عنوان: **اجرای وظایف زمان بندی شده با Quartz.NET - قسمت اول**
نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۲۴ ۱۰:۱۳
آدرس: www.dotnettips.info
گروه‌ها: Quartz.NET, Schedule, زمانبندی

مقدمه

اگر قصد اجرای برخی کارها به صورت زمانبندی شده و در فواصل زمانی مشخص را دارید، این مقاله به شما کمک خواهد کرد تا به بهترین شکل ممکن آن را انجام دهید. کارهایی مانند ارسال خبرنامه، فرستادن SMS تبریک تولد یا هماهنگ سازی داده‌ها بین دو منبع داده از جمله اعمالی هستند که باید به صورت زمانبندی شده انجام شوند.

کتابخانه‌ی Quartz.NET، از کتابخانه ای با نام Quartz و از زبان Java به NET. منتقل شده است. Quartz.NET، رایگان و باز متن است و از طریق آدرس http://quartznet.sourceforge.net در دسترس است. از طریق NuGet نیز می‌توانید با تایپ عبارت quartz در فرم مربوطه، این کتابخانه را نصب کنید. این کتابخانه را در برنامه‌های Desktop و Web (حتی یک Shared Server) تست کردم و به خوبی انجام وظیفه می‌کند.

شروع کار با Quartz.NET

ضمن در اختیار قرار دادن امکانات فوق العاده و انعطاف پذیری بسیار، کار با این کتابخانه آسان و از فرایندی منطقی تبعیت می‌کند. فرایند اجرای یک روال زمانبندی شده از طریق Quartz.NET، از چهار مرحله‌ی اصلی تشکیل شده است.

1) پیاده سازی اینترفیس IJob
2) مشخص کردن جزئیات روال با اینترفیس IJobDetail
3) مشخص کردن تنظیمات زمان با استفاده از اینترفیس ITrigger
4) مدیریت اجرا با استفاده از اینترفیس IScheduler

مثالی را بررسی می‌کنیم. در این مثال قصد داریم تا عبارتی را همراه با تاریخ و زمان جاری در یک فایل ذخیره کنیم. این پیغام باید 3 بار و در فواصل زمانی 10 ثانیه به فایل اضافه شود. در پایان، فایلی خواهیم داشت که در سه خط، یک عبارت، همراه با تاریخ و زمان‌های مختلف را که 10 ثانیه با یکدیگر اختلاف دارند در خود ذخیره کرده است. ابتدا کار زمانبندی شده را با ارائه‌ی پیاده سازی برای متد Execute اینترفیس IJob این کتابخانه ایجاد می‌کنیم. وارد کردن فضای نام Quartz را فراموش نکنید.

```
namespace SchedulerDemo.Jobs
{
    using System;
    using System.IO;
    using Quartz;

    public class HelloJob : IJob
    {
        public void Execute(IJobExecutionContext context)
        {
            // for web apps
            // string path = System.Web.Hosting.HostingEnvironment.MapPath("~/Data/Log.txt");

            // for desktop apps
            string path = @"C:\Log.txt";

            using (StreamWriter sw = new StreamWriter(path, true))
            {
                sw.WriteLine("Message from HelloJob " + DateTime.Now.ToString());
            }
        }
    }
}
```

در اینترفیس IJob در ASP.NET، به شی HttpContext دسترسی ندارید، بنابراین در صورتی که قصد داشته باشید از متدی مانند Server.MapPath استفاده کنید، توفیقی به دست نخواهید آورد. در عوض می‌توانید از متد System.Web.Hosting.HostingEnvironment.MapPath استفاده کنید.

حال، زمان انجام تنظیمات مختلف برای اجرای روال مربوطه است. بهتر است تا interfaceیی ایجاد و متدی با نام Run در آن

داشته باشیم.

```
namespace SchedulerDemo.Interfaces
{
    public interface ISchedule
    {
        void Run();
    }
}
```

حال، پیاده سازی خود را برای این interface ارائه می‌دهیم.

```
namespace SchedulerDemo.Jobs
{
    using System;
    using Quartz;
    using Quartz.Impl;
    using SchedulerDemo.Interfaces;
    using SchedulerDemo.Jobs;

    public class HelloSchedule : ISchedule
    {
        public void Run()
        {
            //DateTimeOffset startTime = DateBuilder.NextGivenSecondDate(null, 2);
            DateTimeOffset startTime = DateBuilder.FutureDate(2, IntervalUnit.Second);

            IJobDetail job = JobBuilder.Create<HelloJob>()
                                       .WithIdentity("job1")
                                       .Build();

            ITrigger trigger = TriggerBuilder.Create()
                                             .WithIdentity("trigger1")
                                             .StartAt(startTime)
                                             .WithSimpleSchedule(x =>
x.WithIntervalInSeconds(10).WithRepeatCount(2))
                                             .Build();

            ISchedulerFactory sf = new StdSchedulerFactory();
            IScheduler sc = sf.GetScheduler();
            sc.ScheduleJob(job, trigger);

            sc.Start();
        }
    }
}
```

معرفی فضاهای نام Quartz و Quartz.Impl را فراموش نکنید.
از حالا، به روالی که قرار است به صورت زمانبندی شده اجرا شود، "وظیفه" می‌گوییم.
ابتدا باید مشخص کنیم که وظیفه در چه زمانی پس از اجرای برنامه شروع به اجرا کند. از آنجا که پایه و اساس زمانبندی، بر تاریخ و ساعت استوار است، کتابخانه‌ی Quartz.NET، روش‌ها و امکانات بسیاری را برای تعیین زمان در اختیار قرار می‌دهد. با بررسی تمامی آنها، ساده‌ترین و منعطف‌ترین را به شما معرفی می‌کنم. کلاس DateBuilder که همراه با Quartz.NET وجود دارد، امکان تعیین زمان را به اشکال مختلف می‌دهد. در خط 14، از متد FutureDate این کلاس استفاده شده است که خوانایی بهتری نسبت به بقیه‌ی متدها دارد. پارامتر اول این متد، عدد، و پارامتر دوم، واحد زمانی را می‌پذیرد.

```
DateTimeOffset startTime = DateBuilder.FutureDate(2, IntervalUnit.Second);
```

در اینجا، زمان آغاز وظیفه را 2 ثانیه پس از آغاز برنامه تعریف کرده ایم. واحدهای زمانی دیگر شامل میلی ثانیه، دقیقه، ساعت، روز، ماه، هفته و سال هستند. کلاس DateBuilder، متدهای مختلفی برای تعیین زمان را در اختیار قرار می‌دهد. تعیین زمان آغاز به روش دیگر را به صورت کامنت شده در خط 13 مشاهده می‌کنید.
وظیفه‌ی ایجاد شده در خط 16 تا 18 معرفی شده است.

```
IJobDetail job = JobBuilder.Create<HelloJob>()
                           .WithIdentity("job1")
                           .Build();
```

پشتیبانی Quartz.NET از سینتکس fluent، کدنویسی را ساده و لذت بخش می‌کند. با استفاده از متد Create کلاس JobBuilder، وظیفه را معرفی می‌کنیم. متد Create، یک متد Generic است که نام کلاسی که اینترفیس IJob را پیاده سازی کرده است می‌پذیرد. یک نام را با استفاده از متد WithIdentity به وظیفه نسبت می‌دهیم (البته این کار، اختیاری است) و در انتها، متد Build را فراخوانی می‌کنیم. خروجی متد Build، از نوع IJobDetail است.
و حالا نوبت به تنظیمات زمان رسیده است. در Quartz.NET، این مرحله، "ایجاد trigger" نام دارد. خطوط 20 تا 24 به این کار اختصاص دارند.

```
ITrigger trigger = TriggerBuilder.Create()
                                 .WithIdentity("trigger1")
                                 .StartAt(startTime)
                                 .WithSimpleSchedule(x =>
x.WithIntervalInSeconds(10).WithRepeatCount(2))
                                 .Build();
```

ابتدا متد Create کلاس TriggerBuilder را فراخوانی می‌کنیم، سپس با استفاده از متد WithIdentity، یک نام به trigger اختصاص می‌دهیم (البته این کار، اختیاری است). با متد StartAt، زمان شروع وظیفه را که در ابتدا با استفاده از کلاس DateBuilder ایجاد کردیم تعیین می‌کنیم. مهمترین قسمت، تعیین دفعات و فواصل زمانی اجرای وظیفه است. همان طور که احتمالاً حدس زده اید، Quartz.NET مجموعه ای غنی از روش‌های مختلف برای تعیین بازه‌ی زمانی اجرا را در اختیار قرار می‌دهد. آسان‌ترین راه، استفاده از متد WithSimpleSchedule است. با استفاده از یک عبارت Lambda که ورودی آن از نوع کلاس SimpleScheduleBuilder است، دفعات و فواصل زمانی اجرا را تعیین می‌کنیم. متد WithIntervalInSeconds، برای تعیین فواصل زمانی در بازه‌ی ثانیه استفاده می‌شود. متد WithRepeatCount نیز برای تعیین دفعات اجرا است. وظیفه‌ی ما، 3 مرتبه و در فواصل زمانی 10 ثانیه اجرا می‌شود. مطمئن باشید اشتباه نکردم! بله، سه مرتبه. تعداد دفعات اجرا برابر است با عددی که برای متد WithRepeatCount تعیین می‌کنید، به علاوه‌ی یک. منطقی است، چون مرتبه‌ی اول اجرا زمانی است که با استفاده از متد StartAt تعیین کرده اید. در پایان، متد Build را فراخوانی می‌کنیم. خروجی متد Build، از نوع ITrigger است.
آخرین کار (خطوط 26 تا 30)، ایجاد شی از اینترفیس IScheduler، فراخوانی متد ScheduleJob آن، و پاس دادن اشیای job و trigger که در قسمت قبل ایجاد شده اند به این متد است. در انتها، متد ()Start را برای آغاز وظیفه فراخوانی می‌کنیم.

```
ISchedulerFactory sf = new StdSchedulerFactory();
IScheduler sc = sf.GetScheduler();
sc.ScheduleJob(job, trigger);

sc.Start();
```

حال شما یک وظیفه تعریف کرده اید که در هر جای برنامه به صورت زیر، قابل فراخوانی است.

```
ISchedule myTask = new HelloSchedule();
myTask.Run();
```

کتابخانه ای که با آن سر و کار داریم بسیار غنی است و امکانات بسیاری دارد. در قسمت بعد، با برخی امکانات دیگر این کتابخانه آشنا می‌شوید.

نظرات خوانندگان

نویسنده: محمد
تاریخ: ۱۳۹۱/۰۵/۲۴ ۱۳:۱۱

با سلام و تشکر از مطب مفید شما

من در مورد کارهای زمان بندی شده قبلا سرچ کردم و به این نتیجه رسیدم که دات نت تا قبل از ورژن 4 ،استانداردی برای انجام کارهای زمان بندی شده نداشت ولی در ورژن 4 فضای نام system.threading.tasks.taskscheduler را ارائه داده است. در صورت امکان تفاوت‌های میان این استاندارد و مطالب فوق را بیان کنید

با تشکر مجدد

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۲۴ ۱۵:۴۴

System.Threading.Tasks.TaskScheduler، یک کلاس abstract هست. باید برای اون پیاده سازی صورت بگیره و به خودیه خود، کار خاصی انجام نمیده. برای اجرای وظایف به طور همزمان یا ایجاد یک صف از وظایف استفاده میشه، یا میشه با استفاده از اون، Thread جاری رو ذخیره کرد تا بعداً بشه در هنگام پایان یافتن کار یک Thread، به کنترل‌های UI از طریق اون دسترسی داشت. در مبحث این مقاله جایی نداره.

