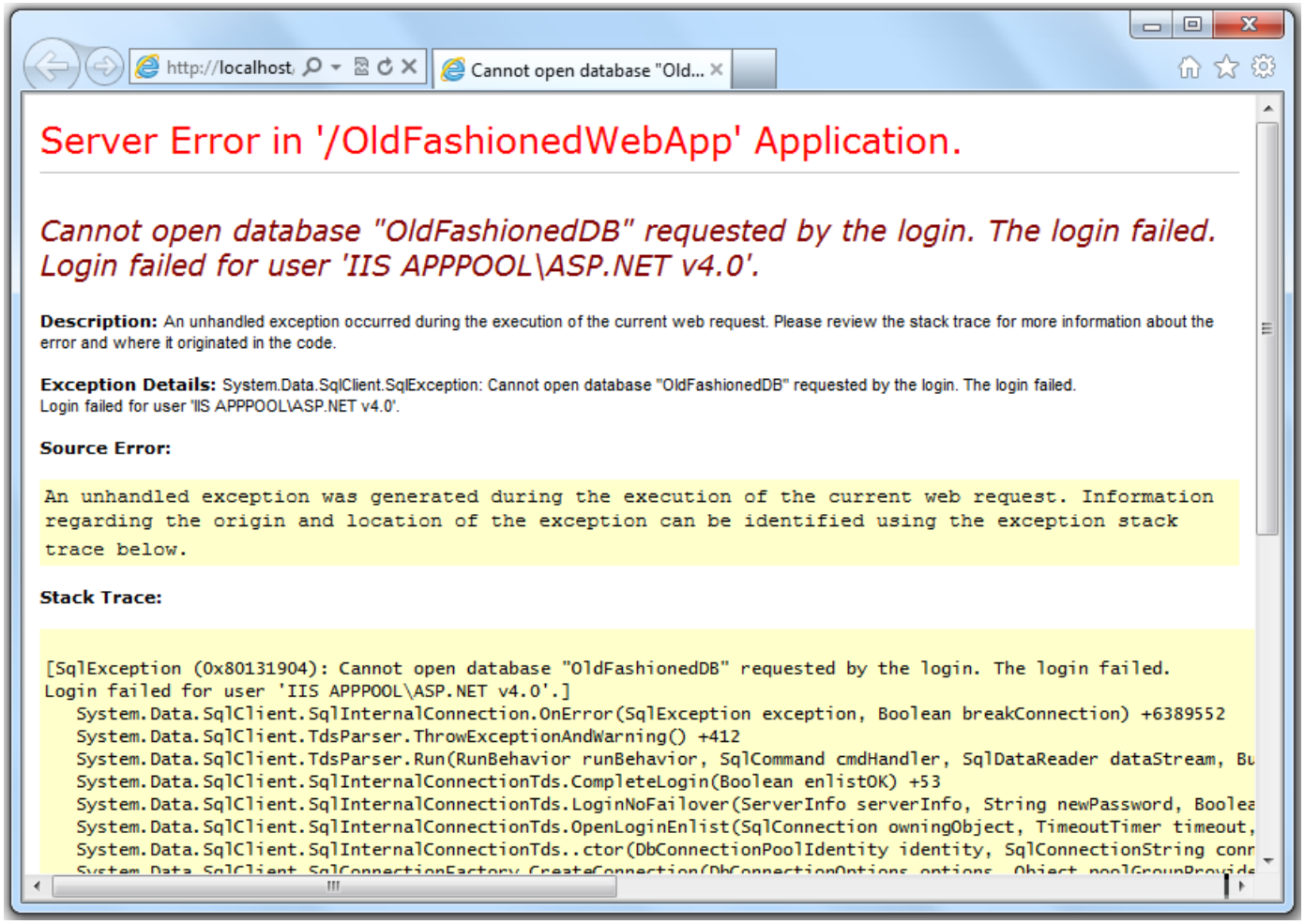


پیغام خطای دوم واضح است، نیاز است پروفایل کاربر را بارگذاری کنیم. انجام اینکار زیاد مشکل نیست، هر Application Pool در IIS تنظیماتی برای بارگذاری پروفایل کاربر دارد که از قسمت Advanced Settings قابل دسترسی است. متاسفانه پس از انتشار سرویس پک 1 برای Windows 7 مسائل کمی پیچیده‌تر شد. در حال حاظر فعال کردن **loadUserProfile** برای بارگذاری

کامل پروفایل کاربر به تنهایی کافی نیست، و باید **setProfileEnvironment** را هم فعال کنیم. برای اطلاعات بیشتر در این باره به مستندات KB 2547655 مراجعه کنید. بدین منظور باید فایل **applicationHost.config** را ویرایش کنید. فایل مذکور در مسیر *C:\Windows\System32\inetsrv\config* قرار دارد. همانطور که در مستندات KB 2547655 توضیح داده شده، باید پرچم هر دو تنظیمات را برای **ASP.NET 4.0** فعال کنیم:

```
<add name="ASP.NET v4.0" autoStart="true" managedRuntimeVersion="v4.0"
managedPipelineMode="Integrated">
    <processModel identityType="ApplicationPoolIdentity" loadUserProfile="true"
setProfileEnvironment="true" />
</add>
```

پس از بروز رسانی این تنظیمات، Application Pool را Restart کنید و اپلیکیشن را مجددا اجرا نمایید. اگر همه چیز طبق انتظار پیش برود، با خطای جدیدی مواجه خواهیم شد:



جای هیچ نگرانی نیست، چرا که این پیغام خطا انتظار می‌رود. همانطور که در ابتدا گفته شد، دو خاصیت LocalDb باعث بروز این خطاها می‌شوند و ما هنوز به خاصیت دوم نپرداخته ایم. بصورت پیش فرض وهله‌های LocalDb خصوصی (private) هستند و در Windows account جاری اجرا می‌شوند. بنابراین ApplicationPoolIdentity در IIS به وهله‌های دیتابیس دسترسی نخواهد داشت. در قسمت دوم این مقاله، راه‌های مختلفی را برای رفع این مشکل بررسی می‌کنیم.

عنوان: استفاده از LocalDb در IIS، قسمت دوم: مالکیت وهله ها
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۹ ۱۸:۵
آدرس: www.dotnettips.info
گروه‌ها: IIS, Tips, LocalDB

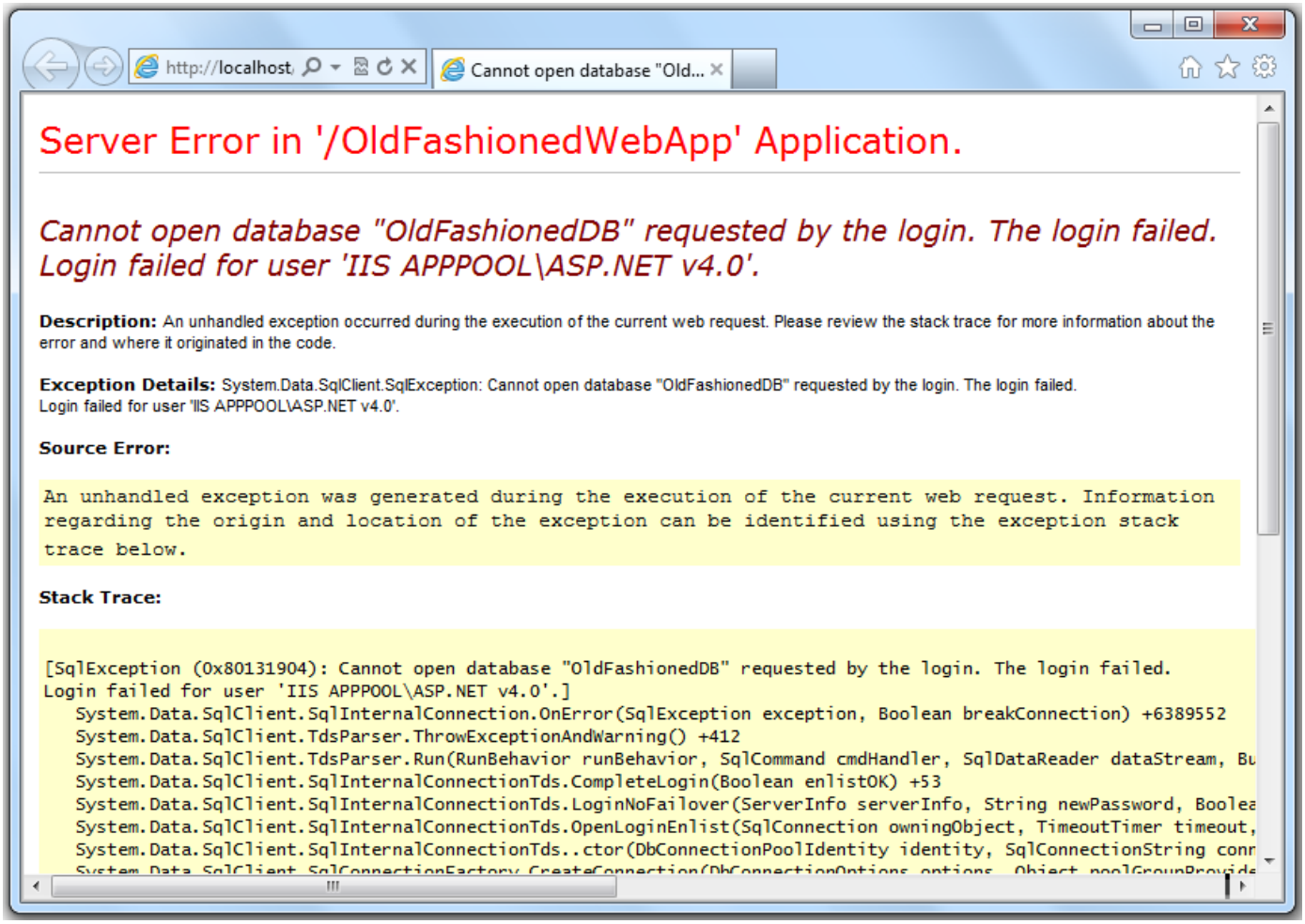
در قسمت قبلی این مقاله گفتیم که دو خاصیت از LocalDb هنگام استفاده از Full IIS باعث بروز خطا می‌شوند:

LocalDb نیاز دارد که پروفایل کاربر بارگذاری شده باشد
بصورت پیش فرض وهله LocalDb متعلق به یک کاربر بوده، و خصوصی است

در قسمت قبل دیدیم چگونه باید پروفایل کاربر را بدرستی بارگذاری کنیم. در این مقاله به مالکیت وهله‌ها (instance ownership) می‌پردازیم.

مشکل وهله خصوصی

در پایان قسمت قبلی، اپلیکیشن وب را در این حالت رها کردیم:

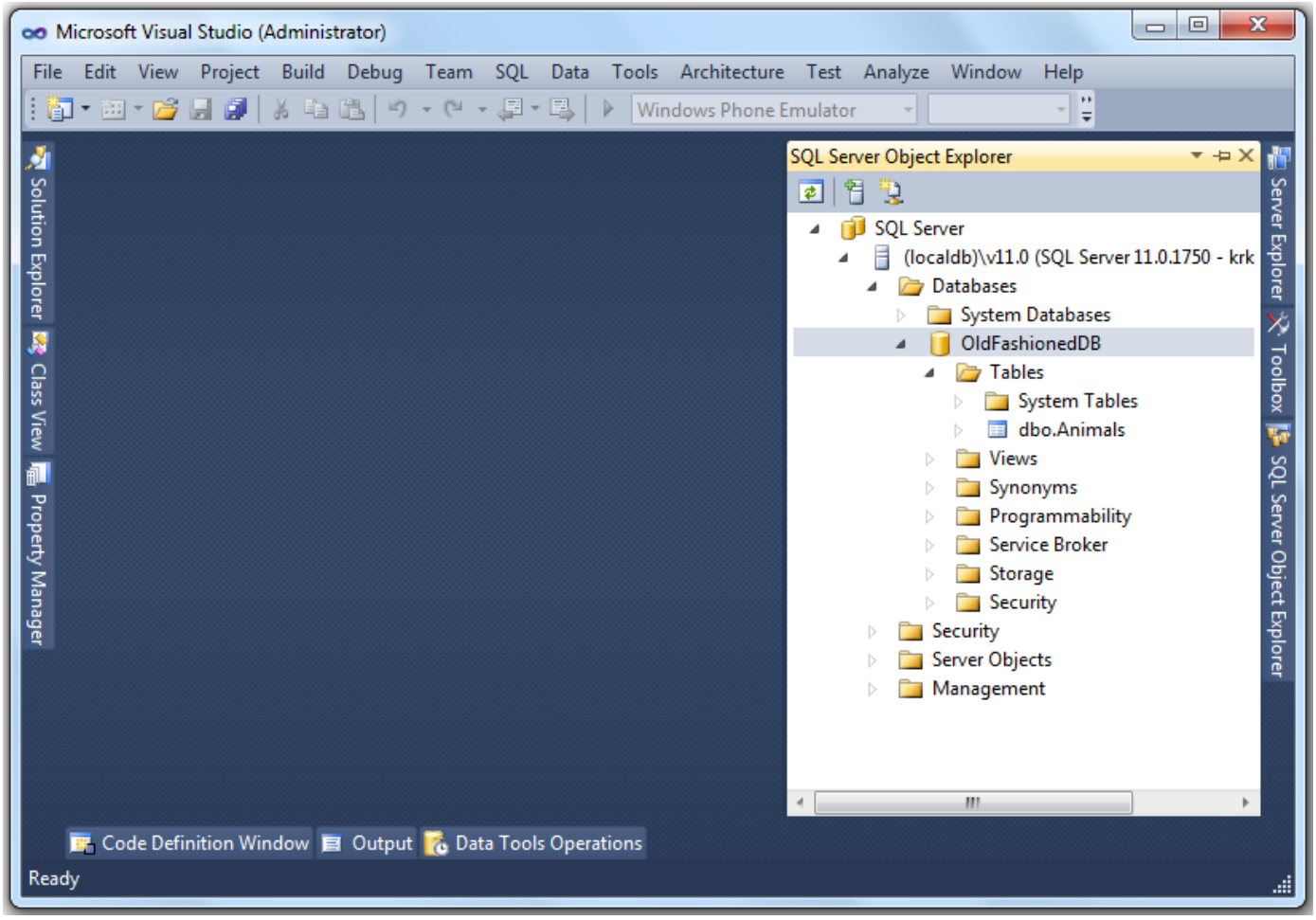


همانطور که مشاهده می‌کنید با خطای زیر مواجه هستیم:

System.Data.SqlClient.SqlException: Cannot open database "OldFashionedDB" requested by the login. The login failed.

.'Login failed for user 'IIS APPPOOL\ASP.NET v4.0

این بار پیغام خطا واضح و روشن است. LocalDb با موفقیت اجرا شده و اپلیکیشن وب هم توانسته به آن وصل شود، اما این کانکشن سپس قطع شده چرا که دسترسی به وهله جاری وجود نداشته است. اکانت ApplicationPoolIdentity (در اینجا **IIS APPPOOL\ASP.NET v4.0**) نتوانسته به دیتابیس LocalDb وارد شود، چرا که دیتابیس مورد نظر در رشته اتصال اپلیکیشن (OldFashionedDB) وجود ندارد. عجیب است، چرا که وصل شدن به همین دیتابیس با رشته اتصال جاری در ویژوال استودیو با موفقیت انجام می‌شود.



همانطور که در تصویر بالا مشاهده می‌کنید از ابزار SQL Server Object Explorer استفاده شده است. این ابزار توسط SQL Server Data Tools معرفی شد و در نسخه‌های بعدی ویژوال استودیو هم وجود دارد و توسعه یافته است. چطور ممکن است ویژوال استودیو براحتی بتواند به دیتابیس وصل شود، اما اپلیکیشن وب ما با همان رشته اتصال نمی‌تواند دیتابیس را باز کند؟ در هر دو صورت رشته اتصال ما بدین شکل است:

```
Data Source=(localdb)\v11.0;Initial Catalog=OldFashionedDB;Integrated Security=True
```

پاسخ این است که در اینجا، **دو وهله** از LocalDb وجود دارد. بر خلاف وهله‌های SQL Server Express که بعنوان سرویس‌های ویندوزی اجرا می‌شوند، وهله‌های LocalDb بصورت پروسس‌های کاربری (user processes) اجرا می‌شوند. هنگامی که کاربران مختلفی سعی می‌کنند به LocalDb متصل شوند، برای هر کدام از آنها پروسس‌های مجزایی اجرا خواهد شد. هنگامی که در ویژوال استودیو به **(localdb)\v11.0** وصل می‌شویم، وهله ای از LocalDb ساخته شده و در حساب کاربری ویندوز جاری اجرا می‌شود.

اما هنگامی که اپلیکیشن وب ما در IIS می‌خواهد به همین دیتابیس وصل شود، وهله دیگری ساخته شده و در **ApplicationPoolIdentity** اجرا می‌شود. گرچه ویژوال استودیو و اپلیکیشن ما هر دو از یک رشته اتصال استفاده می‌کنند، اما در عمل هر کدام به وهله‌های متفاوتی از LocalDb دسترسی پیدا خواهند کرد. پس مسلما دیتابیسی که توسط وهله ای در ویژوال استودیو ساخته شده است، برای اپلیکیشن وب ما در IIS در دسترس نخواهد بود.

یک مقایسه خوب از این وضعیت، پوشه **My Documents** در ویندوز است. فرض کنید در ویژوال استودیو کدی بنویسیم که در این پوشه یک فایل جدید می‌سازد. حال اگر با حساب کاربری دیگری وارد ویندوز شویم و به پوشه My Documents برویم این فایل را نخواهیم یافت. چرا که پوشه My Documents برای هر کاربر متفاوت است. بهمین شکل، وهله‌های LocalDb برای هر کاربر متفاوت است و به پروسس‌ها و دیتابیس‌های مختلفی اشاره می‌کنند.

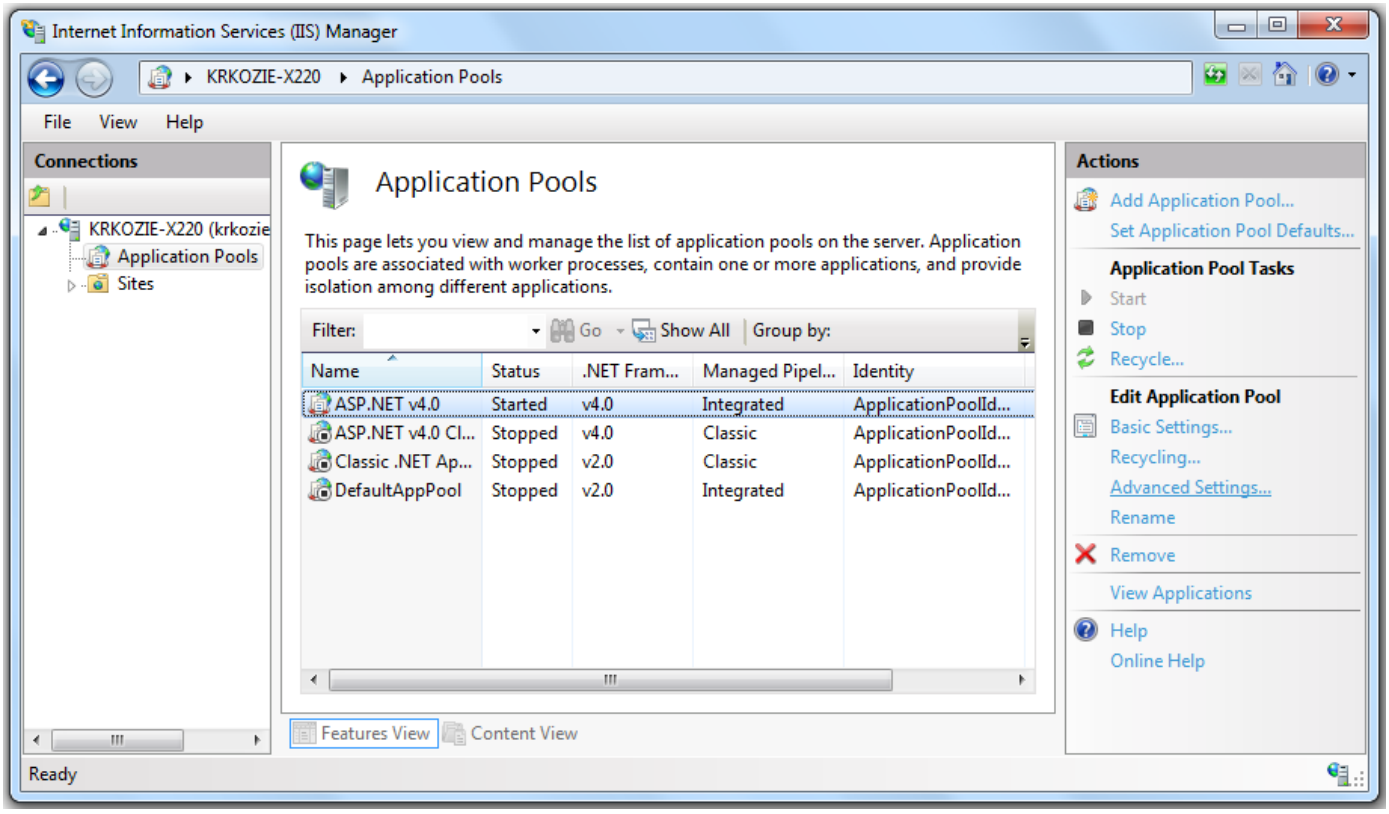
به همین دلیل است که اپلیکیشن وب ما می‌تواند بدون هیچ مشکلی روی IIS Express اجرا شود و دیتابیس را باز کند. چرا که IIS Express درست مانند LocalDb یک پروسس کاربری است. IIS Express توسط ویژوال استودیو راه اندازی می‌شود و روی حساب کاربری جاری اجرا می‌گردد، پس پروسس آن با پروسس خود ویژوال استودیو یکسان خواهد بود و هر دو زیر یک اکانت کاربری اجرا خواهند شد.

