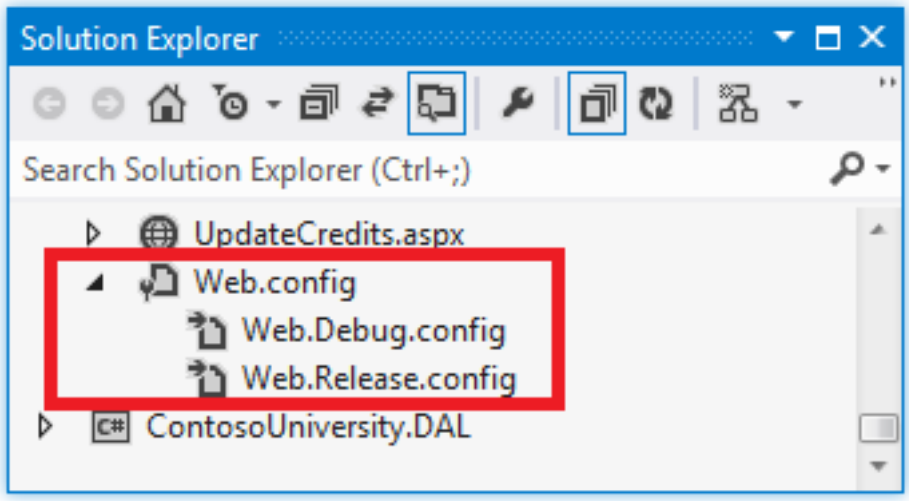
عنوان: **Web.config File Transformation #1**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۲/۱۷ ۱۳:۳۵
آدرس: www.dotnettips.info
گروه‌ها: Software deployment, Web.config Transformation

یکی از مشکلات برنامه نویسان اختلاف بین فایل web.config تولید شده در سیستم خودشان و مقصد نهایی برنامه می‌باشد. در این مطلب به نحوه خودکار سازی تغییرات، برای توسعه بر روی مقصد نهایی برنامه می‌پردازیم. اکثر برنامه‌ها تنظیماتی در فایل web.config خود دارند که زمان عرضه برای مقصد نهایی می‌بایست تغییر کنند. پردازش خودکار این تغییرات کمک می‌کند تا از خطاهای ناشی از تغییرات دستی در زمان عرضه نهایی جلوگیری شود.

فایل‌های پیش فرض انتقالی

در پنجره Solution Explorer فایل web.config را بوسیله ایکون کنار آن باز کنید تا دو فایل پیش فرض web.Debug.config و web.Release.config که برای build configurationهای برنامه ایجاد شده‌اند، مشاهده نمایید.



شما می‌توانید برای ایجاد build configuration دیگری، با راست کلیک کردن بر روی فایل web.config و انتخاب گزینه Add Config Transforms یک فایل دیگر را ایجاد نمایید. البته اگر این گزینه غیر فعال است، ابتدا می‌بایست یک build configuration جدید را ایجاد نمایید.

غیرفعال کردن حالت اشکال زدایی (debug)

در زمان عرضه نهایی برنامه دیگر لازم نیست تا امکان اشکال زدایی فعال باشد؛ به همین خاطر در فایل web.Release.config در تگ compilation مانند کد زیر از دستور RemoveAttributes استفاده می‌کنیم تا آن خاصیت را حذف نماید.

```
<compilation xdt:Transform="RemoveAttributes(debug)" />
```

محدود کردن صفحات خطای برنامه فقط برای برنامه نویس

یکی دیگر از تنظیمات، عدم نمایش خطاهای برنامه به کاربر نهایی و انتقال آن به یک صفحه رخداد خطا می‌باشد:

```
<customErrors mode="Off" defaultRedirect="~/GenericErrorPage.aspx">
```

```
  <error statusCode="404" redirect="~/GenericErrorPage.aspx" />
</customErrors>
```

اما در زمان کد نویسی در محیط ویژوال استادیو می‌خواهیم mode را بر روی Off تنظیم نماییم تا خطا نمایش داده شود:

```
<customErrors mode="RemoteOnly" xdt:Transform="Replace" defaultRedirect="~/GenericErrorPage.aspx">
  <error statusCode="404" redirect="~/GenericErrorPage.aspx" />
</customErrors>
```

برای این کار از دستور Replace استفاده می‌کنیم تا تگ customErrors با مد RemoteOnly را جایگزین مد Off در زمان Release نماید.

تنظیم رشته اتصال پایگاه داده اصلی

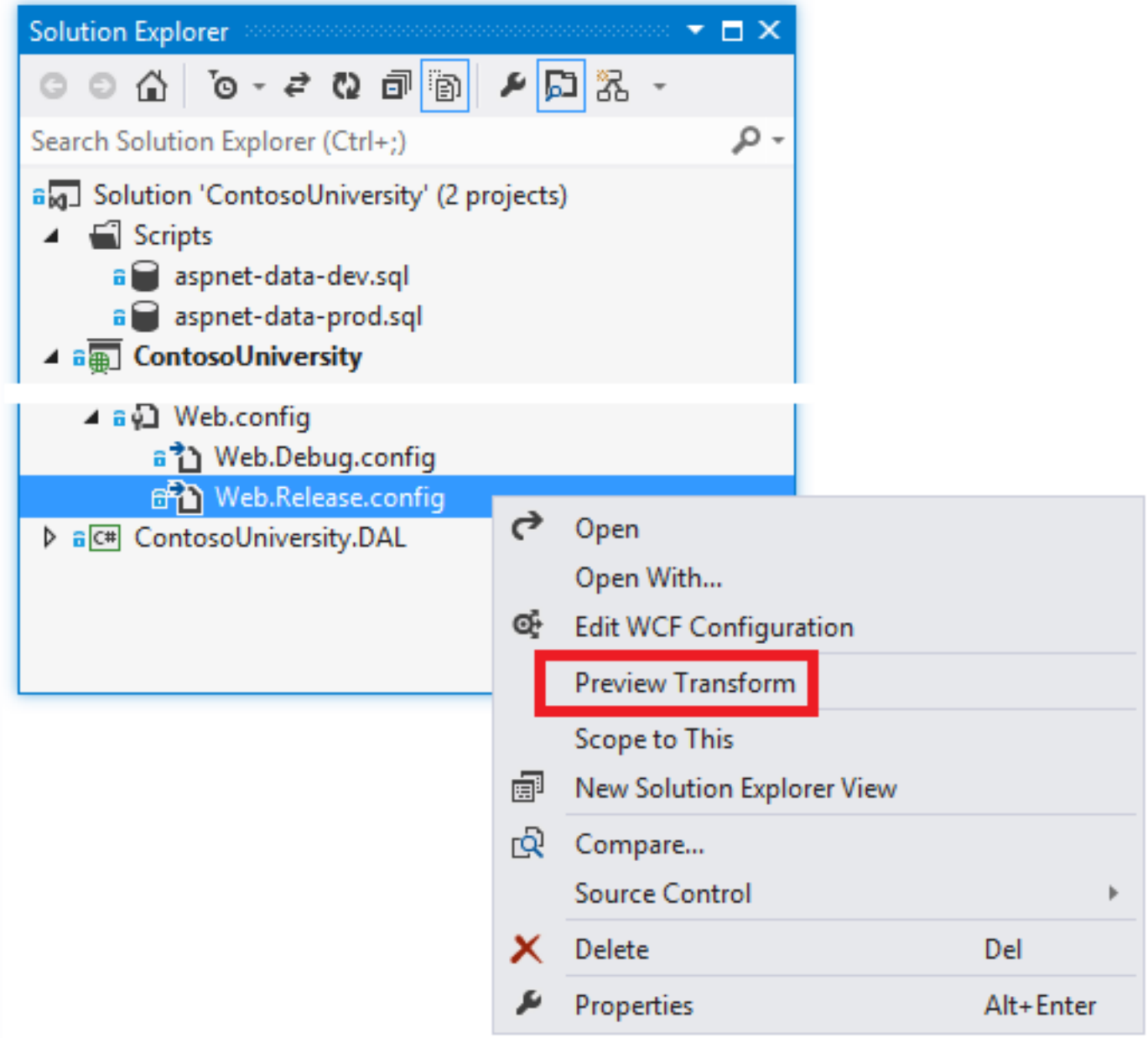
مهمترین قسمت فایل web.config تگ connection string می‌باشد که باید به رشته اتصال پایگاه داده نهایی برنامه، تغییر یابد.

```
<connectionStrings>
      <add name="TestContext"
        connectionString="Data Source=Server1;Password=****;User ID=sa; Initial Catalog=Test;Integrated
Security=True"
        xdt:Transform="SetAttributes" xdt:Locator="Match(name)"/>
    </connectionStrings>
```

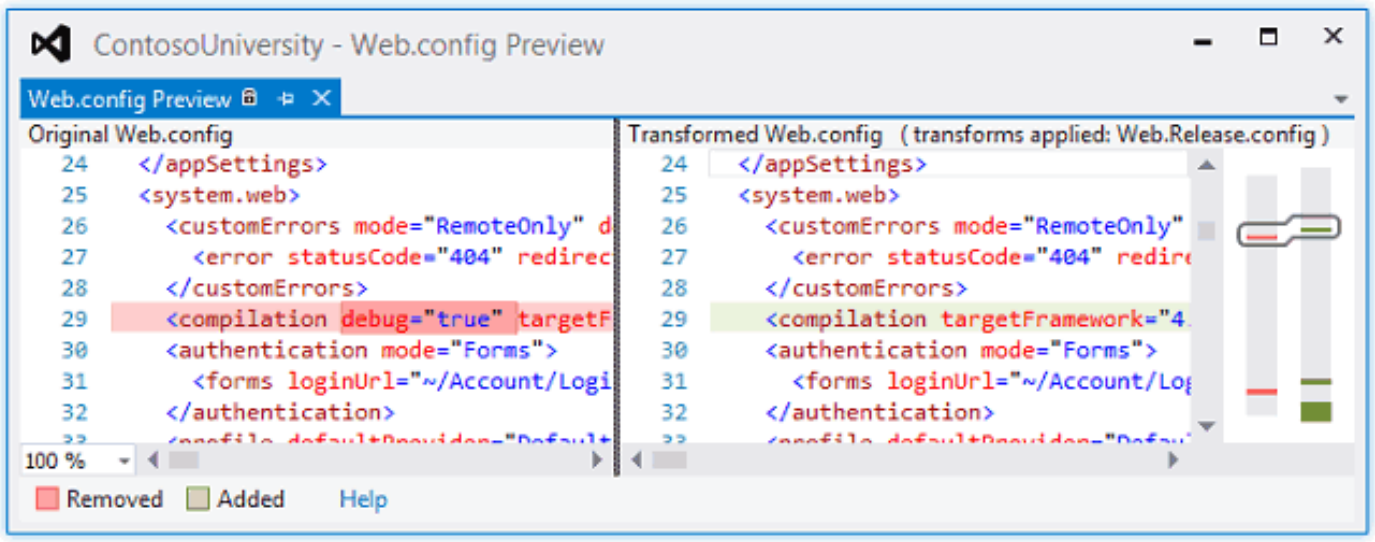
برای این کار از دستور SetAttributes مانند بالا استفاده می‌کنیم. اما چون ممکن است چندین connection string داشته باشیم، از دستور Match با مقدار name استفاده می‌کنیم، تا فقط رشته اتصالی که نام آن برابر رشته جاری می‌باشد را تغییر دهد که در اینجا TestContext را جستجو می‌نماید.