نویسنده: محمد
تاریخ: ۱۳۹۱/۰۵/۲۴ ۱۶:۴۱

ممنون از شما

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۵/۲۵ ۱:۱۰

جناب مهندس راد سلام
روشی که اینجا شما فرمودین نمونش قبلا حدود 4 سال پیش من دیدم که آقای کیوان نیری توی سایتشون اموزش داده بودن (روشی شبیه همین روش با انعطاف پذیری بالا) اما مشکلی که توی این روش برای وب وجود داره اناجام کارهایی هست که باید حتما انجام بشه مثل واریز سود بانکی، در صورتی که سرور ری استارت بشه اطلاعات جاب‌های ما از بین میره
که البته من خودم روش اقای نیری با دیتابیس ترکیب کردم و میشه گفت بدون اشکال چند سالی هست داره کار میکنه
اگر این روش که فرمودین راهی داره برای ری استارت شدن وب ممنون می‌شم شرح بدین

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۲۵ ۹:۱۳

سلام برادر.
Quartz.NET رو دست کم نگیرید. بسیار بسیار قدرتمنده. این کتابخانه برای ردیابی وظایف از مفهومی با عنوان JobStore استفاده می‌کنه. به طور پیش فرض داده‌های مربوط به ردیابی وضعیت اجرای وظیفه‌ها در حافظه‌ی RAM قرار می‌گیرند که Quartz.NET اون رو به عنوان RAMJobStore میشناسه. اما برای سناریوهایی همانند آنچه که شما گفتید، پشتیبانی از پیش تعبیه شده برای ذخیره‌ی داده‌های ردیابی رو در پایگاه داده هم فراهم می‌کنه و این قابلیت رو با عنوان AdoJobStore میشناسه.
این کتابخانه در حال حاضر از پایگاه‌های داده‌ی SQLite، MySql، Oracle، SqlServer و Firebird پشتیبانی می‌کنه . فایل اسکریپت برای تولید جداول مورد نیاز در بسته‌ی دانلودی اون هست.

در ضمن، Quartz.NET با خودش سرویسی به همراه داره که میتونه به عنوان سرویس‌های سیستم عامل معرفی بشه تا هرگاه سیستم بالا اومد، اون سرویس به طور خودکار وظیفه‌ها رو از یک فایل با فرمت XML میخونه و اجرا می‌کنه.

در مورد 4 سال پیش که فرمودید، بله بنده هم پیاده سازی‌های مختلفی دیدم. حتی یکی از هموطنان، با استفاده از Cache، برای برای برنامه‌های ASP.NET، قابلیت زمانبندی (هر چند بسیار محدود) ارائه داده بود. استفاده از Timer هم روتین‌ترین چیزی هست که به ذهن همه میرسه اما در قسمت پرسش و پاسخ سایت Quartz.NET، دلایل مطلوب نبودن استفاده‌ی صِرف از Timer در قسمت "Why not just use System.Timers.Timer?" گفته شده.

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۵/۲۵ ۱۰:۳۶

ممنونم از پاسخ شما
فکر کنم از کد تایمر منظورتون پیاده‌سازی آقای سالار خلیلی هست دیگه؟
اگه اینجوری باشه خیلی عالیه مستنداتشو خوندم جالبه
ممنون از پست مفیدتون

نویسنده: بهمن خلفی
تاریخ: ۱۳۹۱/۰۵/۲۵ ۱۲:۴۳

با سلام و عرض ادب و سپاس فراوان
جناب راد از حوصله ای که به خرج دادید و مطلب درخواستی را با توضیحات بسیار کامل مطرح نمودید تشکر و سپاسگذارم
من قبلا از کد آقای نیری استفاده کردم خوب بود ولی کافی نیست یه سری اشکالات ریزی داره که نمیشه برای کارهای بزرگ و حساس استفاده کرد
ولی با خواندن مطلب شما و مراجعه به رفرنس مطلب مطمئن شدم که این امکان بسیار قدرتمند و قابل مقایسه با موارد مشابه خودش نیست
امیدوارم که همواره کامیاب باشید

نویسنده: علی معصومی
تاریخ: ۱۳۹۱/۰۵/۲۵ ۲۳:۵۷

سلام من یک وبسایتی د ارم که در فواصل زمانی مشخص از وب سرویس پیامک‌های واردرو دریافت میکنه و روی اونا پردازش انجام میده ادر حال حاضر این کارو دارم به وسیله یک سرویس و تایمر روی یک سرور اختصاصی انجام میدم که هزینه بالایی داره برام به نظر شما این کارو میتونم با همین که شما آموزش دادید انجام بدم؟

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۲۶ ۱۱:۱۴

بله، می‌تونید.

نویسنده: مهران
تاریخ: ۱۳۹۱/۰۷/۱۰ ۱۶:۵۳

سلام
یک کلاس با اسم Abzar وجود دارد میخواهم متدی که با اسم ADD() در این کلاس وجود دارد در هر پنج دقیقه یک بار اجرا شود.
توجه داشته باشید که در Page-Load نباید باشد.
و این برنامه برای یک وب سایت است

کاربر هیچ کاری با این کلاس و متد ندارد و هیچ وقت در صفحه ای باز نمیشود. ولی این متد در طول شبانه روز در هر 5 دقیقه یک بار اجرا میشود

آیا با **Quartz.NET میتوان این کار را انجام داد ؟**

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۷/۱۱ ۱۱:۲۱

شما باید از قوانین استفاده از این کتابخانه پیروی کنید. پیاده سازی اینترفیس‌های لازم برای استفاده از این کتابخانه ضروری هست...
کاری که قصد دارید انجام بدید کمترین کاریه که Quartz.NET می‌تونه انجام بده. در نقطه‌ی آغاز برنامه (روال Application_Start فایل Global.asax)، می‌تونید تعیین کنید که شروع اجرا در تاریخ و ساعت خاصی باشه و در بازه‌های زمانی مشخصی اجرا بشه.
ضمناً، این کتابخانه از فرمت cron هم برای تعیین زمان پشتیبانی می‌کنه که اون رو خیلی قدرتمند می‌کنه.

نویسنده: مهران
تاریخ: ۱۳۹۱/۰۷/۱۱ ۲۱:۱

این کتابخانه کمی ناشناخته است و مطلب زیادی در موردش پیدا نکردم، آن چیزی هم که هست بیشتر انگلیسی است که بیشتر گیج کننده است.
اگر لطف کنید دقیق‌تر (با جزییات) بفرمایید چه کار باید بکنم ممنون میشوم .

یا مثالی با توجه به توضیحی که دادم بگذارید خیلی ممنون میشوم.

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۷/۱۲ ۹:۳۲

به نظر من، دقیق‌تر از این امکان توضیح وجود نداره.

نویسنده: جلال
تاریخ: ۱۳۹۱/۰۸/۱۵ ۲۰:۲۰

سلام.
ممنون بابت معرفی این کتابخانه‌ی مفید. جایی فرمودید که « *در اینترفیس IJob در ASP.NET، به شی HttpContext دسترسی ندارید.* » که درست نیست. شما در همه جای برنامه ASP.NET به کمک HttpContext.Current به HttpContext جاری دسترسی دارید. آیا استفاده از این روش مشکل خاصی داره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۱۵ ۲۱:۲۲

نه در همه جای یک برنامه ASP.NET. در یک سری از روال‌های global.asax.cs دسترسی به HttpContext وجود ندارد. روال‌هایی که دخالتی در پردازش درخواست وب جاری ندارند (هرجایی که Request context وجود نداشته باشد).
از همین روال‌ها هم اتفاقا برای اجرا و آغاز jobها در وب استفاده می‌شوند.

نویسنده: mahdi1391
تاریخ: ۱۳۹۱/۰۹/۱۲ ۱۶:۴۷

با سلام خدمت دوستان و اساتید عزیز. ممنون از این مقالهٔ ارزشمند، واقعاً عالی و کاربردی است.
من دقیقاً با مشکلی که در یکی از بندهای این کامنت (جناب بهروز راد) به آن اشاره کردند، مواجه هستم. مشکلی که من دارم در Shared Server است.
فرض کنید یک وظیفه در حال اجرا است، و در هاست‌های اشتراکی چندین بار در روز اتفاق می‌افتد که IIS ریستارت شود، Recycling انجام شود یا حتی سرور ریستارت شود؛ لطفاً در این مورد بیشتر توضیح بفرمائید که چطور آن وظیفه را دوباره به اجرا

دربیاورم.
فعلاً مجبورم آن وظیفه را دستی فراخوانی کنم.
ممنون.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۱۲ ۱۷:۲۳

یک فید برای سایت درست کنید. این فید را در فیدبرنر ثبت کنید. مابقی آن خودکار است. دیگر گوگل دست از سر شما برنخواهد داشت!

نویسنده: mahdi1391
تاریخ: ۱۳۹۱/۰۹/۱۲ ۲۰:۱۶

با تشکر از پاسخ شما.
شاید بهتر باشد مشکل را بگویم تا شاید سناریوی بهتری برای آن وجود داشته باشد.
در هاست‌های اشتراکی (مخصوصلاً هاستینگ‌های داخل کشور)، اگر مدت کوتاهی کسی به سایت شما سر نزند، به صورت خودکار سایت از حافظه خارج میشود (که دلایل آن واضح است) و پس از درخواست نمایش صفحه، حداقل 30 ثانیه زمان لازم است تا دوباره چرخهٔ اجرا انجام و صفحه اولیه نمایش داده شود. حال من دنبال راهی هستم که همیشه سایت در حال اجرا باشد.
راه حل هایی مثل این را پیدا کردم که برای هاست‌های اشتراکی کاربرد ندارد.
ممنون.

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۹/۱۲ ۲۰:۵۳

این روش رو تست کردی؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۱۲ ۲۰:۵۷

فیدبرنر ساعتی چندبار به سایت شما سر می‌زند. این سر زدن یعنی ارسال درخواست به سایت. در این حالت اگر سایت درون حافظه هم نباشد، مجددا اجرا خواهد شد.

نویسنده: mahdi1391
تاریخ: ۱۳۹۱/۰۹/۱۲ ۲۱:۱۱

ممنون از جناب نصیری و جناب صاحب بابت پاسخ گویی.
تست میکنم.
با تشکر.

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۹/۱۳ ۱۰:۵

من از Pingdom استفاده می‌کنم. می‌تونید فواصل زمانی پینگ از سایت رو برای این سرویس روی 5 دقیقه تنظیم کنید.

نویسنده: محسن موسوی
تاریخ: ۱۳۹۱/۰۹/۱۳ ۱۴:۵۱

دوست عزیز من قبلا از این روش استفاده کردم و جواب گرفتم:به این ترتیب که
در global

```
void Application_Start(object sender, EventArgs e)
```

```
    {
        // Schedule Start

    }

 void Application_End(object sender, EventArgs e)
    {
        //  Code that runs on application shutdown
        var   scheduler = new StdSchedulerFactory().GetScheduler();
        scheduler.Shutdown(true);
```

```
DoWebRequest ("SiteAddress");
 }
```

```
public string DoWebRequest(string url)
  {
WebRequest request = WebRequest.Create(url);
request.Credentials = CredentialCache.DefaultCredentials;
HttpWebResponse response = (HttpWebResponse)(request.GetResponse());
Stream dataStream = response.GetResponseStream();
StreamReader reader = new StreamReader(dataStream);
string responseFromServer = reader.ReadToEnd();
reader.Close();
dataStream.Close();
response.Close();
return responseFromServer;
}
```

نویسنده: مسعود زیانی
تاریخ: ۱۳۹۱/۰۹/۱۴ ۱۶:۲۰

با سلام و تشکر بابت مطلب عالیتون
من میخوام به تعداد بالا پیامک ارسال کنم و هر اگه هر کدوم ناموفق بود مثلا هر 3 ثانیه باز تا مثلا 4 بار تلاش کنه که باز پیامک رو ارسال کنه...اینکار رو با تایمر انجما میدم اما متاسفانه تو تعداد بالا سرورم استاپ میشه و فقط در تعدادهای کم مثلا زیر 300 تا درست کار میکنه
با این کلاس شما اینکار من کاملا قابل هندل هست؟

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۹/۱۵ ۷:۷

مشکل توقف Server شما ارتباطی با این بحث نداره و باید در روش و کدهایی که استفاده می‌کنید دلیل رو جستجو کنید.
اما برای ارسال زمانبندی شده‌ی SMS، بله می‌تونید از این کتابخانه استفاده کنید.

نویسنده: مسعود زیانی
تاریخ: ۱۳۹۱/۰۹/۱۶ ۲۳:۱۵

خیلی ممنون

نویسنده: محمد حسین فخراوری
تاریخ: ۱۳۹۱/۱۱/۲۳ ۱۵:۲۳

با سلام
اقای راد چرا برنامه روی هاست بعد مدتی از کار میافته.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۲۳ ۱۷:۲۵

لطفا نظرات همین مطلب رو از ابتدا مطالعه کنید؛ بحث شده قبلا.

نویسنده: محمد حسین فخراوری
تاریخ: ۱۳۹۱/۱۱/۲۴ ۱۱:۵۵

اقای نصیری در سایت pingdom ایا ثبتنامش پولی؟
و http://feedburner.google.com چطوری ثبتنام کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۲۴ ۱۲:۲۹

- فیدبرنر جزو خدمات گوگل است. همینقدر که یک اکانت گوگل یا جی‌میل دارید به خدمات فیدبرنر هم دسترسی خواهید داشت.
- برای سرویس‌های دیگر که به سایت ping ارسال می‌کنند، از جستجوی گوگل شروع کنید.