راه حل ها

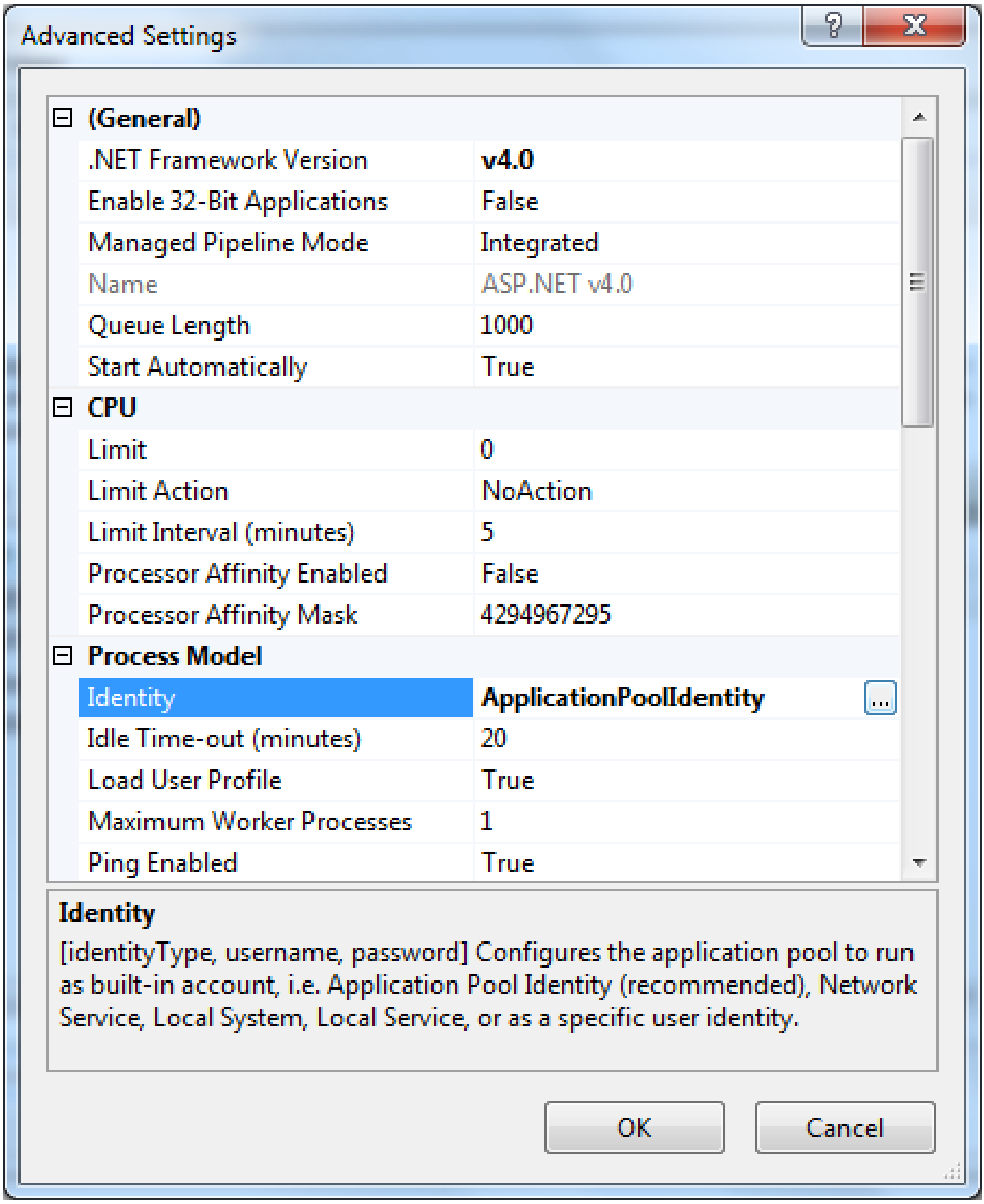
درک ماهیت مشکل جاری، راه حال‌های مختلفی را برای رفع آن بدست می‌دهد. از آنجا که هر راه حل مزایا و معایب خود را دارد، بجای معرفی یک راه حال واحد چند راهکار را بررسی می‌کنیم.

رویکرد 1: اجرای IIS روی کاربر جاری ویندوز

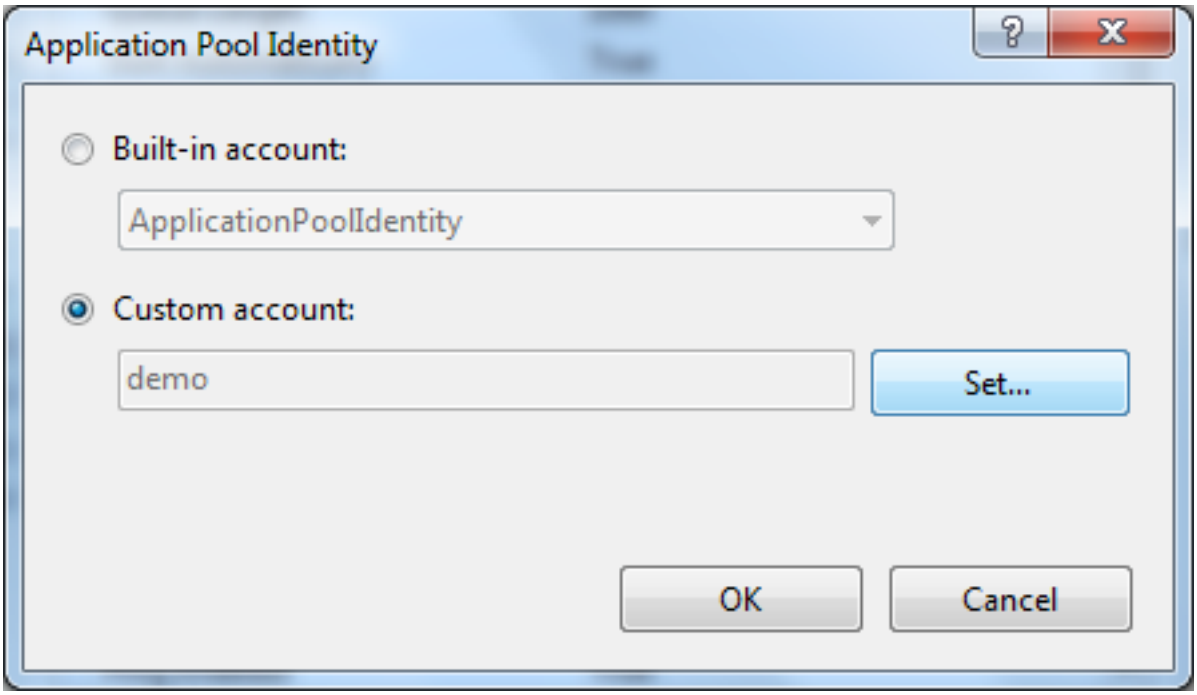
اگر مشکل، حساب‌های کاربری مختلف است، چرا خود IIS را روی کاربر جاری اجرا نکنیم؟ در این صورت ویژوال استودیو و اپلیکیشن ما هر دو به یک وهله از LocalDb وصل خواهند شد و همه چیز بدرستی کار خواهد کرد. ایجاد تغییرات لازم نسبتا ساده است. IIS را اجرا کنید و Application Pool مناسب را انتخاب کنید، یعنی همان گزینه که برای اپلیکیشن شما استفاده می‌شود.



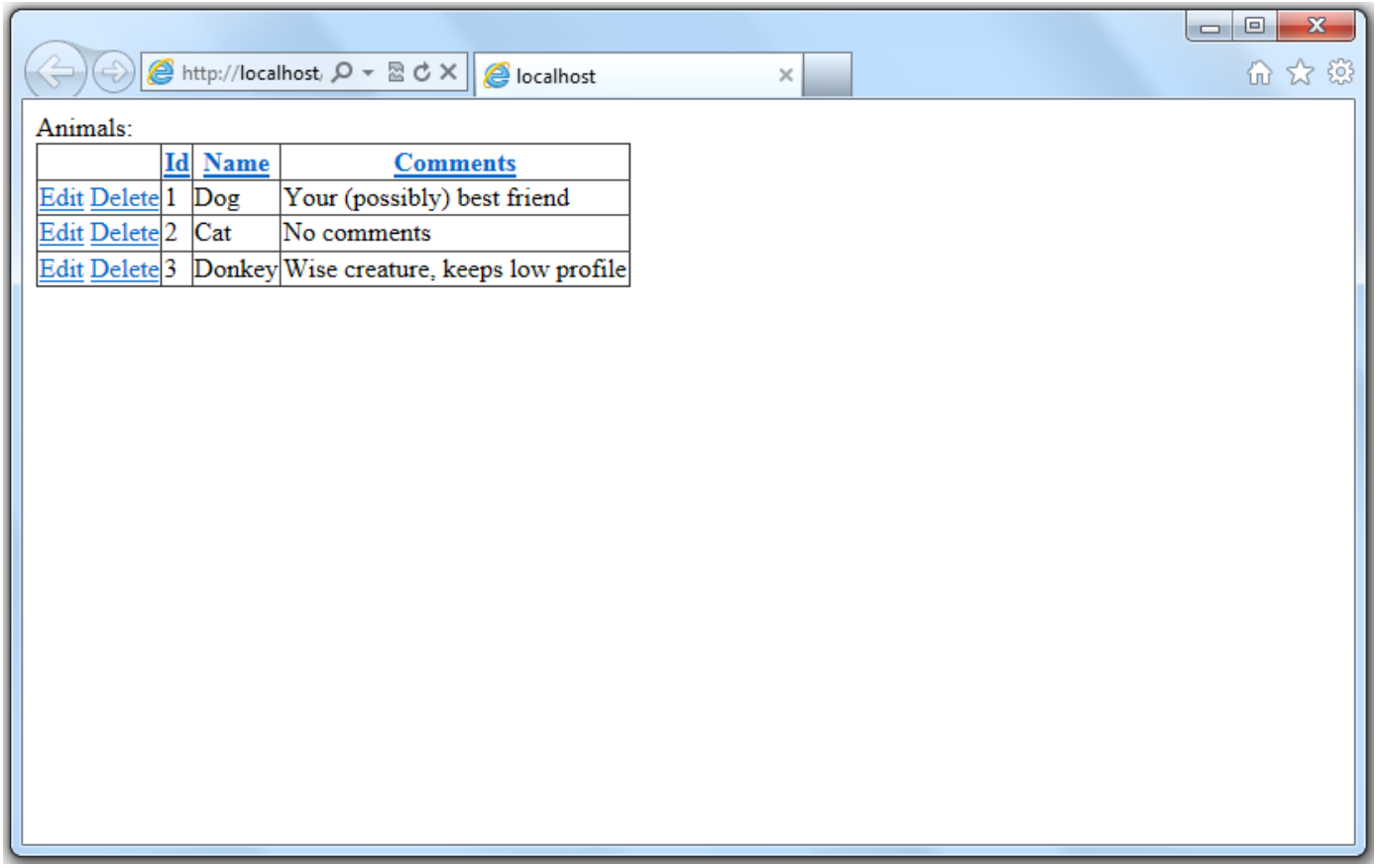
قسمت Advanced Settings را باز کنید:



روی دکمه سه نقطه کنار خاصیت **Identity** کلیک کنید تا پنجره Application Pool Identity باز شود:



در این قسمت می‌توانید از حساب کاربری جاری استفاده کنید. روی دکمه Set کلیک کنید و نام کاربری و رمز عبور خود را وارد نمایید. حال اگر اپلیکیشن را مجددا اجرا کنید، همه چیز باید بدرستی اجرا شود.



خوب، معایب این رویکرد چیست؟ مسلما اجرای اپلیکیشن وب روی اکانت کاربری جاری، ریسک‌های امنیتی متعددی را معرفی می‌کند. اگر کسی بتواند اپلیکیشن وب ما را هک کند، به تمام منابع سیستم که اکانت کاربری جاری به آنها دسترسی دارد، دسترسی خواهد داشت. اما اجرای اپلیکیشن مورد نظر روی ApplicationPoolIdentity امنیت بیشتری را ارائه می‌کند، چرا که اکانت‌های ApplicationPoolIdentity دسترسی بسیار محدودتری به منابع سیستم محلی دارند. بنابراین استفاده از این روش بطور کلی توصیه نمی‌شود، اما در سناریوهای خاصی با در نظر داشتن ریسک‌های امنیتی می‌تواند رویکرد خوبی باشد.

رویکرد 2: استفاده از وهله مشترک

یک راه حال دیگر استفاده از قابلیت instance sharing است. این قابلیت به ما این امکان را می‌دهد تا یک وهله LocalDb را بین کاربران یک سیستم به اشتراک بگذاریم. وهله به اشتراک گذاشته شده، توسط یک نام عمومی (public name) قابل دسترسی خواهد بود.

ساده‌ترین راه برای به اشتراک گذاشتن وهله‌های LocalDb استفاده از ابزار SqlLocalDB.exe است. بدین منظور Command Prompt را بعنوان مدیر سیستم باز کنید و فرمان زیر را اجرا نمایید:

```
sqllocaldb share v11.0 IIS_DB
```

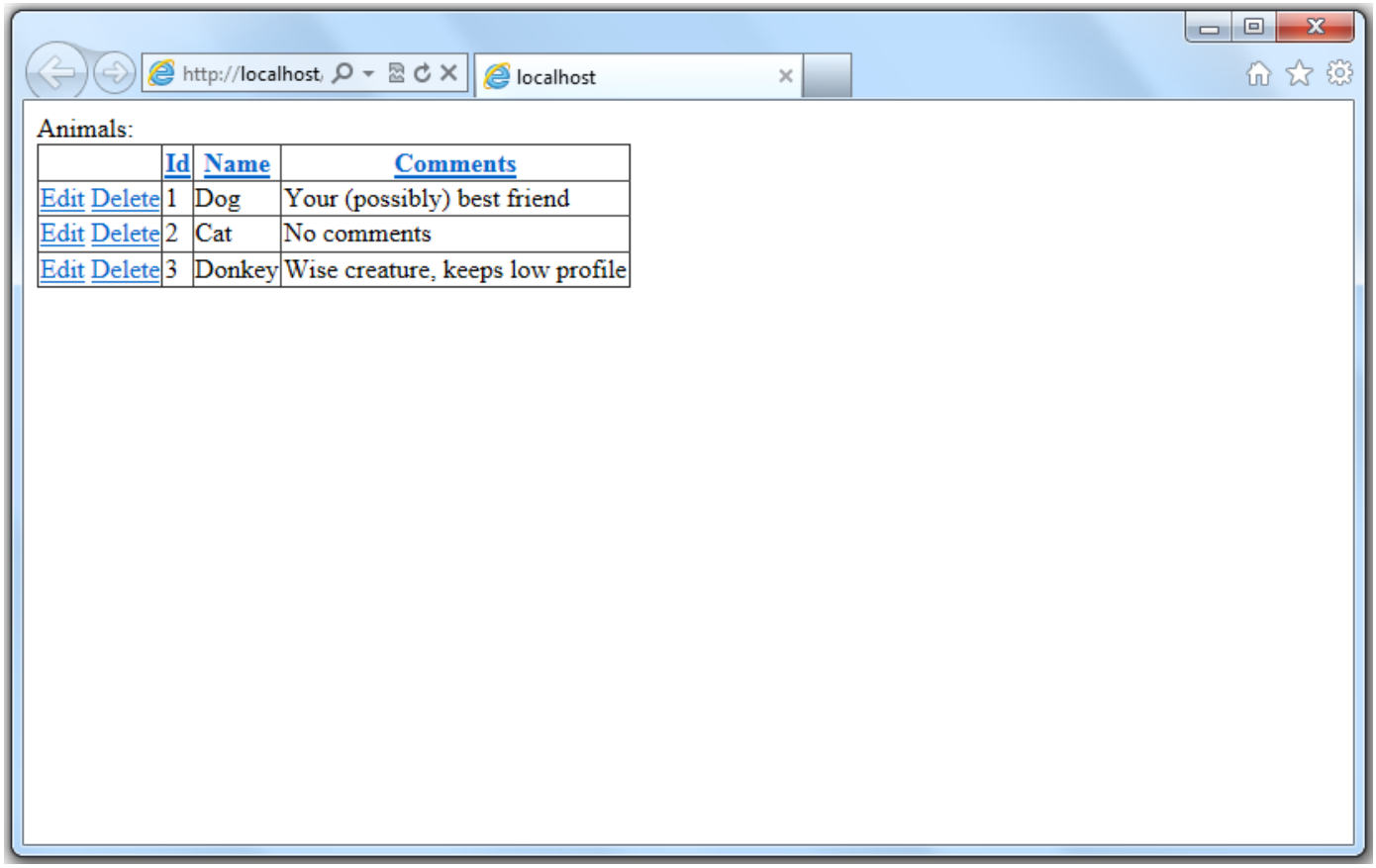
این فرمان وهله خصوصی LocalDb را با نام عمومی **IIS_DB** به اشتراک می‌گذارد. حال تمام کاربران سیستم می‌توانند با آدرس **(localdb)\.\IIS_DB** به این وهله وصل شوند. این فرمت آدرس دهی سرور دیتابیس، مشخص می‌کند که از یک وهله shared استفاده می‌کنیم. رشته اتصال جدید مانند لیست زیر خواهد بود:

```
Data Source=(localdb)\.\IIS_DB;Initial Catalog=OldFashionedDB;Integrated Security=True
```

پیش از آنکه اپلیکیشن وب ما بتواند به این وهله متصل شود، باید لاگین‌های مورد نیاز برای ApplicationPoolIdentity را ایجاد کنیم. راه اندازی وهله ساده است، کافی است دیتابیس را در SQL Server Object Explorer باز کنید. این کار اتصالی به دیتابیس برقرار می‌کند و آن را زنده نگاه می‌دارد. برای ایجاد لاگین مورد نظر، می‌توانیم در SQL Server Object Explorer یک کوئری اجرا کنیم:

```
create login [IIS APPPOOL\ASP.NET v4.0] from windows;
exec sp_addsrvrolemember N'IIS APPPOOL\ASP.NET v4.0', sysadmin
```

اسکریپت بالا به اکانت ApplicationPoolIdentity سطح دسترسی کامل می‌دهد. در صورت امکان بهتر است از سطوح دسترسی محدودتری استفاده کنید، مثلا دسترسی به دیتابیس یا جداولی مشخص. حالا می‌توانید اپلیکیشن را مجددا اجرا کنید و همه چیز بدون خطا باید کار کند.



معایب این روش چیست؟ مشکل اصلی در این رویکرد این است که پیش از آنکه اپلیکیشن ما بتواند به وهله مشترک دسترسی داشته باشد، باید وهله مورد نظر را راه اندازی و اجرا کنیم. بدین منظور، حساب کاربری ویندوزی که مالکیت وهله را دارد باید به آن وصل شود و کانکشن را زنده نگه دارد، در غیر اینصورت وهله LocalDb قابل دسترسی نخواهد بود.

رویکرد 3: استفاده از SQL Server Express

از آنجا که نسخه کامل SQL Server Express بعنوان یک سرویس ویندوزی اجرا می‌شود، شاید بهترین راه استفاده از همین روش باشد. کافی است یک نسخه از SQL Server Express را نصب کنیم، دیتابیس مورد نظر را در آن بسازیم و سپس به آن متصل شویم. برای این کار حتی می‌توانید از ابزار جدید SQL Server Data Tools استفاده کنید، چرا که با تمام نسخه‌های SQL Server سازگار است. در صورت استفاده از نسخه‌های کامل تر، رشته اتصال ما بدین شکل تغییر خواهد کرد:

```
Data Source=.\SQLEXPRESS;Initial Catalog=OldFashionedDB;Integrated Security=True
```

مسلما در این صورت نیز، لازم است اطمینان حاصل کنیم که ApplicationPoolIdentity به وهله SQL Server Express دسترسی کافی دارد. برای این کار می‌توانیم از اسکریپت قبلی استفاده کنیم:

```
create login [IIS APPPOOL\ASP.NET v4.0] from windows;
exec sp_addsrvrolemember N'IIS APPPOOL\ASP.NET v4.0', sysadmin
```

حال اجرای مجدد اپلیکیشن باید با موفقیت انجام شود. استفاده از این روش مسلما امکان استفاده از LocalDb را از ما می‌گیرد. ناگفته نماند که وهله‌های SQL Server Express همیشه در حال اجرا خواهند بود چرا که بصورت سرویس‌های ویندوزی اجرا می‌شوند. همچنین استفاده از این روش ممکن است شما را با مشکلاتی هم مواجه کند. مثلا خرابی رجیستری ویندوز می‌تواند SQL Server Express را از کار بیاندازد و مواردی از این دست. راهکارهای دیگری هم وجود دارند که در این مقاله به آنها نپرداختیم. مثلا می‌توانید از AttachDbFilename استفاده کنید یا از اسکریپت‌های T-SQL برای استفاده از وهله خصوصی ASP.NET کمک بگیرید. اما این روش‌ها دردسرهای زیادی دارند، بهمین دلیل از آنها صرفنظر کردیم.

مطالعه بیشتر درباره LocalDb

Introducing LocalDB and LocalDB Q&A

Upgrading .NET Framework 4 to support LocalDB connections and using SQL Server Management Studio to work with LocalDB

Using LocalDB in Visual Studio 2010

Where are LocalDB database files located

نظرات خوانندگان

نویسنده: رضا گرمارودی
تاریخ: ۱۳۹۲/۱۰/۲۹ ۲۰:۳

سلام؛ ممنون از مقاله جالبتون. یک سوال داشتم. مقالات شما رو در خصوص Identity و Localdb دنبال کردم. شما تجربه کاری با WindowsAzor داریو و اینجا هم از Localdb صحبت کردین. می‌خواستم برای هر کاربر یک دیتابیس مجزا داشته باشم اما هاست‌ها نهایتا دو یا سه تا دیتابیس sqlserver در اختیار شما می‌گذارند. LocalDb همان طور که از اسمش برمیاد Local هست یعنی نمیشه روی هاست ولو این که از این دیتابیس تنها یک نفر آنلاین استفاده کنه؟ استفاده از windowsAzor چطور؟ مشکل من و حل می‌کنه یا شما راه حل بهتری معرفی می‌کنید؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۹ ۲۲:۳۱

SQL Server CE داخل پروسه برنامه اجرا میشه (و پروسه مجزایی نداره). نسخه LocalDB خارج از پروسه برنامه اما به صورت یک child process اجرا میشه. برنامه وب شما می‌تونه با هر تعداد فایل sdf مربوط به SQL CE یا فایل mdf بانک اطلاعاتی LocalDB کار کنه (در رشته اتصالی آن AttachDbFileName قابل تعریف هست).