پیش نمایش connection string نهایی

برای مشاهده پیش نمایش نهایی web.config، بر روی فایل web.Release.config در پنجره Solution Explorer راست کلیک کنید و گزینه Preview Transform را انتخاب کنید.



هر دو فایل web.config اصلی و نهایی در کنار هم آورده می‌شود و اختلاف بین این دو را که برجسته‌تر شده است، می‌توانید مشاهده نمایید.


ContosoUniversity - Web.config Preview
Web.config Preview
Original Web.config
Transformed Web.config (transforms applied: Web.Release.config)
100 %
Removed
Added
Help

ادامه دارد...

نظرات خوانندگان

نویسنده: امیر
تاریخ: ۱۳۹۱/۱۲/۲۲ ۲۱:۲۴

یعنی ما تنظیمات رو روی سرور داریم بعد میایم خودمون تنظیم میکنیم، یا اینکه با این کار خودش خودکار روی سرور ست میشه.
لطفا کمی توضیح بدین

نویسنده: سید امیر سجادی
تاریخ: ۱۳۹۲/۰۲/۰۵ ۹:۱۷

این کار زمان publish سایت انجام میشه و تنظیماتی که روی فایل web.config انجام شده رو اعمال میکنه.

نویسنده: آتوسا فتوحی
تاریخ: ۱۳۹۲/۱۰/۲۲ ۲۲:۲۹

با سلام.
این تنظیمات رو باید در Web.Config نوشت یا در Web.Release.Config

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۲/۱۰/۲۳ ۱۶:۲۱

دستورات انتقال را می‌توانید در فایل Web.Debug.Config و یا Web.Release.Config بنویسید

نویسنده: آتوسا فتوحی
تاریخ: ۱۳۹۲/۱۰/۲۴ ۱۱:۳۴

سلام.
چجوری Connection String رو هنگام انتقال فایل Web.Config باید Encrypt کرد ؟

عنوان: **ابزاری برای تغییر خودکار ارجاعات پروژه برای کار با SVN SourceControl**
نویسنده: بهزاد
تاریخ: ۱۳۹۲/۰۲/۰۲ ۱۱:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: Subversion, SVN, Visual Studio, Software deployment, VisualSVN

ما در شرکت برای Source Control از SVN استفاده می‌کنیم، مزایای سورس کنترل آنقدر واضح است که دیگه من اینجا چیزی ازش نمیگم
اما برای استفاده از سورس کنترل یک مشکلی وجود دارد، اگر شما تعدادی پروژه را به کاربران خاصی بدین و تعدادی رو ندین، اون کاربر وقتی پروژه‌ها را می‌گیره با مشکل ارجاعات پروژه‌ها مواجه است. چرا که برخی از پروژه‌های ارجاعی، روی کامپیوتر برنامه نویس 1 وجود نداره. برعکس ش هم همین طوره، چون اون کاربر، پروژه‌های ارجاعی رو نداره، باید به جاش به اسمبلی نهایی اون پروژه ارجاع بده. بنابراین وقتی مدیر پروژه‌ها رو می‌گیره، باز ارجاعات اشتباه هستند!
ما اینجا برای رفع این مشکل ابزاری درست کردیم، به اسم SolutionExplorer.
این ابزار فایل solution رو به همراه پوشه حاوی فایل‌های اسمبلی می‌گیره. اگر پروژه ای به اسمبلی ای ارجاع داده باشه که پروژه اش توی solution باشه، ارجاع به اسمبلی رو تبدیل میکنه به ارجاع به پروژه و برعکسش، اگر پروژه ای به پروژه دیگه ای ارجاع داده باشه که توی solution وجود نداشته باشه، توی پوشه اسمبلی ها، دنبال اسمبلی ای می‌گرده که اسمش شبیه اسم پروژه ارجاعی باشه و اگر پیدا کنه، ارجاع رو عوض می‌کنه
البته برای جلوگیری از به هم ریختگی، نرم افزار از فایل‌های پروژه ای که دستکاری می‌کنه، پشتیبان می‌گیره دانلود پروژه

توجه:
* این برنامه از تمامی جهات تست نشده است، با ریسک خودتون ازش استفاده کنید (ما تو شرکت دیگه ریسکی نداریم :))
* سیستم نامگذاری اسمبلی‌ها و پروژه‌های ما ممکنه فرق کنه
* اگر به مشکلی برخوردید، لطفا زیر همین مطلب برام بنویسید
* انتخاب پوشه اسمبلی ها، الزامی نیست

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۰۲ ۱۱:۴۰

ممنون. ایده خوبی هست.

یک روش دیگر هم استفاده از نیوگت هست برای مدیریت لوکال وابستگی‌ها

Creating and then using a NuGet local repository

How to access NuGet when NuGet.org is down

نویسنده: بهزاد
تاریخ: ۱۳۹۲/۰۲/۰۲ ۱۱:۴۲

در مورد وابستگی‌های نوگت کاری نکردیم، فقط در مورد پروژه‌ها و اسمبلی هاست

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۰۲ ۱۲:۳۰

نیوگت لوکال روی شبکه می‌تونید تعریف کنید بر اساس وابستگی‌های داخلی خودتون. یعنی کاملا مستقل از نیوگت روی اینترنت.

نویسنده: یوسف نژاد
تاریخ: ۱۳۹۲/۰۲/۰۳ ۸:۲۹

شما میتونین خروجی تمام پروژه‌های ریفرنس داده شده در پروژه‌های دیگه رو به یک مسیر مشخص و مشترک تنظیم کنید. تمام پروژه‌ها هم ریفرنس خودشون رو از اون مسیر مشخص بگیرن. سپس فایلهای dll. یا exe. موردنظر رو بصورت multi-check out تنظیم کنید. بعدش هرکسی که آخرین نسخه از اون کتابخونه رو داره توسعه میده هر روز چکین کنه و بقیه هم هر روز get latest کنن. کاری که ما داریم به راحتی در شرکت خودمون انجام میدیم.

نویسنده: سیروس
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۳:۳۱

ما هم از این روش استفاده می‌کنیم.

نویسنده: منیژه محمدی
تاریخ: ۱۳۹۲/۰۸/۲۶ ۲۱:۱۳

در مورد TFS چطور ؟ در مورد ان پیشنهادی ندارید؟

عنوان: **Web.config File Transformation #2**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۲/۰۲/۱۹ ۱۶:۵۵
آدرس: www.dotnettips.info
گروه‌ها: Visual Studio, Software deployment, Web.config Transformation

در مطلب قبلی Web.config File Transformation #1 با مفهوم انتقال وب کانفیگ و برخی از روش‌های آن آشنا شدید در ادامه به موارد دیگری خواهم پرداخت.

قواعد انتقال وب کانفیگ

در کل دو ویژگی اصلی در انتقال وب کانفیگ وجود دارد که یک xdt:Transform و دیگری xdt:Locator می باشد. این دو در واقع چگونگی تغییر فایل انتقالی در زمان deploy آن را تعیین می‌کنند. این ویژگی‌ها از نوع xml می‌باشد که در فضای نام XML-Document- Transform تعریف شده و با پسوند xdt شروع می‌شود.

قاعده ویژگی Locator

این ویژگی برای جستجو در فایل وب کانفیگ استفاده می‌شود

:Condition

عبارت condition برای تعیین شرط قاعده Locator استفاده می‌شود

```
Locator="Condition(XPath expression)"
```

مثال:

```
<configuration xmlns:xdt="...">
  <connectionStrings>
    <add name="AWLT" connectionString="newstring"
       providerName="newprovider"
       xdt:Transform="Replace"
       xdt:Locator="Condition(@name='oldname'
         or @providerName='oldprovider')" />
  </connectionStrings>
</configuration>
```

مثال بالا نشان می‌دهد که چگونه دنبال connectionstring ی بگردیم که نام آن oldname یا نام ارائه دهنده آن oldprovider باشد.

:Match

برای انتخاب عنصر یا عناصری که مقدار آن برابر ویژگی یا ویژگی‌های تعیین شده باشد. در صورتی که چندین ویژگی یافت شود، فقط عناصری که همه ویژگی‌های تعیین شده آن‌ها برابر باشد انتخاب می‌شود. نام ویژگی‌ها را می‌بایست با کما از هم جدا نمود.

```
Locator="Match(comma-delimited list of one or more attribute names)"
```

مثال:

```
<configuration xmlns:xdt="...">
  <connectionStrings>
    <add name="AWLT" connectionString="newstring"
       providerName="newprovider"
       xdt:Transform="Replace"
       xdt:Locator="Match(name)" />
  </connectionStrings>
</configuration>
```

مثال فوق دنبال عناصری که ویژگی نام آنها برابر AWLT باشد می‌گردد.

:XPath

عبارتی از مسیرهای عناصر xml را جستجو می‌کند.

```
Locator="XPath(XPath expression)"
```

مثال:

```
<configuration xmlns:xdt="...">
  <connectionStrings>
    <add name="AWLT" connectionString="newstring"
       providerName="newprovider"
       xdt:Transform="Replace"
       xdt:Locator="XPath(configuration/connectionStrings[@name='AWLT'
         or @providerName='System.Data.SqlClient'])" />
  </connectionStrings>
</configuration>
```

در این مثال همانند مثال Condition همان عناصر را با استفاده از XPath جستجو می‌نماید.

قاعده ویژگی Transform

این ویژگی نحوه انتقال عنصر در فایل وب کانفیگ را مشخص می‌سازد.

:Replace

عناصر تعیین شده با عناصر انتقالی جایگزین می‌شود

```
Transform="Replace"
```

:Insert

عنصر انتقالی به عنوان فرزند عنصر تعیین شده اضافه می‌گردد.

```
Transform="Insert"
```

مثال:

```
<configuration xmlns:xdt="...">
  <connectionStrings>
    <add name="AWLT" connectionString="newstring"
       providerName="newprovider"
       xdt:Transform="Insert" />
  </connectionStrings>
</configuration>
```

connectionstring انتقالی را به لیست رشته‌های اتصال فایل انتقالی اضافه می‌نماید.

:InsertBefore

عنصر انتقالی را قبل از مسیر تعیین شده با XPath قرار می‌دهد.

```
Transform="InsertBefore(XPath expression)"
```

مثال:

```
<configuration xmlns:xdt="...">
  <authorization>
    <allow roles="Admins"
      xdt:Transform="InsertBefore(/configuration/system.web/authorization/deny[@users='*'])" />
  </authorization>
</configuration>
```

عنصر انتقالی که همان allow می‌باشد که قبل از عنصر مسیر configuration/system.web/authorization/deny برای همه کاربر

است قرار می‌دهد.

:InsertAfter

عنصر تعیین شده را بعد از مسیر تعیین شده با XPath قرار می‌دهد.

```
Transform="InsertAfter(XPath expression)"
```

مثال:

```
<configuration xmlns:xdt="...">
  <authorization>
    <deny users="UserName"
      xdt:Transform="InsertAfter
        (/configuration/system.web/authorization/allow[@roles='Admins'])" />
  </authorization>
</configuration>
```

:Remove

عنصر تعیین شده را از فایل انتقالی حذف می‌نماید. اگر چندین عنصر یافت شود اولین عنصر حذف خواهد شد.

```
Transform="Remove"
```

مثال

```
<configuration xmlns:xdt="...">
  <connectionStrings>
    <add xdt:Transform="Remove" />
  </connectionStrings>
</configuration>
```

همه عناصر add عنصر connectionstring را انتخاب و اولین add را حذف می‌نماید.