نویسنده: محمد حسین فخراوری
تاریخ: ۱۳۹۱/۱۱/۲۴ ۱۳:۳۵

ممنون اقای نصیری .
یکی از دوستان روشی گفتن publicstringDoWebRequest(stringurl) در Application_End
ایا این روش هم درسته و میتوان روش حساب کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۲۴ ۱۳:۳۹

خیر. سرور می‌تونه ری استارت یا خاموش بشه. امکان خیلی از اتفاقات دیگر هم هست که نیاز به راه اندازی مجدد و رسیدن اولین درخواست رو داره. در کل روش ارسال ping به سرور از طریق یک برنامه خارجی مطمئن‌ترین است.

نویسنده: محمد حسین فخراوری
تاریخ: ۱۳۹۱/۱۱/۲۷ ۱۳:۲۱

با سلام
دوستان اگر ممکن راهنمای ثبت سایت در این 2 سایت بالا که اقای نصیری گفتن بزارین.
هر کاری میکنم نمیشه

نویسنده: محسن
تاریخ: ۱۳۹۱/۱۱/۲۸ ۱:۳

از سایت پالس‌می هم می‌تونی استفاده کنی.

نویسنده: علی
تاریخ: ۱۳۹۲/۰۴/۳۱ ۱۵:۲۷

بعد از اجرا این پیغام رو میده:

Could not load file or assembly 'Common.Logging' or one of its dependencies. This assembly is built by a runtime newer than the currently loaded runtime and cannot be loaded.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۳۱ ۱۶:۵۷

پیغام خطایی که دریافت کردید به این معنا است که اسمبلی دریافتی با یک شماره بالاتر دات نت کامپایل شده و در شماره نگارش مدنظر شما که پایین‌تر هست قابل استفاده نیست.
بهتر است سورس کد کتابخانه را دریافت و کامپایل یا اسمبلی‌های قدیمی‌تر این کتابخانه را دریافت کنید.

نویسنده: سید محمد
تاریخ: ۱۳۹۲/۰۷/۰۶ ۱۰:۲

بنده یک مشکل عجیب با quartz پیدا کردم
من یک schedule برای ایمیل نوشتم که بتونه ایمیل انبوه با فاصله زمانی ارسال کند

منتها من برای اینکه ارسال ایمیل شود باید لیست ایمیل‌ها را در هر بار به job بفرستم و با هر بار فرستادن , آن ایمیل از لیست ایمیل‌ها پاک شود .برای اینکه بتوانم لیست ایمیل‌ها را با هر بار اجرای job حفظ کنم که متوجه شوم چه ایمیل هایی مانده است از اتریبیوت PersistJobDataAfterExecution و DisallowConcurrentExecution بالای سر job استفاده کردم .
در job گفتم اگر تعداد لیست ایمیل‌ها به صفر رسید schedule متوقف شود
در لوکال مشکلی ندارد ولی در عملی متوجه شدم گویا مقدار لیست ایمیل‌ها حفظ نمی‌شود و مجدد ایمیل زده می‌شود.لطفا کمک کنید

```
[PersistJobDataAfterExecution]
   [DisallowConcurrentExecution]
   public class SendGroupEmailJob : IJobBase
   {
       private List<MailAddress> lstMails;

       public void Execute(IJobExecutionContext context)
       {
           int result = 0;
           if (context.JobDetail.JobDataMap["UserEmailList"] != null)
           {
               lstMails = context.JobDetail.JobDataMap["UserEmailList"] as List<MailAddress>;

               if (lstMails.Count == 0)
               {

                   context.Scheduler.UnscheduleJob(new TriggerKey(context.Trigger.Key.Name));


               }
               else
               {

                   JobDataMap map = context.JobDetail.JobDataMap;
                   result = EmailHandler.Send(lstMails[0],
                       map.GetString("Subject"),
                      map.GetString("Body").Replace("[FullName]",
lstMails[0].DisplayName).Replace("[Email]", lstMails[0].Address),
                     context.JobDetail.JobDataMap["Attachment"] as List<string>,
                        MailPriority.High,
                        true,
                        Encoding.UTF8,
                        DeliveryNotificationOptions.None,
                        map.GetString("SenderEmail"),
                        map.GetString("SenderName"),
                        map.GetString("BccEmail"),
                        map.GetString("Prefix"),
                        map.GetBoolean("IsSSL"),
                        map.GetBoolean("IsCredential"),
                        map.GetString("Server"),
                        map.GetInt("Port"),
                        map.GetInt("TimeOut"),
                        map.GetString("PassWord"));
                   lstMails.RemoveAt(0);

               }


           }

       }
   }
```

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۷/۰۶ ۱۰:۳۹

lstMails رو استاتیک تعریف کن. به علاوه هنگام بررسی count آن یک lock هم باید قرار بدی. ضمنا بررسی count آن‌را ببر قبل از انتساب به آن. الان داری یک لیست کامل رو به یک لیست به ظاهر موجود انتساب می‌دی بعد مقدار تعداد آیتم‌هاش رو حساب می‌کنی. خوب، این مقدار همیشه مساوی است با تعداد لیست جدید انتساب داده شده. همچنین برنامه‌های وب ممکن است به دلایل زیادی ری‌استارت شوند؛ یعنی از دست رفتن تمام متغیرهای درون حافظه. بنابراین بهتر است از دیتابیس استفاده کنید.

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۹/۰۳ ۱۲:۰

لطفا یک نمونه سمپل بذارید

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۹/۰۳ ۱۲:۲۲

یک سری به سورسش بزنید. پر است از مثال .

نویسنده: عصاری
تاریخ: ۱۳۹۲/۱۰/۲۶ ۱۴:۴۳

با سلام؛ من برای ارسال پیامک‌های زمانبندی شده، از همین روش استفاده کردم، اما یک مشکل دارم.
اونم اینه که در صورتی که چند job به طور همزمان اجرا بشن تداخل پیدا میکنن. برای توضیح بیشتر، زمانی که چند job رو با عبارت cron یکسان ثبت میکنم -> مثلا:

```
0 33 10 * * ?
```

پس از اجرا و ارسال پیام که باید تراکنش مالی اون ثبت بشه(لازم به توضیح است که من عملیات ارسال پیام و ثبت تراکنشات مالی رو در متد execute نوشتم)، ثبت نمیشه و اجرای بقیه jobها ادامه پیدا میکنه. ممنون میشم بنده رو راهنمایی کنین.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۷ ۱:۳۱

به صورت پیش فرض جاب‌ها به صورت موازی پردازش می‌شوند. اگر می‌خواهید سریالی شوند از ویژگی DisallowConcurrentExecution استفاده کنید (روی نام کلاس قرارش بدید).

نویسنده: عصاری
تاریخ: ۱۳۹۲/۱۰/۲۸ ۱۰:۱

من از(DisallowConcurrentExecution) استفاده کردم اما بازم jobها به صورت موازی اجرا میشن!؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۸ ۱۰:۱۹

طراحی lock و مباحث همزمانی (lock, mutex, semaphore) و Threading را باید خودتان پیاده سازی کنید.
DisallowConcurrentExecution یعنی یک وهله از یک کلاس خاص در زمان اجرا ایجاد شده و چندین وهله همزمان از آن ایجاد نخواهد شد. روش دیگر اینکار استفاده از TriggerListener, JobListener or SchedulerListener است. یکی که تمام شد، بعدی شروع شود.

نویسنده: بهاره فیضی
تاریخ: ۱۳۹۳/۰۳/۲۹ ۱۱:۴۰

سلام؛ من سعی کردم این کدها رو تست کنم بنابراین قدم به قدم پیش رفتم. بعد وظیفه رو به شکل زیر فراخوانی کردم

```
protected void Application_Start(object sender, EventArgs e)
        {
            Interface1 myTask = new HelloSchedule();
            myTask.Run();
        }
```

دو بار web application رو اجرا کردم و نتیجه به شکل زیر بود

```
Message from HelloJob 6/19/2014 10:40:15 AM
Message from HelloJob 6/19/2014 10:40:25 AM
Message from HelloJob 6/19/2014 10:40:35 AM
Message from HelloJob 6/19/2014 10:40:45 AM
Message from HelloJob 6/19/2014 10:40:55 AM
Message from HelloJob 6/19/2014 11:24:36 AM
Message from HelloJob 6/19/2014 11:24:45 AM
Message from HelloJob 6/19/2014 11:24:55 AM
```

یعنی دفعه اول 5 بار وظیفه اجرا شده دفعه دوم 3 بار.
ولی هدف من از اعمال زمانبندی اینه که بی نهایت بار وظیفه اجرا بشه، به طور مثال در یک سایت آگهی شبیه ایستگاه، مثلاً دو روز قبل از انقضای آگهی به فرد اطلاع داده بشه که آگهی شما داره منقضی میشه.
البته من در لوکال تست کردم.
من بهتر از

```
Application_Start
```

جایی برای فراخوانی پیدا نکردم.
لطفاً راهنمایی کنید.

نویسنده: محبوبه عصاری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۱۰:۴۸

با سلام
ممنون از راهنماییتون، ممکن هست برای استفاده از joblistener مثالی بذارید؟ برای ارسال پیامک به صورت ترتیبی چطور باید این لیست رو به joblistener اضافه نمود و چطور باید بعد از هر ارسال ، ارسال بعدی را فراخوانی نمود؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۱۱:۳۳

برای ارسال پیامک به صورت ترتیبی از حلقه استفاده کنید. زمانیکه job اجرا شد، ابتدا لیست اشخاصی را که باید پیامک دریافت کنند از دیتابیس واکشی کرده و سپس در طی یک حلقه، به آن‌ها پیام ارسال کنید.