نویسنده: رضا گرمارودی
تاریخ: ۱۳۹۲/۱۱/۰۱ ۱۶:۵۷

Attachdb در رشته اتصالی موجود است و بر روی لوکال مشکلی ندارم . اما بر روی هاست بانک را Attach نمی‌کند. با سه تا هاست مختلف تست گرفتم ولی نشد! ممکنه LocalDb و Sql Server CE تنها بر روی لوکال کار کنند و نه بر روی هاست؟ یا ارائه دهنده سرویس می‌بایست این سرویس‌ها را بر روی سرور خود نصب کنه ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۰۱ ۱۷:۵

SQL CE با bin deploy کار می‌کنه و نیازی به نصب نداره (البته نصاب داره؛ ولی مهم و الزامی نیست). فایل‌هاش رو بسته نیوگت زیر به پروژه اضافه می‌کنه:

http://www.nuget.org/packages/Microsoft.SqlServer.Compact

اما LocalDB رو باید سرور دار حتما خودش یکبار نصب کنه.

عنوان: **استفاده از EF در اپلیکیشن های N-Tier : قسمت اول**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۰۷ ۲:۳۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, Tips, ASP.NET Web API, N-Layer Architecture, Entity Framework 6

تمام اپلیکیشن‌ها را نمی‌توان در یک پروسس بسته بندی کرد، بدین معنا که تمام اپلیکیشن روی یک سرور فیزیکی قرار گیرد. در عصر حاظر معماری بسیاری از اپلیکیشن‌ها چند لایه است و هر لایه روی سرور مجزایی توزیع می‌شود. بعنوان مثال یک معماری کلاسیک شامل سه لایه نمایش (presentation)، اپلیکیشن (application) و داده (data) است. لایه بندی منطقی (logical layering) یک اپلیکیشن می‌تواند در یک App Domain واحد پیاده سازی شده و روی یک کامپیوتر میزبانی شود. در این صورت لازم نیست نگران مباحثی مانند پراکسی ها، مرتب سازی (serialization)، پروتوکل‌های شبکه و غیره باشیم. اما اپلیکیشن‌های بزرگی که چندین کلاینت دارند و در مراکز داده میزبانی می‌شوند باید تمام این مسائل را در نظر بگیرند. خوشبختانه پیاده سازی چنین اپلیکیشن هایی با استفاده از Entity Framework و دیگر تکنولوژی‌های مایکروسافت مانند WCF, Web API ساده‌تر شده است. منظور از n-Tier معماری اپلیکیشن هایی است که لایه‌های نمایش، منطق تجاری و دسترسی داده هر کدام روی سرور مجزایی میزبانی می‌شوند. این تفکیک فیزیکی لایه‌ها به بسط پذیری، مدیریت و نگهداری اپلیکیشن‌ها در دراز مدت کمک می‌کند، اما معمولا تاثیری منفی روی کارایی کلی سیستم دارد. چرا که برای انجام عملیات مختلف باید از محدوده ماشین‌های فیریکی عبور کنیم.

معماری N-Tier چالش‌های بخصوصی را برای قابلیت‌های change-tracking در EF اضافه می‌کند. در ابتدا داده‌ها توسط یک آبجکت EF Context بارگذاری می‌شوند اما این آبجکت پس از ارسال داده‌ها به کلاینت از بین می‌رود. تغییراتی که در سمت کلاینت روی داده‌ها اعمال می‌شوند ردیابی (track) نخواهند شد. هنگام بروز رسانی، آبجکت Context جدیدی برای پردازش اطلاعات ارسالی باید ایجاد شود. مسلما آبجکت جدید هیچ چیز درباره Context پیشین یا مقادیر اصلی موجودیت‌ها نمی‌داند.

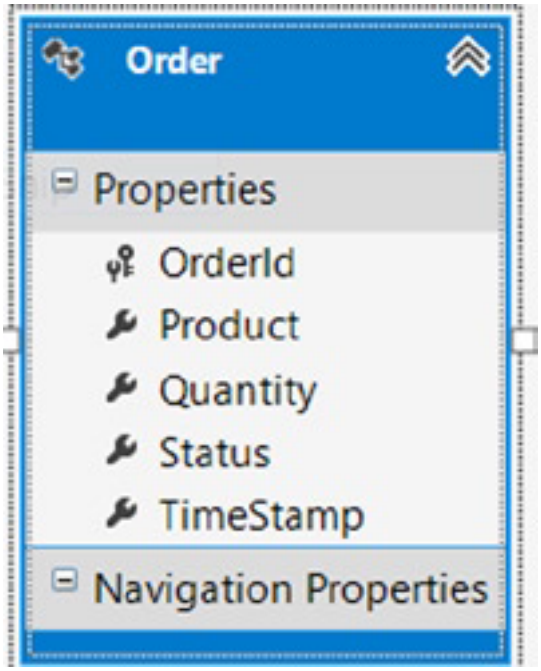
در نسخه‌های قبلی Entity Framework توسعه دهندگان با استفاده از قالب ویژه ای بنام Self-Tracking Entities می‌توانستند تغییرات موجودیت‌ها را ردیابی کنند. این قابلیت در نسخه EF 6 از رده خارج شده است و گرچه هنوز توسط ObjectContext پشتیبانی می‌شود، آبجکت DbContext از آن پشتیبانی نمی‌کند.

در این سری از مقالات روی عملیات پایه CRUD تمرکز می‌کنیم که در اکثر اپلیکیشن‌های n-Tier استفاده می‌شوند. همچنین خواهیم دید چگونه می‌توان تغییرات موجودیت‌ها را ردیابی کرد. مباحثی مانند همزمانی (concurrency) و مرتب سازی (serialization) نیز بررسی خواهند شد. در قسمت یک این سری مقالات، به بروز رسانی موجودیت‌های منفصل (disconnected) توسط سرویس‌های Web API نگاهی خواهیم داشت.

بروز رسانی موجودیت‌های منفصل با Web API

سناریویی را فرض کنید که در آن برای انجام عملیات CRUD از یک سرویس Web API استفاده می‌شود. همچنین مدیریت داده‌ها با مدل Code-First پیاده سازی شده است. در مثال جاری یک کلاینت Console Application خواهیم داشت که یک سرویس Web API را فراخوانی می‌کند. توجه داشته باشید که هر اپلیکیشن در Solution مجزایی قرار دارد. تفکیک پروژه‌ها برای شبیه سازی یک محیط n-Tier انجام شده است.

فرض کنید مدلی مانند تصویر زیر داریم.



همانطور که می‌بینید مدل جاری، سفارشات یک اپلیکیشن فرضی را معرفی می‌کند. می‌خواهیم مدل و کد دسترسی به داده‌ها را در یک سرویس Web API پیاده سازی کنیم، تا هر کلاینتی که از HTTP استفاده می‌کند بتواند عملیات CRUD را انجام دهد. برای ساختن سرویس مورد نظر مراحل زیر را دنبال کنید.

در ویژوال استودیو پروژه جدیدی از نوع ASP.NET Web Application بسازید و قالب پروژه را Web API انتخاب کنید. نام پروژه را به Recipe1.Service تغییر دهید.

کنترلر جدیدی از نوع WebApi Controller با نام OrderController به پروژه اضافه کنید.

کلاس جدیدی با نام Order در پوشه مدل‌ها ایجاد کنید و کد زیر را به آن اضافه نمایید.

```
public class Order
{
    public int OrderId { get; set; }
    public string Product { get; set; }
    public int Quantity { get; set; }
    public string Status { get; set; }
    public byte[] TimeStamp { get; set; }
}
```

با استفاده از NuGet Package Manager کتابخانه Entity Framework 6 را به پروژه اضافه کنید.

حال کلاسی با نام Recipe1Context ایجاد کنید و کد زیر را به آن اضافه نمایید.

```
public class Recipe1Context : DbContext
{
    public Recipe1Context() : base("Recipe1ConnectionString") { }

    public DbSet<Order> Orders { get; set; }

    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Order>().ToTable("Orders");
        // Following configuration enables timestamp to be concurrency token
        modelBuilder.Entity<Order>().Property(x => x.TimeStamp)
            .IsConcurrencyToken()
            .HasDatabaseGeneratedOption(DatabaseGeneratedOption.Computed);
    }
}
```

فایل Web.config پروژه را باز کنید و رشته اتصال زیر را به قسمت ConnectionStrings اضافه نمایید.

```
<connectionStrings>
  <add name="Recipe1ConnectionString"
    connectionString="Data Source=.;
    Initial Catalog=EFRecipes;
    Integrated Security=True;
    MultipleActiveResultSets=True"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

فایل Global.asax را باز کنید و کد زیر را به آن اضافه نمایید. این کد بررسی Entity Framework Compatibility را غیرفعال می‌کند.

```
protected void Application_Start()
{
    // Disable Entity Framework Model Compatibilty
    Database.SetInitializer<Recipe1Context>(null);
    ...
}
```

در آخر کد کنترلر Order را با لیست زیر جایگزین کنید.

```
public class OrderController : ApiController
{
    // GET api/order
    public IEnumerable<Order> Get()
    {
        using (var context = new Recipe1Context())
        {
            return context.Orders.ToList();
        }
    }

    // GET api/order/5
    public Order Get(int id)
    {
        using (var context = new Recipe1Context())
        {
            return context.Orders.FirstOrDefault(x => x.OrderId == id);
        }
    }

    // POST api/order
    public HttpResponseMessage Post(Order order)
    {
        // Cleanup data from previous requests
        Cleanup();

        using (var context = new Recipe1Context())
        {
            context.Orders.Add(order);
            context.SaveChanges();
            // create HttpResponseMessage to wrap result, assigning Http Status code of 201,
            // which informs client that resource created successfully
            var response = Request.CreateResponse(HttpStatusCode.Created, order);
            // add location of newly-created resource to response header
            response.Headers.Location = new Uri(Url.Link("DefaultApi",
                new { id = order.OrderId }));
            return response;
        }
    }

    // PUT api/order/5
    public HttpResponseMessage Put(Order order)
    {
        using (var context = new Recipe1Context())
        {
            context.Entry(order).State = EntityState.Modified;
            context.SaveChanges();
            // return Http Status code of 200, informing client that resouce updated successfully
            return Request.CreateResponse(HttpStatusCode.OK, order);
        }
    }

    // DELETE api/order/5
```

```
    public HttpResponseMessage Delete(int id)
    {
        using (var context = new Recipe1Context())
        {
            var order = context.Orders.FirstOrDefault(x => x.OrderId == id);
            context.Orders.Remove(order);
            context.SaveChanges();
            // Return Http Status code of 200, informing client that resouce removed successfully
            return Request.CreateResponse(HttpStatusCode.OK);
        }
    }

    private void Cleanup()
    {
        using (var context = new Recipe1Context())
        {
            context.Database.ExecuteSqlCommand("delete from [orders]");
        }
    }
}
```

قابل ذکر است که هنگام استفاده از Entity Framework در MVC یا Web API، بکارگیری قابلیت Scaffolding بسیار مفید است. این فریم ورک‌های ASP.NET می‌توانند کنترلرهایی کاملا اجرایی برایتان تولید کنند که صرفه جویی چشمگیری در زمان و کار شما خواهد بود.

در قدم بعدی اپلیکیشن کلاینت را می‌سازیم که از سرویس Web API استفاده می‌کند.

در ویژوال استودیو پروژه جدیدی از نوع Console Application بسازید و نام آن را به Recipe1.Client تغییر دهید.
کلاس موجودیت Order را به پروژه اضافه کنید. همان کلاسی که در سرویس Web API ساختیم.

نکته: قسمت هایی از اپلیکیشن که باید در لایه‌های مختلف مورد استفاده قرار گیرند - مانند کلاس‌های موجودیت‌ها - بهتر است در لایه مجزایی قرار داده شده و به اشتراک گذاشته شوند. مثلا می‌توانید پروژه ای از نوع Class Library بسازید و تمام موجودیت‌ها را در آن تعریف کنید. سپس لایه‌های مختلف این پروژه را ارجاع خواهند کرد.

فایل program.cs را باز کنید و کد زیر را به آن اضافه نمایید.

```
private HttpClient _client;
private Order _order;

private static void Main()
{
    Task t = Run();
    t.Wait();

    Console.WriteLine("\nPress <enter> to continue...");
    Console.ReadLine();
}

private static async Task Run()
{
    // create instance of the program class
    var program = new Program();
    program.ServiceSetup();
    program.CreateOrder();
    // do not proceed until order is added
    await program.PostOrderAsync();
    program.ChangeOrder();
    // do not proceed until order is changed
    await program.PutOrderAsync();
    // do not proceed until order is removed
    await program.RemoveOrderAsync();
}

private void ServiceSetup()
{
    // map URL for Web API cal
    _client = new HttpClient { BaseAddress = new Uri("http://localhost:3237/") };
    // add Accept Header to request Web API content
```

```
    // negotiation to return resource in JSON format
    _client.DefaultRequestHeaders.Accept.
        Add(new MediaTypeWithQualityHeaderValue("application/json"));
}

private void CreateOrder()
{
    // Create new order
    _order = new Order { Product = "Camping Tent", Quantity = 3, Status = "Received" };
}

private async Task PostOrderAsync()
{
    // leverage Web API client side API to call service
    var response = await _client.PostAsJsonAsync("api/order", _order);
    Uri newOrderUri;

    if (response.IsSuccessStatusCode)
    {
        // Capture Uri of new resource
        newOrderUri = response.Headers.Location;
        // capture newly-created order returned from service,
        // which will now include the database-generated Id value
        _order = await response.Content.ReadAsAsync<Order>();
        Console.WriteLine("Successfully created order. Here is URL to new resource: {0}",
newOrderUri);
    }
    else
        Console.WriteLine("{0} ({1})", (int)response.StatusCode, response.ReasonPhrase);
}

private void ChangeOrder()
{
    // update order
    _order.Quantity = 10;
}

private async Task PutOrderAsync()
{
    // construct call to generate HttpPut verb and dispatch
    // to corresponding Put method in the Web API Service
    var response = await _client.PutAsJsonAsync("api/order", _order);

    if (response.IsSuccessStatusCode)
    {
        // capture updated order returned from service, which will include new quanity
        _order = await response.Content.ReadAsAsync<Order>();
        Console.WriteLine("Successfully updated order: {0}", response.StatusCode);
    }
    else
        Console.WriteLine("{0} ({1})", (int)response.StatusCode, response.ReasonPhrase);
}

private async Task RemoveOrderAsync()
{
    // remove order
    var uri = "api/order/" + _order.OrderId;
    var response = await _client.DeleteAsync(uri);

    if (response.IsSuccessStatusCode)
        Console.WriteLine("Sucessfully deleted order: {0}", response.StatusCode);
    else
        Console.WriteLine("{0} ({1})", (int)response.StatusCode, response.ReasonPhrase);
}
```

اگر اپلیکیشن کلاینت را اجرا کنید باید با خروجی زیر مواجه شوید:

```
Successfully created order: http://localhost:3237/api/order/1054
```

```
Successfully updated order: OK
```

```
Sucessfully deleted order: OK
```

شرح مثال جاری

با اجرای اپلیکیشن Web API شروع کنید. این اپلیکیشن یک کنترلر Web API دارد که پس از اجرا شما را به صفحه خانه هدایت می‌کند. در این مرحله اپلیکیشن در حال اجرا است و سرویس‌های ما قابل دسترسی هستند.

حال اپلیکیشن کنسول را باز کنید. روی خط اول کد program.cs یک breakpoint تعریف کرده و اپلیکیشن را اجرا کنید. ابتدا آدرس سرویس Web API را پیکربندی کرده و خاصیت Accept Header را مقدار دهی می‌کنیم. با این کار از سرویس مورد نظر درخواست می‌کنیم که داده‌ها را با فرمت JSON بازگرداند. سپس یک آبجکت Order می‌سازیم و با فراخوانی متد PostAsJsonAsync آن را به سرویس ارسال می‌کنیم. این متد روی آبجکت HttpClient تعریف شده است. اگر به اکشن متد Post در کنترلر Order یک breakpoint اضافه کنید، خواهید دید که این متد سفارش جدید را بعنوان یک پارامتر دریافت می‌کند و آن را به لیست موجودیت‌ها در Context جاری اضافه می‌نماید. این عمل باعث می‌شود که آبجکت جدید بعنوان Added علامت گذاری شود، در این مرحله Context جاری شروع به ردیابی تغییرات می‌کند. در آخر با فراخوانی متد SaveChanges داده‌ها را ذخیره می‌کنیم. در قدم بعدی کد وضعیت 201 (Created) و آدرس منبع جدید را در یک آبجکت HttpResponseMessage قرار می‌دهیم و به کلاینت ارسال می‌کنیم. هنگام استفاده از Web API باید اطمینان حاصل کنیم که کلاینت‌ها درخواست‌های ایجاد رکورد جدید را بصورت POST ارسال می‌کنند. درخواست‌های HTTP Post بصورت خودکار به اکشن متد متناظر نگاشت می‌شوند.

در مرحله بعد عملیات بعدی را اجرا می‌کنیم، تعداد سفارش را تغییر می‌دهیم و موجودیت جاری را با فراخوانی متد PutAsJsonAsync به سرویس Web API ارسال می‌کنیم. اگر به اکشن متد Put در کنترلر سرویس یک breakpoint اضافه کنید، خواهید دید که آبجکت سفارش بصورت یک پارامتر دریافت می‌شود. سپس با فراخوانی متد Entry و پاس دادن موجودیت جاری بعنوان رفرنس، خاصیت State را به Modified تغییر می‌دهیم، که این کار موجودیت را به Context جاری می‌چسباند. حال فراخوانی متد SaveChanges یک اسکریپت بروز رسانی تولید خواهد کرد. در مثال جاری تمام فیلدهای آبجکت Order را بروز رسانی می‌کنیم. در شماره‌های بعدی این سری از مقالات، خواهیم دید چگونه می‌توان تنها فیلدهایی را بروز رسانی کرد که تغییر کرده اند. در آخر عملیات را با بازگرداندن کد وضعیت 200 (OK) به اتمام می‌رسانیم.

در مرحله بعد، عملیات نهایی را اجرا می‌کنیم که موجودیت Order را از منبع داده حذف می‌کند. برای اینکار شناسه (Id) رکورد مورد نظر را به آدرس سرویس اضافه می‌کنیم و متد DeleteAsync را فراخوانی می‌کنیم. در سرویس Web API رکورد مورد نظر را از دیتابیس دریافت کرده و متد Remove را روی Context جاری فراخوانی می‌کنیم. این کار موجودیت مورد نظر را بعنوان Deleted علامت گذاری می‌کند. فراخوانی متد SaveChanges یک اسکریپت Delete تولید خواهد کرد که نهایتا منجر به حذف شدن رکورد می‌شود.

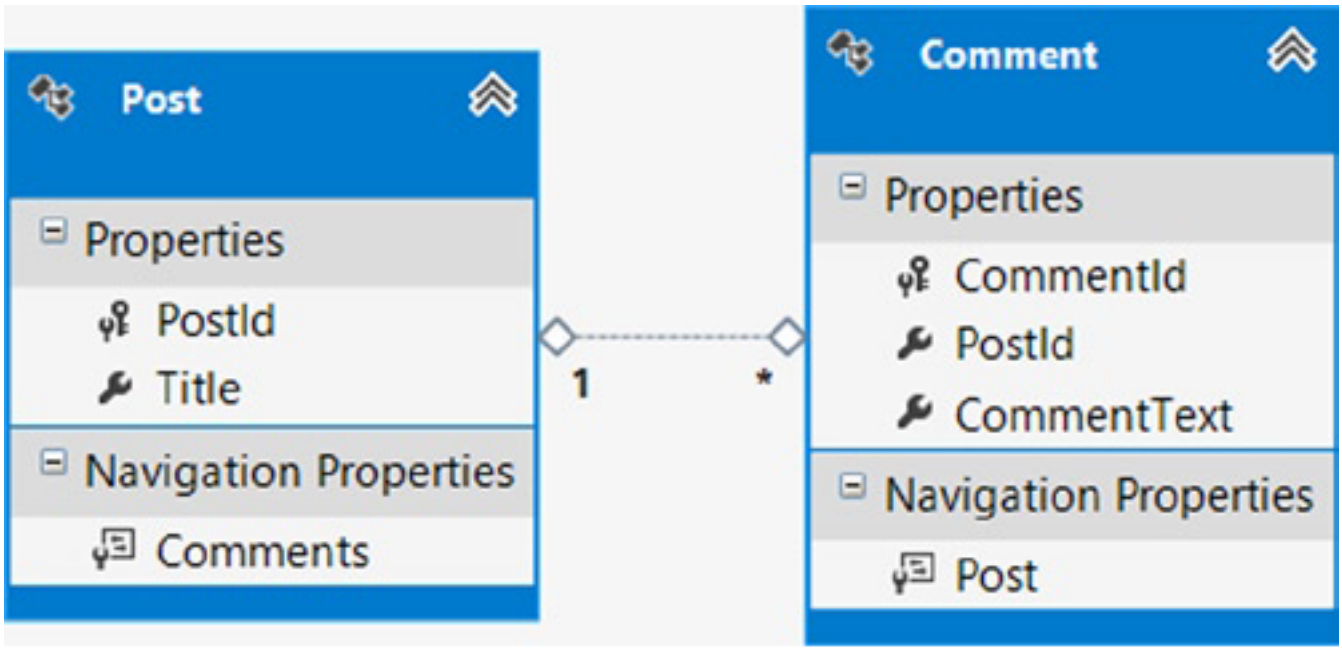
در یک اپلیکیشن واقعی بهتر است کد دسترسی داده‌ها از سرویس Web API تفکیک شود و در لایه مجزایی قرار گیرد.

عنوان: **استفاده از EF در اپلیکیشن های N-Tier : قسمت دوم**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۰۷ ۷:۴۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, Tips, WCF, N-Layer Architecture, Entity Framework 6

در قسمت قبل معماری اپلیکیشن‌های N-Tier و بروز رسانی موجودیت‌های منفصل توسط Web API را بررسی کردیم. در این قسمت بروز رسانی موجودیت‌های منفصل توسط WCF را بررسی می‌کنیم.

بروز رسانی موجودیت‌های منفصل توسط WCF

سناریویی را در نظر بگیرید که در آن عملیات CRUD توسط WCF پیاده سازی شده اند و دسترسی داده‌ها با مدل Code-First انجام می‌شود. فرض کنید مدل اپلیکیشن مانند تصویر زیر است.



همانطور که می‌بینید مدل ما متشکل از پست‌ها و نظرات کاربران است. برای ساده نگاه داشتن مثال جاری، اکثر فیلدها حذف شده اند. مثلا متن پست ها، نویسنده، تاریخ و زمان انتشار و غیره. می‌خواهیم تمام کد دسترسی داده‌ها را در یک سرویس WCF پیاده سازی کنیم تا کلاینت‌ها بتوانند عملیات CRUD را توسط آن انجام دهند. برای ساختن این سرویس مراحل زیر را دنبال کنید.
در ویژوال استودیو پروژه جدیدی از نوع Class Library بسازید و نام آن را به Recipe2 تغییر دهید.
با استفاده از NuGet Package Manager کتابخانه Entity Framework 6 را به پروژه اضافه کنید.
سه کلاس با نام‌های Post, Comment و Recipe2Context به پروژه اضافه کنید. کلاس‌های Post و Comment موجودیت‌های مدل ما هستند که به جداول متناظرشان نگاشت می‌شوند. کلاس Recipe2Context آبجکت DbContext ما خواهد بود که بعنوان درگاه عملیاتی EF عمل می‌کند. دقت کنید که خاصیت‌های لازم WCF یعنی DataContract و DataMember در کلاس‌های موجودیت‌ها بدرستی استفاده می‌شوند. لیست زیر کد این کلاس‌ها را نشان می‌دهد.