:RemoveAll

عنصر یا عناصر تعیین شده را از فایل انتقالی حذف می‌نماید.

```
Transform="RemoveAll"
```

مثال:

```
<configuration xmlns:xdt="...">
  <connectionStrings>
    <add xdt:Transform="RemoveAll" />
  </connectionStrings>
</configuration>
```

همه عناصر add را از فایل انتقالی حذف می‌نماید.

:RemoveAttribute

ویژگی تعیین شده را از عناصر انتخاب شده حذف می‌نماید.

```
Transform="RemoveAttributes(comma-delimited list of one or more attribute names)"
```

مثال:

```
<configuration xmlns:xdt="...">
```

```
  <compilation
    xdt:Transform="RemoveAttributes(debug,batch)">
  </compilation>
</configuration>
```

ویژگی debug و batch را از عنصر compilation وب کانفیگ انتقالی حذف می‌نماید.

:SetAttribute

ویژگی تعیین شده را با مقادیر انتقالی مقدار دهی می‌کند.

```
Transform="SetAttributes(comma-delimited list of one or more attribute names)"
```

مثال:

```
<configuration xmlns:xdt="...">
  <compilation
    batch="false"
    xdt:Transform="SetAttributes(batch)">
  </compilation>
</configuration>
```

ویژگی batch وب کانفیگ انتقالی را با مقدار false مقدار دهی می‌کند.

ادامه دارد...

عنوان: **بازسازی کامل پوشه packages بسته‌های NuGet به صورت خودکار**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۷ ۱۶:۰
آدرس: www.dotnettips.info
گروه‌ها: NuGet, Software deployment

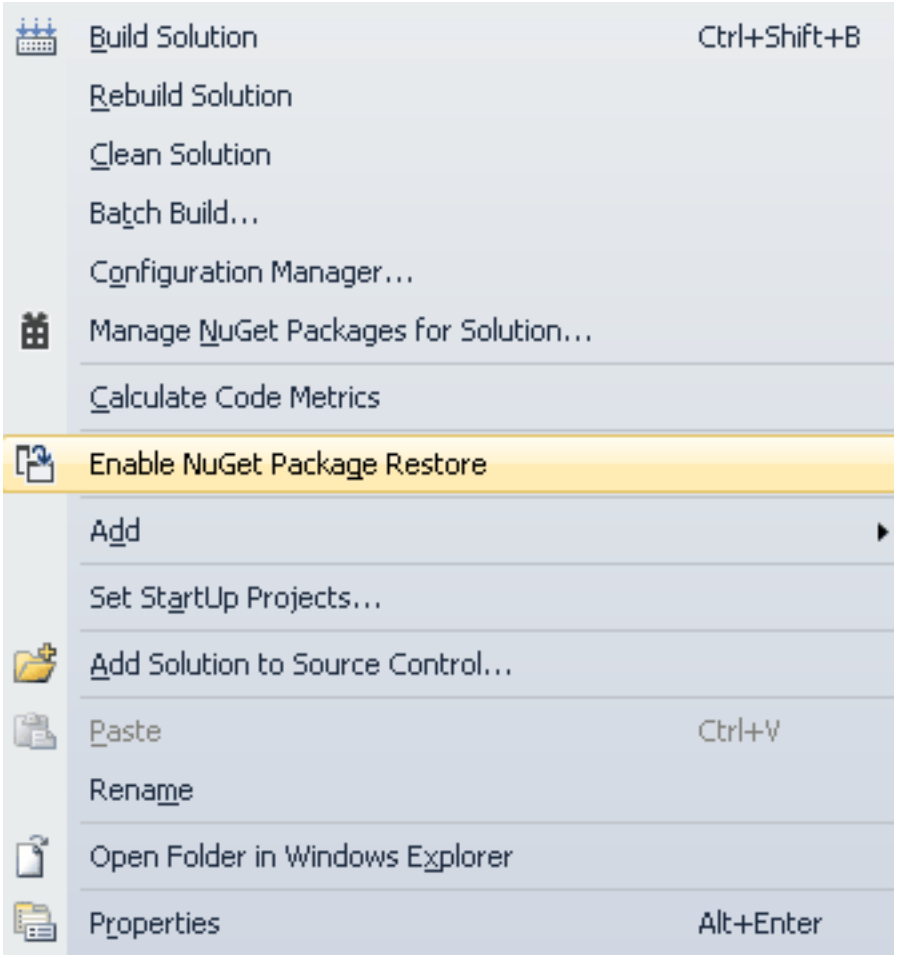
اگر قصد انتشار عمومی پروژه خود را دارید، نیازی به ارائه پوشه packages آن نیست. استفاده کننده نهایی به روشی که در ادامه توضیح داده خواهد شد، می‌تواند ارجاعات کل Solution را به یکباره به روز نماید؛ البته اگر تنها فایل یا فایل‌های packages.config پروژه‌های موجود پیوست شده و موجود باشند.

یک آزمایش

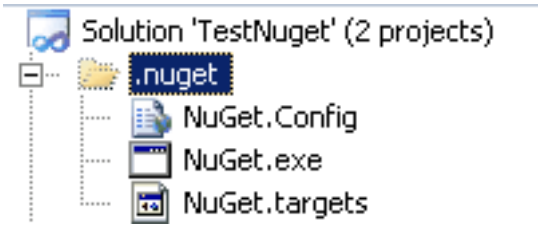
یک پروژه کنسول جدید را آغاز می‌کنیم. سپس به Solution آن یک Class library آزمایشی را نیز اضافه خواهیم کرد.
اکنون در خط فرمان پاورشل نیوگت (Tools > Library Package Manager > Package Manager Console) به هر یک از این دو پروژه، ارجاعی را به بسته structuremap از طریق دستور زیر اضافه می‌کنیم:

```
Install-Package structuremap
```

اکنون یکبار پروژه را کامپایل کرده و سپس VS.NET را خاتمه می‌دهیم.
در ادامه پوشه‌های packages و همچنین bin و obj را کلا حذف می‌کنیم؛ اما فایل‌های متنی packages.config پروژه‌ها را نگه خواهیم داشت.
مجددا به VS.NET مراجعه خواهیم کرد. اینبار بر روی Solution کلیک راست کرده و گزینه «Enable NuGet Package Restore» را انتخاب می‌کنیم:



یک پوشه حاوی NuGet.exe به Solution جاری اضافه خواهد شد:



اکنون اگر پروژه را Build کنیم، تمام ارجاعات را به صورت خودکار از اینترنت (و یا کش موجود بر روی سیستم) دریافت و به Solution اضافه می‌کند.
به علاوه پوشه Packages نیز مجددا بازسازی خواهد شد.

پس از اینکار نهایتا برای اطمینان خاطر می‌توان دستور ذیل را در خط فرمان پاورشل نیوگت صادر کرد:

```
Update-Package -Safe
```

این دستور به یکباره کلیه ارجاعات موجود packages را بررسی کرده و به روز خواهد کرد. پارامتر safe آن اختیاری است. اگر عنوان شود، سعی می‌کند همان شماره نگارشی را که در فایل‌های packages.config موجود است، دریافت و به روز نماید. در غیر اینصورت، آخرین فایل و آخرین نگارش موجود را دریافت و به روز رسانی خواهد کرد. به این ترتیب می‌توان به صرفه جویی زمانی قابل توجهی در یک پروژه با ارجاعات زیاد، رسید.
برای نمونه دستور update-package را بر روی یک پروژه MVC4 اجرا کنید تا این صرفه جویی زمانی را بهتر بتوانید حس کنید !

نتیجه گیری
لطفا حین ارائه عمومی پروژه خود، پوشه‌های bin، obj و همچنین packages آن‌را حذف کنید. استفاده کننده صرفا با داشتن فایل‌های packages.config به کمک روشی که عنوان شد می‌تواند ارجاعات کل Solution را بازیابی کند.

نظرات خوانندگان

نویسنده: توحید عزیزی
تاریخ: ۱۳۹۲/۰۲/۲۸ ۰:۱۲

بسیار عالی. سپاسگزارم.

نویسنده: Ara
تاریخ: ۱۳۹۲/۰۲/۳۰ ۱۴:۵۳

سلام
خسته نباشید
می‌خواستم بدونم برای اینکه یک پکیج لوکالی استفاده بشه چه باید کرد !
مثلا مثلا DLL های Framework رو بشه Package کرد و همکاران شرکت پس تغییر یک dll بتونن اون رو تو سیستم خودشون آپدیت کنند !

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۳۰ ۱۵:۴

برای استفاده لوکال در شبکه داخلی، امکان تهیه یک NuGet Server لوکال وجود داره: « Hosting Your Own NuGet Feeds ». حتی شبیه به سایت گالری NuGet رو هم میشه لوکال نصب کرد؛ سورس باز است .

نویسنده: Ara
تاریخ: ۱۳۹۲/۰۳/۱۰ ۲۱:۳۳

ممنون
وب هاست نمی‌خواستم ، در مورد نحوه ساختن Package می‌خواستم بدونم، که فکر می‌کردم دنگ فنگ زیادی داره
نمی‌دونستم براحتی دستور زیره
nuget pack ProjectFile.csproj
که پکیج هام رو ساختم و گذاشتم تو یک فولدر تو ویژوال استودیو تو قسمت
Options/Package Manger/Package Resources
اضافه اش کردم
البته این هم خوب بود برای ویرایش و غیره
http://docs.nuget.org/docs/creating-packages/using-a-gui-to-build-packages

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۲/۰۴/۰۵ ۱۳:۲۱

با سلام.
چرا وقتی که بر روی سولوشن کلیک راست میکنم گزینه Enable NuGet Package Restore وجود ندارد. nuget ولی نصب شده است. با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۰۵ ۱۳:۳۴

- این گزینه در آخرین نگارش‌های NuGet اضافه شده. نیاز است خود NuGet رو به روز کنید .
- همچنین اگر مطابق شکل دوم، پوشه nuget. در پروژه موجود باشد، این گزینه ظاهر نخواهد شد.