نویسنده: محبوبه عصاری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۱۲:۱

دقیقا این کاری بود که من قبلا انجام داده بودم ولی مشکلی که پیش می‌آمد این بود که مثلا اگر در یک ارسال بایستی 200 پیامک ارسال گردد قبل از اینکه ارسال این 200 پیامک به اتمام برسد زمان اجرا به پایان رسیده و تابع execute مجددا فراخوانی می‌شود و از انجایی که هنوز وضعیت این رکورد در دیتابیس به ارسال شده تغییر پیدا نکرده مجددا این 200 پیامک ارسال می‌گردد. این مشکل حتی زمانی که از حلقه for هم استفاده نمی‌شود وجود دارد و در تعداد ارسال بالا به مشکل می‌خورد .
در زیر کدی که برای ارسال استفاده نموده ام را قرار دادم.
با تشکر از شما

```
namespace SchedulerDemo.Jobs
{
    using System;
```

```
    using System.Linq;
    using System.IO;
    using Quartz;
    using System.Collections.Generic;
    using System.Configuration;

    [PersistJobDataAfterExecution]
    [DisallowConcurrentExecution]
    public class SendJob : IJob
    {
        public void Execute(IJobExecutionContext context)
        {

            using (var db = new DALModel.DALEntities())
            {

                byte status = (byte)AllEnums.Sms.Status.InProgress;
                var item = db.SentBoxes.Where(p => p.Status == status && p.IsDeleted==false &&
p.UserInfo.IsDeleted==false &&  p.HasTime == true && p.SendInTime == false && p.SendDateX <=
DateTime.Now).OrderBy(p=>p.Id).FirstOrDefault();
                Cls_SMS.ClsSend sms_Batch = new Cls_SMS.ClsSend();

                if (item != null)
                {
                    decimal smsCount = 0;
                    if (item.UserInfo.CalculateType ==
Convert.ToByte(AllEnums.FinancialTransaction.CalculationUnit.Message))
                    {
                        smsCount = Convert.ToDecimal(Function.GetSmsCount(item.Price, item.UserId));
                    }
                    else
                    {
                        smsCount = Convert.ToDecimal(item.CorrectCount);
                    }
                    decimal adminCredit = Function.GetAdminCreditLink1000();
                    if (adminCredit != -1 && adminCredit >= smsCount)
                    {

                        if ((item.UserInfo.Credit - (item.UserInfo.LowCredit)) >= item.Price)
                        {
                            item.SendInTime = true;
                            db.SaveChanges();
                            string numberList = item.NumberList;
                            int position = item.NumberList.LastIndexOf(',');
                            numberList = item.NumberList.Substring(0, position);
                            List<string> receivers_List = new List<string>();
                            receivers_List = (numberList).Split(',').ToList();
                            string[] ret2 = new string[2];
                            string[] DestAdd = new string[receivers_List.Count];
                            DestAdd = receivers_List.ToArray();
                            ret2 = sms_Batch.SendSMS_Batch(item.Message, DestAdd,
item.UserInfoSenderNumber.AllNumber.Number, ConfigurationManager.AppSettings["SmsUserNameLink1000"],
ConfigurationManager.AppSettings["SmsPasswordLink1000"],
ConfigurationManager.AppSettings["SmsIPAddressLink1000"],
ConfigurationManager.AppSettings["SmsCompanyLink1000"], false, item.Id);
                            var sentBoxUpdate = db.SentBoxes.FirstOrDefault(p => p.Id == item.Id);
                            sentBoxUpdate.Status = Convert.ToByte(AllEnums.Sms.Status.Send);
                            sentBoxUpdate.FinancialTransactionId = db.FinancialTransactions.Where(p =>
p.UserId == item.UserId).Max(p => p.Id);

                            if (ret2 != null)
                            {
                                sentBoxUpdate.RetValue0 = ret2[0];
                                sentBoxUpdate.RetValue1 = ret2[1];
                            }
                            db.SaveChanges();
                        }
                        else
                        {
                            byte statusFailedForAccount = (byte)AllEnums.Sms.Status.FailedForAccount;
                            item.SendInTime = true;
                            item.Status = statusFailedForAccount;
                            item.FailedCount = item.CorrectCount;
                            item.FailedList = item.NumberList;
                            db.SaveChanges();

                        }
                    }
                    else
                    {
                        byte statusFaildForError = (byte)AllEnums.Sms.Status.FaildForError;
```

```
                        item.SendInTime = true;
                        item.Status = statusFaildForError;
                        item.FailedCount = item.CorrectCount;
                        item.FailedList = item.NumberList;
                        db.SaveChanges();

                    }
                }


            }
        }
    }
}

namespace SchedulerDemo.Interfaces
{
    public interface IScheduleSend
    {
        void RunSendSms();
    }
}

namespace SchedulerDemo.Jobs
{
    using System;
    using Quartz;
    using Quartz.Impl;
    using SchedulerDemo.Interfaces;
    using SchedulerDemo.Jobs;

    public class SendSchedule : IScheduleSend
    {
        public void RunSendSms()
        {
            DateTimeOffset startTime = DateBuilder.FutureDate(2, IntervalUnit.Second);

            IJobDetail job = JobBuilder.Create<SendJob>()
                                       .WithIdentity("jobSendSmsInTime")
                                       .Build();

            ITrigger trigger = TriggerBuilder.Create()
                                             .WithIdentity("triggerSendSmsInTime")
                                             .StartAt(startTime)
                                             .WithSimpleSchedule(x =>
x.WithIntervalInMinutes(5).RepeatForever())
                                             .Build();

            ISchedulerFactory sf = new StdSchedulerFactory();
            IScheduler sc = sf.GetScheduler();
            sc.ScheduleJob(job, trigger);

            sc.Start();
        }
    }
}
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۱۲:۱۴

از DNT Scheduler استفاده کنید. در اینجا اگر یک job هنوز در حال اجرا باشد، وهله‌ی جدیدی از آن آغاز نخواهد شد:

```
var taskToRun = _tasks.Where(x => !x.IsRunning && x.RunAt(now)).OrderBy(x => x.Order).ToList();
```

در قسمت IsRunning این بررسی به صورت خودکار انجام می‌شود.

نویسنده: بهاره فیضی
تاریخ: ۱۳۹۳/۰۵/۰۲ ۱۱:۳

سلام.
من یه وظیفه نوشتم که می‌یاد هر 1 دقیقه 1 بار یه متن رو در Table درج می‌کنه.

این کد کلاس

```
using Quartz;
using System.Data;
using System.Data.OleDb;
using System.Configuration;


namespace WebApplication1
{

    public class TestQuartzClass:IJob
    {


        public void Execute(IJobExecutionContext context)
        {
            //Page myPage = (Page)HttpContext.Current.Handler;
            //TextBox MyTextBox=(TextBox)myPage.FindControl("txt");

            string sql = "Insert into tbl_Test (Content) values (@Content)";
            ExecuteNoneQuery(System.Data.CommandType.Text, sql, new OleDbParameter[]{
                //new OleDbParameter("@Content", MyTextBox.Text)
                new OleDbParameter("@Content","Hello world!")
            });

        }

        public int ExecuteNoneQuery(CommandType commandType, string commandText, params
OleDbParameter[] commandParameters)
        {
            using (OleDbConnection con = new
OleDbConnection(ConfigurationManager.ConnectionStrings["ConStr"].ConnectionString))
            {
                OleDbCommand cmd = new OleDbCommand();
                cmd.Connection = con;
                cmd.CommandType = commandType;
                cmd.CommandText = commandText;
                cmd.Parameters.AddRange(commandParameters);
                con.Open();
                int retVal = cmd.ExecuteNonQuery();
                con.Close();
                return retVal;
            }
        }
    }


}
```

و این هم کد صفحه ای که با کلیک دکمه وظیفه شروع به کار می‌کنه

```
using System;
using Quartz;
using Quartz.Impl;

namespace WebApplication1
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        public static void ConfigureQuartzJobs()
        {
            // construct a scheduler factory
            ISchedulerFactory schedFact = new StdSchedulerFactory();

            // get a scheduler
            IScheduler sched = schedFact.GetScheduler();
            sched.Start();
            IJobDetail job = JobBuilder.Create<TestQuartzClass>()
```

```
                .WithIdentity("SendJob")
                .Build();
            var trigger = TriggerBuilder.Create()
                .WithIdentity("SendTrigger")
                .WithSimpleSchedule(x => x.WithIntervalInMinutes(1).RepeatForever())
                //.StartAt(startTime)
                .StartNow()
                .Build();

            sched.ScheduleJob(job, trigger);
        }

        protected void btn_Click(object sender, EventArgs e)
        {
            ConfigureQuartzJobs();
        }
    }
}
```

همونطور که می‌بینید متن رو به شکل زیر پاس دادم.

```
new OleDbParameter("@Content","Hello world!")
```

ولی من می‌خوام این متن رو از TextBox بگیرم.
و برای اینکه در کلاس بتونم به کنترلهای صفحه دسترسی داشته باشم کدهای زیر رو به متد Exceute اضافه کردم

```
Page myPage = (Page)HttpContext.Current.Handler;
            TextBox MyTextBox=(TextBox)myPage.FindControl("txt");
```

ولی به محض اینکه این کدها رو اضافه می‌کنم دیگه برنامه کار نمی‌کنه.

متن داخل تکست باکس رو هم قصد داشتم به شکل زیر پاس بدم.

```
new OleDbParameter("@Content", MyTextBox.Text)
```

لطفاً راهنمایی کنید.
من یک نمونه هم به منظور تست آماده کردم که از لینک زیر می‌تونید دانلود کنید:
http://www.4shared.com/rar/1Fu_jpOOba/WebApplication1.html

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۵/۰۲ ۱۱:۱۴

ترد اجرایی یک وظیفه‌ی پس زمینه با ترد اجرایی یک درخواست وب یکی نیست. بنابراین نمی‌تونید به اشیاء رابط کاربری در تردی دیگر دسترسی داشته باشید. در برنامه‌های دسکتاپ اگر این‌کار را انجام بدید، برنامه آنی کرش می‌کنه. در وب هم فرقی نمی‌کنه؛ اصول یکی هست.

کاری که می‌خواهید انجام بدید کلا نباید از این طریق انجام بشه. اگر هدفتون مثلا پیاده سازی **auto-save** هست که در بعضی از سایت‌ها دیدید که هر از چند ثانیه یکبار متن رو ذخیره می‌کنند، این رو با یک تایمر جاوا اسکریپتی سمت کاربر و Ajax انجام می‌دن و نه با یک وظیفه‌ی پس زمینه.

نویسنده: بهاره فیضی
تاریخ: ۱۳۹۳/۰۵/۰۲ ۱۲:۳۵

ممنون که پاسخ دادید.

نه من این رو تستی انجام دادم.
هدف اصلی من ارسال ایمیل انبوه بود به این صورت که تعداد زیادی ایمیل بر اساس صنف‌های کاری در یک جدول ذخیره شدن. مثلاً صنف ساختمانی، ...
حالا من می‌خوام یک صفحه داشته باشم
ادمین بیاد از دراپ دان گروه رو انتخاب کنه و بر حسب صنف حالا مثلاً 100 تا ایمیل بهش نشون داده میشه.
بعد به فرض یه متن رو در یک تکست باکس وارد کنه و این متن به این افراد ایمیل بشه.
در این لینک توضیحات کاملتری رو داده بودم ولی هنوز جوابی نگرفتم.
http://forums.asp.net/t/1998923.aspx?send+mass+email+using+quartz+net
از ویندوز سرویس هم نمی‌تونم استفاده کنم چون هاستها اشتراکی هستند و ادمین هاست اجازه نصب نمی‌ده.

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۵/۰۲ ۱۲:۴۱

یک جدول طراحی کنید به نام mass mail که در آن یک Content به علاوه فیلد IsDone وجود داره. مدیر سیستم متن خودش رو به صورت معمول در این جدول ثبت کنه. در وظیفه‌ی پس زمینه، رکوردهایی را که IsDone آن‌ها false است یکی یکی یافته و پردازش کنید. بعد از اتمام کار، IsDone هر رکورد را true کنید.

مزیت این روش اینه که دو ترد کاملا مجزای رابط کاربری و ترد وظیفه‌ی پس زمینه، هیچ تداخل و تماسی با هم نخواهند داشت تا سبب کرش برنامه شوند.

نویسنده: بهاره فیضی
تاریخ: ۱۳۹۳/۰۵/۰۲ ۱۲:۵۶

واقعاً از تون ممنونم. چه روش خوب و ساده ای گفتید. اصلاً به ذهن خودم نرسیده بود.

نویسنده: بهاره فیضی
تاریخ: ۱۳۹۳/۰۵/۱۲ ۲۲:۱۷

سلام.
من کدهای زیر رو در رویداد کلیک یک دکمه نوشتم که کاربر با کلیک دکمه اقدام به ارسال ایمیل به صورت گروهی می‌کنه.
حالا یکی از کاربرا با این ارور مواجه شده

```
.Unable to store Job: 'DEFAULT.SendJob', because one already exists with this identification

Description: An unhandled exception occurred during the execution of the current web request. Please review the
.stack trace for more information about the error and where it originated in the code

Exception Details: Quartz.ObjectAlreadyExistsException: Unable to store Job: 'DEFAULT.SendJob', because one
.already exists with this identification

:Source Error

;()Line 374: .Build
:Line 375
;(Line 376: sched.ScheduleJob(job, trigger
:Line 377
```

که البته من متوجه شدم دلیل این به خاطر اینه که هنوز جاب قبلی پایان نیافته کاربر دوباره روی دکمه کلیک می‌کنه و چون هنوز جاب قبلی اتمام نیافته با این ارور مواجه میشیم. بنابراین من یک دکمه گذاشتم و کدهای زیر رو نوشتم در رویداد کلیک این دکمه

```
var scheduler = new StdSchedulerFactory().GetScheduler();
        scheduler.DeleteJob(new JobKey("SendJob"));
```

که با کلیک این دکمه جاب قبلی متوقف میشه و دیگه اون ارور رو نمی‌گیریم. ولی مشکل اینجاست که چطور میشه بدون اینکه اصلاً با اون ارور کاربر مواجه بشه پیغام بدیم که شما در حال حاضر مجاز به ارسال نیستید و مثلاً 10 دقیقه بعد دوباره اقدام به ارسال فرمایید.

یا اینکه چطور میشه یک جاب که تموم شد به کاربر پیغام بدیم که جاب پایان یافته، در این صورت دیگه کاربر در زمان نامناسب اقدام به ایجاد مجدد جاب نخواهد کرد.

ممنون.

نویسنده: بهاره فیضی
تاریخ: ۱۳۹۳/۰۵/۱۳ ۱۳:۲۱

فعلاً با کد زیر مشکل موقتاً حل شد، یعنی کاربر باید با کلیک روی دکمه مطمئن بشه که حالا می‌تونه اقدام به ارسال بعدی کنه یا نمی‌تونه.
جالبه متوجه شدم اون کدهای متوقف کردن جاب هم در جا عمل نمی‌کنن، یعنی معلوم نیست کی متوقف بشه. پس اینجا به درد کار ما نمی‌خوره.
اگه راه بهتری به نظرتون رسید ممنون می‌شم در میون بزارید.