```
[DataContract(IsReference = true)]
public class Post
{
    public Post()
    {
        comments = new HashSet<Comments>();
    }

    [DataMember]
    public int PostId { get; set; }
```

```
    [DataMember]
    public string Title { get; set; }
    [DataMember]
    public virtual ICollection<Comment> Comments { get; set; }
}

[DataContract(IsReference=true)]
public class Comment
{
    [DataMember]
    public int CommentId { get; set; }
    [DataMember]
    public int PostId { get; set; }
    [DataMember]
    public string CommentText { get; set; }
    [DataMember]
    public virtual Post Post { get; set; }
}

public class EFRecipesEntities : DbContext
{
    public EFRecipesEntities() : base("name=EFRecipesEntities") {}

    public DbSet<Post> posts;
    public DbSet<Comment> comments;
}
```

یک فایل App.config به پروژه اضافه کنید و رشته اتصال زیر را به آن اضافه نمایید.

```
<connectionStrings>
  <add name="Recipe2ConnectionString"
    connectionString="Data Source=.;
    Initial Catalog=EFRecipes;
    Integrated Security=True;
    MultipleActiveResultSets=True"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

حال یک پروژه WCF به Solution جاری اضافه کنید. برای ساده نگاه داشتن مثال جاری، نام پیش فرض Service1 را بپذیرید. فایل IService1.cs را باز کنید و کد زیر را با محتوای آن جایگزین نمایید.

```
[ServiceContract]
public interface IService1
{
    [OperationContract]
    void Cleanup();
    [OperationContract]
    Post GetPostByTitle(string title);
    [OperationContract]
    Post SubmitPost(Post post);
    [OperationContract]
    Comment SubmitComment(Comment comment);
    [OperationContract]
    void DeleteComment(Comment comment);
}
```

فایل Service1.svc.cs را باز کنید و کد زیر را با محتوای آن جایگزین نمایید. بیاد داشته باشید که پروژه Recipe2 را ارجاع کنید و فضای نام آن را وارد نمایید. همچنین کتابخانه EF 6 را باید به پروژه اضافه کنید.

```
public class Service1 : IService
{
    public void Cleanup()
    {
        using (var context = new EFRecipesEntities())
        {
            context.Database.ExecuteSqlCommand("delete from [comments]");
            context. Database.ExecuteSqlCommand ("delete from [posts]");
        }
    }

    public Post GetPostByTitle(string title)
```

```
    {
        using (var context = new EFRecipesEntities())
        {
            context.Configuration.ProxyCreationEnabled = false;
            var post = context.Posts.Include(p => p.Comments).Single(p => p.Title == title);
            return post;
        }
    }

    public Post SubmitPost(Post post)
    {
        context.Entry(post).State =
            // if Id equal to 0, must be insert; otherwise, it's an update
            post.PostId == 0 ? EntityState.Added : EntityState.Modified;
        context.SaveChanges();
        return post;
    }

    public Comment SubmitComment(Comment comment)
    {
        using (var context = new EFRecipesEntities())
        {
            context.Comments.Attach(comment);
            if (comment.CommentId == 0)
            {
                // this is an insert
                context.Entry(comment).State = EntityState.Added);
            }
            else
            {
                // set single property to modified, which sets state of entity to modified, but
                // only updates the single property – not the entire entity
                context.entry(comment).Property(x => x.CommentText).IsModified = true;
            }
            context.SaveChanges();
            return comment;
        }
    }

    public void DeleteComment(Comment comment)
    {
        using (var context = new EFRecipesEntities())
        {
            context.Entry(comment).State = EntityState.Deleted;
            context.SaveChanges();
        }
    }
}
```

در آخر پروژه جدیدی از نوع Windows Console Application به Solution جاری اضافه کنید. از این اپلیکیشن بعنوان کلاینتی برای تست سرویس WCF استفاده خواهیم کرد. فایل program.cs را باز کنید و کد زیر را با محتوای آن جایگزین نمایید. روی نام پروژه کلیک راست کرده و گزینه Add Service Reference را انتخاب کنید، سپس ارجاعی به سرویس Service1 اضافه کنید. رفرنسی هم به کتابخانه کلاس‌ها که در ابتدای مراحل ساختید باید اضافه کنید.

```
class Program
{
    static void Main(string[] args)
    {
        using (var client = new ServiceReference2.Service1Client())
        {
            // cleanup previous data
            client.Cleanup();
            // insert a post
            var post = new Post { Title = "POCO Proxies" };
            post = client.SubmitPost(post);
            // update the post
            post.Title = "Change Tracking Proxies";
            client.SubmitPost(post);
            // add a comment
            var comment1 = new Comment { CommentText = "Virtual Properties are cool!", PostId =
post.PostId };
            var comment2 = new Comment { CommentText = "I use ICollection<T> all the time", PostId =
post.PostId };
            comment1 = client.SubmitComment(comment1);
            comment2 = client.SubmitComment(comment2);
            // update a comment
```

```
            comment1.CommentText = "How do I use ICollection<T>?";
            client.SubmitComment(comment1);
            // delete comment 1
            client.DeleteComment(comment1);
            // get posts with comments
            var p = client.GetPostByTitle("Change Tracking Proxies");
            Console.WriteLine("Comments for post: {0}", p.Title);
            foreach (var comment in p.Comments)
            {
                Console.WriteLine("\tComment: {0}", comment.CommentText);
            }
        }
    }
}
```

اگر اپلیکیشن کلاینت (برنامه کنسول) را اجرا کنید با خروجی زیر مواجه می‌شوید.

```
Comments for post: Change Tracking Proxies
Comment: I use ICollection<T> all the time
```

شرح مثال جاری

ابتدا با اپلیکیشن کنسول شروع می‌کنیم، که کلاینت سرویس ما است. نخست در یک بلاک {} using وهله ای از کلاینت سرویس مان ایجاد می‌کنیم. درست همانطور که وهله ای از یک EF Context می‌سازیم. استفاده از بلوک‌های using توصیه می‌شود چرا که متد Dispose بصورت خودکار فراخوانی خواهد شد، چه بصورت عادی چه هنگام بروز خطا. پس از آنکه وهله ای از کلاینت سرویس را در اختیار داشتیم، متد Cleanup را صدا می‌زنیم. با فراخوانی این متد تمام داده‌های تست پیشین را حذف می‌کنیم. در چند خط بعدی، متد SubmitPost را روی سرویس فراخوانی می‌کنیم. در پیاده سازی فعلی شناسه پست را بررسی می‌کنیم. اگر مقدار شناسه صفر باشد، خاصیت State موجودیت را به Added تغییر می‌دهید تا رکورد جدیدی ثبت کنیم. در غیر اینصورت فرض بر این است که چنین موجودیتی وجود دارد و قصد ویرایش آن را داریم، بنابراین خاصیت State را به Modified تغییر می‌دهیم. از آنجا که مقدار متغیرهای int بصورت پیش فرض صفر است، با این روش می‌توانیم وضعیت پست‌ها را مشخص کنیم. یعنی تعیین کنیم رکورد جدیدی باید ثبت شود یا رکوردی موجود بروز رسانی گردد. رویکردی بهتر آن است که پارامتری اضافی به متد پاس دهیم، یا متدی مجزا برای ثبت رکوردهای جدید تعریف کنیم. مثلا رکوردی با نام InsertPost. در هر حال، بهترین روش بستگی به ساختار اپلیکیشن شما دارد.

اگر پست جدیدی ثبت شود، خاصیت PostId با مقدار مناسب جدید بروز رسانی می‌شود و وهله پست را باز می‌گردانیم. ایجاد و بروز رسانی نظرات کاربران مشابه ایجاد و بروز رسانی پست‌ها است، اما با یک تفاوت اساسی: بعنوان یک قانون، هنگام بروز رسانی نظرات کاربران تنها فیلد متن نظر باید بروز رسانی شود. بنابراین با فیلدهای دیگری مانند تاریخ انتشار و غیره اصلا کاری نخواهیم داشت. بدین منظور تنها خاصیت CommentText را بعنوان Modified علامت گذاری می‌کنیم. این امر منجر می‌شود که Entity Framework عبارتی برای بروز رسانی تولید کند که تنها این فیلد را در بر می‌گیرد. توجه داشته باشید که این روش تنها در صورتی کار می‌کند که بخواهید یک فیلد واحد را بروز رسانی کنید. اگر می‌خواستیم فیلدهای بیشتری را در موجودیت Comment بروز رسانی کنیم، باید مکانیزمی برای ردیابی تغییرات در سمت کلاینت در نظر می‌گرفتیم. در مواقعی که خاصیت‌های متعددی می‌توانند تغییر کنند، معمولا بهتر است کل موجودیت بروز رسانی شود تا اینکه مکانیزمی پیچیده برای ردیابی تغییرات در سمت کلاینت پیاده گردد. بروز رسانی کل موجودیت بهینه‌تر خواهد بود.

برای حذف یک دیدگاه، متد Entry را روی آبجکت DbContext فراخوانی می‌کنیم و موجودیت مورد نظر را بعنوان آرگومان پاس می‌دهیم. این امر سبب می‌شود که موجودیت مورد نظر بعنوان Deleted علامت گذاری شود، که هنگام فراخوانی متد SaveChanges اسکریپت لازم برای حذف رکورد را تولید خواهد کرد.

در آخر متد GetPostByTitle یک پست را بر اساس عنوان پیدا کرده و تمام نظرات کاربران مربوط به آن را هم بارگذاری می‌کند. از آنجا که ما کلاس‌های POCO را پیاده سازی کرده ایم، Entity Framework آبجکتی را بر می‌گرداند که Dynamic Proxy نامیده می‌شود. این آبجکت پست و نظرات مربوط به آن را در بر خواهد گرفت. متاسفانه WCF نمی‌تواند آبجکت‌های پروکسی را مرتب سازی (serialize) کند. اما با غیرفعال کردن قابلیت ایجاد پروکسی‌ها (ProxyCreationEnabled=false) ما به Entity Framework

می‌گوییم که خود آبجکت‌های اصلی را بازگرداند. اگر سعی کنید آبجکت پروکسی را سریال کنید با پیغام خطای زیر مواجه خواهید شد:

```
The underlying connection was closed: The connection was closed unexpectedly
```

می توانیم غیرفعال کردن تولید پروکسی را به متد سازنده کلاس سرویس منتقل کنیم تا روی تمام متدهای سرویس اعمال شود.

در این قسمت دیدیم چگونه می‌توانیم از آبجکت‌های POCO برای مدیریت عملیات CRUD توسط WCF استفاده کنیم. از آنجا که هیچ اطلاعاتی درباره وضعیت موجودیت‌ها روی کلاینت ذخیره نمی‌شود، متدهایی مجزا برای عملیات CRUD ساختیم. در قسمت‌های بعدی خواهیم دید چگونه می‌توان تعداد متدهایی که سرویس مان باید پیاده سازی کند را کاهش داد و چگونه ارتباطات بین کلاینت و سرور را ساده‌تر کنیم.

نظرات خوانندگان

نویسنده: جلال
تاریخ: ۱۳۹۲/۱۲/۱۰ ۲۰:۲۰

در این سناریو، فرض را بر این گذاشته اید که موجودیتها یا جدید هستند و یا ویرایش شده اند و بنابراین حتی اگر یک پست کامنتهایی داشته باشد که ویرایش نشده اند، برای انها دستور update صادر میشود و این، در مواردیکه تعداد کامنتها(که البته همیشه این موجودیتهای اینگونه به سادگی کامنت نیستند) زیاد باشد، روی کارایی تأثیر منفی خواهد داشت، چه راهی برای تشخیص موجودیتهایی که سمت کلاینت تغییری نکرده اند، پیشنهاد میدهید؟

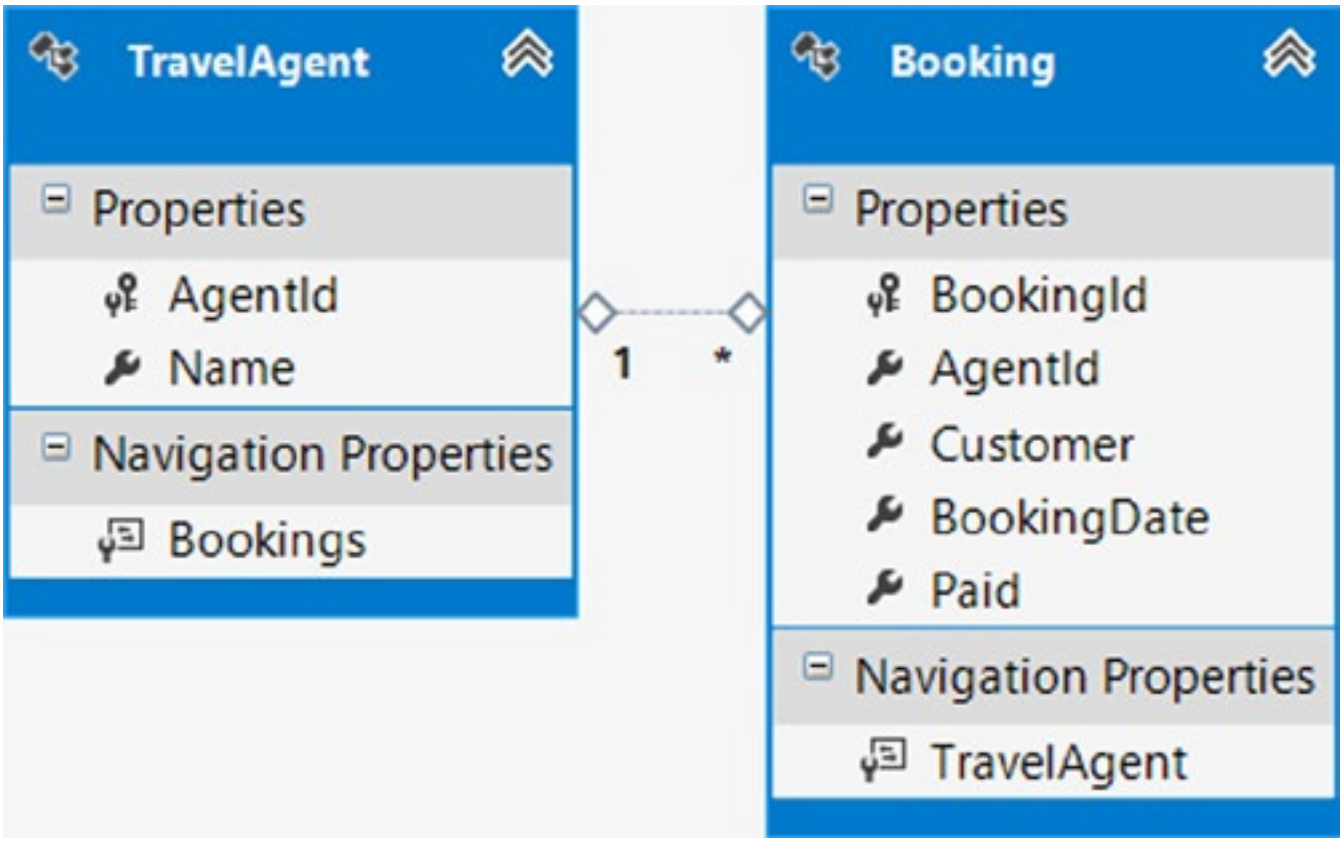
عنوان: **استفاده از EF در اپلیکیشن های N-Tier : قسمت سوم**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۰۸ ۱۰:۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, Tips, ASP.NET Web API, N-Layer Architecture, Web API, Entity Framework 6

در قسمت قبلی بروز رسانی موجودیت‌های منفصل با WCF را بررسی کردیم. در این قسمت خواهیم دید چگونه می‌توان تغییرات موجودیت‌ها را تشخیص داد و عملیات CRUD را روی یک Object Graph اجرا کرد.

تشخیص تغییرات با Web API

فرض کنید می‌خواهیم از سرویس‌های Web API برای انجام عملیات CRUD استفاده کنیم، اما بدون آنکه برای هر موجودیت متدهایی مجزا تعریف کنیم. به بیان دیگر می‌خواهیم عملیات مذکور را روی یک Object Graph انجام دهیم. مدیریت داده‌ها هم با مدل Code-First پیاده سازی می‌شود. در مثال جاری یک اپلیکیشن کنسول خواهیم داشت که بعنوان یک کلاینت سرویس را فراخوانی می‌کند. هر پروژه نیز در Solution مجزایی قرار دارد، تا یک محیط n-Tier را شبیه سازی کنیم.

مدل زیر را در نظر بگیرید.



همانطور که می‌بینید مدل ما آژانس‌های مسافرتی و رزرواسیون آنها را ارائه می‌کند. می‌خواهیم مدل و کد دسترسی داده‌ها را در یک سرویس Web API پیاده سازی کنیم تا هر کلاینتی که به HTTP دسترسی دارد بتواند عملیات CRUD را انجام دهد. برای ساختن سرویس مورد نظر مراحل زیر را دنبال کنید:

در ویژوال استودیو پروژه جدیدی از نوع ASP.NET Web Application بسازید و قالب پروژه را Web API انتخاب کنید. نام پروژه را به Recipe3.Service تغییر دهید.

کنترلر جدیدی بنام TravelAgentController به پروژه اضافه کنید.

دو کلاس جدید با نام‌های TravelAgent و Booking بسازید و کد آنها را مطابق لیست زیر تغییر دهید.

```
public class TravelAgent
{
    public TravelAgent()
    {
        this.Bookings = new HashSet<Booking>();
    }

    public int AgentId { get; set; }
    public string Name { get; set; }
    public virtual ICollection<Booking> Bookings { get; set; }
}

public class Booking
{
    public int BookingId { get; set; }
    public int AgentId { get; set; }
    public string Customer { get; set; }
    public DateTime BookingDate { get; set; }
    public bool Paid { get; set; }
    public virtual TravelAgent TravelAgent { get; set; }
}
```

با استفاده از NuGet Package Manager کتابخانه Entity Framework 6 را به پروژه اضافه کنید.

کلاس جدیدی بنام Recipe3Context بسازید و کد آن را مطابق لیست زیر تغییر دهید.

```
public class Recipe3Context : DbContext
{
    public Recipe3Context() : base("Recipe3ConnectionString") { }
    public DbSet<TravelAgent> TravelAgents { get; set; }
    public DbSet<Booking> Bookings { get; set; }

    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        modelBuilder.Entity<TravelAgent>().HasKey(x => x.AgentId);
        modelBuilder.Entity<TravelAgent>().ToTable("TravelAgents");
        modelBuilder.Entity<Booking>().ToTable("Bookings");
    }
}
```

فایل Web.config پروژه را باز کنید و رشته اتصال زیر را به قسمت ConnectionStrings اضافه کنید.

```
<connectionStrings>
  <add name="Recipe3ConnectionString"
    connectionString="Data Source=.;
    Initial Catalog=EFRecipes;
    Integrated Security=True;
    MultipleActiveResultSets=True"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

فایل Global.asax را باز کنید و کد زیر را به متد Application_Start اضافه نمایید. این کد بررسی Model Compatibility در EF را غیرفعال می‌کند. همچنین به JSON serializer می‌گوییم که self-referencing loop خاصیت‌های پیمایشی را نادیده بگیرد. این حلقه بدلیل ارتباط bidirectional بین موجودیت‌ها بوجود می‌آید.

```
protected void Application_Start()
{
    // Disable Entity Framework Model Compatibilty
    Database.SetInitializer<Recipe1Context>(null);

    // The bidirectional navigation properties between related entities
    // create a self-referencing loop that breaks Web API's effort to
    // serialize the objects as JSON. By default, Json.NET is configured
    // to error when a reference loop is detected. To resolve problem,
    // simply configure JSON serializer to ignore self-referencing loops.
    GlobalConfiguration.Configuration.Formatters.JsonFormatter
        .SerializerSettings.ReferenceLoopHandling =
        Newtonsoft.Json.ReferenceLoopHandling.Ignore;
    ...
```

```
}
```

فایل RouteConfig.cs را باز کنید و قوانین مسیریابی را مانند لیست زیر تغییر دهید.

```
public static void Register(HttpConfiguration config)
{
    config.Routes.MapHttpRoute(
      name: "ActionMethodSave",
      routeTemplate: "api/{controller}/{action}/{id}",
      defaults: new { id = RouteParameter.Optional });
}
```

در آخر کنترلر TravelAgent را باز کنید و کد آن را مطابق لیست زیر بروز رسانی کنید.