نویسنده: منیژه محمدی
تاریخ: ۱۳۹۲/۰۸/۲۶ ۲۰:۵۷

سلام من وقتی این خط را اجرا میکنم Install-Package structuremap این خطا را بهم میدهد

```
PM> Install-Package structuremap
'Install-Package : The remote name could not be resolved: 'az320820.vo.msecnd.net
At line:1 char:16
+ Install-Package <<<< structuremap +
+ CategoryInfo : NotSpecified: (:) [Install-Package], WebException +
+ FullyQualifiedErrorId : NuGetCmdletUnhandledException,NuGet.PowerShell.Commands.InstallPackageCommand +
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۲۶ ۲۱:۱۰

remote name could not be resolved یعنی مشکل DNS و یا تنظیمات اتصالی را دارید به احتمال زیاد. آدرس زیر را در **IE** امتحان کنید (از این جهت که تنظیمات اینترنت IE به برنامه‌های دات نت به صورت پیش فرض اعمال می‌شوند). اگر باز نشد، مشکل اتصالی دارید:

https://az320820.vo.msecnd.net/packages/structuremap.2.6.4.1.nupkg

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۰ ۱۳:۱۳

یک نکته‌ی تکمیلی

مطلب جاری توسط تیم NuGet منسوخ شده اعلام گردیده و در نگارش‌های جدید آن، به صورت خودکار و بدون نیاز به هیچگونه تنظیم اضافه‌تری انجام می‌شود. بیشتر در اینجا

نویسنده: وحید نصیری
تاریخ: ۱۳۹۴/۰۱/۰۵ ۱۳:۰

یک نکته

اگر حین ارائه‌ی برنامه‌ی خود فایل exe مربوط به nuget را ارائه ندهید، پیام خطای یافت نشدن آن‌را در حین Build مشاهده خواهید کرد. برای رفع آن تنها کافی است فایل NuGet.targets را گشوده و دریافت خودکار nuget.exe را فعال کنید:

```
<DownloadNuGetExe Condition=" '$(DownloadNuGetExe)' == '' ">true</DownloadNuGetExe>
```

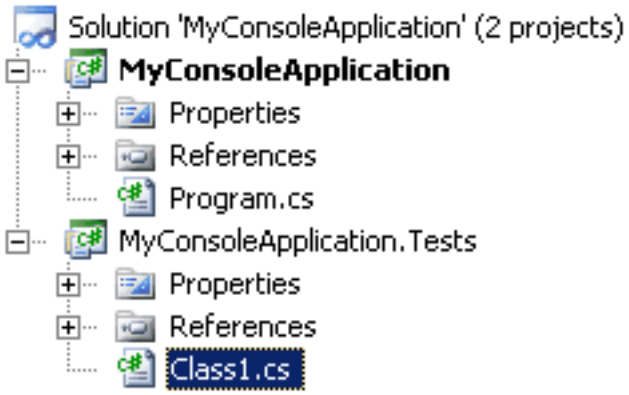
این تنظیم به صورت پیش فرض غیرفعال است.

عنوان: **ایجاد نصاب یک قالب پروژه جدید چند پروژه‌ای در ویژوال استودیو**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۹ ۸:۵
آدرس: www.dotnettips.info
برچسب‌ها: Visual Studio, Software deployment

در ویژوال استودیو ذیل منوی File، گزینه‌ای وجود دارد به نام Export template که کار آن تهیه یک قالب، بر اساس ساختار پروژه جاری است. این قابلیت جهت تهیه قالب‌های سفارشی، برای کاهش زمان تهیه پروژه‌ها بسیار مفید است. به این ترتیب می‌توان بسیاری از نکات مدنظر را، در یک قالب ویژه لحاظ کرد و به دفعات بدون نیاز به copy/paste مداوم فایل‌ها و تنظیمات اولیه، بسیار سریع یک پروژه جدید دلخواه را ایجاد نمود.
اما ... این قالب تهیه شده، صرفا بر اساس یکی از چندین پروژه Solution جاری تهیه می‌شود و همچنین نصب و توزیع آن نیز دستی است. در ادامه قصد داریم با نحوه تهیه یک قالب جدید پروژه متشکل از چندین پروژه، به همراه تهیه فایل VSI نصاب آن، آشنا شویم.

تهیه یک ساختار نمونه

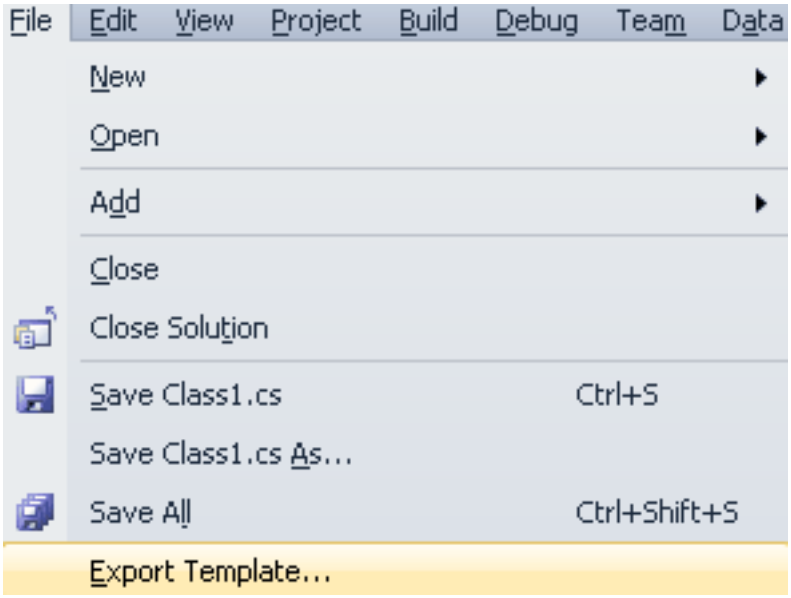
یک پروژه جدید کنسول را به نام MyConsoleApplication ایجاد کنید. سپس به Solution جاری، یک Class library جدید را به نام مثلا MyConsoleApplication.Tests اضافه نمائید. تا اینجا به شکل زیر خواهیم رسید:



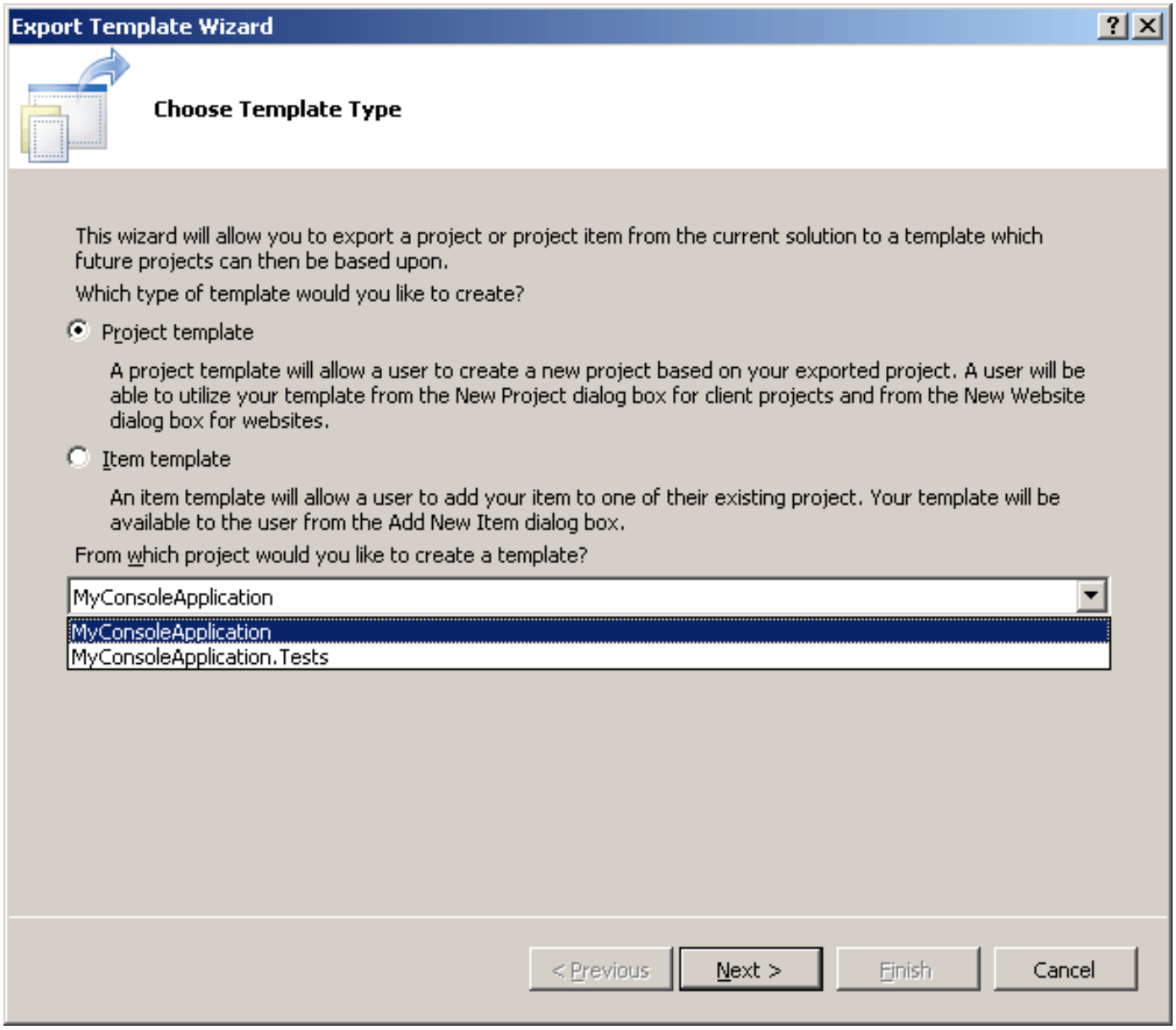
اکنون قصد داریم از این پروژه خاص، یک قالب تهیه کنیم؛ تا هربار نخواهیم یک چنین مراحلی را تکرار کنیم.

تهیه قالب به ازای هر پروژه در Solution

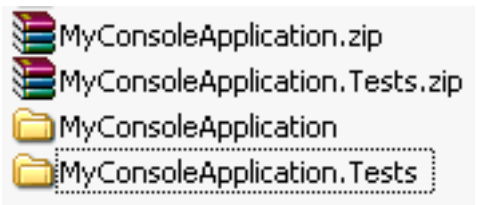
در همین حال که Solution باز است، به منوی File و گزینه Export template مراجعه کنید.



در اینجا تنها امکان انتخاب یک پروژه وجود دارد. به همین جهت این مرحله را باید به ازای هر تعداد پروژه موجود در `Solution` یکبار تکرار کرد.



اکنون در پوشه My Documents\Visual Studio 2010\My Exported Templates دو فایل zip به نام‌های MyConsoleApplication.zip و MyConsoleApplication.Tests.zip وجود دارند. هر دو فایل را توسط برنامه‌های مخصوص گشودن فایل‌های Zip گشوده و تبدیل به دو پوشه باز شده MyConsoleApplication و MyConsoleApplication.Tests کنید.

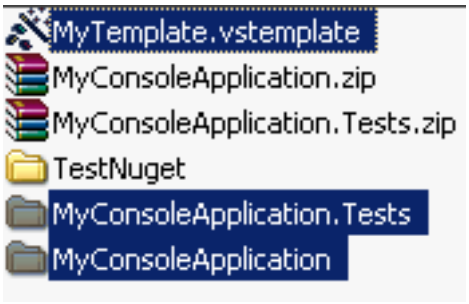


افزودن فایل MyTemplate.vstemplate چند پروژه‌ای

در همین پوشه جاری که اکنون حاوی دو پوشه باز شده است، یک فایل متنی جدید را با محتوای ذیل به نام MyTemplate.vstemplate ایجاد کنید:

```
<VSTemplate Version="3.0.0" Type="ProjectGroup"
xmlns="http://schemas.microsoft.com/developer/vstemplate/2005">
  <TemplateData>
    <Name>MyConsoleApplication</Name>
    <Description>MyConsoleApplication Desc</Description>
    <ProjectType>CSharp</ProjectType>
  </TemplateData>
  <TemplateContent>
    <ProjectCollection>
      <ProjectTemplateLink ProjectName="MyConsoleApplication">
      MyConsoleApplication\MyTemplate.vstemplate</ProjectTemplateLink>
      <ProjectTemplateLink ProjectName="MyConsoleApplication.Tests">
      MyConsoleApplication.Tests\MyTemplate.vstemplate</ProjectTemplateLink>
    </ProjectCollection>
  </TemplateContent>
</VSTemplate>
```

در اینجا به ازای هر پروژه، یک ProjectTemplateLink ایجاد خواهد شد که به فایل MyTemplate.vstemplate موجود در قالب آن اشاره می‌کند.
در ادامه این دو پوشه باز شده و فایل MyTemplate.vstemplate فوق را انتخاب کرده:



و همگی را تبدیل به یک فایل zip جدید کنید؛ مثلا به نام MyConsoleApplicationTemplates.zip.

تهیه فایل نصاب از قالب پروژه جدید

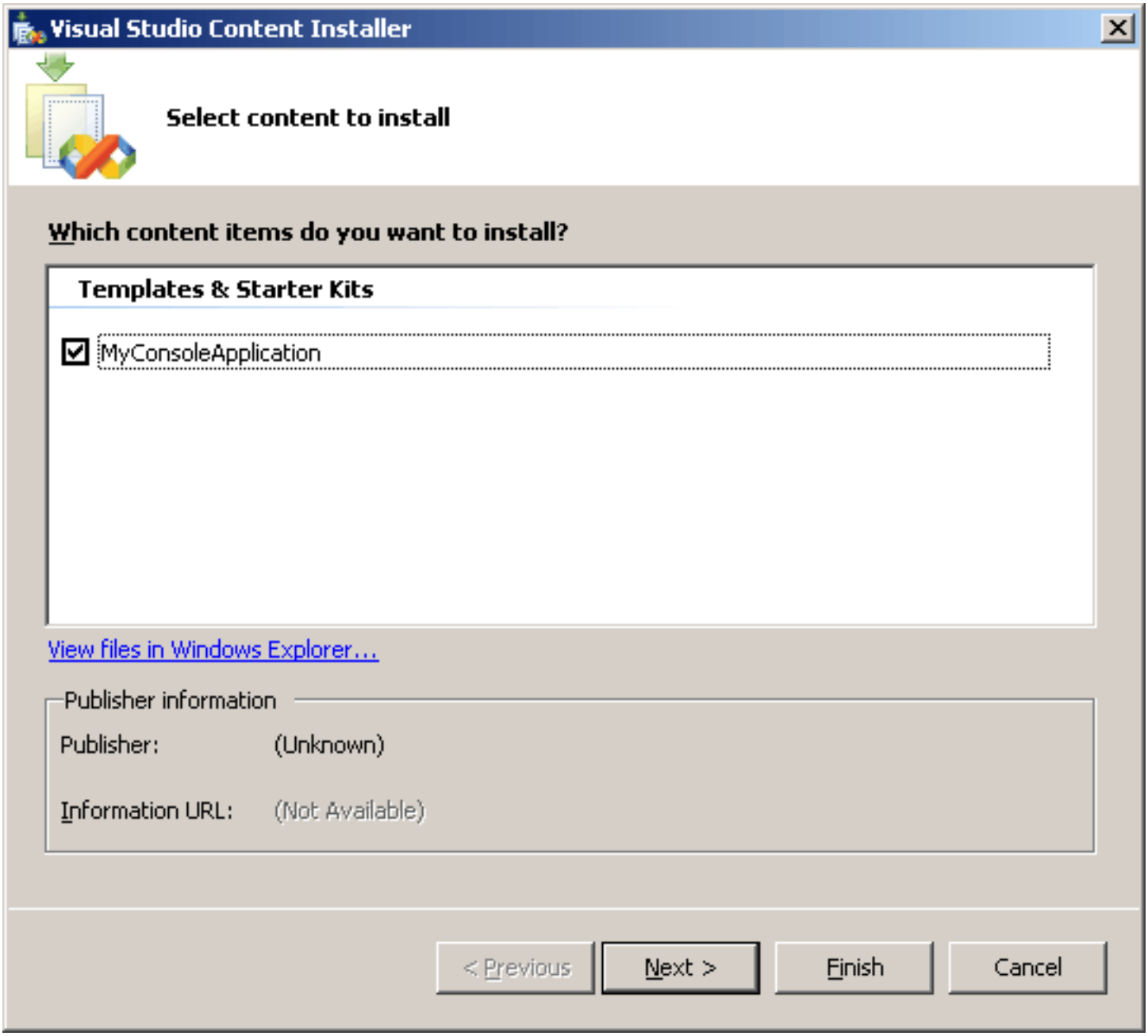
تا اینجا موفق شدیم، چندین قالب پروژه تهیه شده را به هم متصل کرده و تبدیل به یک فایل zip نهایی کنیم. مرحله بعد ایجاد فایلی است متنی به نام MyConsoleApplicationTemplates.vscontent با محتویات زیر:

```
<VSContent xmlns="http://schemas.microsoft.com/developer/vscontent/2005">
  <Content>
    <FileName>MyConsoleApplicationTemplates.zip</FileName>
    <DisplayName>MyConsoleApplication</DisplayName>
    <Description>A C# project that ...</Description>
    <FileContentType>VSTemplate</FileContentType>
    <ContentVersion>1.0</ContentVersion>
    <Attributes>
      <Attribute name="ProjectType" value="Visual C#" />
      <Attribute name="ProjectSubType" value="Web" />
      <Attribute name="TemplateType" value="Project" />
    </Attributes>
  </Content>
</VSContent>
```

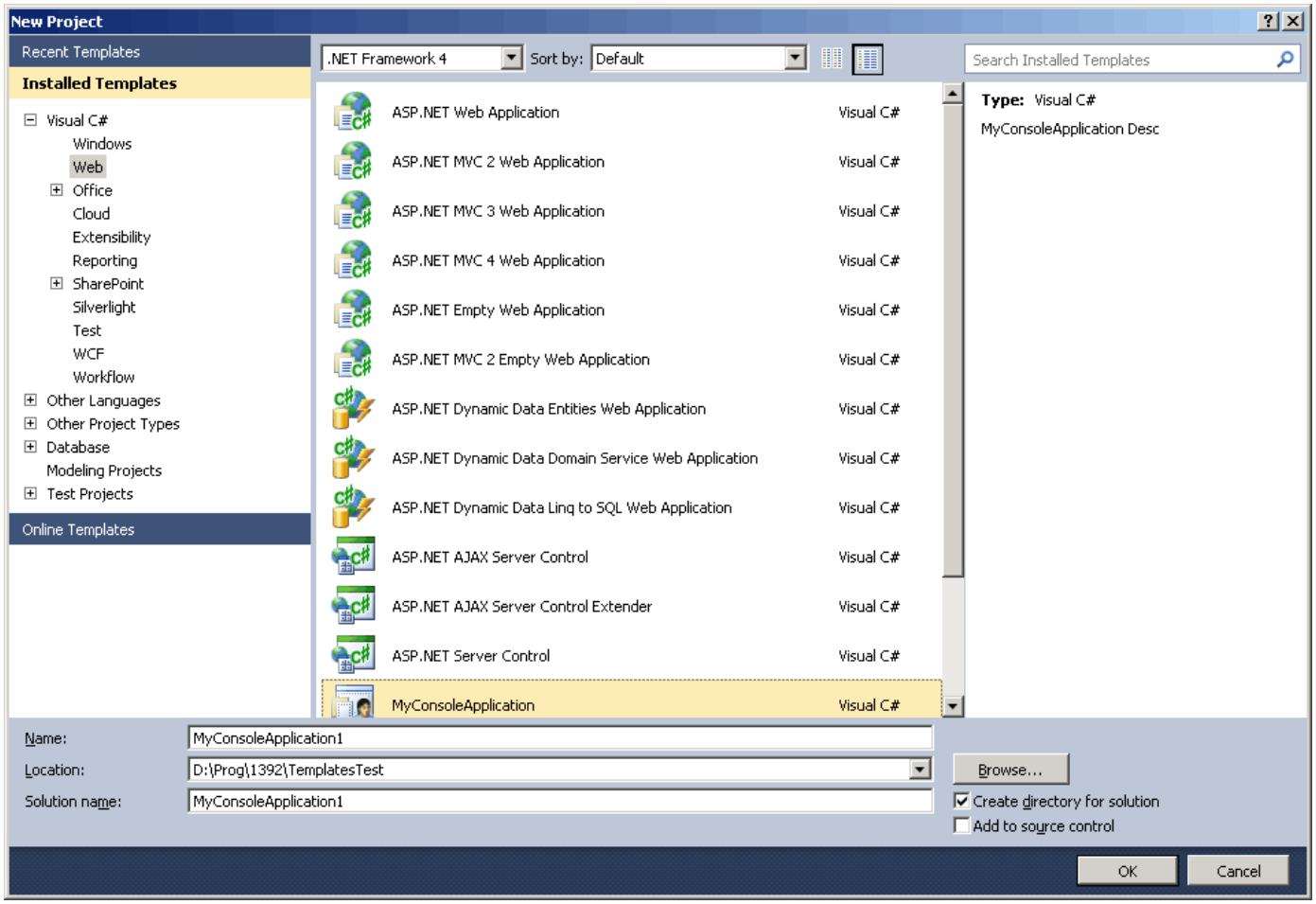
در اینجا توسط قسمت Attributes مشخص می‌کنیم که قالب پروژه جدید باید در صفحه new project، در کدام مدخل قرار گیرد. بنابراین مطابق تنظیمات فوق، قالب جدید ذیل پروژه‌های وب سی‌شارپ قرار خواهد گرفت. مقدار FileName آن دقیقا معادل نام

فایل zip ایی است که در مرحله قبل ایجاد کردیم.

مرحله بعد انتخاب دو فایل MyConsoleApplicationTemplates.vscontent و MyConsoleApplicationTemplates.zip و تبدیل ایندو به یک فایل zip جدید است. پس از ایجاد فایل جدید، پسوند آن‌را به VSI تغییر دهید؛ برای مثال نام آن‌را به MyConsoleApplicationTemplates.vsi تغییر دهید. اکنون این فایل نهایی با دوبار کلیک بر روی آن قابلیت اجرا و نصب خودکار را پیدا می‌کند.



پس از نصب، بلافاصله ذیل قسمت پروژه‌های وب قابل دسترسی و استفاده خواهد بود:



بنابراین به صورت خلاصه:

1) به ازای هر پروژه، یک فایل قالب zip معادل آن باید تهیه شود.

2) تمام این فایل‌های zip را گشوده و تبدیل به پوشه‌های متناظری کنید.

3) یک فایل MyTemplate.vstemplate را در پوشه ریشه مرحله 2 جهت تعریف ProjectTemplateLinkها اضافه کنید.

4) فایل جدید MyTemplate.vstemplate مرحله 3 و تمام پوشه‌های قالب‌های باز شده مرحله 2 را zip کنید.

5) سپس یک فایل vscontent نصاب را تهیه و آن‌را با فایل zip مرحله 4 مجددا zip کرده و پسوند آن‌را به VSI تغییر دهید.

اکنون می‌توان این فایل VSI را توزیع کرد.

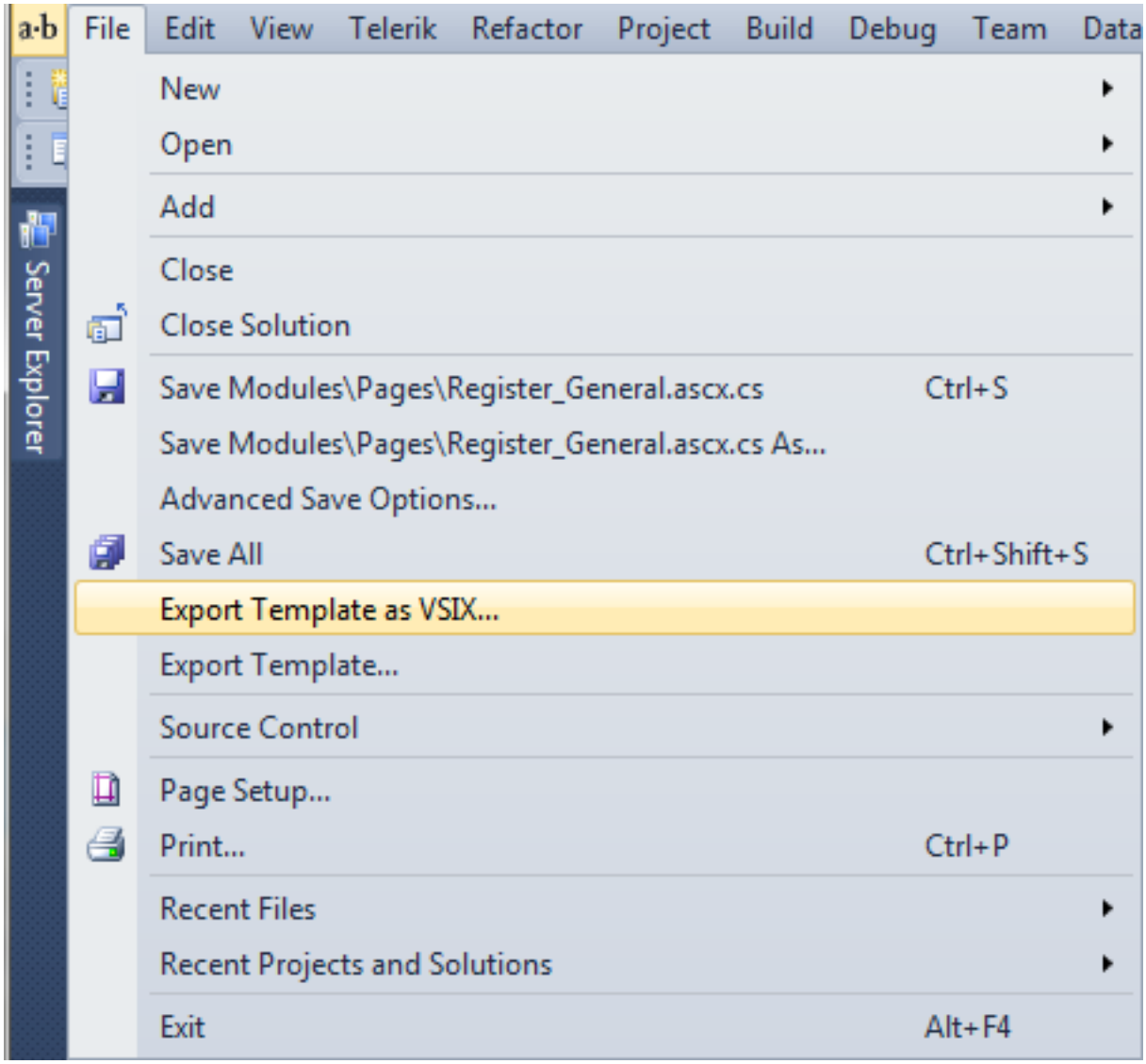
نظرات خوانندگان

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۱:۵۷

سلام و ممنون

بنده از این روش استفاده کرده بودم و نهایتا برای خودکار سازی این اعمال از افزونه ExportTemplate(vsix).vsix ویژوال استودیو استفاده کردم

طریقه استفاده اون هم به این صورت هستش که پس از نصب گزینه Export Template as VSIX... در منوی فایل ظاهر میشه و با کلیک بر روی اون تمامی پروژه‌های موجود در Solution جاری رو لیست می‌کنه و می‌تونید انتخاب کنید و Export کنید



نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۲:۲

نکته تکمیلی اینکه این امکان وجود داره که پس از انتخاب پروژه(ها)، میتونید برای نصاب خودتون آیکون قرار بدید، لایسنس گذاری کنید و در ضمن ساخت نصاب عمل import شدن به VS هم به طور خودکار انجام بشه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۳:۱۰

- ممنون. افزونه خیلی کاربردی و مفیدی است.
- البته در حالت دستی عنوان شده امکان تعریف آیکون و غیره هم هست. در متن، لینک داده شده به مراجع تولید فایل‌های vstemplate و vscontent که برای نمونه یک مدخل اضافه‌تر برای آیکون پیدا می‌کند :