```
protected void Button6_Click(object sender, EventArgs e)
    {
        var scheduler = new StdSchedulerFactory().GetScheduler();
        var jobs = scheduler.GetCurrentlyExecutingJobs();
        //foreach (var j in jobs)
        //{
        //    Console.WriteLine("Progress of {0} is {1}",
        //        j.JobDetail.Key,
        //        j.JobDetail.JobDataMap["progress"]);
        //    Label620.Text += j.JobDetail.Key.ToString();

        //}
        if (jobs.Count ==0)
        {



            Label620.Text = "آماده برای ارسال ایمیل!";

        }
        else
        {

            Label620.Text = "لطفاً بعد از چند دقیقه مجدداً تلاش نمایید!";
        }

    }
```

عنوان: **اجرای وظایف زمان بندی شده با Quartz.NET - قسمت دوم**
نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۲۶ ۲۱:۲۲
آدرس: www.dotnettips.info
گروه‌ها: Quartz.NET, Schedule, زمانبندی

در این قسمت، نحوه‌ی استفاده از قابلیت‌های کتابخانه‌ی Quartz.NET را در قالب پرسش و پاسخ ادامه می‌دهیم.

ابتدا یک توضیح کلی:
برای مدیریت وظیفه‌ها در Quartz.NET، در هر جای پروژه می‌توانید به صورت ذیل به مدیر وظیفه‌ها دسترسی داشته باشید.

```
var scheduler = new StdSchedulerFactory().GetScheduler();
```

از حالا به بعد، هر جا که در کدها کلمه‌ی scheduler را دیدید، ایجاد آن از طریق خط قبل بوده است.
سعی کنید همیشه هنگام ایجاد اشیا از نوع IJobDetail و ITrigger، از متد WithIdentity (همان طور که در قسمت قبل مشاهده کردید) برای نامگذاری وظایف و triggerها استفاده کنید تا بتوانید بعداً با استفاده از نامشان به آنها ارجاع پیدا کرده و مدیریتشان کنید

1) سوال: چگونه می‌توان در یک زمان دلخواه (مثلاً در زمان کلیک بر روی یک دکمه)، اجرای یک وظیفه را متوقف کرد؟
جواب: برای توقف تمامی وظایف می‌توان از متد ()Shutdown شی scheduler استفاده کرد:

```
scheduler.Shutdown(true);
```

در صورتی که مقدار true را به متد Shutdown پاس دهید، تا زمانی که وظایفی که در وسط اجرای خود هستند کارشان به پایان نرسد، کنترل اجرای برنامه به خط بعد نمی‌رود، اما در صورتی که این متد را بدون پارامتر فراخوانی کنید یا مقدار false را به آن پاس دهید، کنترل اجرای برنامه به دستور بعد می‌رود و وظایف در پشت صحنه به کار خود ادامه می‌دهند. دقت داشته باشید که منظور از ادامه‌ی کار یک وظیفه، ادامه‌ی کار آن در وضعیت جاری خود است. به بیان واضح تر، اگر مرتبه‌ی اجرای یک وظیفه 20 مرتبه بود و در مرتبه‌ی دوم اجرای آن، متد ()Shutdown به هر صورتی فراخوانی شد، مرتبه‌های دیگر به هیچ وجه اجرا نمی‌شوند.
اگر چند وظیفه به طور همزمان در حال اجرا باشند و قصد داشته باشید تا یکی از آنها را متوقف کنید، یکی از دو حالت زیر وجود دارد:

1) یک وظیفه به چند trigger نسبت داده شده است.
2) هر وظیفه فقط یک trigger دارد.
در صورتی که قصد دارید وظیفه از تمامی triggerها گرفته شود (معمولاً هم همین رفتار مد نظر است)، از متد DeleteJob استفاده کنید؛ و اگر قصد دارید تا اجرای وظیفه توسط یک trigger مشخص لغو شود و triggerهای دیگر مختص آن وظیفه به کار خود ادامه دهند، از متد UnscheduleJob استفاده کنید. اگر از متد DeleteJob استفاده می‌کنید، نام وظیفه را با ایجاد نمونه ای از کلاس JobKey برای آن مشخص کنید و در صورتی که از متد UnscheduleJob استفاده می‌کنید، نام trigger را با ایجاد نمونه ای از کلاس TriggerKey تعیین کنید.

```
scheduler.DeleteJob(new JobKey("job1"));
// or
scheduler.UnscheduleJob(new TriggerKey("trigger1"));
```

2) سوال: چگونه می‌توان اجرای وظیفه‌ها را به حالت تعلیق در آورد؟
جواب: برای به تعلیق در آوردن اجرای تمامی وظایف، از متد ()StandBy استفاده کنید:

```
scheduler.Standby();
```

برای ادامه‌ی کار، متد ()Start را مجدداً فراخوانی کنید.

در صورتی که قصد دارید اجرای وظیفه ای خاص را به حالت تعلیق در آورید، از متد ()PauseJob استفاده کنید. نام وظیفه را با ایجاد نمونه ای از کلاس JobKey برای آن مشخص کنید:

```
scheduler.PauseJob(new JobKey("job1"));
```

برای ادامه‌ی وظیفه، از متد ()ResumeJob استفاده کنید. نام وظیفه را با ایجاد نمونه ای از کلاس JobKey برای آن مشخص کنید:

```
scheduler.ResumeJob(new JobKey("job1"));
```

برای تعلیق اجرای تمامی وظایف، متد ()PauseAll، و برای ادامه‌ی کار تمامی وظایف، متد ()ResumeAll را فراخوانی کنید.

3) سوال: چگونه می‌توان یک وظیفه‌ی در حال اجرا را آپدیت کرد و تغییر مشخصات داد؟
جواب: با استفاده از متد AddJob و تنظیم پارامتر دوم آن به مقدار true:

```
IJobDetail job = JobBuilder.Create<NewJob>()
                           .WithIdentity("job1")
                           .Build();

scheduler.AddJob(job, true);
```

اگر قبلاً کلاسی با عنوان OldJob برای وظیفه ای با نام job1 تعریف شده است، با استفاده از قطعه کد بالا می‌توان کلاس NewJob را به جای آن معرفی کرد. البته به شرطی که نام وظیفه‌ی جدید با نام وظیفه‌ی قدیم، یکسان باشد. پارامتر دوم متد AddJob مشخص می‌کند که آیا در صورتی که نام وظیفه ای که قرار است در فرایند زمانبندی قرار بگیرد با نام یکی از وظایف موجود یکسان باشد، وظیفه‌ی جدید، جایگزین وظیفه‌ی قدیم شود یا خیر.

4) سوال: چگونه می‌توان یک trigger در حال اجرا را آپدیت کرد و تغییر مشخصات داد؟
جواب: یک trigger جدید ایجاد و با استفاده از متد ()RescheduleJob، جایگزین trigger قدیمی کنید:

```
ITrigger trigger = TriggerBuilder.Create()
                                 .WithIdentity("newTrigger")
                                 .StartNow()
                                 .ForJob("job1")
                                 .WithSimpleSchedule(x =>
x.WithIntervalInSeconds(5).WithRepeatCount(20))
                                 .Build();

scheduler.RescheduleJob(new TriggerKey("oldTrigger"), trigger);
```

نام trigger جدید می‌تواند با نام trigger قدیم یکسان باشد. در تکه کد قبل، triggerیی با نام newTrigger ایجاد و زمان اجرای آن به حال تنظیم شده است. با استفاده از متد ()ForJob و تعیین نام وظیفه، trigger جدید را به وظیفه ای با نام job1 نسبت داده ایم. بازه‌ی زمانی اجرا، هر 5 ثانیه و 21 مرتبه خواهد بود. در متد ()RescheduleJob و در پارامتر اول آن، نام trigger قدیمی را با ایجاد شی ای از کلاس TriggerKey مشخص کرده ایم و به پارامتر دوم، شی ایجاد شده برای trigger جدید را پاس داده ایم.

5) سوال: چگونه می‌توان تعداد دفعات اجرای یک وظیفه را بی نهایت تعیین کرد؟
پاسخ: با استفاده از متد ()RepeatForever در هنگام ایجاد trigger:

```
ITrigger trigger = TriggerBuilder.Create()
                                 .WithIdentity("trigger1")
                                 .StartAt(startTime)
                                 .ForJob("job1")
                                 .WithSimpleSchedule(x => x.WithIntervalInSeconds(5).RepeatForever())
                                 .Build();
```

6) سوال: چگونه می‌توان تعداد دفعات اجرای تمامی وظایف را به دست آورد؟

پاسخ: با استفاده از متد ()GetMetaData:

```
SchedulerMetaData metaData = scheduler.GetMetaData();
int numberOfJobsExecuted = metaData.NumberOfJobsExecuted;
```

متد ()GetMetaData، اطلاعاتی در مورد مدیر وظایف می‌دهد. نوع برگشتی این متد، SchedulerMetaData است. یکی از خصیصه‌های این نوع، NumberOfJobsExecuted نام دارد که تعداد دفعات اجرای تمامی وظایف تا زمان حال را برگشت می‌دهد.

7) سوال: چگونه می‌توان زمان آغاز به کار مدیر زمانبندی را متوجه شد؟
پاسخ: یکی دیگر از خصیصه‌های نوع SchedulerMetaData، RunningSince نام دارد که بدین منظور استفاده می‌شود.

```
SchedulerMetaData metaData = scheduler.GetMetaData();
DateTimeOffset? runningSince = metaData.RunningSince;
```

ادامه دارد...

نظرات خوانندگان

نویسنده: رضا
تاریخ: ۱۳۹۱/۰۵/۲۷ ۱۰:۳۸

آقای راد خیلی ممنون از مطلب مفیدتون. اگر بخواهیم مثلاً یک قسمت از UI رو هر چند دقیقه آپدیت کنیم، نحوه انجام ای کار به چه صورتی خواهد بود؟

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۲۷ ۱۳:۱۹

البته نگفتید که منظورتون desktop یا وب هست، با این فرض که در مورد desktop می‌پرسید، یکی از روش‌ها اینه که یک متغیر عمومی تعریف کنید که ارجاعی به فرمی که قرار هست آپدیت باشه داشته باشه:

```
using System.Windows.Forms;

public static class GlobalData
{
    public static Form ScheduleForm { get; set; }
}
```

در سازنده‌ی فرم می‌تونید اون رو به فرم جاری مقداردهی کنید:

```
GlobalData.ScheduleForm = this;
```

با این فرض که قرار هست عنوان یک دکمه در فرم با نام myButton به My Text تغییر کنه، کلاس پیاده ساز اینترفیس IJob به صورت زیر خواهد بود.