```
public class TravelAgentController : ApiController
{
    // GET api/travelagent
    [HttpGet]
    public IEnumerable<TravelAgent> Retrieve()
    {
        using (var context = new Recipe3Context())
        {
            return context.TravelAgents.Include(x => x.Bookings).ToList();
        }
    }

    /// <summary>
    /// Update changes to TravelAgent, implementing Action-Based Routing in Web API
    /// </summary>
    public HttpResponseMessage Update(TravelAgent travelAgent)
    {
        using (var context = new Recipe3Context())
        {
            var newParentEntity = true;
            // adding the object graph makes the context aware of entire
            // object graph (parent and child entities) and assigns a state
            // of added to each entity.
            context.TravelAgents.Add(travelAgent);
            if (travelAgent.AgentId > 0)
            {
                // as the Id property has a value greater than 0, we assume
                // that travel agent already exists and set entity state to
                // be updated.
                context.Entry(travelAgent).State = EntityState.Modified;
                newParentEntity = false;
            }

            // iterate through child entities, assigning correct state.
            foreach (var booking in travelAgent.Bookings)
            {
                if (booking.BookingId > 0)
                    // assume booking already exists if ID is greater than zero.
                    // set entity to be updated.
                    context.Entry(booking).State = EntityState.Modified;
            }

            context.SaveChanges();
            HttpResponseMessage response;
            // set Http Status code based on operation type
            response = Request.CreateResponse(newParentEntity ? HttpStatusCode.Created :
HttpStatusCode.OK, travelAgent);
            return response;
        }
    }

    [HttpDelete]
    public HttpResponseMessage Cleanup()
    {
        using (var context = new Recipe3Context())
        {
            context.Database.ExecuteSqlCommand("delete from [bookings]");
            context.Database.ExecuteSqlCommand("delete from [travelagents]");
        }
        return Request.CreateResponse(HttpStatusCode.OK);
    }
```

```
}
```

در قدم بعدی کلاینت پروژه را می‌سازیم که از سرویس Web API مان استفاده می‌کند.

در ویژوال استودیو پروژه جدیدی از نوع Console application بسازید و نام آن را به Recipe3.Client تغییر دهید.
فایل program.cs را باز کنید و کد آن را مطابق لیست زیر بروز رسانی کنید.

```
internal class Program
{
    private HttpClient _client;
    private TravelAgent _agent1, _agent2;
    private Booking _booking1, _booking2, _booking3;
    private HttpResponseMessage _response;

    private static void Main()
    {
        Task t = Run();
        t.Wait();
        Console.WriteLine("\nPress <enter> to continue...");
        Console.ReadLine();
    }

    private static async Task Run()
    {
        var program = new Program();
        program.ServiceSetup();
        // do not proceed until clean-up is completed
        await program.CleanupAsync();
        program.CreateFirstAgent();
        // do not proceed until agent is created
        await program.AddAgentAsync();
        program.CreateSecondAgent();
        // do not proceed until agent is created
        await program.AddSecondAgentAsync();
        program.ModifyAgent();
        // do not proceed until agent is updated
        await program.UpdateAgentAsync();
        // do not proceed until agents are fetched
        await program.FetchAgentsAsync();
    }

    private void ServiceSetup()
    {
        // set up infrastructure for Web API call
        _client = new HttpClient {BaseAddress = new Uri("http://localhost:6687/")};
        // add Accept Header to request Web API content negotiation to return resource in JSON format
        _client.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));
    }

    private async Task CleanupAsync()
    {
        // call cleanup method in service
        _response = await _client.DeleteAsync("api/travelagent/cleanup/");
    }

    private void CreateFirstAgent()
    {
        // create new Travel Agent and booking
        _agent1 = new TravelAgent {Name = "John Tate"};
        _booking1 = new Booking
        {
            Customer = "Karen Stevens",
            Paid = false,
            BookingDate = DateTime.Parse("2/2/2010")
        };

        _booking2 = new Booking
        {
            Customer = "Dolly Parton",
            Paid = true,
            BookingDate = DateTime.Parse("3/10/2010")
        };

        _agent1.Bookings.Add(_booking1);
        _agent1.Bookings.Add(_booking2);
```

```
    }

    private async Task AddAgentAsync()
    {
        // call generic update method in Web API service to add agent and bookings
        _response = await _client.PostAsync("api/travelagent/update/",
            _agent1, new JsonMediaTypeFormatter());

        if (_response.IsSuccessStatusCode)
        {
            // capture newly created travel agent from service, which will include
            // database-generated Ids for each entity
            _agent1 = await _response.Content.ReadAsAsync<TravelAgent>();
            _booking1 = _agent1.Bookings.FirstOrDefault(x => x.Customer == "Karen Stevens");
            _booking2 = _agent1.Bookings.FirstOrDefault(x => x.Customer == "Dolly Parton");

            Console.WriteLine("Successfully created Travel Agent {0} and {1} Booking(s)",
            _agent1.Name, _agent1.Bookings.Count);
        }
        else
            Console.WriteLine("{0} ({1})", (int) _response.StatusCode, _response.ReasonPhrase);
    }

    private void CreateSecondAgent()
    {
        // add new agent and booking
        _agent2 = new TravelAgent {Name = "Perry Como"};
        _booking3 = new Booking {
            Customer = "Loretta Lynn",
            Paid = true,
            BookingDate = DateTime.Parse("3/15/2010")};
        _agent2.Bookings.Add(_booking3);
    }

    private async Task AddSecondAgentAsync()
    {
        // call generic update method in Web API service to add agent and booking
        _response = await _client.PostAsync("api/travelagent/update/", _agent2, new
JsonMediaTypeFormatter());

        if (_response.IsSuccessStatusCode)
        {
            // capture newly created travel agent from service
            _agent2 = await _response.Content.ReadAsAsync<TravelAgent>();
            _booking3 = _agent2.Bookings.FirstOrDefault(x => x.Customer == "Loretta Lynn");
            Console.WriteLine("Successfully created Travel Agent {0} and {1} Booking(s)",
                _agent2.Name, _agent2.Bookings.Count);
        }
        else
            Console.WriteLine("{0} ({1})", (int) _response.StatusCode, _response.ReasonPhrase);
    }

    private void ModifyAgent()
    {
        // modify agent 2 by changing agent name and assigning booking 1 to him from agent 1
        _agent2.Name = "Perry Como, Jr.";
        _agent2.Bookings.Add(_booking1);
    }

    private async Task UpdateAgentAsync()
    {
        // call generic update method in Web API service to update agent 2
        _response = await _client.PostAsync("api/travelagent/update/", _agent2, new
JsonMediaTypeFormatter());
        if (_response.IsSuccessStatusCode)
        {
            // capture newly created travel agent from service, which will include Ids
            _agent1 = _response.Content.ReadAsAsync<TravelAgent>().Result;
            Console.WriteLine("Successfully updated Travel Agent {0} and {1} Booking(s)", _agent1.Name,
_agent1.Bookings.Count);
        }
        else
            Console.WriteLine("{0} ({1})", (int) _response.StatusCode, _response.ReasonPhrase);
    }

    private async Task FetchAgentsAsync()
    {
        // call Get method on service to fetch all Travel Agents and Bookings
        _response = _client.GetAsync("api/travelagent/retrieve").Result;
        if (_response.IsSuccessStatusCode)
        {
```

```
            // capture newly created travel agent from service, which will include Ids
            var agents = await _response.Content.ReadAsAsync<IEnumerable<TravelAgent>>();

            foreach (var agent in agents)
            {
                Console.WriteLine("Travel Agent {0} has {1} Booking(s)", agent.Name,
agent.Bookings.Count());
            }
        }
        else
            Console.WriteLine("{0} ({1})", (int) _response.StatusCode, _response.ReasonPhrase);
    }
}
```

در آخر کلاس‌های TravelAgent و Booking را به پروژه کلاینت اضافه کنید. اینگونه کدها بهتر است در لایه مجزایی قرار گیرند و بین پروژه‌ها به اشتراک گذاشته شوند.

اگر اپلیکیشن کنسول (کلاینت) را اجرا کنید با خروجی زیر مواجه خواهید شد.

```
Successfully created Travel Agent John Tate and 2 Booking(s)
Successfully created Travel Agent Perry Como and 1 Booking(s)
Successfully updated Travel Agent Perry Como, Jr. and 2 Booking(s)
Travel Agent John Tate has 1 Booking(s)
Travel Agent Perry Como, Jr. has 2 Booking(s)
```

شرح مثال جاری

با اجرای اپلیکیشن Web API شروع کنید. این اپلیکیشن یک کنترلر MVC Web Controller دارد که پس از اجرا شما را به صفحه خانه هدایت می‌کند. در این مرحله سایت در حال اجرا است و سرویس‌ها قابل دسترسی هستند.

سپس اپلیکیشن کنسول را باز کنید، روی خط اول کد فایل program.cs یک breakpoint قرار دهید و آن را اجرا کنید. ابتدا آدرس سرویس Web API را نگاشت می‌کنیم و با تنظیم مقدار خاصیت Accept Header از سرویس درخواست می‌کنیم که اطلاعات را با فرمت JSON بازگرداند.

بعد از آن با استفاده از آبجکت HttpClient متد DeleteAsync را فراخوانی می‌کنیم که روی کنترلر TravelAgent تعریف شده است. این متد تمام داده‌های پیشین را حذف میکند.

در قدم بعدی سه آبجکت جدید می‌سازیم: یک آژانس مسافرتی و دو رزرواسیون. سپس این آبجکت‌ها را با فراخوانی متد PostAsync روی آبجکت HttpClient به سرویس ارسال می‌کنیم. اگر به متد Update در کنترلر TravelAgent یک breakpoint اضافه کنید، خواهید دید که این متد آبجکت آژانس مسافرتی را بعنوان یک پارامتر دریافت می‌کند و آن را به موجودیت TravelAgents در Context جاری اضافه می‌نماید. این کار آبجکت آژانس مسافرتی و تمام آبجکت‌های فرزند آن را در حالت Added اضافه می‌کند و باعث می‌شود که context جاری شروع به ردیابی (tracking) آنها کند.

نکته: قابل ذکر است که اگر موجودیت‌های متعددی با مقداری یکسان در خاصیت کلید اصلی (Primary-key value) دارید باید مجموعه آبجکت‌های خود را Add کنید و نه Attach. در مثال جاری چند آبجکت Booking داریم که مقدار کلید اصلی آنها صفر است (Bookings with Id = 0). اگر از Attach استفاده کنید EF پیغام خطایی صادر می‌کند چرا که چند موجودیت با مقادیر کلید اصلی یکسان به context جاری اضافه کرده اید.

بعد از آن بر اساس مقدار خاصیت Id مشخص می‌کنیم که موجودیت‌ها باید بروز رسانی شوند یا خیر. اگر مقدار این فیلد بزرگتر از صفر باشد، فرض بر این است که این موجودیت در دیتابیس وجود دارد بنابراین خاصیت EntityState را به Modified تغییر می‌دهیم. علاوه بر این فیلدی هم با نام newParentEntity تعریف کرده ایم که توسط آن بتوانیم کد وضعیت مناسبی به کلاینت بازگردانیم. در صورتی که مقدار فیلد Id در موجودیت TravelAgent برابر با یک باشد، مقدار خاصیت EntityState را به همان

Added رها می‌کنیم.

سپس تمام آبجکت‌های فرزند آژانس مسافرتی (رزرواسیون ها) را بررسی میکنیم و همین منطق را روی آنها اعمال می‌کنیم. یعنی در صورتی که مقدار فیلد Id آنها بزرگتر از 0 باشد وضعیت EntityState را به Modified تغییر می‌دهیم. در نهایت متد SaveChanges را فراخوانی می‌کنیم. در این مرحله برای موجودیت‌های جدید اسکریپت‌های Insert و برای موجودیت‌های تغییر کرده اسکریپت‌های Update تولید می‌شود. سپس کد وضعیت مناسب را به کلاینت بر می‌گردانیم. برای موجودیت‌های اضافه شده کد وضعیت 201 (Created) و برای موجودیت‌های بروز رسانی شده کد وضعیت 200 (OK) باز می‌گردد. کد 201 به کلاینت اطلاع می‌دهد که رکورد جدید با موفقیت ثبت شده است، و کد 200 از بروز رسانی موفقیت آمیز خبر می‌دهد. هنگام تولید سرویس‌های REST-based بهتر است همیشه کد وضعیت مناسبی تولید کنید.

پس از این مراحل، آژانس مسافرتی و رزرواسیون جدیدی می‌سازیم و آنها را به سرویس ارسال می‌کنیم. سپس نام آژانس مسافرتی دوم را تغییر می‌دهیم، و یکی از رزرواسیون‌ها را از آژانس اولی به آژانس دومی منتقل می‌کنیم. اینبار هنگام فراخوانی متد Update تمام موجودیت‌ها شناسه ای بزرگتر از 1 دارند، بنابراین وضعیت EntityState آنها را به Modified تغییر می‌دهیم تا هنگام ثبت تغییرات دستورات بروز رسانی مناسب تولید و اجرا شوند.

در آخر کلاینت ما متد Retreive را روی سرویس فراخوانی می‌کند. این فراخوانی با کمک متد GetAsync انجام می‌شود که روی آبجکت HttpClient تعریف شده است. فراخوانی این متد تمام آژانس‌های مسافرتی بهمراه رزرواسیون‌های متناظرشان را دریافت می‌کند. در اینجا با استفاده از متد Include تمام رکوردهای فرزند را بهمراه تمام خاصیت هایشان (properties) بارگذاری می‌کنیم.

دقت کنید که مرتب کننده JSON تمام خواص عمومی (public properties) را باز می‌گرداند، حتی اگر در کد خود تعداد مشخصی از آنها را انتخاب کرده باشید.

نکته دیگر آنکه در مثال جاری از قراردادهای توکار Web API برای نگاشت درخواست‌های HTTP به اکشن متدها استفاده نکرده ایم. مثلا بصورت پیش فرض درخواست‌های POST به متدهایی نگاشت می‌شوند که نام آنها با "Post" شروع می‌شود. در مثال جاری قواعد مسیریابی را تغییر داده ایم و رویکرد مسیریابی RPC-based را در پیش گرفته ایم. در اپلیکیشن‌های واقعی بهتر است از قواعد پیش فرض استفاده کنید چرا که هدف Web API ارائه سرویس‌های REST-based است. بنابراین بعنوان یک قاعده کلی بهتر است متدهای سرویس شما به درخواست‌های متناظر HTTP نگاشت شوند. و در آخر آنکه بهتر است لایه مجزایی برای میزبانی کدهای دسترسی داده ایجاد کنید و آنها را از سرویس Web API تفکیک نمایید.

نظرات خوانندگان

نویسنده: وحید
تاریخ: ۱۳۹۲/۱۱/۱۱ ۱۱:۶

با سلام شما فرمودید: " و در آخر آنکه بهتر است لایه مجزایی برای میزبانی کدهای دسترسی داده ایجاد کنید و آنها را از سرویس Web API تفکیک نمایید. " برای برقراری امنیت در این سرویس چه باید کرد؟ اگر شخصی آدرس سرویس ما رو داشت و در خواست‌های را به آن ارسال کرد چگونه آن را نسبت به بقیه کاربران تمیز کند؟ چون در حقیقت webapi را در پروژه جدیدی در solution قرار دادیم و جدا هاست می‌شود. ممنون

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۱۱ ۱۱:۴۲

برای برقراری امنیت، تعیین هویت و اعتبارسنجی در وب API عموما یا از Forms authentication استفاده می‌شود و یا از ASP.NET Identity . زیر ساخت آن یکی است و مشترک.

عنوان: **طریقه حذف Work item از Team Foundation Server**
نویسنده: حمیدرضا معینی
تاریخ: ۱۳۹۲/۱۱/۰۹ ۸:۴۰
آدرس: www.dotnettips.info
گروه‌ها: Tips, TFS, Agile Planning Tools

چنانچه برای مدیریت پروژه از Team Foundation Server استفاده می‌کنید، صرف نظر از اینکه نسخه اکسپرس یا کامل آنرا روی سرور نصب کرده اید و یا از نسخه رایگانی که مایکروسافت در آدرس http://tfs.visualstudio.com در اختیار کاربران قرار داده استفاده می‌کنید، یقیناً میدانید که Backlog item و Work item می‌توانند در وضعیت‌های مختلفی قرار بگیرند. این وضعیت‌ها بسته به متدلوژی مورد استفاده، از لحاظ تعداد و یا عنوان با یکدیگر متفاوتند.
به عنوان مثال:

New

Approved

Committed

Done

Removed

در وضعیت‌های فوق چنانچه یک Work item اشتباهاً به Backlog افزوده شود، تنها راه اینکه اعلام کنیم این Work item از دستور کار خارج شده است، اینست که آنرا در وضعیت Removed قرار دهیم.

در هیچ کجای رابط کاربری (تحت وب و Team explorer در ویژوال استدیو) امکانی برای حذف یک Work item در نظر گرفته نشده است.

در ادامه روش حذف فیزیکی یک Work item از TFS را خواهیم دید:

1- ابتدا وارد خط فرمان (Command Prompt) شده و به مسیر زیر بروید:

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

2- با استفاده از دستور witadmin بهمراه سوییچ destroywi، می‌توانید Work item مورد نظر را با استفاده از شناسه آن حذف کنید:

```
witadmin destroywi /collection:CollectionURL /id:id
```

به عنوان مثال چنانچه آدرس CollectionURL پروژه شما https://mytfs.visualstudio.com/DefaultCollection و شناسه Work item مورن نظر 357 باشد، خواهیم داشت:

```
witadmin   destroywi   /collection:"https://mytfs.visualstudio.com/DefaultCollection "   /id:357
```

چنانچه این قابلیت نظر شما را جلب کرد، توصیه می‌کنم نگاهی هم به سایر سوییچ‌های فرمان witadmin بیاندازید.

عنوان: **استفاده از EF در اپلیکیشن های N-Tier : قسمت چهارم**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۰۹ ۸:۴۰
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, Tips, ASP.NET Web API, N-Layer Architecture, Entity Framework 6, ASP.NET Web API 2

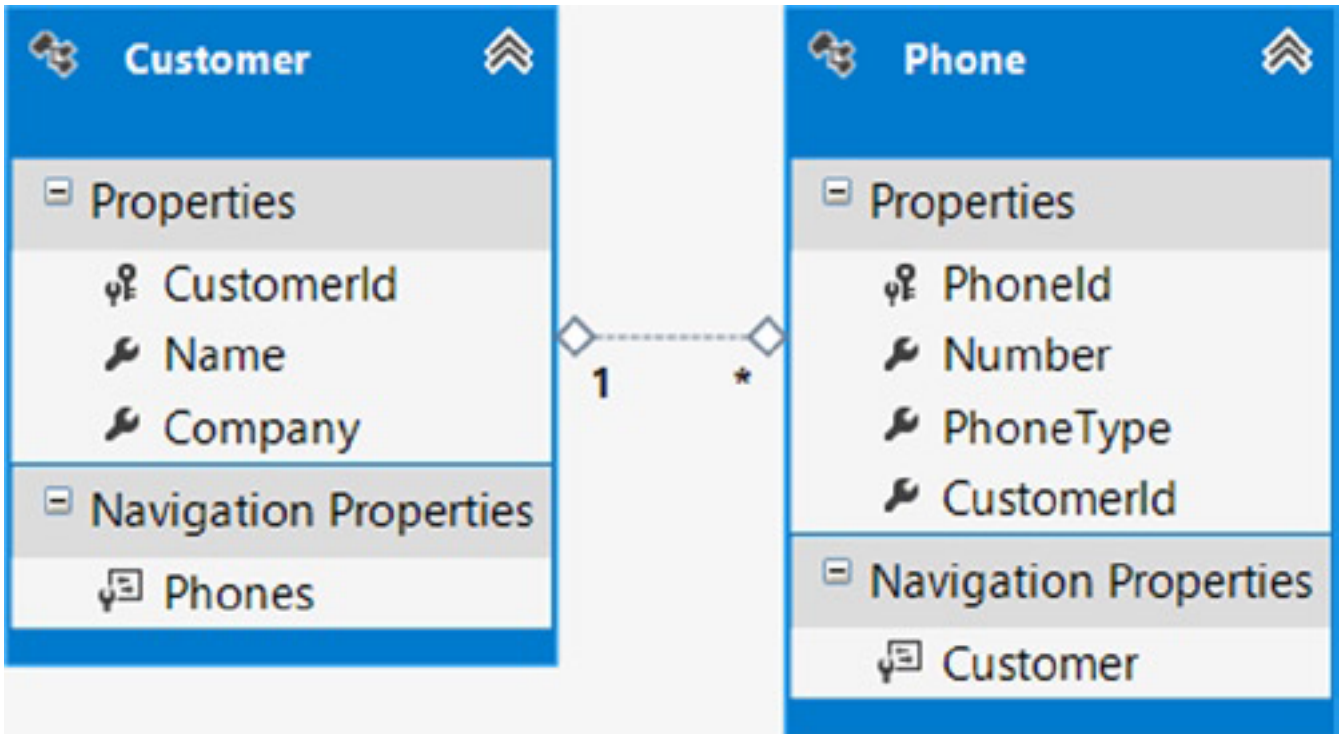
در قسمت قبل تشخیص تغییرات توسط Web API را بررسی کردیم. در این قسمت نگاهی به پیاده سازی Change-tracking در سمت کلاینت خواهیم داشت.

ردیابی تغییرات در سمت کلاینت توسط Web API

فرض کنید می‌خواهیم از سرویس‌های REST-based برای انجام عملیات CRUD روی یک Object graph استفاده کنیم. همچنین می‌خواهیم رویکردی در سمت کلاینت برای بروز رسانی کلاس موجودیت‌ها پیاده سازی کنیم که قابل استفاده مجدد (reusable) باشد. علاوه بر این دسترسی داده‌ها توسط مدل Code-First انجام می‌شود.

در مثال جاری یک اپلیکیشن کلاینت (برنامه کنسول) خواهیم داشت که سرویس‌های ارائه شده توسط پروژه Web API را فراخوانی می‌کند. هر پروژه در یک Solution مجزا قرار دارد، با این کار یک محیط n-Tier را شبیه سازی می‌کنیم.

مدل زیر را در نظر بگیرید.



همانطور که می‌بینید مدل مثال جاری مشتریان و شماره تماس آنها را ارائه می‌کند. می‌خواهیم مدل‌ها و کد دسترسی به داده‌ها را در یک سرویس Web API پیاده سازی کنیم تا هر کلاینتی که به HTTP دسترسی دارد بتواند از آن استفاده کند. برای ساخت سرویس مذکور مراحل زیر را دنبال کنید.

در ویژوال استودیو پروژه جدیدی از نوع ASP.NET Web Application بسازید و قالب پروژه را Web API انتخاب کنید. نام پروژه را به Recipe4.Service تغییر دهید.

کنترلر جدیدی با نام CustomerController به پروژه اضافه کنید.

کلاسی با نام BaseEntity ایجاد کنید و کد آن را مطابق لیست زیر تغییر دهید. تمام موجودیت‌ها از این کلاس پایه مشتق خواهند

شد که خاصیتی بنام TrackingState را به آنها اضافه می‌کند. کلاینت‌ها هنگام ویرایش آبجکت موجودیت‌ها باید این فیلد را مقدار دهی کنند. همانطور که می‌بینید این خاصیت از نوع TrackingState enum مشتق می‌شود. توجه داشته باشید که این خاصیت در دیتابیس ذخیره نخواهد شد. با پیاده سازی enum وضعیت ردیابی موجودیت‌ها بدین روش، وابستگی‌های EF را برای کلاینت از بین می‌بریم. اگر قرار بود وضعیت ردیابی را مستقیما از EF به کلاینت پاس دهیم وابستگی‌های بخصوصی معرفی می‌شدند. کلاس DbContext اپلیکیشن در متد OnModelCreating به EF دستور می‌دهد که خاصیت TrackingState را به جدول موجودیت نگاشت نکند.