```
<Icon>__Template_small.png</Icon>
```

در کل بد نیست یک برنامه نویس بدونه پشت صحنه این اعمال به چه صورتی هست.

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۳:۴۱

بله دقیقا با نظر شما موافقم

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۷:۵۳

سلام ... خیلی ممنون بابت این افزونه ...
گویا این افزونه روی vs 2012 کار نمیکنه! ... راهی هست که بشه کار کنه؟!

نویسنده: بهمن خلفی
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۸:۰

با سلام و عرض تشکر
یک سوال : اینکه بخواهیم از روشی که شما ارائه دادید استفاده کنیم ولی اگر بخواهیم بدین گونه باشد که :
پس از درست کردن قالب مد نظر پروژه بخواهیم یک ساختاری مانند پروژه MVC یا یک ساختار دلخواه داشته باشیم بصورتی که همواره میخواهیم در قالب پروژه ، 3 فولدر وجود داشته باشند که فولدر اول همنام پروژه + Module باشد : اگر نام پروژه هنگام ایجاد یا Add کردن Factor باشد یک فولدر در داخل همان پروژه جدید بنام FactorModule ایجاد شود و دو فولدر دیگه FactorAdmin و FactorUser در آن ایجاد شود و به همین ترتیب...؟!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۸:۱

فایل extension.vsixmanifest افزونه رو باید ویرایش کنید (فایل vsix در اصل یک فایل zip است). مثلا VisualStudio Version آن الان 10 است که باید بشود 11. بعد در SupportedFrameworkRuntimeEdition آن باید MaxVersion به 4.5 تنظیم شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۹ ۲۰:۴

امکان سفارشی سازی قالب ساز با کدنویسی هم میسر است. نیاز است اینترفیس IWizard پیاده سازی شود. در اینجا هر نوع کدی رو که لازم بود می‌شود در متد ProjectFinishedGenerating آن تدارک دید. مثلا پوشه درست کند، تنظیمات پروژه را تغییر دهد و امثال آن.
- یک مثال از پیاده سازی اینترفیس IWizard:
Creating custom project template with wizard for Visual Studio

- مثلا پروژه sharp-architecture از همین روش استفاده می‌کنه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۹ ۲۲:۴۳

یک نکته: روش دیگر ساخت قالب، استفاده از برنامه Templify است.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۳۰ ۰:۲۱

یک نکته تکمیلی دیگر:
با نصب SDK ویژوال استودیو (^ و ^) یک قالب جدید تولید فایل‌های VSIX به مجموعه قالب‌های پروژه‌ها اضافه می‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۳۰ ۱۱:۴۰

یک نکته مهم!
اگر روش فوق را امتحان کنید (چه استفاده از افزونه یاد شده یا حتی روش دستی مقدماتی فوق)، هر نامی را که در ابتدای کار ایجاد Solution جدید وارد کنید، به زیر پروژه‌های اضافه شده اعمال نمی‌شود. یعنی همان نام ابتدایی خودشان را خواهند داشت که این مورد اصلا جالب نیست.
برای رفع آن نیاز است از متغیری به نام $safeprojectname$ استفاده شود (هرجایی که نام پروژه به صورت مستقیم استفاده شده، حتی نام پوشه‌ها یا فایل‌ها) به همراه ReplaceParameters=true. یک مثال را در این مورد در پیوست ذیل می‌توانید دریافت کنید:

MyConsoleApplicationTemplates.zip

روش نصب دستی این قالب با کپی کردن آن در پوشه My Documents\Visual Studio xyz\Templates\ProjectTemplates است.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۳۰ ۱۱:۴۴

و یا از $safeprojectname$ باید استفاده شود به روشی که در اینجا توضیح دادم .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۰۴ ۱۵:۲

نکات مطرح شده در این مطلب، تبدیل به یک پروژه شد: « Solution template generator »

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۲/۰۳/۳۱ ۸:۴۵

با سلام.

یک پروژه class library در solution خود دارم. چگونه می‌شود فضای نام پیش فرض این پروژه و فایل‌های درون آنرا براساس $safeprojectname$ تغییر داد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۳۱ ۸:۵۶

- اینکار باید جداگانه (جدای از پروژه در حال کار) و به صورت دستی انجام شود (یک search و replace است).

- یا پروژه « Solution template generator » این‌کارها رو به صورت خودکار انجام می‌ده.

عنوان: **اطلاع از بروز رسانی نرم افزار ساخته شده**
نویسنده: رضابیات
تاریخ: ۱۳۹۲/۰۵/۱۶ ۱۴:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: Windows forms, xml, Software deployment, Check update application

برای شما هم پیش آمده که نرم افزاری را تهیه و منتشر کرده باشید و تمایل داشته باشید که استفاده کنندگان از وجود نسخه بروز شده مطلع شوند. یک راه ساده این است که اطلاعات نسخه جدید نرم افزار را داخل فایلی ذخیره کنیم و در وب سایت پشتیبانی نرم افزار قرار دهیم. حال بایستی اطلاعات این فایل را در زمان اجرای برنامه بررسی کنیم و در صورت وجود نسخه جدید از نرم افزار به کاربر اطلاع رسانی کنیم.
ابتدا فایل اطلاعات بروز رسانی نرم افزار را تهیه می‌کنیم و در وب سایت پشتیبانی نرم افزار قرار میدهیم. در اینجا از قالب Xml استفاده شده. که در آن Vertsion نسخه در دسترس نرم افزار است و URL هم مسیر وب سایت و یا فایل بروز رسانی است.

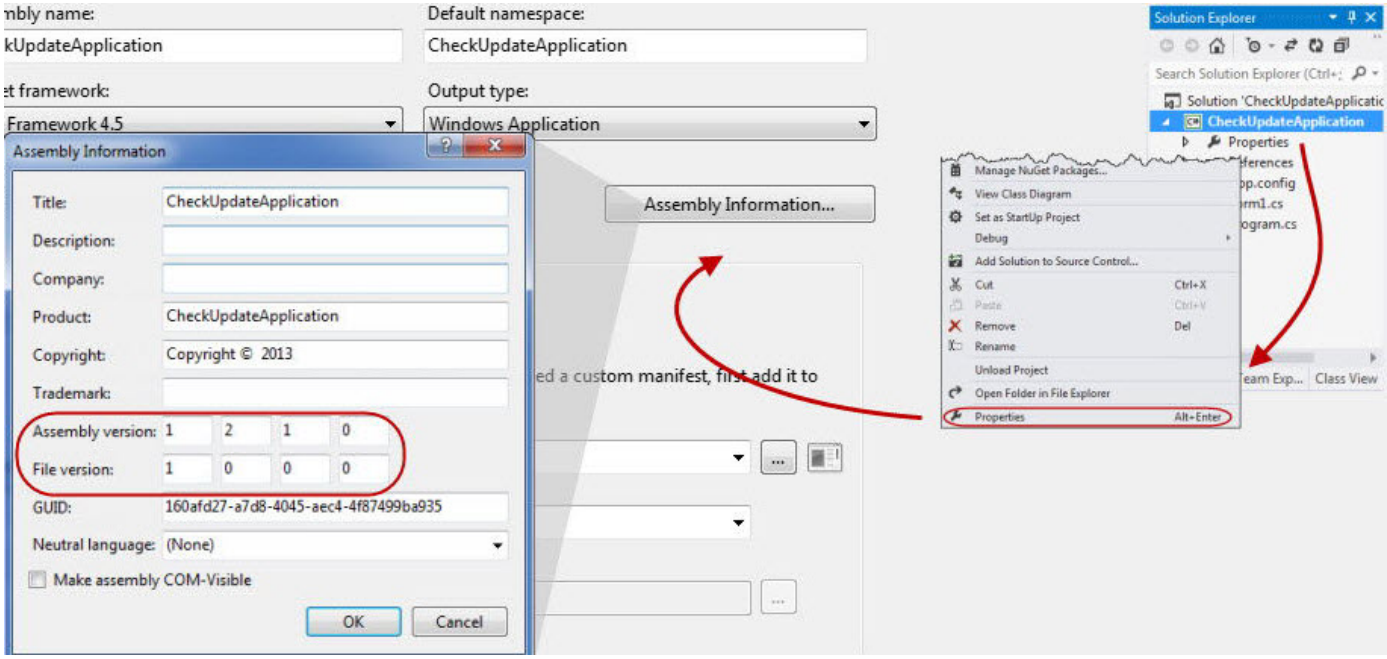
```
<?xml version="1.0" encoding="utf-8"?>
<AccountingApplication>
  <Version>1.5.2</Version>
  <URL>http://www.myappsupport.ir</URL>
</AccountingApplication>
```

نرم افزار را ساخته و کد زیر را در محل مناسبی کد نویسی می‌کنیم. این کد در ابتدا فایل Xml را خوانده و اطلاعات مورد نیاز را از آن دریافت می‌کند. سپس با استخراج نسخه اسمبلی برنامه و مقایسه این دو با هم از وجود نسخه جدید نرم افزار مطلع میشود.