```
namespace SchedulerDemo.Jobs
{
    using System.Linq;
    using System.Windows.Forms;
    using Quartz;

    public class HelloJob : IJob
    {
        private delegate void ButtonTextWriter(string buttonId, string text);

        MainForm form = GlobalData.ScheduleForm as MainForm;

        private void SetButtonText(string buttonId, string text)
        {
            (form.Controls.Find(buttonId, true).FirstOrDefault() as Button).Text = text;
        }

        public void Execute(IJobExecutionContext context)
        {
            form.BeginInvoke(new ButtonTextWriter(SetButtonText), new object[] { "myButton", "My Text"
});
        }
    }
}
```

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۶/۱۳ ۲۰:۱

سلام ... اگر دیده باشید سرویس‌های بلاگ نویسی سرویسی دارن که تو اون میشه یه پستو تایم ارسال بهش داد و در اون روز و ساعت پست تو وبلاگ نمایش داده میشه ... حالا آیا با Quartz.net میشه این کارو کرد؟ (تو asp.net mvc) ... اگه بشه فکر میکنم اگه راجع به این قضیه پستی بزنین خیلی خوب میشه :دی ... ممنون ...

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۶/۱۳ ۲۰:۱۱

این کار ارتباطی با کتابخانه‌های زمانبندی نداره. اونها پست‌ها رو بر اساس تاریخ و ساعت نشون میدن => پست هایی که تاریخ نمایش اونها از زمان جاری قدیمی‌تر هست نشون داده میشن و پست هایی که تاریخ نمایش اونها از تاریخ جاری جدیدتر هست نشون داده نمیشن.

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۶/۱۴ ۰:۳۷

در تاریخ و زمان مشخصی نمایش داده میشن بدون اینکه ادمین وبلاگ به وبلاگش سر بزنه پست در اون زمان مشخص شده در ویلاگ قرار میگیره و همگان اونو میبینن! ... این زمان بندی نیست الان؟! ... یا مثال دیگه : سیستم‌های یادآور یا ToDo که با توجه به زمان و ایمیلی که مشخص کردین براتون میلو ارسال میکنن (اینا منظورم بود :دی)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۴ ۰:۵۲

همانطور که عنوان شد طراحی دیتابیس است نه استفاده از ابزارهای جانبی. یک وبلاگ در این حالت شبیه به کوئری زیر کار می‌کند (هر فراخوانی صفحه‌ای معادل است با یک کوئری از بانک اطلاعاتی):

```
select * from tblPosts where (showDate is null) or (showDate<=getdate())
```

فیلد showDate اگر نال بود، یعنی یک مطلب معمولی است که درجا نمایش پیدا می‌کند. اگر تاریخی برای آن مشخص شده بود، بر اساس تاریخ جاری یک مقایسه صورت گرفته و رکوردها انتخاب و نمایش داده می‌شوند.

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۶/۱۴ ۰:۵۸

ممنون ... ولی بنده منظورم این نبود که الگوریتمش چجوریه! ... منظورم این بود که چیکار کنیم که پشت پرده و بدون این که توسط کاربر Request به پیجی داشته باشیم این کار انجام بشه! ... تو Application_start باید این کارو کرد؟! ... اگه آره چجوری؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۴ ۱:۴

سؤال شما این بود: «... تو وبلاگ نمایش داده میشه ... » این یعنی ارسال یک کوئری به بانک اطلاعاتی و دریافت پاسخ.
و چرا پشت پرده باید انجام بشه؟ یک وبلاگ بر اساس درخواست یک کاربر هست که صفحاتش نمایش داده میشه. یعنی همون کوئری فوق.

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۶/۱۴ ۱:۱۰

کاربر میگه که میخوام 4 ساعت دیگه نمایش بدی! ... یعنی 4 ساعت دیگه باید این Task انجام بشه! ... حالا شما فکر کنین هزار تا کاربر داریم! هر کدوم یه تایمی رو در آینده میدن که پستشون به بازدیدکنندگان ویلاگشون نمایش داده بشه! ... خوب الان یه جورایی ما تو برناممون یه ترد بینهایت میخوایم که به لیست Taskها در زمان‌های مشخص شده رسیدگی کنه! ... بدون این که دیگه کاربر دخالتی داشته باشه! و همیشه هم این ترد باید کار کنه! ... پس اینجوری باید این ترد تو Application_start تنظیم بشه و تا وقتی سرور روشنه این کارارو انجام بده! ... اگه درست نمیگم تصحیح کنید!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۴ ۱:۱۵

ما داریم در مورد یک وبلاگ بحث می‌کنیم. task نمایش در اینجا یعنی باز کردن یک صفحه توسط کاربر نهایی. در این حالت ابتدا یک کوئری به بانک اطلاعاتی ارسال شده و بر اساس فیلد showDate که پیشتر توسط نویسنده‌ها تنظیم شده (یا خیر ... حالت اختیاری)، اطلاعات نمایش خواهند یافت.
نیازی به ترد اضافی نیست. نیازی به ابزار زمانبندی نیست. فقط یک کوئری ساده روی فیلد showDate یک رکورد موجود است.

نویسنده: mgh
تاریخ: ۱۳۹۱/۰۶/۳۱ ۲۰:۱۶

دوست عزیز برنامه‌های زمان بندی ابزاری بسیار زیبا و مفید هستند در صورتی که در مکان و زمان مناسب از آنها بهره ببریم.
جهت انجام این کار بسیار ساده ای که مد نظر شماست، ایجاد یک Task به مانند پیچاندن لقمه دور سر است.
این کاری که شما مثال زدید رو جناب نصیری خیلی روان و ساده توضیح دادند. هیچ کار اضافی هم نیاز نیست. تنها یک کوئری هست
موفق باشید :)

نویسنده: محمد فخراوری
تاریخ: ۱۳۹۱/۰۹/۰۵ ۱:۱۴

با سلام
ممنون بابت راهنمایی‌های اقای راد.
اقای راد در مورد پست اول خودتون اومدید یه کلاس نوشتید و در پست دوم در مورد dll توضیح دادین.
اگر ممکن یه سمل بزارین

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۹/۰۵ ۱۰:۲۴

در پست اول، گام به گام، کدهای یک مثال توضیح داده شده. شما فقط Copy & Paste کنید.

نویسنده: حسین
تاریخ: ۱۳۹۱/۰۹/۰۵ ۱۹:۵۳

ممنون از مقالتون

فقط یه سوالی که برام پیش اومده ,اینه که ایا این کتابخانه از چند نخی برای اجرای کارها استفاده می‌کنه ؟

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۹/۰۶ ۷:۱۱

این کتابخانه، متن باز هست. می‌تونید کدهاش رو بررسی کنید.

نویسنده: محمد فخراوری
تاریخ: ۱۳۹۱/۱۰/۱۶ ۱۷:۲۰

با سلام
اگر میشه سمپلی بزارید .

نویسنده: حسین

تاریخ: ۱۳۹۱/۱۰/۲۹ ۹:۴۱

سلام.
من توی برنامم از این کتابخانه استفاده کردم و به صورت تستی گذاشتم که هر یک دقیقه یک کار خاصی رو انجام بده. ولی یک مشکلی که دارم اینه که وقتی برنامه رو می‌بندم به طور کامل از برنامه خارج نمیشه و هنوز پشت صحنه عمل مربوطه رو انجام میده و بایستی دکمه Stop رو بزنم تا اجراش متوقف بشه.
میخواستم بدونم راه گرفتن یک Scheduler خاص چیست و چطور باید Shutdown اش کرد؟
ممنون.

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۱۰/۲۹ ۱۰:۵۱

چون Threadیی که ایجاد میشه، اصطلاحاً Foreground هست و باید به Background تبدیل بشه.
جزئیات بیشتر...
در ضمن توجه داشته باشید که اگر در وسط اجرای یک فرایند اون رو به زور متوقف کنید، مشکل هرز رفتن منابع رو خواهید داشت. در نظر بگیرید که فایلی رو باز کردید و قصد نوشتن در اون رو دارید، در همین هنگام پروسه رو قطع می‌کنید. فایل باز می‌مونه و پروسس‌های دیگه نمی‌تونن به اون دسترسی داشته باشن. به نظر من بهتره رفتار پیش فرض Quartz.NET رو تغییر ندید و بر روی منطق برنامه‌ی خودتون معطوف باشید.

نویسنده: M.Q
تاریخ: ۱۳۹۱/۱۱/۱۷ ۲۲:۷

با سلام؛ صرف نظر از اینکه ما کدهای زمانبندی رو چطور می‌نویسیم یا از چه کتابخانه‌ها و روش هایی استفاده می‌کنیم، برای پیاده سازی برنامه ای هیچ قسمت یوزر اینترفیس ای ندارد (همانند وبلاگ‌ها یا برنامه های تیکتینگ) و صرفا اطلاعاتی را از دیتابیس (موجود) فراخوانی و عملیاتی روی آنها انجام می‌هد، (مثلا ارسال ایمیل یا پیامک)، از چه نوع application ای استفاده کنیم؟ (webapp - winapp - winservice).

من برای انجام اینکار از winapp استفاده نمی‌کنم چون نمی‌خواهم یک برنامه را روی ویندوز اجرا کنم. همچنین از webapp استفاده نمی‌کنم بنا به دلایلی خاص. در حال حاظر از windows serviceها استفاده می‌کنم. آیا راه حل مناسب‌تری وجود دارد؟

با تشکر از شما

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۱۷ ۲۳:۱۳

سرور متعلق به خودتون است؟ دسترسی بالایی دارید مثل دسترسی نصب و همچنین دسترسی اجرای برنامه به صورت سرویس؟ اگر بله، استفاده از سرویس‌های ویندوز nt روش متداولی است. اگر نه، برنامه‌های وب با حداقل دسترسی همین کار رو می‌تونند انجام بدن.

نویسنده: M.Q
تاریخ: ۱۳۹۱/۱۱/۱۸ ۱۰:۴۶

ممنون از شما آقا سعید

بله در بعضی موارد سرور می‌تواند از خود من باشد و در بعضی موارد دسترسی ریموت برای نصب و کنترل سرویس را دارم.

درمورد برنامه‌های وب برای این کار : من یکبار این کار را انجام دادم و از **cache expiration** استفاده کردم. ولی چون این

برنامه‌ی وب هیچ request ای نداشت برنامه down شد و عملیات زمان بندی متوقف شد.

پس میشه گفت استفاده از سرویس‌های ویندوزی، راه حل متداولی هست؟

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۱۸ ۱۰:۵۳

- تولید درخواست به یک سایت کار سختی نیست. از این سرویس‌هایی که مدام سایت رو ping می‌کنند زیاد هست. مثل سرویس‌های فید خوان. مثل سایت‌هایی که بررسی می‌کنند سایت شما آپ است یا داون.

- نه الزاما. سرویس task scheduler ویندوز همیشه در حال اجرا است؛ دقیق و منظم. این سرویس می‌تونه خیلی راحت یک فایل exe معمولی یا حتی یک فایل bat رو در زمان‌های مشخصی اجرا کنه. برای کارهای مدیریت سیستم زیاد ازش استفاده می‌شه.

نویسنده: احمدی
تاریخ: ۱۳۹۲/۰۵/۲۲ ۱۰:۲۹

با سلام؛ در قسمت تبلیغات سایتی کاربر بازه نمایش تبلیغ خود را وارد می‌کنه و من در مدیریت تائید می‌کنم در زمان تائید در قسمت تبلیغات نامه ای برای ایشان ارسال می‌کنم که از این تاریخ تبلیغ شما نمایش داده می‌شود
مشکل من در اینجاست که می‌خواهم در زمان خاتمه تاریخ تبلیغ هم نامه ای برای شخص ارسال شود و به ایشان اطلاع داده شود که تبلیغ پایان یافته است.
راه حلی که من برای آن در نظر گرفته بودم این است که در زمان تائید اولیه تبلیغ با کوارتز یک Job درست کنم و با futuredate از وی بخواهم در روز اتمام تبلیغ ایمیل بزند
سوال من این است که :
1- آیا فعال بودن یک job آن هم در بازه هایی مانند 6 ماه یا بیشتر به تعداد بالا سرعت سرور را کاهش نمی‌دهد؟
2- از آنجا که تبلیغ‌ها با هم متفاوت است و به فراخور آن نامه‌های ارسالی متفاوت است چگونه می‌توانم به ازای هر تبلیغ به صورت داینامیک Job مربوطه را تخصیص دهم یعنی آن که دستی with identity نزنم(هارد کد نزنم)
پیشاپیش سپاسگزارم

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۲۲ ۱۰:۴۶

نیازی نیست به ازای هر تبلیغ یک جاب درست کنید. کلا یک جاب درست کنید که مثلا هر 5 دقیقه یکبار از دیتابیس کوئری بگیره. مواردی رو که منقضی شده (مقایسه تاریخ سرور با تاریخ فیلد انقضاء تبلیغ) پیدا کنه و براشون ایمیل ارسال کنه. به این صورت سربار خیلی خیلی کمی خواهید داشت.

نویسنده: منصوره
تاریخ: ۱۳۹۲/۱۱/۰۲ ۹:۵۴

سلام من تازه برنامه نویسی asp رو شروع کردم. در سایتی که در حال طراحی آن هستم یک مدیر سیستم دارم و یک سری مشتری که هرکدام نام کاربری و پسورد خودشان را دارند. می‌خواهم برنامه ام به گونه ای باشد که هرروز به طور خودکار بررسی کند اگر تولد یک مشتری است به او اس ام اس دهد یا اگر قرارداد مشتری در حال اتمام است به او اس ام اس دهد و به اطلاعش برساند. من برای این کار یک برنامه کوارتز نوشته ام ولی مشکلی که دارم این است که نمی‌دونم این برنامه را از کجا صدا بزنم. نمی‌تونم هر مشتری که لاگین می‌شود این برنامه رو صدا بزنم. چون ممکن است در طول روز مشتری چندین دفعه لاگین شود. می‌خواهم به گونه ای باشد که دفعه اول که مدیر سیستم وارد سایت شد همون بار اول جاب فراخوانی شود و تا بی نهایت ادامه یابد.جاب را به گونه ای تنظیم کرده ام که روزی یک بار اجرا می‌شود. اگر جاب را همون دفعه اولی که مدیر سیستم وارد می‌شود

صدا بزنم درست است؟ بعد از اینکه مدیر سیستم صفحه اش را می‌بندد اجرای برنامه متوقف می‌شود یا ادامه می‌یابد؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۰۲ ۱۰:۲

در نظرات قسمت اول بهش اشاره شده. این کدها باید در Application_Start فایل global.asax.cs فراخوانی شوند.

نویسنده: منصوره
تاریخ: ۱۳۹۲/۱۱/۰۲ ۱۱:۱۹

با تشکر ولی این فایل ISchedule را نمی‌شناسد!؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۰۲ ۱۱:۲۹

دومین قطعه کدی که در مقاله‌ی قسمت اول معرفی شده تعریف اینترفیس ISchedule است. قطعه کد اول HelloJob هست از بالا و قطعه کد بعدی ISchedule.

نویسنده: منصوره
تاریخ: ۱۳۹۲/۱۱/۰۲ ۱۱:۳۴

من کلاس ISchedule رو تعریف کردم. برای تعریف global روی نام پروژه راست کلیک کردم و add neww item را انتخاب کردم و از آنجا global را انتخاب کرده ام . آیا این روش درست است؟