```
public abstract class BaseEntity
{
    protected BaseEntity()
    {
        TrackingState = TrackingState.Nochange;
    }

    public TrackingState TrackingState { get; set; }
}

public enum TrackingState
{
    Nochange,
    Add,
    Update,
    Remove,
}
```

کلاس‌های موجودیت Customer و PhoneNumber را ایجاد کنید و کد آنها را مطابق لیست زیر تغییر دهید.

```
public class Customer : BaseEntity
{
    public int CustomerId { get; set; }
    public string Name { get; set; }
    public string Company { get; set; }
    public virtual ICollection<Phone> Phones { get; set; }
}

public class Phone : BaseEntity
{
    public int PhoneId { get; set; }
    public string Number { get; set; }
    public string PhoneType { get; set; }
    public int CustomerId { get; set; }
    public virtual Customer Customer { get; set; }
}
```

با استفاده از NuGet Package Manager کتابخانه Entity Framework 6 را به پروژه اضافه کنید.

کلاسی با نام Recipe4Context ایجاد کنید و کد آن را مطابق لیست زیر تغییر دهید. در این کلاس از یکی از قابلیت‌های جدید EF 6 بنام "Configuring Unmapped Base Types" استفاده کرده ایم. با استفاده از این قابلیت جدید هر موجودیت را طوری پیکربندی می‌کنیم که خاصیت TrackingState را نادیده بگیرند. برای اطلاعات بیشتر درباره این قابلیت EF 6 به این لینک مراجعه کنید.

```
public class Recipe4Context : DbContext
{
    public Recipe4Context() : base("Recipe4ConnectionString") { }
    public DbSet<Customer> Customers { get; set; }
    public DbSet<Phone> Phones { get; set; }

    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        // Do not persist TrackingState property to data store
        // This property is used internally to track state of
        // disconnected entities across service boundaries.
        // Leverage the Custom Code First Conventions features from Entity Framework 6.
        // Define a convention that performs a configuration for every entity
        // that derives from a base entity class.
        modelBuilder.Types<BaseEntity>().Configure(x => x.Ignore(y => y.TrackingState));
        modelBuilder.Entity<Customer>().ToTable("Customers");
        modelBuilder.Entity<Phone>().ToTable("Phones");
}
}
```

فایل Web.config پروژه را باز کنید و رشته اتصال زیر را به قسمت ConnectionStrings اضافه نمایید.

```
<connectionStrings>
  <add name="Recipe4ConnectionString"
    connectionString="Data Source=.;
    Initial Catalog=EFRecipes;
    Integrated Security=True;
    MultipleActiveResultSets=True"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

فایل Global.asax را باز کنید و کد زیر را به متد Application_Start اضافه نمایید. این کد بررسی Entity Framework Model Compatibility را غیرفعال می‌کند و به JSON serializer دستور می‌دهد که self-referencing loop خواص پیمایشی را نادیده بگیرد. این حلقه بدلیل رابطه bidirectional بین موجودیت‌های Customer و PhoneNumber بوجود می‌آید.

```
protected void Application_Start()
{
    // Disable Entity Framework Model Compatibilty
    Database.SetInitializer<Recipe1Context>(null);
    // The bidirectional navigation properties between related entities
    // create a self-referencing loop that breaks Web API's effort to
    // serialize the objects as JSON. By default, Json.NET is configured
    // to error when a reference loop is detected. To resolve problem,
    // simply configure JSON serializer to ignore self-referencing loops.
    GlobalConfiguration.Configuration.Formatters.JsonFormatter
        .SerializerSettings.ReferenceLoopHandling =
            Newtonsoft.Json.ReferenceLoopHandling.Ignore;
    ...
}
```

کلاسی با نام EntityStateFactory بسازید و کد آن را مطابق لیست زیر تغییر دهید. این کلاس مقدار خاصیت TrackingState که به کلاینت‌ها ارائه می‌شود را به مقادیر متناظر کامپوننت‌های ردیابی EF تبدیل می‌کند.

```
public static EntityState Set(TrackingState trackingState)
{
    switch (trackingState)
    {
        case TrackingState.Add:
            return EntityState.Added;
        case TrackingState.Update:
            return EntityState.Modified;
        case TrackingState.Remove:
            return EntityState.Deleted;
        default:
            return EntityState.Unchanged;
    }
}
```

در آخر کد کنترلر CustomerController را مطابق لیست زیر بروز رسانی کنید.

```
public class CustomerController : ApiController
{
    // GET api/customer
    public IEnumerable<Customer> Get()
    {
        using (var context = new Recipe4Context())
        {
            return context.Customers.Include(x => x.Phones).ToList();
        }
    }

    // GET api/customer/5
    public Customer Get(int id)
    {
        using (var context = new Recipe4Context())
        {
            return context.Customers.Include(x => x.Phones).FirstOrDefault(x => x.CustomerId == id);
        }
```

```
    }

    [ActionName("Update")]
    public HttpResponseMessage UpdateCustomer(Customer customer)
    {
        using (var context = new Recipe4Context())
        {
            // Add object graph to context setting default state of 'Added'.
            // Adding parent to context automatically attaches entire graph
            // (parent and child entities) to context and sets state to 'Added'
            // for all entities.
            context.Customers.Add(customer);
            foreach (var entry in context.ChangeTracker.Entries<BaseEntity>())
            {
                entry.State = EntityStateFactory.Set(entry.Entity.TrackingState);
                if (entry.State == EntityState.Modified)
                {
                    // For entity updates, we fetch a current copy of the entity
                    // from the database and assign the values to the orginal values
                    // property from the Entry object. OriginalValues wrap a dictionary
                    // that represents the values of the entity before applying changes.
                    // The Entity Framework change tracker will detect
                    // differences between the current and original values and mark
                    // each property and the entity as modified. Start by setting
                    // the state for the entity as 'Unchanged'.
                    entry.State = EntityState.Unchanged;
                    var databaseValues = entry.GetDatabaseValues();
                    entry.OriginalValues.SetValues(databaseValues);
                }
            }

        context.SaveChanges();
    }

    return Request.CreateResponse(HttpStatusCode.OK, customer);
}

    [HttpDelete]
    [ActionName("Cleanup")]
    public HttpResponseMessage Cleanup()
    {
        using (var context = new Recipe4Context())
        {
            context.Database.ExecuteSqlCommand("delete from phones");
            context.Database.ExecuteSqlCommand("delete from customers");
            return Request.CreateResponse(HttpStatusCode.OK);
        }
    }
}
```

حال اپلیکیشن کلاینت (برنامه کنسول) را می‌سازیم که از این سرویس استفاده می‌کند.

در ویژوال استودیو پروژه جدیدی از نوع Console Application بسازید و نام آن را به Recipe4.Client تغییر دهید.

فایل program.cs را باز کنید و کد آن را مطابق لیست زیر تغییر دهید.

```
internal class Program
{
    private HttpClient _client;
    private Customer _bush, _obama;
    private Phone _whiteHousePhone, _bushMobilePhone, _obamaMobilePhone;
    private HttpResponseMessage _response;

    private static void Main()
    {
        Task t = Run();
        t.Wait();
        Console.WriteLine("\nPress <enter> to continue...");
        Console.ReadLine();
    }

    private static async Task Run()
    {
        var program = new Program();
        program.ServiceSetup();
        // do not proceed until clean-up completes
        await program.CleanupAsync();
```

```
        program.CreateFirstCustomer();
        // do not proceed until customer is added
        await program.AddCustomerAsync();
        program.CreateSecondCustomer();
        // do not proceed until customer is added
        await program.AddSecondCustomerAsync();
        // do not proceed until customer is removed
        await program.RemoveFirstCustomerAsync();
        // do not proceed until customers are fetched
        await program.FetchCustomersAsync();
    }

    private void ServiceSetup()
    {
        // set up infrastructure for Web API call
        _client = new HttpClient { BaseAddress = new Uri("http://localhost:62799/") };
        // add Accept Header to request Web API content negotiation to return resource in JSON format
        _client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue
        ("application/json"));
    }
    private async Task CleanupAsync()
    {
        // call the cleanup method from the service
        _response = await _client.DeleteAsync("api/customer/cleanup/");
    }

    private void CreateFirstCustomer()
    {
        // create customer #1 and two phone numbers
        _bush = new Customer
        {
            Name = "George Bush",
            Company = "Ex President",
            // set tracking state to 'Add' to generate a SQL Insert statement
            TrackingState = TrackingState.Add,
        };
        _whiteHousePhone = new Phone
        {
            Number = "212 222-2222",
            PhoneType = "White House Red Phone",
            // set tracking state to 'Add' to generate a SQL Insert statement
            TrackingState = TrackingState.Add,
        };
        _bushMobilePhone = new Phone
        {
            Number = "212 333-3333",
            PhoneType = "Bush Mobile Phone",
            // set tracking state to 'Add' to generate a SQL Insert statement
            TrackingState = TrackingState.Add,
        };
        _bush.Phones.Add(_whiteHousePhone);
        _bush.Phones.Add(_bushMobilePhone);
    }

    private async Task AddCustomerAsync()
    {
        // construct call to invoke UpdateCustomer action method in Web API service
        _response = await _client.PostAsync("api/customer/updatecustomer/", _bush, new
JsonMediaTypeFormatter());
        if (_response.IsSuccessStatusCode)
        {
            // capture newly created customer entity from service, which will include
            // database-generated Ids for all entities
            _bush = await _response.Content.ReadAsAsync<Customer>();
            _whiteHousePhone = _bush.Phones.FirstOrDefault(x => x.CustomerId == _bush.CustomerId);
            _bushMobilePhone = _bush.Phones.FirstOrDefault(x => x.CustomerId == _bush.CustomerId);
            Console.WriteLine("Successfully created Customer {0} and {1} Phone Numbers(s)",
            _bush.Name, _bush.Phones.Count);
            foreach (var phoneType in _bush.Phones)
            {
                Console.WriteLine("Added Phone Type: {0}", phoneType.PhoneType);
            }
        }
        else
            Console.WriteLine("{0} ({1})", (int)_response.StatusCode, _response.ReasonPhrase);
    }

    private void CreateSecondCustomer()
    {
        // create customer #2 and phone numbers
        _obama = new Customer
```

```
        {
            Name = "Barack Obama",
            Company = "President",
            // set tracking state to 'Add' to generate a SQL Insert statement
            TrackingState = TrackingState.Add,
        };
        _obamaMobilePhone = new Phone
        {
            Number = "212 444-4444",
            PhoneType = "Obama Mobile Phone",
            // set tracking state to 'Add' to generate a SQL Insert statement
            TrackingState = TrackingState.Add,
        };
        // set tracking state to 'Modifed' to generate a SQL Update statement
        _whiteHousePhone.TrackingState = TrackingState.Update;
        _obama.Phones.Add(_obamaMobilePhone);
        _obama.Phones.Add(_whiteHousePhone);
    }

    private async Task AddSecondCustomerAsync()
    {
        // construct call to invoke UpdateCustomer action method in Web API service
        _response = await _client.PostAsync("api/customer/updatecustomer/", _obama, new
JsonMediaTypeFormatter());
        if (_response.IsSuccessStatusCode)
        {
            // capture newly created customer entity from service, which will include
            // database-generated Ids for all entities
            _obama = await _response.Content.ReadAsAsync<Customer>();
            _whiteHousePhone = _bush.Phones.FirstOrDefault(x => x.CustomerId == _obama.CustomerId);
            _bushMobilePhone = _bush.Phones.FirstOrDefault(x => x.CustomerId == _obama.CustomerId);
            Console.WriteLine("Successfully created Customer {0} and {1} Phone Numbers(s)",
            _obama.Name, _obama.Phones.Count);
            foreach (var phoneType in _obama.Phones)
            {
                Console.WriteLine("Added Phone Type: {0}", phoneType.PhoneType);
            }
        }
        else
            Console.WriteLine("{0} ({1})", (int)_response.StatusCode, _response.ReasonPhrase);
    }

    private async Task RemoveFirstCustomerAsync()
    {
        // remove George Bush from underlying data store.
        // first, fetch George Bush entity, demonstrating a call to the
        // get action method on the service while passing a parameter
        var query = "api/customer/" + _bush.CustomerId;
        _response = _client.GetAsync(query).Result;

        if (_response.IsSuccessStatusCode)
        {
            _bush = await _response.Content.ReadAsAsync<Customer>();
            // set tracking state to 'Remove' to generate a SQL Delete statement
            _bush.TrackingState = TrackingState.Remove;
            // must also remove bush's mobile number -- must delete child before removing parent
            foreach (var phoneType in _bush.Phones)
            {
                // set tracking state to 'Remove' to generate a SQL Delete statement
                phoneType.TrackingState = TrackingState.Remove;
            }
            // construct call to remove Bush from underlying database table
            _response = await _client.PostAsync("api/customer/updatecustomer/", _bush, new
JsonMediaTypeFormatter());
            if (_response.IsSuccessStatusCode)
            {
                Console.WriteLine("Removed {0} from database", _bush.Name);
                foreach (var phoneType in _bush.Phones)
                {
                    Console.WriteLine("Remove {0} from data store", phoneType.PhoneType);
                }
            }
            else
                Console.WriteLine("{0} ({1})", (int)_response.StatusCode, _response.ReasonPhrase);
        }
        else
        {
            Console.WriteLine("{0} ({1})", (int)_response.StatusCode, _response.ReasonPhrase);
        }
    }
```

```
    private async Task FetchCustomersAsync()
    {
        // finally, return remaining customers from underlying data store
        _response = await _client.GetAsync("api/customer/");
        if (_response.IsSuccessStatusCode)
        {
            var customers = await _response.Content.ReadAsAsync<IEnumerable<Customer>>();
            foreach (var customer in customers)
            {
                Console.WriteLine("Customer {0} has {1} Phone Numbers(s)",
                customer.Name, customer.Phones.Count());
                foreach (var phoneType in customer.Phones)
                {
                    Console.WriteLine("Phone Type: {0}", phoneType.PhoneType);
                }
            }
        }
        else
        {
            Console.WriteLine("{0} ({1})", (int)_response.StatusCode, _response.ReasonPhrase);
        }
    }
}
```

در آخر کلاس‌های Customer, Phone و BaseEntity را به پروژه کلاینت اضافه کنید. چنین کدهایی بهتر است در لایه مجزایی قرار گیرند و بین لایه‌های مختلف اپلیکیشن به اشتراک گذاشته شوند.

اگر اپلیکیشن کلاینت را اجرا کنید با خروجی زیر مواجه خواهید شد.

```
Successfully created Customer Geroge Bush and 2 Phone Numbers(s)
Added Phone Type: White House Red Phone
Added Phone Type: Bush Mobile Phone
Successfully created Customer Barrack Obama and 2 Phone Numbers(s)
Added Phone Type: Obama Mobile Phone
Added Phone Type: White House Red Phone
Removed Geroge Bush from database
Remove Bush Mobile Phone from data store
Customer Barrack Obama has 2 Phone Numbers(s)
Phone Type: White House Red Phone
Phone Type: Obama Mobile Phone
```

شرح مثال جاری

با اجرای اپلیکیشن Web API شروع کنید. این اپلیکیشن یک MVC Web Controller دارد که پس از اجرا شما را به صفحه خانه هدایت می‌کند. در این مرحله سایت در حال اجرا است و سرویس‌ها قابل دسترسی هستند.

سپس اپلیکیشن کنسول را باز کنید و روی خط اول کد فایل program.cs یک breakpoint قرار داده و آن را اجرا کنید. ابتدا آدرس سرویس را نگاشت می‌کنیم و از سرویس درخواست می‌کنیم که اطلاعات را با فرمت JSON بازگرداند.

سپس توسط متد DeleteAsync که روی آبجکت HttpClient تعریف شده است اکشن متد Cleanup را روی سرویس فراخوانی می‌کنیم. این فراخوانی تمام داده‌های پیشین را حذف می‌کند.

در قدم بعدی یک مشتری بهمراه دو شماره تماس می‌سازیم. توجه کنید که برای هر موجودیت مشخصا خاصیت TrackingState

را مقدار دهی می‌کنیم تا کامپوننت‌های Change-tracking در EF عملیات لازم SQL برای هر موجودیت را تولید کنند.

سپس توسط متد PostAsync که روی آبجکت HttpClient تعریف شده اکشن متد UpdateCustomer را روی سرویس فراخوانی می‌کنیم. اگر به این اکشن متد یک breakpoint اضافه کنید خواهید دید که موجودیت مشتری را بعنوان یک پارامتر دریافت می‌کند و آن را به context جاری اضافه می‌نماید. با اضافه کردن موجودیت به کانتکست جاری کل object graph اضافه می‌شود و EF شروع به ردیابی تغییرات آن می‌کند. دقت کنید که آبجکت موجودیت باید Add شود و نه Attach.

قدم بعدی جالب است، هنگامی که از خاصیت DbChangeTracker استفاده می‌کنیم. این خاصیت روی آبجکت context تعریف شده و یک IEnumerable<DbEntityEntry> را با نام Entries ارائه می‌کند. در اینجا بسادگی نوع پایه EntityType را تنظیم میکنیم. این کار به ما اجازه می‌دهد که در تمام موجودیت هایی که از نوع BaseEntity هستند پیمایش کنیم. اگر بیاد داشته باشید این کلاس، کلاس پایه تمام موجودیت‌ها است. در هر مرحله از پیمایش (iteration) با استفاده از کلاس EntityStateFactory مقدار خاصیت TrackingState را به مقدار متناظر در سیستم ردیابی EF تبدیل می‌کنیم. اگر کلاینت مقدار این فیلد را به Modified تنظیم کرده باشد پردازش بیشتری انجام می‌شود. ابتدا وضعیت موجودیت را از Modified به Unchanged تغییر می‌دهیم. سپس مقادیر اصلی را با فراخوانی متد GetDatabaseValues روی آبجکت Entry از دیتابیس دریافت می‌کنیم. فراخوانی این متد مقادیر موجود در دیتابیس را برای موجودیت جاری دریافت می‌کند. سپس مقادیر بدست آمده را به کلکسیون OriginalValues اختصاص می‌دهیم. پشت پرده، کامپوننت‌های EF Change-tracking بصورت خودکار تفاوت‌های مقادیر اصلی و مقادیر ارسالی را تشخیص می‌دهند و فیلدهای مربوطه را با وضعیت Modified علامت گذاری می‌کنند. فراخوانی‌های بعدی متد SaveChanges تنها فیلدهایی که در سمت کلاینت تغییر کرده اند را بروز رسانی خواهد کرد و نه تمام خواص موجودیت را.

در اپلیکیشن کلاینت عملیات افزودن، بروز رسانی و حذف موجودیت‌ها توسط مقداردهی خاصیت TrackingState را نمایش داده ایم.

متد UpdateCustomer در سرویس ما مقادیر TrackingState را به مقادیر متناظر EF تبدیل می‌کند و آبجکت‌ها را به موتور change-tracking ارسال می‌کند که نهایتا منجر به تولید دستورات لازم SQL می‌شود.

نکته: در اپلیکیشن‌های واقعی بهتر است کد دسترسی داده‌ها و مدل‌های دامنه را به لایه مجزایی منتقل کنید. همچنین پیاده سازی فعلی change-tracking در سمت کلاینت می‌تواند توسعه داده شود تا با انواع جنریک کار کند. در این صورت از نوشتن مقادیر زیادی کد تکراری جلوگیری خواهید کرد و از یک پیاده سازی می‌توانید برای تمام موجودیت‌ها استفاده کنید.

نظرات خوانندگان

نویسنده: امیرحسین
تاریخ: ۱۳۹۲/۱۱/۱۰ ۰:۴

میشه در مورد async کمی توضیح بدین که چرا و به چه دلیلی استفاده شده ؟

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۱۰ ۱:۲۵

الزامی به استفاده از قابلیت‌های async نیست، اما توصیه میشه در مواقعی که امکانش هست و مناسب است از این قابلیت استفاده کنید. لزوما کارایی (performance) بهتری بدست نمیارید ولی مسلما تجربه کاربری بهتری خواهید داشت. عملیاتی که بصورت async اجرا میشن ریسمان جاری (current thread) رو قفل نمی‌کنند، بنابراین اجرای اپلیکیشن ادامه پیدا می‌کنه و پاسخگویی بهتری بدست میارید. برای مطالعه بیشتر به این لینک مراجعه کنید.