```
...
using System.Xml;
namespace CheckUpdateApplication
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void CheckUpdate_Click(object sender, EventArgs e)
        {
            Version NewVersion = null;
            string DownloadPath = "";
            try
            {
                XmlTextReader xmlRead = new
XmlTextReader("http://www.myappsupport.ir/AccUpdateVersion.xml");
                xmlRead.MoveToContent();
                string elmName = "";
                if ((xmlRead.NodeType == XmlNodeType.Element) && (xmlRead.Name ==
"AccountingApplication"))
                {
                    while (xmlRead.Read())
                    {
                        if (xmlRead.NodeType == XmlNodeType.Element)
                        {
                            elmName = xmlRead.Name;
                        }
                        else
                        {
                            if ((xmlRead.NodeType == XmlNodeType.Text) && (xmlRead.HasValue))
                            {
                                switch (elmName)
                                {
                                    case "Version":
                                        NewVersion = new Version(xmlRead.Value);
                                        break;
                                    case "URL":
                                        DownloadPath = xmlRead.Value;
                                        break;
                                }
                            }
                        }
                    }
                }
                Version AppVertion =
```

```
System.Reflection.Assembly.GetExecutingAssembly().GetName().Version;
                if (AppVertion.CompareTo(NewVersion) < 0)
                {
                    DialogResult Result = MessageBox.Show("نسخه " +
                        NewVersion.Major.ToString() + "." +
                        NewVersion.Build.ToString() + " نسخه" ,"؟در دسترس میباشد مایل به دانلود هستید
",جدید
                        MessageBoxButtons.YesNo,MessageBoxIcon.Question);
                    if (Result == DialogResult.Yes)
                    {
                        System.Diagnostics.Process.Start(DownloadPath);
                    }
                }
                else
                {
                    MessageBox.Show("نرم افزار بروز میباشد");
                }
            }
            catch (Exception E)
            {
                MessageBox.Show(E.Message);
            }
        }
    }
}
```

به روش زیر هم نسخه اسمبلی برنامه را می‌شود تغییر داد.



سورس برنامه نمونه CheckUpdateApplicationSample.rar

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۱۶ ۱۸:۹

ممنون از شما. یک روش برای اینکه مستقیما با XML Reader کار نکنیم می‌تونه استفاده از روش‌های سریالایز کردن کلاس‌ها باشه. دردسرش کمتره.

یک سؤال: این فلش‌های انحنا دار رو با چه برنامه‌ای ایجاد کردید؟

نویسنده: رضابیات
تاریخ: ۱۳۹۲/۰۵/۱۷ ۰:۳۴

با تشکر از توجه و راهنمایی شما. نرم افزارهای زیادی برای این کار وجود داره ولی من خیلی وقته از Snagit و Snagit Editor استفاده میکنم. این نرم افزار بیشتر برای فیلم و عکس گرفتن از دسکتاب استفاده میشه ولی امکانات فراوانی دیگری هم در این نرم افزار وجود داره. من خودم نسخه Snagit 11.2.1 را از سایت http://www.softgozar.com دانلود کردم.

نویسنده: مصطفی
تاریخ: ۱۳۹۲/۰۶/۰۹ ۹:۳۴

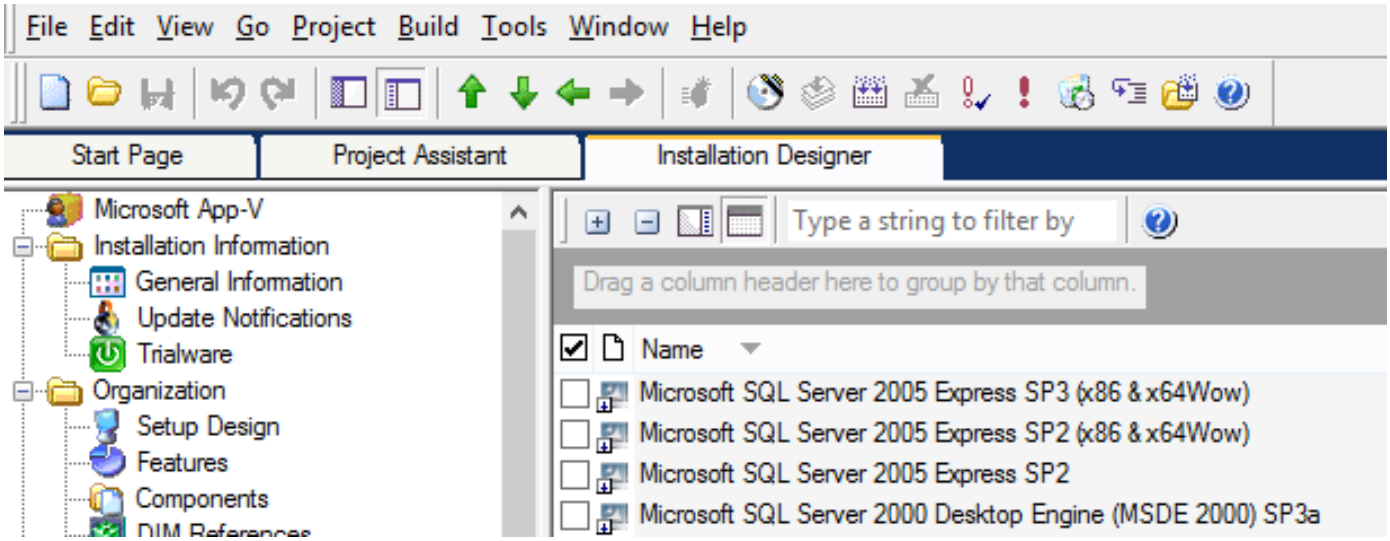
سلام
میخواستم بدونم فرق File Version و Assembly Version چیه؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۰۹ ۲۲:۱۲

Assembly Version برای مصرف کنندگان اسمبلی شما مهمه (و فقط در دنیای CLR دارای اهمیت هست). مثلا شخصی ارجاعی به اسمبلی نگارش خاصی داره. AssemblyFileVersion در قسمت خواص فایل در ویندوز قابل مشاهده است و بیشتر برای برنامه‌های ست آپ مفیده. اطلاعات بیشتر

عنوان: **افزودن SQL Server CE 4.0 به لیست توزیع مجدد در InstallShield 2010**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۲/۱۰/۲۵ ۲۱:۳۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server CE, Software deployment, Embedded Database, InstallShield, InstallShield-SetupPrerequisites

در برنامه‌ی ساخت نصاب InstallShield، در قسمت افزودن بسته‌های نصبی برای برنامه‌ی ساخته شده



بسته‌ی نصب SQL Server CE 3.5 SP2 وجود دارد:

اما برای برنامه‌های جدیدتر نیاز به افزودن بسته‌ی نصب دیتابیس SQL Server CE نسخه 4 است که با عدم وجود این بسته روبرو هستیم. در ادامه با نحوه‌ی افزودن این بسته‌ها آشنا خواهید شد.

اینگونه بسته‌ها در کنار برنامه‌ی ساخت نصاب و در پوشه‌ی SetupPrerequisites نگهداری شده و با نوع *.prq ذخیره می‌شوند. این نوع فایل‌ها از نوع xml هستند و در واقع یک نوع کار نگاشت را انجام می‌دهند. برای نمونه محتویات یکی از این فایل‌ها را در زیر می‌بینید:

```
<?xml version="1.0" encoding="UTF-8"?>
<SetupPrereq>
<conditions>
<condition Type="32" Comparison="2" Path="HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Microsoft SQL Server
Compact Edition\v3.5\ENU" FileName="DesktopRuntimeVersion" ReturnValue="3.5.8080.0"></condition>
</conditions>
<operatingsystemconditions>
<operatingsystemcondition MajorVersion="5" MinorVersion="0" PlatformId="2" CSDVersion=""
ServicePackMajorMin="3"></operatingsystemcondition>
<operatingsystemcondition MajorVersion="5" MinorVersion="1" PlatformId="2" CSDVersion="" Bits="1"
ProductType="1"></operatingsystemcondition>
<operatingsystemcondition MajorVersion="6" MinorVersion="0" PlatformId="2" CSDVersion=""
Bits="1"></operatingsystemcondition>
<operatingsystemcondition MajorVersion="5" MinorVersion="2" PlatformId="2" CSDVersion="" Bits="1"
ProductType="2|3"></operatingsystemcondition>
<operatingsystemcondition MajorVersion="6" MinorVersion="1" PlatformId="2" CSDVersion=""
```