مطالعه بیشتر

Using Asynchronous Methods in ASP.NET 4.5

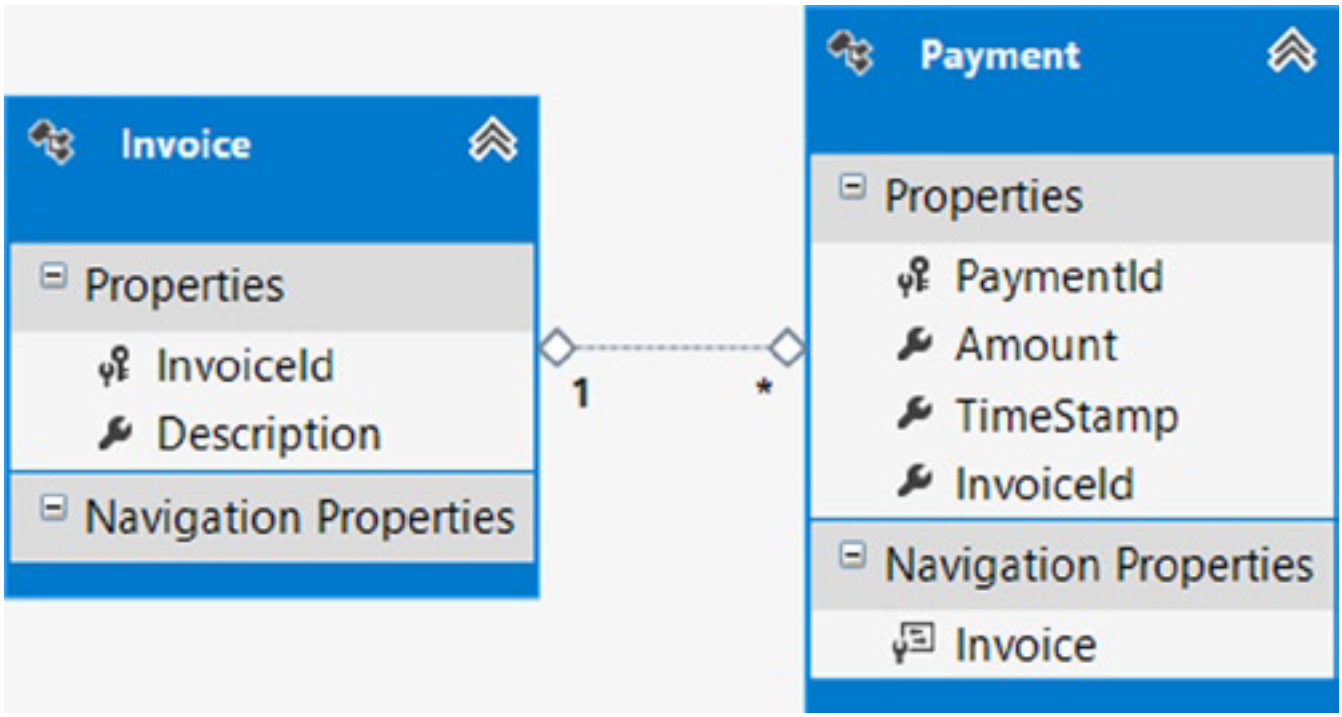
Async and Await

عنوان: استفاده از EF در اپلیکیشن های N-Tier : قسمت پنجم
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۱۰ ۲:۵۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, Tips, N-Layer Architecture, Entity Framework 6

در قسمت قبل پیاده سازی change-tracking در سمت کلاینت توسط Web API را بررسی کردیم. در این قسمت نگاهی به حذف موجودیت‌های منفصل یا disconnected خواهیم داشت.

حذف موجودیت‌های منفصل

فرض کنید موجودیتی را از یک سرویس WCF دریافت کرده اید و می‌خواهید آن را برای حذف علامت گذاری کنید. مدل زیر را در نظر بگیرید.



همانطور که می‌بینید مدل ما صورت حساب‌ها و پرداخت‌های متناظر را ارائه می‌کند. در اپلیکیشن جاری یک سرویس WCF پیاده سازی کرده ایم که عملیات دیتابیسی کلاینت‌ها را مدیریت می‌کند. می‌خواهیم توسط این سرویس آبجکتی را (در اینجا یک موجودیت پرداخت) حذف کنیم. برای ساده نگاه داشتن مثال جاری، مدل‌ها را در خود سرویس تعریف می‌کنیم. برای ایجاد سرویس مذکور مراحل زیر را دنبال کنید.

در ویژوال استودیو پروژه جدیدی از نوع WCF Service Library بسازید و نام آن را به Recipe5 تغییر دهید.

روی پروژه کلیک راست کنید و گزینه Add New Item را انتخاب کنید. سپس گزینه‌های Data -> ADO.NET Entity Data Model را برگزینید.

از ویزارد ویژوال استودیو برای اضافه کردن یک مدل با جداول Invoice و Payment استفاده کنید. برای ساده نگه داشتن مثال جاری، فیلد پیمایشی Payments را از موجودیت Invoice حذف کرده ایم (برای این کار روی خاصیت پیمایشی Payments کلیک راست کنید و گزینه Delete From Model را انتخاب کنید.) روی خاصیت TimeStamp موجودیت Payment کلیک راست کنید و گزینه Properties را انتخاب کنید. سپس مقدار Concurrency Mode آن را به Fixed تغییر دهید. این کار باعث می‌شود که مقدار این فیلد برای کنترل همزمانی بررسی شود. بنابراین مقدار TimeStamp در عبارت WHERE تمام دستورات بروز رسانی و حذف درج خواهد شد.

فایل IService1.cs را باز کنید و تعریف سرویس را مانند لیست زیر تغییر دهید.

```
[ServiceContract]
public interface IService1
{
    [OperationContract]
    Payment InsertPayment();
    [OperationContract]
    void DeletePayment(Payment payment);
}
```

فایل Service1.cs را باز کنید و پیاده سازی سرویس را مانند لیست زیر تغییر دهید.

```
public class Service1 : IService1
{
    public Payment InsertPayment()
    {
        using (var context = new EFRecipesEntities())
        {
            // delete the previous test data
            context.Database.ExecuteSqlCommand("delete from [payments]");
            context.Database.ExecuteSqlCommand("delete from [invoices]");
            var payment = new Payment { Amount = 99.95M, Invoice =
                new Invoice { Description = "Auto Repair" } };
            context.Payments.Add(payment);
            context.SaveChanges();
            return payment;
        }
    }

    public void DeletePayment(Payment payment)
    {
        using (var context = new EFRecipesEntities())
        {
            context.Entry(payment).State = EntityState.Deleted;
            context.SaveChanges();
        }
    }
}
```

برای تست این سرویس به یک کلاینت نیاز داریم. یک پروژه جدید از نوع Console Application به راه حل جاری اضافه کنید و کد آن را مطابق لیست زیر تغییر دهید. فراموش نکنید که ارجاعی به سرویس هم اضافه کنید. روی پروژه کلاینت کلیک راست کرده و Add Service Reference را انتخاب نمایید. ممکن است پیش از آنکه بتوانید سرویس را ارجاع کنید، نیاز باشد پروژه سرویس را ابتدا اجرا کنید (کلیک راست روی پروژه سرویس و انتخاب گزینه Debug -> Start Instance).

```
class Program
{
    static void Main()
    {
        var client = new Service1Client();
        var payment = client.InsertPayment();
        client.DeletePayment(payment);
    }
}
```

اگر روی خط اول متد Main() یک breakpoint قرار دهید می‌توانید مراحل ایجاد و حذف یک موجودیت Payment را دنبال کنید.

شرح مثال جاری

در مثال جاری برای بروز رسانی و حذف موجودیت‌های منفصل از الگویی رایج استفاده کرده ایم که در سرویس‌های WCF و Web API استفاده می‌شود.

در کلاینت با فراخوانی متد InsertPayment یک پرداخت جدید در دیتابیس ذخیره می‌کنیم. این متد، موجودیت Payment ایجاد شده را باز می‌گرداند. موجودیتی که به کلاینت باز می‌گردد از DbContext منفصل (disconnected) است، در واقع در چنین وضعیتی آبجکت context ممکن است در فضای پروسس دیگری قرار داشته باشد، یا حتی روی کامپیوتر دیگری باشد.

برای حذف موجودیت Payment از متد DeletePayment استفاده می‌کنیم. این متد به نوبه خود با فراخوانی متد Entry روی آبجکت context و پاس دادن موجودیت پرداخت بعنوان آرگومان، موجودیت را پیدا می‌کند. سپس وضعیت موجودیت را به EntityState.Deleted تغییر می‌دهیم که این کار آبجکت را برای حذف علامت گذاری می‌کند. فراخوانی‌های بعدی متد SaveChanges() موجودیت را از دیتابیس حذف خواهد کرد.

آبجکت پرداختی که برای حذف به context الحاق کرده ایم تمام خاصیت هایش مقدار دهی شده اند، درست مانند هنگامی که این موجودیت به دیتابیس اضافه شده بود. اما از آنجا که از *foreign key association* استفاده می‌کنیم، تنها فیلدهای کلید موجودیت، خاصیت همزمانی (concurrency) و TimeStamp برای تولید عبارت where مناسب لازم هستند که نهایتا منجر به حذف موجودیت خواهد شد. تنها استثنا درباره این قاعده هنگامی است که موجودیت شما یک یا چند خاصیت از نوع پیچیده یا Complex Type داشته باشد. از آنجا که خاصیت‌های پیچیده، اجزای ساختاری یک موجودیت محسوب می‌شوند نمی‌توانند مقادیر null بپذیرند. یک راه حل ساده این است که هنگامی که EF مشغول ساختن عبارت SQL Delete لازم برای حذف موجودیت بر اساس کلید و خاصیت همزمانی آن است، وهله جدیدی از نوع داده پیچیده خود بسازید. اگر فیلدهای complex type را با مقادیر null رها کنید، فراخوانی متد SaveChanges() با خطا مواجه خواهد شد.

اگر از یک *independent association* استفاده می‌کنید که در آن کثرت (multiplicity) موجودیت مربوطه یک، یا صفر به یک است، EF انتظار دارد که کلیدهای موجودیت‌ها بدرستی مقدار دهی شوند تا بتواند عبارت where مناسب را برای دستورات بروز رسانی و حذف تولید کند. اگر در مثال جاری از یک رابطه independent association بین موجودیت‌های Invoice و Payment استفاده می‌کردیم، لازم بود تا خاصیت پیمایشی Invoice را با وهله ای از صورت حساب مقدار دهی کنیم که خاصیت InvoiceId آن نیز بدرستی مقدار دهی شده باشد. در این صورت عبارت where نهایی شامل فیلدهای PaymentId, TimeStamp و InvoiceId خواهد بود.

نکته: هنگام پیاده سازی معماری‌های n-Tier با Entity Framework، استفاده از رویکرد Foreign Key Association برای موجودیت‌های مرتبط باید با ملاحظات جدی انجام شود. پیاده سازی رویکرد Independent Association مشکل است و می‌تواند کد شما را بسیار پیچیده کند. برای مطالعه بیشتر درباره این رویکردها و مزایا و معایب آنها به این لینک مراجعه کنید که توسط یکی از برنامه نویسان تیم EF نوشته شده است.

اگر موجودیت شما تعداد متعددی Independent Association دارد، مقدار دهی تمام آنها می‌تواند خسته کننده شود. رویکردی ساده‌تر این است که وهله مورد نظر را از دیتابیس دریافت کنید و آن را برای حذف علامت گذاری نمایید. این روش کد شما را ساده‌تر می‌کند، اما هنگامی که آبجکت را از دیتابیس دریافت می‌کنید EF کوئری جاری را بازنویسی می‌کند تا تمام روابط یک، یا صفر به یک بارگذاری شوند. مگر آنکه از گزینه *NoTracking* روی context خود استفاده کنید. اگر در مثال جاری رویکرد Independent Association را پیاده سازی کرده بودیم، هنگامی که موجودیت Payment را از دیتابیس دریافت می‌کنیم (قبل از علامت گذاری برای حذف) EF یک Object state entry برای موجودیت پرداخت و یک Relationship entry برای رابطه بین Payment و Invoice می‌ساخت. سپس وقتی که موجودیت پرداخت را برای حذف علامت گذاری می‌کنیم، EF رابطه بین پرداخت و صورت حساب را هم برای حذف علامت گذاری می‌کند. در اینجا عبارت where تولید شده مانند قبل، شامل فیلدهای PaymentId, TimeStamp و InvoiceId خواهد بود.

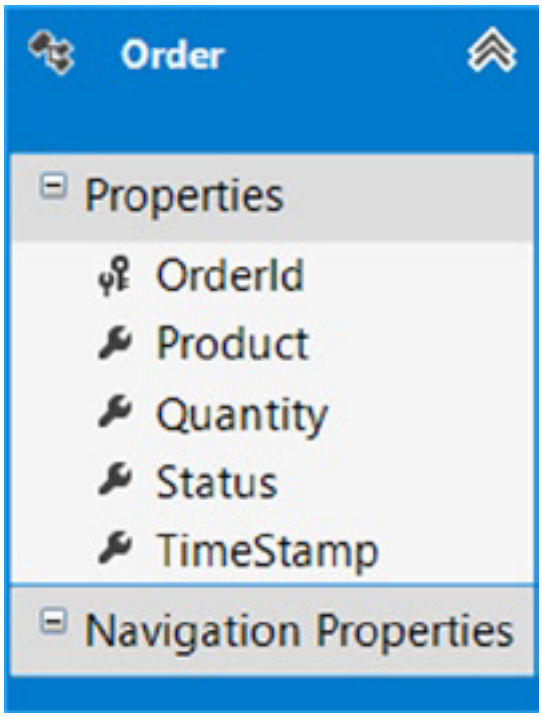
یک گزینه دیگر برای حذف موجودیت‌ها در Independent Associations این است که تمام موجودیت‌های مرتبط را مشخصا بارگذاری کنیم (eager loading) و کل Object graph را برای حذف به سرویس WCF یا Web API بفرستیم. در مثال جاری می‌توانستیم موجودیت صورتحساب مرتبط با موجودیت پرداخت را مشخصا بارگذاری کنیم. اگر می‌خواستیم موجودیت Payment را حذف کنیم، می‌توانستیم کل گراف را که شامل هر دو موجودیت می‌شود به سرویس ارسال کنیم. اما هنگام استفاده از چنین روشی باید بسیار دقت کنید، چرا که این رویکرد پهنای باند بیشتری مصرف می‌کند و زمان پردازش بیشتری هم برای مرتب سازی (serialization) صرف می‌کند. بنابراین هزینه این رویکرد نسبت به سادگی کدی که بدست می‌آید به مراتب بیشتر است.

عنوان: **استفاده از EF در اپلیکیشن های N-Tier : قسمت ششم**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۱۱ ۰:۲۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, Tips, WCF, N-Layer Architecture, Web API, Entity Framework 6

در قسمت قبل رویکردهای مختلف برای حذف موجودیت‌های منفصل را بررسی کردیم. در این قسمت مدیریت همزمانی یا Concurrency را بررسی خواهیم کرد.

فرض کنید می‌خواهیم مطمئن شویم که موجودیتی که توسط یک کلاینت WCF تغییر کرده است، تنها در صورتی بروز رسانی شود که شناسه (token) همزمانی آن تغییر نکرده باشد. به بیان دیگر شناسه ای که هنگام دریافت موجودیت بدست می‌آید، هنگام بروز رسانی باید مقداری یکسان داشته باشد.

مدل زیر را در نظر بگیرید.



می‌خواهیم یک سفارش (order) را توسط یک سرویس WCF بروز رسانی کنیم در حالی که اطمینان حاصل می‌کنیم موجودیت سفارش از زمانی که دریافت شده تغییری نکرده است. برای مدیریت این وضعیت دو رویکرد تقریبا متفاوت را بررسی می‌کنیم. در هر دو رویکرد از یک ستون همزمانی استفاده می‌کنیم، در این مثال فیلد TimeStamp.

در ویژوال استودیو پروژه جدیدی از نوع WCF Service Library بسازید و نام آن را به Recipe6 تغییر دهید.
روی نام پروژه کلیک راست کنید و گزینه Add New Item را انتخاب کنید. سپس گزینه‌های Data -> Entity Data Model را برگزینید. از ویزارد ویژوال استودیو برای اضافه کردن مدل جاری و جدول Orders استفاده کنید. در EF Designer روی فیلد TimeStamp کلیک راست کنید و گزینه Properties را انتخاب کنید. سپس مقدار CuncurrencyMode آنرا به Fixed تغییر دهید.
فایل IService1.cs را باز کنید و تعریف سرویس را مطابق لیست زیر بروز رسانی کنید.

```
[ServiceContract]
public interface IService1
{
    [OperationContract]
```

```
    Order InsertOrder();
    [OperationContract]
    void UpdateOrderWithoutRetrieving(Order order);
    [OperationContract]
    void UpdateOrderByRetrieving(Order order);
}
```

فایل Service1.cs را باز کنید و پیاده سازی سرویس را مطابق لیست زیر تکمیل کنید.

```
public class Service1 : IService1
{
    public Order InsertOrder()
    {
        using (var context = new EFRecipesEntities())
        {
            // remove previous test data
            context.Database.ExecuteSqlCommand("delete from [orders]");
            var order = new Order
            {
                Product = "Camping Tent",
                Quantity = 3,
                Status = "Received"
            };
            context.Orders.Add(order);
            context.SaveChanges();
            return order;
        }
    }

    public void UpdateOrderWithoutRetrieving(Order order)
    {
        using (var context = new EFRecipesEntities())
        {
            try
            {
                context.Orders.Attach(order);
                if (order.Status == "Received")
                {
                    context.Entry(order).Property(x => x.Quantity).IsModified = true;
                    context.SaveChanges();
                }
            }
            catch (OptimisticConcurrencyException ex)
            {
                // Handle OptimisticConcurrencyException
            }
        }
    }

    public void UpdateOrderByRetrieving(Order order)
    {
        using (var context = new EFRecipesEntities())
        {
            // fetch current entity from database
            var dbOrder = context.Orders
            .Single(o => o.OrderId == order.OrderId);
            if (dbOrder != null &&
                // execute concurrency check
                StructuralComparisons.StructuralEqualityComparer.Equals(order.TimeStamp,
dbOrder.TimeStamp))
            {
                dbOrder.Quantity = order.Quantity;
                context.SaveChanges();
            }
            else
            {
                // Add code to handle concurrency issue
            }
        }
    }
}
```

برای تست این سرویس به یک کلاینت نیاز داریم. پروژه جدیدی از نوع Console Application به راه حل جاری اضافه کنید و کد آن را مطابق لیست زیر تکمیل کنید. با کلیک راست روی نام پروژه و انتخاب گزینه Add Service Reference سرویس پروژه را هم ارجاع کنید. دقت کنید که ممکن است پیش از آنکه بتوانید سرویس را ارجاع کنید نیاز باشد روی آن کلیک راست کرده و از منوی

Debug گزینه Start Instance را انتخاب کنید تا وهله از سرویس به اجرا در بیاید.

```
class Program
{
    static void Main(string[] args)
    {
        var service = new Service1Client();
        var order = service.InsertOrder();
        order.Quantity = 5;
        service.UpdateOrderWithoutRetrieving(order);
        order = service.InsertOrder();
        order.Quantity = 3;
        service.UpdateOrderByRetrieving(order);
    }
}
```

اگر به خط اول متد Main() یک breakpoint اضافه کنید و اپلیکیشن را اجرا کنید می‌توانید افزودن و بروز رسانی یک Order با هر دو رویکرد را بررسی کنید.

شرح مثال جاری

متد InsertOrder() داده‌های پیشین را حذف می‌کند، سفارش جدیدی می‌سازد و آن را در دیتابیس ثبت می‌کند. در آخر موجودیت جدید به کلاینت باز می‌گردد. موجودیت بازگشتی هر دو مقدار OrderId و TimeStamp را دارا است که توسط دیتابیس تولید شده اند. سپس در کلاینت از دو رویکرد نسبتا متفاوت برای بروز رسانی موجودیت استفاده می‌کنیم.

در رویکرد نخست، متد UpdateOrderWithoutRetrieving() موجودیت دریافت شده از کلاینت را Attach می‌کند و چک می‌کند که مقدار فیلد Status چیست. اگر مقدار این فیلد "Received" باشد، فیلد Quantity را با EntityState.Modified علامت گذاری می‌کنیم و متد SaveChanges() را فراخوانی می‌کنیم. EF دستورات لازم برای بروز رسانی را تولید می‌کند، که فیلد quantity را مقدار دهی کرده و یک عبارت where هم دارد که فیلدهای OrderId و TimeStamp را چک می‌کند. اگر مقدار TimeStamp توسط یک دستور بروز رسانی تغییر کرده باشد، بروز رسانی جاری با خطا مواجه خواهد شد. برای مدیریت این خطا ما بدنه کد را در یک بلاک try/catch قرار می‌دهیم، و استثنای OptimisticConcurrencyException را مهار می‌کنیم. این کار باعث می‌شود اطمینان داشته باشیم که موجودیت Order دریافت شده از متد InsertOrder() تاکنون تغییری نکرده است. دقت کنید که در مثال جاری تمام خواص موجودیت بروز رسانی می‌شوند، صرفنظر از اینکه تغییر کرده باشند یا خیر.

رویکرد دوم نشان می‌دهد که چگونه می‌توان وضعیت همزمانی موجودیت را پیش از بروز رسانی مشخصا دریافت و بررسی کرد. در اینجا می‌توانید مقدار TimeStamp موجودیت را از دیتابیس بگیرید و آن را با مقدار موجودیت کلاینت مقایسه کنید تا وجود تغییرات مشخص شود. این رویکرد در متد UpdateOrderByRetrieving() نمایش داده شده است. گرچه این رویکرد برای تشخیص تغییرات خواص موجودیت‌ها و یا روابط شان مفید و کارآمد است، اما بهترین روش هم نیست. مثلا ممکن است از زمانی که موجودیت را از دیتابیس دریافت می‌کنید، تا زمانی که مقدار TimeStamp آن را مقایسه می‌کنید و نهایتا متد SaveChanges() را صدا میزنید، موجودیت شما توسط کلاینت دیگری بروز رسانی شده باشد.

مسلما رویکرد دوم هزینه برتر از رویکرد اولی است، چرا که برای مقایسه مقادیر همزمانی موجودیت ها، یکبار موجودیت را از دیتابیس دریافت می‌کنید. اما این رویکرد در مواقعی که Object graphهای بزرگ یا پیچیده (complex) دارید بهتر است، چون پیش از ارسال موجودیت‌ها به context در صورت برابر نبودن مقادیر همزمانی پروسس را لغو می‌کنید.

نظرات خوانندگان

نویسنده: Senator
تاریخ: ۱۳۹۲/۱۱/۱۶ ۱۸:۵۴

خیلی ممنون.
عالی بود ...

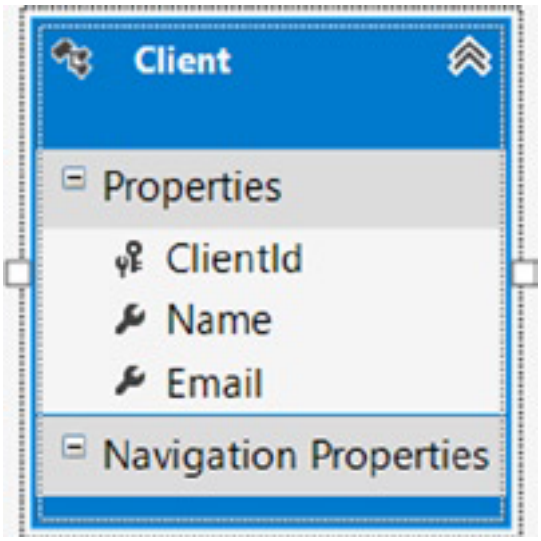
عنوان: **استفاده از EF در اپلیکیشن های N-Tier : قسمت هفتم**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۱۱ ۱:۱۰
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, Tips, WCF, N-Layer Architecture, Entity Framework 6

در قسمت قبلی مدیریت همزمانی در بروز رسانی‌ها را بررسی کردیم. در این قسمت مرتب سازی (serialization) پراکسی‌ها در سرویس‌های WCF را بررسی خواهیم کرد.

مرتب سازی پراکسی‌ها در سرویس‌های WCF

فرض کنید یک پراکسی دینامیک (dynamic proxy) از یک کوئری دریافت کرده اید. حال می‌خواهید این پراکسی را در قالب یک آبجکت CLR سریال کنید. هنگامی که آبجکت‌های موجودیت را بصورت POCO-based پیاده سازی می‌کنید، EF بصورت خودکار یک آبجکت دینامیک مشتق شده را در زمان اجرا تولید می‌کند که *dynamix proxy* نام دارد. این آبجکت برای هر موجودیت POCO تولید می‌شود. این آبجکت پراکسی بسیاری از خواص مجازی (virtual) را بازنویسی می‌کند تا عملیاتی مانند ردیابی تغییرات و بارگذاری تنبل را انجام دهد.

فرض کنید مدلی مانند تصویر زیر دارید.



ما از کلاس *ProxyDataContractResolver* برای deserialize کردن یک آبجکت پراکسی در سمت سرور و تبدیل آن به یک آبجکت POCO روی کلاینت WCF استفاده می‌کنیم. مراحل زیر را دنبال کنید.

در ویژوال استودیو پروژه جدیدی از نوع WCF Service Application بسازید. یک Entity Data Model به پروژه اضافه کنید و جدول Clients را مطابق مدل مذکور ایجاد کنید.
کلاس POCO تولید شده توسط EF را باز کنید و کلمه کلیدی virtual را به تمام خواص اضافه کنید. این کار باعث می‌شود EF کلاس‌های پراکسی دینامیک تولید کند. کد کامل این کلاس در لیست زیر قابل مشاهده است.

```
public class Client
{
    public virtual int ClientId { get; set; }
    public virtual string Name { get; set; }
    public virtual string Email { get; set; }
}
```

نکته: بیاد داشته باشید که هرگاه مدل EDMX را تغییر می‌دهید، EF بصورت خودکار کلاس‌های موجودیت‌ها را مجددا ساخته و تغییرات مرحله قبلی را بازنویسی می‌کند. بنابراین یا باید این مراحل را هر بار تکرار کنید، یا قالب T4 مربوطه را ویرایش کنید. اگر هم از مدل Code-First استفاده می‌کنید که نیازی به این کارها نخواهد بود.

ما نیاز داریم که DataContractSerializer از یک کلاس ProxyDataContractResolver استفاده کند تا پراکسی کلاینت را به موجودیت کلاینت برای کلاینت سرویس WCF تبدیل کند. یعنی client proxy به client entity تبدیل شود، که نهایتا در اپلیکیشن کلاینت سرویس استفاده خواهد شد. بدین منظور یک ویژگی operation behavior می‌سازیم و آن را به متد ()GetClient در سرویس الحاق می‌کنیم. برای ساختن ویژگی جدید از کدی که در لیست زیر آمده استفاده کنید. بیاد داشته باشید که کلاس ProxyDataContractResolver در فضای نام Entity Framework قرار دارد.

```
using System.ServiceModel.Description;
using System.ServiceModel.Channels;
using System.ServiceModel.Dispatcher;
using System.Data.Objects;

namespace Recipe8
{
    public class ApplyProxyDataContractResolverAttribute :
        Attribute, IOperationBehavior
    {
        public void AddBindingParameters(OperationDescription description,
            BindingParameterCollection parameters)
        {
        }
        public void ApplyClientBehavior(OperationDescription description,
            ClientOperation proxy)
        {
            DataContractSerializerOperationBehavior
                dataContractSerializerOperationBehavior =
                    description.Behaviors
                    .Find<DataContractSerializerOperationBehavior>();
                dataContractSerializerOperationBehavior.DataContractResolver =
                    new ProxyDataContractResolver();
        }
        public void ApplyDispatchBehavior(OperationDescription description,
            DispatchOperation dispatch)
        {
            DataContractSerializerOperationBehavior
                dataContractSerializerOperationBehavior =
                    description.Behaviors
                    .Find<DataContractSerializerOperationBehavior>();
            dataContractSerializerOperationBehavior.DataContractResolver =
                new ProxyDataContractResolver();
        }
        public void Validate(OperationDescription description)
        {
        }
    }
}
```

فایل IService1.cs را باز کنید و کد آن را مطابق لیست زیر تکمیل نمایید.

```
[ServiceContract]
public interface IService1
{
    [OperationContract]
    void InsertTestRecord();
    [OperationContract]
    Client GetClient();
    [OperationContract]
    void Update(Client client);
}
```

فایل IService1.svc.cs را باز کنید و پیاده سازی سرویس را مطابق لیست زیر تکمیل کنید.

```
public class Client
{
    [ApplyProxyDataContractResolver]
    public Client GetClient()
```

```
    {
        using (var context = new EFRecipesEntities())
        {
            context.Cofiguration.LazyLoadingEnabled = false;
            return context.Clients.Single();
        }
    }
    public void Update(Client client)
    {
        using (var context = new EFRecipesEntities())
        {
            context.Entry(client).State = EntityState.Modified;
            context.SaveChanges();
        }
    }
    public void InsertTestRecord()
    {
        using (var context = new EFRecipesEntities())
        {
            // delete previous test data
            context.ExecuteSqlCommand("delete from [clients]");
            // insert new test data
            context.ExecuteStoreCommand(@"insert into
                [clients](Name, Email) values ('Armin Zia','armin.zia@gmail.com')");
        }
    }
}
```

حال پروژه جدیدی از نوع Console Application به راه حل جاری اضافه کنید. این اپلیکیشن، کلاینت تست ما خواهد بود. پروژه سرویس را ارجاع کنید و کد کلاینت را مطابق لیست زیر تکمیل نمایید.

```
using Recipe8Client.ServiceReference1;

namespace Recipe8Client
{
    class Program
    {
        static void Main(string[] args)
        {
            using (var serviceClient = new Service1Client())
            {
                serviceClient.InsertTestRecord();

                var client = serviceClient.GetClient();
                Console.WriteLine("Client is: {0} at {1}", client.Name, client.Email);

                client.Name = "Armin Zia";
                client.Email = "arminzia@live.com";
                serviceClient.Update(client);

                client = serviceClient.GetClient();
                Console.WriteLine("Client changed to: {0} at {1}", client.Name, client.Email);
            }
        }
    }
}
```

اگر اپلیکیشن کلاینت را اجرا کنید با خروجی زیر مواجه خواهید شد.

```
Client is: Armin Zia at armin.zia@gmail.com
Client changed to: Armin Zia at arminzia@live.com
```

شرح مثال جاری

مایکروسافت استفاده از آبجکت‌های POCO با WCF را توصیه می‌کند چرا که مرتب سازی (serialization) آبجکت موجودیت را ساده‌تر می‌کند. اما در صورتی که از آبجکت‌های POCO ای استفاده می‌کنید که *changed-based notification* دارند (یعنی خواص موجودیت را با virtual علامت گذاری کرده اید و کلکسیون‌های خواص پیمایشی از نوع ICollection هستند)، آنگاه EF برای موجودیت‌های بازگشتی کوئری‌ها *پراکسی‌های دینامیک* تولید خواهد کرد.

استفاده از پراکسی‌های دینامیک با WCF دو مشکل دارد. مشکل اول مربوط به سریال کردن پراکسی است. کلاس DataContractSerializer تنها قادر به serialize/deserialize کردن انواع شناخته شده (known types) است. در مثال جاری این یعنی موجودیت Client. اما از آنجا که EF برای موجودیت‌ها پراکسی می‌سازد، حالا باید آبجکت پراکسی را سریال کنیم، نه خود کلاس Client را. اینجا است که از DataContractResolver استفاده می‌کنیم. این کلاس می‌تواند حین سریال کردن آبجکت ها، نوعی را به نوع دیگر تبدیل کند. برای استفاده از این کلاس ما یک ویژگی سفارشی ساختیم، که پراکسی‌ها را به کلاس‌های POCO تبدیل می‌کند. سپس این ویژگی را به متد GetClient() اضافه کردیم. این کار باعث می‌شود که پراکسی دینامیکی که توسط متد GetClient() برای موجودیت Client تولید می‌شود، به درستی سریال شود.

مشکل دوم استفاده از پراکسی‌ها با WCF مربوط به بارگذاری تبل یا Lazy Loading می‌شود. هنگامی که DataContractSerializer موجودیت‌ها را سریال می‌کند، تمام خواص موجودیت را دستیابی خواهد کرد که منجر به اجرای lazy-loading روی خواص پیمایشی می‌شود. مسلما این رفتار را نمی‌خواهیم. برای رفع این مشکل، مشخصا قابلیت بارگذاری تنبل را خاموش یا غیرفعال کرده ایم.

عنوان: **خلاص شدن از شر deep null check**
نویسنده: رفیعی
تاریخ: ۱۳۹۳/۰۱/۳۰ ۱۲:۴۵
آدرس: www.dotnettips.info
گروه‌ها: C#, Tips

آیا تا به حال مجبور به نوشتن کدی شبیه قطعه کد زیر شده اید؟

```
var store = GetStore();
string postCode = null;
if (store != null && store.Address != null && store.Address.PostCode != null)
     postCode = store.Address.PostCode.ToString();
```

بله! من مطمئن هستم برای شما هم پیش آمده است.

هدف بازیابی و یا محاسبه یک مقدار است، اما برای انجام این کار می‌بایست به چندین شیء میانی دسترسی پیدا کنیم که البته ممکن است در حالت پیش فرض خود قرار داشته باشند و حاوی هیچ مقداری نباشند. بنابراین برای جلوگیری از وقوع NullException ، مجبوریم تمامی اشیائی که در مسیر قرار دارند را بررسی کنیم که null نباشند. مثال بالا کاملا گویا ست. گاهی اوقات حتی ممکن است فراخوانی یک متد، تبدیل نوع با استفاده از as و یا دسترسی به عناصر یک مجموعه وجود داشته باشد. متاسفانه مدیریت تمامی این حالات باعث حجیم شدن کدها و در نتیجه کاهش خوانایی آنها می‌شود. بنابراین باید به دنبال یک راه حل مناسب بود.

استفاده از یک متد الحاقی شرطی (Conditional extensions)

از نظر بسیاری از برنامه نویس‌ها راه حل، استفاده از یک متد الحاقی شرطی است. اگر عبارت "c# deep null check" را گوگل کنید، پیاده سازی‌های متنوعی را پیدا خواهید کرد. اگر چه این متدها نام‌های متفاوتی دارند اما همه آن‌ها از یک ایده کلی مشترک استفاده می‌کنند:

```
public static TResult IfNotNull<TResult, TSource>(
    this TSource source,
    Func<TSource, TResult> onNotDefault)
    where TSource : class
{
    if (onNotDefault == null) throw new ArgumentNullException("onNotDefault");
    return source == null ? default(TResult) : onNotDefault(source);
}
```

همانطور که می‌بینید این متد الحاقی مقداری از نوع TResult را بر می‌گرداند. اگر source که در اینجا با توجه به الحاقی بودن متد به معنای شی جاری است، null باشد مقدار پیش فرض نوع خروجی(TResult) بازگردانده می‌شود و در غیر این صورت دیلیگیت onNotDefault فراخوانی می‌گردد.

بعد از افزودن متد الحاقی IfNotNull به پروژه می‌توانیم مثال ابتدای مطلب را به صورت زیر بنویسیم :

```
var postCode =
    GetStore()
        .IfNotNull(x => x.Address)
        .IfNotNull(x => x.PostCode)
        .IfNotNull(x => x.ToString());
```

این روش مزایای بسیاری دارد اما در موارد پیچیده دچار مشکل می‌شویم. برای مثال در نظر بگیرید قصد داریم در طول مسیر، متدی را فراخوانی کنیم و مقدار بازگشتی را در یک متغیر موقتی ذخیره کنیم و بر اساس آن ادامه مسیر را طی کنیم. متاسفانه این کارها هم اکنون امکان پذیر نیست. پس به نظر می‌رسد باید کمی متد الحاقی IfNotNull را بهبود ببخشیم.

برای بهبود عملکرد متد الحاقی IfNotNull علاوه بر موارد ذکر شده حداقل دو مورد به نظر من می‌رسد:
این متد فقط با انواع ارجاعی (reference types) کار می‌کند و می‌بایست برای کار با انواع مقداری (value types) اصلاح شود.

با انواع داده ای مثل string چه باید کرد؟ در مورد این نوع داده‌ها تنها مطمئن شدن از null نبودن کافی نیست. برای مثال در مورد string ، گاهی اوقات ما می‌خواهیم از خالی نبودن آن نیز مطمئن شویم. و یا در مورد collectionها تنها null نبودن کافی نیست بلکه زمانی که نیاز به محاسبه مجموع و یا یافتن بزرگترین عضو است، باید از خالی نبودن مجموعه و وجود حداقل یک عضو در آن مطمئن باشیم.

برای حل این مشکلات می‌توانیم متد الحاقی IfNotNull را به متد الحاقی IfNotDefault تبدیل کنیم:

```
public static TResult IfNotDefault<TResult, TSource>(
    this TSource source,
    Func<TSource, TResult> onNotDefault,
    Predicate<TSource> isNotDefault = null)
{
    if (onNotDefault == null) throw new ArgumentNullException("onNotDefault");
    var isDefault = isNotDefault == null
        ? EqualityComparer<TSource>.Default.Equals(source, default(TSource))
        : !isNotDefault(source);
   return isDefault ? default(TResult) : onNotDefault(source);
}
```

تعریف این متد خیلی با تعریف متد قبلی متفاوت نیست. به منظور پشتیبانی از struct ها، قید where TSource : class حذف شده است. بنابراین دیگر نمی‌توان از مقایسه‌ی ساده null با استفاده از عملگر == استفاده کرد چراکه structها nullable نیستند. پس مجبوریم از EqualityComparer<TSource>.Default بخواهیم که این کار را انجام دهد. متد الحاقی IfNotDefault همچنین شامل یک predicate اختیاری با نام isNotDefault است. در صورتی که مقایسه پیش فرض کافی نبود می‌توان از این predicate استفاده کرد.
در انتها اجازه بدهید چند مثال کاربردی را مرور کنیم:
1- انجام یک سری اعمال بر روی string در صورتی که رشته خالی نباشد:

```
return person
        . IfNotDefault(x => x.Name)
        . IfNotDefault(SomeOperation, x => !string.IsNullOrEmpty(x));
```

محاسبه‌ی مقدار میانگین. متد الحاقی IfNotDefault به زیبایی در یک زنجیره‌ی LINQ کار می‌کند:

```
var avg = students
        .Where(IsNotAGraduate)
        .FirstOrDefault()
        .IfNotDefault(s => s.Grades)
        .IfNotDefault(g => g.Average(), g => g != null && g.Length > 0);
```

برای مطالعه بیشتر

Get rid of deep null checks

Chained null checks and the Maybe monad

Maybe or IfNotNull using lambdas for deep expressions

Dynamically Check Nested Values for IsNull Values

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۳۰ ۱۳:۱۱

با تشکر از شما. در سی‌شارپ 6 ، برای این مورد ویژه قرار است عملگر جدیدی اضافه شود:

Null propagation	customer?.Orders?[5]?.$price	Planned	Planned

بحثی در این مورد

نویسنده: رفیعی
تاریخ: ۱۳۹۳/۰۱/۳۰ ۱۴:۷

بله همین طوره!ممنون از دقت نظرتون. اتفاقاً پیشنهاد این کار رو هم فردی به نام جمشید اسدزاده که احتمالا ایرانیه در قسمت پیشنهادهای مایکروسافت مطرح کرده.

نویسنده: حمید سامانی
تاریخ: ۱۳۹۳/۰۱/۳۰ ۱۷:۱۱

احتمالا ایشون SafeNavigationOperator توی زبان Groovy را قبلا دیده بودن :) .

نویسنده: یاسر مرادی
تاریخ: ۱۳۹۳/۰۱/۳۰ ۱۸:۲۶

متاسفانه روش فوق کد نویسی را تا حد زیادی تحت تاثیر قرار می‌دهد، مگر این که روش استفاده از متد الحاقی شما را به خوبی متوجه نشده باشم
به مثال زیر دقت کنید:

```
public class Customer
    {
        public CustomerInfo Info { get; set; }
        public Int32 GetNameLength()
        {
            return this.IfNotDefault(city => city.Info)
                .IfNotDefault(info => info.CityInfo)
                .IfNotDefault(cityInfo => cityInfo.Name)
                .IfNotDefault(name => name.Length);
        }
    }
    public class CustomerInfo
    {
        public CustomerCityInfo CityInfo { get; set; }
    }
    public class CustomerCityInfo
    {
        public String Name { get; set; }
    }
```

و برای استفاده داریم:

```
Customer customer = new Customer();
            String cityName = customer
                .IfNotDefault(cust => cust.Info)
                .IfNotDefault(info => info.CityInfo)
```

```
                .IfNotDefault(city => city.Name);
            Int32 length = customer.GetNameLength();
```

در حالی که با متد الحاقی زیر داریم

```
public static TValue GetValue<TObj, TValue>(this TObj obj, Func<TObj, TValue> member, TValue
defaultValueOnNull = default(TValue))
        {
            if (member == null)
                throw new ArgumentNullException("member");

            if (obj == null)
                throw new ArgumentNullException("obj");

            try
            {
                return member(obj);
            }
            catch (NullReferenceException)
            {
                return defaultValueOnNull;
            }
        }
```

تعریف ساده‌تر کلاس

```
public class Customer
    {
        public CustomerInfo Info { get; set; }

        public Int32 GetNameLength()
        {
            return this.Info.CityInfo.Name.Length;
        }
    }

    public class CustomerInfo
    {
        public CustomerCityInfo CityInfo { get; set; }
    }

    public class CustomerCityInfo
    {
        public String Name { get; set; }
    }
```

و سادگی در استفاده

```
Customer customer = new Customer();

            String cityName = customer.GetValue(cust => cust.Info.CityInfo.Name, "Not Selected");

            Int32 i = customer.GetValue(cust => cust.GetNameLength());
```

شاید بگویید استفاده از Try-Catch سیستم را کند می‌کند، البته نه در آن حدی که فکر می‌کنید، و اگر قسمتی از کد شما به تعداد زیادی در بازه‌ی زمانی کوتاه فراخوانی می‌شود، می‌توانید آنرا به صورت کاملا عادی بنویسید، چون واقعا تعداد این شرایط زیاد نیست و این مورد سناریوی فراگیری نیست، در عوض خوانایی کد بسیار بسیار بالاتر از حالات عادی است.
در ضمن دقت کنید که تا زمانی که خطای NullReference رخ ندهد، سرعت سیستم در حد همان حداقل نیز کاهش نمی‌یابد، بدین جهت که بسیاری از افراد فکر می‌کنند Try-Catch نوشتن به خودی خود برنامه را کند می‌کند، ولی این رخ دادن خطا و جمع آوری StackTrace و ... است که برنامه را کند می‌کند، که شاید در خیلی از موارد اصلا رخ ندهد.
البته کدهای نوشته صرفا نمونه کد است، به هیچ وجه اصول طراحی در آن رعایت نشده است، بلکه سعی کرده ام مثال واضح‌تری بزنم
موفق و پایدار باشید

نویسنده: رضا عرب
تاریخ: ۱۳۹۳/۰۱/۳۱ ۹:۲۶

با تشکر از شما من یک Extension Method نوشتم که به نظرم کار کردن باهاش راحتتره. GetValueOrDefault

نویسنده: رفیعی
تاریخ: ۱۳۹۳/۰۱/۳۱ ۱۰:۰

بدیهی است راه‌های زیادی برای این کار وجود دارد اگرچه هسته همه اون‌ها خیلی شبیه...
متد الحاقی IfNotDefault چند ویژگی مهم دارد :

همانطور که در متن ذکر شده، بحث فقط چک برای null نبودن نیست بلکه چک برای قرار نداشتن در حالت پیش فرضه! که در انواعی مثل string و collectionها خیلی مهمه.
گاهی اوقات هر کدام از اشیاء در طول زنجیره برای ما مهم هستند. متد الحاقی IfNotDefault این امکان را دارد که هر کدام از اشیاء جداگانه بررسی شوند. روش ارایه شده در C# 6.0 هم همینگونه است.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۲ ۰:۳۲

دو روش دیگر برای حل این مساله

- استفاده از روش‌های AOP مانند Minimizing the null ref with dynamic proxies
- استفاده از expression trees مانند Avoiding nulls with expression trees