```
Bits="1"></operatingsystemcondition>
<operatingsystemcondition MajorVersion="6" MinorVersion="0" PlatformId="2" CSDVersion="" Bits="1"
ProductType="2|3"></operatingsystemcondition>
</operatingsystemconditions>
<files>
<file LocalFile="&lt;ISProductFolder&gt;\SetupPrerequisites\SQL CE 3.5\SSCERuntime_x86-ENU.msi"
URL="http://go.microsoft.com/fwlink/?LinkId=166085&amp;clcid=0x409"
CheckSum="86AF6D36DFF214718DCD35D851249D3D" FileSize="0,3164160"></file>
</files>
<execute file="SSCERuntime_x86-ENU.msi" cmdline="/q /norestart" cmdlinesilent="/q /norestart"
returncodetoreboot="1641,3010,4123" requiresmsiengine="1"></execute>
<properties Id="{A7C4B3C0-F3A0-426A-A043-E13DBA123E52}" Description="This prerequisite installs the
Microsoft SQL Server Compact 3.5 SP2."
AltPrqURL="http://saturn.installshield.com/is/prerequisites/microsoft sql ce 3.5 sp2.prq"></properties>
<behavior Reboot="2"></behavior>
</SetupPrereq>
```

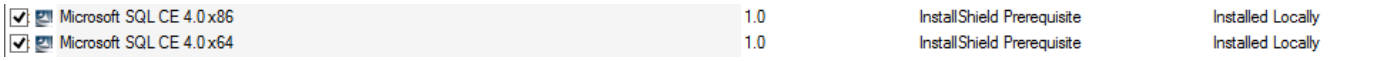
کافیست به ازای هر نسخه‌ی 32 و یا 64 بیتی، فایل xml مورد نظر، با پسوند prq در پوشه‌ی SetupPrerequisites ذخیره شود.
برای نسخه 32 بیتی(Microsoft SQL CE 4.0 x86.prq):

```
<?xml version="1.0" encoding="UTF-8"?>
<SetupPrereq>
    <conditions>
        <condition Type="32" Comparison="2" Path="HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Microsoft SQL
Server Compact Edition\v4.0\ENU" FileName="DesktopRuntimeVersion" ReturnValue="4.0.8482.1"></condition>
    </conditions>
    <operatingsystemconditions>
        <operatingsystemcondition CSDVersion="" Bits="1"></operatingsystemcondition>
    </operatingsystemconditions>
    <files>
        <file LocalFile=".\SSCERuntime_x86-ENU.exe"
URL="http://download.microsoft.com/download/0/5/D/05DCCDB5-57E0-4314-A016-874F228A8FAD/SSCERuntime_x86-
ENU.exe" CheckSum="0A55733CF406FBD05DFCFF5A27A0B4F7" FileSize="0,2379544"></file>
    </files>
    <execute file="SSCERuntime_x86-ENU.exe"></execute>
    <properties Id="{2754916B-119B-4428-9F94-DC9E45072CCC}"></properties>
    <behavior Failure="4" Reboot="2"></behavior>
</SetupPrereq>
```

و برای نسخه 64 بیتی(Microsoft SQL CE 4.0 x64.prq) :

```
<?xml version="1.0" encoding="UTF-8"?>
<SetupPrereq>
    <conditions>
        <condition Type="32" Comparison="2" Path="HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Microsoft SQL
Server Compact Edition\v4.0\ENU" FileName="DesktopRuntimeVersion" ReturnValue="4.0.8482.1"></condition>
    </conditions>
    <operatingsystemconditions>
        <operatingsystemcondition CSDVersion="" Bits="2"></operatingsystemcondition>
    </operatingsystemconditions>
    <files>
        <file LocalFile=".\SSCERuntime_x64-ENU.exe"
URL="http://download.microsoft.com/download/0/5/D/05DCCDB5-57E0-4314-A016-874F228A8FAD/SSCERuntime_x64-
ENU.exe" CheckSum="A417082ECAEDD95AFB41F73DC140C350" FileSize="0,2621240"></file>
    </files>
    <execute file="SSCERuntime_x64-ENU.exe"></execute>
    <properties Id="{7CB7BE3C-614A-403F-94D9-5652285A3EDF}"></properties>
    <behavior Failure="4" Reboot="2"></behavior>
</SetupPrereq>
```

و در نهایت این دو بسته به لیست اضافه خواهد شد:



نظرات خوانندگان

نویسنده: ایمان رضایی پور
تاریخ: ۱۳۹۲/۱۰/۲۷ ۱۱:۴۶

در اینجا بیشترین قابلیت‌ها را SQL CE 3.5 SP2 به خود اختصاص داده است. به نظر شما دلیل آن چیست؟
مگر SQL CE 4 جدیدتر نیست؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۷ ۱۳:۱

Features not supported in SQL Server Compact 4.0

علتش این است که مثلا LINQ to SQL دیگر با SQL CE 4 پشتیبانی نمی‌شود چون خود LINQ to SQL دیگر توسط MS توسعه جدی پیدا نمی‌کند و به EF سوئیچ شده و EF هم پروایدر رسمی برای SQL CE 4 داره.

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۲/۱۰/۲۸ ۸:۵۲

یکی دیگه از دلایلی که بیشتر از نسخه 3.5 صحبت میشه میتونه این باشه که نهایت نسخه ای است که در سری Windows CE و خصوصا 6.0 استفاده میشه. البته آخرین نسخه SQL Server Compact 3.5 SP2 هستش.

عنوان: **تهیه‌ی گزارش از آخرین ارائه‌های یک پروژه در GitHub**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۴/۰۱/۰۳ ۱۱:۲۰
آدرس: www.dotnettips.info
گروه‌ها: JSON, Software deployment, json.net, GitHub

چندی قبل مطلب « اطلاع از بروز رسانی نرم افزار ساخته شده » را در سایت جاری مطالعه کردید. در این روش بسیار متداول، شماره نگارش‌های جدید برنامه در یک فایل XML و مانند آن قرار می‌گیرند و برنامه هربار این فایل را جهت یافتن شماره‌های مندرج در آن اسکن می‌کند. اگر پروژه‌ی شما سورس باز است و در GitHub هاست شده، روش دیگری نیز برای یافتن این اطلاعات وجود دارد. در GitHub می‌توان از طریق آدرسی به شکل https://api.github.com/repos/user_name/project_name/releases به اطلاعات آخرین ارائه‌های یک پروژه (قرار گرفته در برگه‌ی releases آن) با فرمت JSON دسترسی یافت (یک مثال). در ادامه قصد داریم روش استفاده‌ی از آن‌را بررسی کنیم.

ساختار JSON ارائه‌های یک پروژه در GitHub

ساختار کلی اطلاعات ارائه‌های یک پروژه در GitHub چنین شکلی را دارد:

```
[
  {
    release_info,
    "assets": [
      {
      }
    ]
  }
]
```

در اینجا آرایه‌ای از اطلاعات ارائه‌ی یک پروژه ارسال می‌شود. هر ارائه نیز دارای دو قسمت است: لینکی به صفحه‌ی اصلی release در GitHub و سپس آرایه‌ای به نام assets که در آن اطلاعات فایل‌های پیوستی مانند نام فایل، آدرس، اندازه و امثال آن قرار گرفته‌اند.

تهیه‌ی کلاس‌های معادل فرمت JSON ارائه‌های برنامه در GitHub

اگر بخواهیم قسمت‌های مهم خروجی JSON فوق را تبدیل به کلاس‌های معادل دات نتی کنیم، به دو کلاس ذیل خواهیم رسید:

```
using Newtonsoft.Json;
using System;

namespace ApplicationAnnouncements
{
    public class GitHubProjectRelease
    {
        [JsonProperty(PropertyName = "url")]
        public string Url { get; set; }

        [JsonProperty(PropertyName = "assets_url")]
        public string AssetsUrl { get; set; }

        [JsonProperty(PropertyName = "upload_url")]
        public string UploadUrl { get; set; }

        [JsonProperty(PropertyName = "html_url")]
        public string HtmlUrl { get; set; }

        [JsonProperty(PropertyName = "id")]
        public int Id { get; set; }

        [JsonProperty(PropertyName = "tag_name")]
        public string TagName { get; set; }

        [JsonProperty(PropertyName = "target_commitish")]
        public string TargetCommitish { get; set; }

        [JsonProperty(PropertyName = "name")]
```

```
        public string Name { get; set; }

        [JsonProperty(PropertyName = "body")]
        public string Body { get; set; }

        [JsonProperty(PropertyName = "draft")]
        public bool Draft { get; set; }

        [JsonProperty(PropertyName = "prerelease")]
        public bool PreRelease { get; set; }

        [JsonProperty(PropertyName = "created_at")]
        public DateTime CreatedAt { get; set; }

        [JsonProperty(PropertyName = "published_at")]
        public DateTime PublishedAt { get; set; }

        [JsonProperty(PropertyName = "assets")]
        public Asset[] Assets { get; set; }
    }

    public class Asset
    {
        [JsonProperty(PropertyName = "url")]
        public string Url { get; set; }

        [JsonProperty(PropertyName = "id")]
        public int Id { get; set; }

        [JsonProperty(PropertyName = "name")]
        public string Name { get; set; }

        [JsonProperty(PropertyName = "label")]
        public string Label { get; set; }

        [JsonProperty(PropertyName = "content_type")]
        public string ContentType { get; set; }

        [JsonProperty(PropertyName = "state")]
        public string State { get; set; }

        [JsonProperty(PropertyName = "size")]
        public int Size { get; set; }

        [JsonProperty(PropertyName = "download_count")]
        public int DownloadCount { get; set; }

        [JsonProperty(PropertyName = "created_at")]
        public DateTime CreatedAt { get; set; }

        [JsonProperty(PropertyName = "updated_at")]
        public DateTime UpdatedAt { get; set; }
    }
}
```

در اینجا از ویژگی JsonProperty جهت معرفی نام‌های واقعی خواص ارائه شده‌ی توسط GitHub استفاده کرده‌ایم.
پس از تشکیل این کلاس‌ها، مرحله‌ی بعد، دریافت اطلاعات JSON از آدرس API ارائه‌های پروژه در GitHub و سپس نگاشت آن‌ها می‌باشد:

```
using (var webClient = new WebClient())
{
    webClient.Headers.Add("user-agent", "DNTProfiler");
    var jsonData = webClient.DownloadString(url);
    var gitHubProjectReleases = JsonConvert.DeserializeObject<GitHubProjectRelease[]>(jsonData);

    foreach (var release in gitHubProjectReleases)
    {
       foreach (var asset in release.Assets)
       {
           // …
       }
    }
}
```

حین کار با WebClient یا هر روش دیگری که برای دریافت اطلاعات از وب استفاده می‌کنید، حتما نیاز است user-agent را ذکر

کرد. در غیر اینصورت GitHub درخواست شما را برگشت خواهد زد. پس از دریافت اطلاعات JSON، با استفاده از متد JsonConvert.DeserializeObject کتابخانه‌ی JSON.NET ، می‌توان آن‌ها را تبدیل به آرایه‌ای از GitHubProjectRelease کرد.

یک نکته: اگر به صفحه‌ی اصلی ارائه‌های یک پروژه در GitHub دقت کنید، شماره‌ی تعداد بار دریافت یک ارائه مشخص نشده‌است. در این API، عدد DownloadCount، بیانگر تعداد بار دریافت پروژه‌ی شما است.