```
<%@ Application Language="C#" %>
<script runat="server">
void Application_Start(object sender, EventArgs e)
{
}
</script>
```

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

همانطور که قبلا عنوان شد، تعاریف myTask.Run را در Application_Start قرار بدید.

نویسنده: میثم99
تاریخ: ۱۳۹۲/۱۲/۰۲ ۱۹:۳۱

با سلام؛ اگه می‌توانید یک مثال از Listener بگذارید ممنون می‌شوم. به این صورت که چک کند اگر کار قبلی در حال اجرا هست دیگر اجرا نشود و اگر کار به اتمام رسیده بود شروع به اجرای تابع مورد نظر کند.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۲ ۲۱:۳۹

متد GetCurrentlyExecutingJobs لیست jobهای موازی در حال اجرا را می‌دهد:

```
var jobs = scheduler.GetCurrentlyExecutingJobs();
```

از نتیجه‌ی آن برای تصمیم گیری استفاده کنید؛ مثلا اگر این لیست خالی یا نال بود آنگاه job جاری اجرا شود.

نویسنده: میثم99
تاریخ: ۱۳۹۲/۱۲/۰۲ ۲۱:۴۷

ممنون
IJobListener رو پیاده کردم. یک متد داره به نام JobWasExecuted
در ابتدا یک متغییر استاتیک بولین تعریف کردم. وقتی تابع اجرا میشه یک میشود. و در JobWasExecuted اون متغییر رو 0 میکنم.
تو تست خوب جواب داد. کاملا درست کار میکنه وقتی کار اتمام پیدا کنه اجرا میشه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۲ ۲۲:۰

روش دیگر استفاده از MergedJobDataMap هست برای اشتراک گذاری داده‌ها در context آن.

نویسنده: میثم99
تاریخ: ۱۳۹۲/۱۲/۰۲ ۲۲:۸

با تابع GetCurrentlyExecutingJobs() هم انجام شد پیاده سازیش هم راحتره ممنون.

نویسنده: منصوره
تاریخ: ۱۳۹۲/۱۲/۱۹ ۱۳:۸

سلام من در برنامم برای ارسال اس ام اس از کوارتز استفاده کردم درست هم کار میکنه ولی مشکلی که داره اینه که می‌خوام روزی یکبار عمل ارسال اس ام اس انجام بشه بنابراین trigger را به صورت زیر تعریف کرده ام.

```
ITrigger trigger =
TriggerBuilder.Create().WithIdentity("trigger1").StartAt(starttime).WithSimpleSchedule(x =>
x.WithIntervalInHours(24).RepeatForever()).Build ();
```

ولی انگار که این کد کار نمی‌کنه چون تقریبا ساعتی دو بار داره اس ام اس رو میفرسته لطفا راهنمایی ام کنید که مشکل چیه ؟ در ضمن ارسال اس ام اس در قسمت application_start به صورت زیر تعریف شده

```
double h = DateTime.Now.Hour;
  h += 7.5;
  if (h >= 8)
  {
SchedulerDemo.Interfaces.ISchedule mytask = new SchedulerDemo.Jobs.Sendschedule();
mytask.run();
  }
```

اضافه کردن 7.5 به ساعت سیستم به دلیل این است که مطابق با ساعت ایران شود. با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۹ ۱۳:۳۲

در متد application_start لاگ کنید برنامه در چه زمان‌هایی ری‌استارت شده.

نویسنده: منصوره
تاریخ: ۱۳۹۲/۱۲/۱۹ ۱۴:۶

ببخشید من درست متوجه منظورتون نشدم. لاگ سایت را از سرور گرفتم ولی چیزی ازش متوجه نشدم. آیا منظوظرتون این لاگه ؟ میشه بیشتر توضیح بدید ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۹ ۱۴:۹

در یک فایل متنی ساده ثبت کنید، متد application_start چندبار در طول روز فراخوانی شده یا حتی در طول یک ساعت (هر

بار که این متد فراخوانی می‌شود، جایی آن‌را ثبت کنید). استفاده از ELMAH هم برای اینکار مفید است. ممکن است برنامه بیش از حد ری‌استارت می‌شود.

نویسنده: افشین
تاریخ: ۱۳۹۲/۱۲/۲۰ ۱۱:۴

یه برنامه نوشتم، که کاربر میتونه زمان رو انتخاب و Job رو ایجاد کنه... لیست Jobها هم در CheckedBoxList نشون داده میشه تا کاربر هر کدوم رو که دوست داشتم موقتا Pause و Resume کنه.

jobها بدون نقص کار میکنن منتها Jobها بعد از Pause و Resume دیگه فعال نمیشن. مشکل از کجاست؟

```
public void Run()
        {
...
  chkJobs.Items.Add(job.Key, true);
}
```

```
private void chkJobs_ItemCheck(object sender, ItemCheckEventArgs e)
        {
            int index = e.Index;
            var sch = Scheduler.GetScheduler();
            foreach (object item in chkJobs.Items)
            {
                JobKey key = (JobKey)item;

                if (e.NewValue == CheckState.Unchecked)
                {
                    sch.PauseJob(key);
                }
                else
                {
                        sch.ResumeJob(key);
                }
            }
        }
```

```
public class Jobs : IJob
        {
            public void Execute(IJobExecutionContext context)
            {
                JobDataMap datamap = context.JobDetail.JobDataMap;
                switch (Convert.ToInt32(datamap.GetString("OperationType")))
                {
                    case 1:
                        MessageBox.Show(datamap.GetString("OperationValue"));
                        break;
                }

            }
        }
```

نویسنده: منصوره
تاریخ: ۱۳۹۲/۱۲/۲۱ ۹:۵۵

سلام من این کارو انجام دادم متد application_start مرتب ری استارت میشه تقریبا هر 20 دقیقه یه بار. سرور ما یک هاست اشتراکی است. میشه لطفا راهنمایی کنید که مشکل چی می‌تونه باشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۱ ۱۰:۷

- نکات مطلب « چه زمان‌هایی یک برنامه‌ی ASP.NET ری استارت می‌شود؟ » را بررسی کنید.

- همچنین اگر هاست نامبرده تمام سایت‌ها را با یک Application pool مدیریت می‌کند، کرش یکی از چند ده سایت دیگر می‌تواند سبب ری‌استارت شدن سایت شما هم بشود؛ چون برنامه‌ها از همه ایزوله نشده‌اند. راه حل آن ایجاد یک Application pool مجزا به ازای هر سایت هست (توسط هاست‌دار).

نویسنده: منصوره
تاریخ: ۱۳۹۲/۱۲/۲۱ ۱۰:۱۶

ممنون از جوابتون این اتفاق مرتب می‌افتد یعنی هر 17,18 دقیقه یکبار. اگر مشکل از کرش سایتهای دیگه باشه آیا این طور مرتب برنامه ری استارت میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۱ ۱۰:۱۹

بله. زمانیکه تمام سایت‌ها با یک Application pool مدیریت می‌شوند، تمام آن‌ها توسط یک وهله از w3wp.exe اجرا خواهند شد. با تعریف Application poolهای مجزا، هر سایت، یک وهله‌ی مجزا از w3wp.exe را به خود اختصاص خواهد داد. یعنی اگر Task manager سرور را بررسی کنید، به ازای هر سایت، یک w3wp.exe با pid مجزا قابل مشاهده است. به این ترتیب اگر pid=1234 کرش کرد، تاثیری روی pid=4321 نخواهد داشت.

نویسنده: منصوره
تاریخ: ۱۳۹۲/۱۲/۲۱ ۱۰:۳۸

من با سرور تماس گرفتم گفتن که همه سایتها pool جدا دارن. توی مقاله هم که نگاه کردم هیچکدوم از موارد ذکر شده برای سایت ما صدق نمی‌کرد. نمیدونید مشکل دیگه می‌تونه ناشی از چی باشه؟ تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۱ ۱۱:۵

با تنظیمات Application pool همه کار می‌توان کرد. مثلا تنظیم کرد سایت هر 20 دقیقه یکبار پس از بیکاری از حافظه خارج شود. یا تنظیم کرد اگر مصرف حافظه‌ی برنامه به یک حد مشخصی رسید، سایت ری استارت شود. بررسی آن نیاز به بررسی کدهای شما و تنظیمات سرور دارد.

عنوان: **انجام کارهای زمانبندی شده در برنامه‌های ASP.NET توسط DNT Scheduler**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۶ ۱۱:۴۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Schedule

اگر به دو مطلب استفاده از Quartz.Net (^ و ^) و خصوصا نظرات آن دقت کرده باشید به این نتیجه خواهید رسید که ... این کتابخانه‌ی در اصل جاوایی گنگ طراحی شده‌است. در سایت جاری برای انجام کارهای زمانبندی شده (مانند ارسال ایمیل‌های روزانه خلاصه مطالب، تهیه خروجی PDF و XML سایت، تبدیل پیش نویس‌ها به مطالب، بازسازی ایندکس‌های جستجو و امثال آن) از یک Thread timer استفاده می‌شود که حجم نهایی کتابخانه‌ی محصور کننده و مدیریت کننده‌ی وظایف آن جمعا 8 کیلوبایت است؛ متشکل از ... سه کلاس. در ادامه کدهای کامل و نحوه‌ی استفاده از آن را بررسی خواهیم کرد.

دریافت کتابخانه DNT Scheduler و مثال آن

DNTScheduler.7z

در این بسته، کدهای کتابخانه‌ی DNT Scheduler و یک مثال وب فرم را، ملاحظه خواهید کرد. از این جهت که برای ثبت وظایف این کتابخانه، از فایل global.asax.cs استفاده می‌شود، اهمیتی ندارد که پروژه‌ی شما وب فرم است یا MVC. با هر دو حالت کار می‌کند.

نحوه‌ی تعریف یک وظیفه‌ی جدید

کار با تعریف یک کلاس و پیاده سازی ScheduledTaskTemplate شروع می‌شود:

```
public class SendEmailsTask : ScheduledTaskTemplate
```

برای نمونه :

```
using System;

namespace DNTScheduler.TestWebApplication.WebTasks
{
    public class SendEmailsTask : ScheduledTaskTemplate
    {
        /// <summary>
        /// اگر چند جاب در یک زمان مشخص داشتید، این خاصیت ترتیب اجرای آن‌ها را مشخص خواهد کرد
        /// </summary>
        public override int Order
        {
            get { return 1; }
        }

        public override bool RunAt(DateTime utcNow)
        {
            if (this.IsShuttingDown || this.Pause)
                return false;

            var now = utcNow.AddHours(3.5);
            return now.Minute % 2 == 0 && now.Second == 1;
        }

        public override void Run()
        {
            if (this.IsShuttingDown || this.Pause)
                return;

            System.Diagnostics.Trace.WriteLine("Running Send Emails");
        }

        public override string Name
        {
            get { return "ارسال ایمیل"; }
        }
    }
}
```

- در اینجا Order، ترتیب اجرای وظیفه‌ی جاری را در مقایسه با سایر وظیفه‌هایی که قرار است در یک زمان مشخص اجرا شوند، مشخص می‌کند.
- متد RunAt ثانیه‌ای یکبار فراخوانی می‌شود (بنابراین بررسی now.Second را فراموش نکنید). زمان ارسالی به آن UTC است و اگر برای نمونه می‌خواهید بر اساس ساعت ایران کار کنید باید 3.5 ساعت به آن اضافه نمائید. این مساله برای سرورهایی که خارج از ایران قرار دارند مهم است. چون زمان محلی آن‌ها برای تصمیم گیری در مورد زمان اجرای کارها مفید نیست.
در متد RunAt فرصت خواهید داشت تا منطق زمان اجرای وظیفه‌ی جاری را مشخص کنید. برای نمونه در مثال فوق، این وظیفه هر دو دقیقه یکبار اجرا می‌شود. یا اگر خواستید اجرای آن فقط در سال 23 و 33 دقیقه هر روز باشد، تعریف آن به نحو ذیل خواهد بود:

```
public override bool RunAt(DateTime utcNow)
        {
            if (this.IsShuttingDown || this.Pause)
                return false;

            var now = utcNow.AddHours(3.5);
            return now.Hour == 23 && now.Minute == 33 && now.Second == 1;
        }
```

- خاصیت IsShuttingDown موجود در کلاس پایه ScheduledTaskTemplate، توسط کتابخانه‌ی DNT Scheduler مقدار دهی می‌شود. این کتابخانه قادر است زمان خاموش شدن پروسه‌ی فعلی IIS را تشخیص داده و خاصیت IsShuttingDown را true کند. بنابراین در حین اجرای وظیفه‌ای مشخص، به مقدار IsShuttingDown دقت داشته باشید. اگر true شد، یعنی فقط 30 ثانیه وقت دارید تا کار را تمام کنید.
خاصیت Pause هر وظیفه را برنامه می‌تواند تغییر دهد. به این ترتیب در مورد توقف یا ادامه‌ی یک وظیفه می‌توان تصمیم گیری کرد. خاصیت ScheduledTasksCoordinator.Current.ScheduledTasks، لیست وظایف تعریف شده را در اختیار شما قرار می‌دهد.
- در متد Run، منطق وظیفه‌ی تعریف شده را باید مشخص کرد. برای مثال ارسال ایمیل یا تهیه‌ی بک آپ.
- Name نیز نام وظیفه‌ی جاری است که می‌تواند در گزارشات مفید باشد.

همین مقدار برای تعریف یک وظیفه کافی است.

نحوه‌ی ثبت و راه اندازی وظایف تعریف شده

پس از اینکه چند وظیفه را تعریف کردیم، برای مدیریت بهتر آن‌ها می‌توان یک کلاس ثبت و معرفی کلی را مثلا به نام ScheduledTasksRegistry ایجاد کرد:

```
using System;
using System.Net;

namespace DNTScheduler.TestWebApplication.WebTasks
{
    public static class ScheduledTasksRegistry
    {
        public static void Init()
        {
            ScheduledTasksCoordinator.Current.AddScheduledTasks(
                new SendEmailsTask(),
                new DoBackupTask());

            ScheduledTasksCoordinator.Current.OnUnexpectedException = (exception, scheduledTask) =>
            {
                //todo: log the exception.
                System.Diagnostics.Trace.WriteLine(scheduledTask.Name + ":" + exception.Message);
            };

            ScheduledTasksCoordinator.Current.Start();
        }

        public static void End()
        {
```

```
            ScheduledTasksCoordinator.Current.Dispose();
        }

        public static void WakeUp(string pageUrl)
        {
            try
            {
                using (var client = new WebClient())
                {
                    client.Credentials = CredentialCache.DefaultNetworkCredentials;
                    client.Headers.Add("User-Agent", "ScheduledTasks 1.0");
                    client.DownloadData(pageUrl);
                }
            }
            catch (Exception ex)
            {
                //todo: log ex
                System.Diagnostics.Trace.WriteLine(ex.Message);
            }
        }
    }
}
```

- شیء ScheduledTasksCoordinator.Current، نمایانگر تنها وهله‌ی مدیریت وظایف برنامه است.
- توسط متد ScheduledTasksCoordinator.Current.AddScheduledTasks، تنها کافی است کلاس‌های وظایف مشتق شده از ScheduledTaskTemplate، معرفی شوند.
- به کمک متد ScheduledTasksCoordinator.Current.Start، کار Thread timer برنامه شروع می‌شود.
- اگر در حین اجرای متد Run، استثنایی رخ دهد، آن‌را توسط یک Action delegate به نام ScheduledTasksCoordinator.Current.OnUnexpectedException می‌توانید دریافت کنید. کتابخانه‌ی DNT Scheduler برای اجرای وظایف، از یک ترد با سطح تقدم Below normal استفاده می‌کند تا در حین اجرای وظایف، برنامه‌ی جاری با اخلال و کندی مواجه نشده و بتواند به درخواست‌های رسیده پاسخ دهد. در این بین اگر استثنایی رخ دهد، می‌تواند کل پروسه‌ی IIS را خاموش کند. به همین جهت این کتابخانه کار try/catch استثناهای متد Run را نیز انجام می‌دهد تا از این لحاظ مشکلی نباشد.
- متد ScheduledTasksCoordinator.Current.Dispose کار مدیر وظایف برنامه را خاتمه می‌دهد.
- از متد WakeUp تعریف شده می‌توان برای بیدار کردن مجدد برنامه استفاده کرد.

استفاده از کلاس ScheduledTasksRegistry تعریف شده

پس از اینکه کلاس ScheduledTasksRegistry را تعریف کردیم، نیاز است آن‌را به فایل استاندارد global.asax.cs برنامه به نحو ذیل معرفی کنیم:

```
using System;
using System.Configuration;
using DNTScheduler.TestWebApplication.WebTasks;

namespace DNTScheduler.TestWebApplication
{
    public class Global : System.Web.HttpApplication
    {
        protected void Application_Start(object sender, EventArgs e)
        {
            ScheduledTasksRegistry.Init();
        }

        protected void Application_End()
        {
            ScheduledTasksRegistry.End();
            //نکته مهم این روش نیاز به سرویس پینگ سایت برای زنده نگه داشتن آن است
            ScheduledTasksRegistry.WakeUp(ConfigurationManager.AppSettings["SiteRootUrl"]);
        }
    }
}
```

- متد ScheduledTasksRegistry.Init در حین آغاز برنامه فراخوانی می‌شود.
- متد ScheduledTasksRegistry.End در پایان کار برنامه جهت پاکسازی منابع باید فراخوانی گردد.

همچنین در اینجا با فراخوانی ScheduledTasksRegistry.WakeUp، می‌توانید برنامه را مجددا زنده کنید! IIS مجاز است یک سایت ASP.NET را پس از مثلا 20 دقیقه عدم فعالیت (فعالیت به معنای درخواست‌های رسیده به سایت است و نه کارهای پس زمینه)، از حافظه خارج کند (این عدد در application pool برنامه قابل تنظیم است). در اینجا در فایل web.config برنامه می‌توانید آدرس یکی از صفحات سایت را برای فراخوانی مجدد تعریف کنید:

```
<?xml version="1.0"?>
<configuration>
  <appSettings>
      <add key="SiteRootUrl" value="http://localhost:10189/Default.aspx" />
  </appSettings>
</configuration>
```

همینکه درخواست مجددی به این صفحه برسد، مجددا برنامه توسط IIS بارگذاری شده و اجرا می‌گردد. به این ترتیب وظایف تعریف شده، در طول یک روز بدون مشکل کار خواهند کرد.

گزارشگیری از وظایف تعریف شده

برای دسترسی به کلیه وظایف تعریف شده، از خاصیت ScheduledTasksCoordinator.Current.ScheduledTasks استفاده نمائید:

```
var jobsList = ScheduledTasksCoordinator.Current.ScheduledTasks.Select(x => new
{
   TaskName = x.Name,
   LastRunTime = x.LastRun,
   LastRunWasSuccessful = x.IsLastRunSuccessful,
   IsPaused = x.Pause,
}).ToList();
```

لیست حاصل را به سادگی می‌توان در یک Grid نمایش داد.

نظرات خوانندگان

نویسنده: محمد رعیت پیشه
تاریخ: ۱۳۹۲/۱۲/۲۶ ۱۹:۲۲

آیا یکی از کاربردهای این مطلب می‌تونه مثلا ارسال یک ایمیل یک هفته قبل از اتمام زمان شارژ کاربری باشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۶ ۱۹:۳۷

بله. می‌توانید یک وظیفه‌ی جدید تعریف کنید که هر شب ساعت مثلا 11 و 15 دقیقه اجرا شود (نحوه‌ی تعریف متد RunAt). سپس در متد Run آن یک کوئری از دیتابیس گرفته، لیست موارد مدنظر را واکشی کرده و به آن‌ها ایمیل بزنید.

نویسنده: پیمان مهربانی
تاریخ: ۱۳۹۲/۱۲/۲۷ ۱۱:۴۷

کتابخانه quartz scheduler سبکی بود اما نگرانی هایی مانند پیش آمدن همزمانی در اجرای وظایف را دارم.پیش از این از کتابخانه Scheduler خود ویندوز و یا Jonهای SQL Server هم بهره برد.

این روش چه مزایا و معایبی نسبت به روش‌های موجود دارد و آیا توصیه شده برای استفاده در پروژه‌های بزرگ هست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۷ ۱۲:۲

- «پیش آمدن همزمانی در اجرای وظایف»
خاصیت Order را برای وظایفی که قرار است در یک زمان مشخص اجرا شوند، مقدار دهی کنید. 1و 2 و 3 و الی آخر.
- «حتی می‌توان از Scheduler خود ویندوز و یا Jonهای SQL Server هم بهره برد».
بله. به شرطی‌که سرور در اختیار شما باشد و دسترسی کافی برای انجام اینکار را داشته باشید. البته در این حالت خاص، مدیریت آن یکپارچه با یک برنامه‌ی وب نیست.
در سرورهای اشتراکی روش ارائه شده در این مطلب بدون نیاز به سطح دسترسی خاصی کار می‌کند. ضمنا برای ASP.NET نوشته شده است و این قابلیت را دارد که به شما اعلام کند مثلا تا 30 ثانیه دیگر برنامه از سرور unload می‌شود؛ توسط خاصیت IsShuttingDown. همچنین حق تقدم ترد آن طوری تنظیم شده که سبب اختلال در عملیات و عملکرد متداول سایت نشود.
- «آیا توصیه شده برای استفاده در پروژه‌های بزرگ هست؟»
یک به اشتراک گذاری بود از قسمتی از کدهای زیر ساخت سایت جاری که هم اکنون مورد استفاده است (مقدمه بحث).

نویسنده: سلمان کاظمی
تاریخ: ۱۳۹۲/۱۲/۲۸ ۱:۳۲

سوالی که واسه من پیش اومده اینه که من یک نرم افزار Web Form دارم . هدفم اینه که یک روز قبل از تاریخ تولد اعضا ایمیلی با عنوان تولدتان مبارک ارسال شه .حالا من باید این کدهای گفته شده را در برنامه خود بیاورم یا یک وب سرویس بنویسم که اینکارو انجام بده؟اگه بخوام که در برنامه انجام بشه خوب کجا باید نوشت ؟ یا به عبارتی دیگه باید حتما برنامه اجرا بشه و اگه برنامه اصلا Run نشه چی میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۸ ۸:۵۹

- وب سرویس فقط با یک درخواست رسیده کار می‌کند. کار کتابخانه‌ی فوق، اجرا در پس زمینه‌ی برنامه به صورت مداوم است.
- فقط دو حالت وجود دارد که برنامه اجرا نشود:

الف) protected void Application_End فراخوانی شود. متد WakeUp نوشته شده برای این منظور و راه اندازی مجدد برنامه توسط آن، کافی است.
ب) کل سرور ری استارت شود (نه فقط برنامه). در این حالت کافی است آدرس برنامه را به یکی از سرویس‌هایی که هر از چندگاهی برنامه را ping می‌کنند، معرفی کنید.

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۳/۰۱/۰۱ ۲۳:۲۳

خیلی خیلی ممنون ... کتابخانه‌ی ساده و مفیدی است.
سوال من اینه که میشه کاری کرد که آدرس روت تعریف شده در فایل کانفیگ جهت بیدار شدن IIS را در کد نوشت مثل کد زیر :

```
var urlToWakeup = Request.Url.Scheme + Uri.SchemeDelimiter + Request.Url.Host +
            (Request.Url.IsDefaultPort ? "" : ":" + Request.Url.Port);
```

که مجبور نباشی در هر سایتی که طراحی میکنیم آدرس رو در فایل کانفیگ ست کنیم !
پی نوشت : من این کار رو در متد Application_End نوشتم و خطای در دسترس نبودن Request رو دریافت کردم ! (هر چند که میدونستم این خطا رو میده !)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۰۲ ۰:۵۱

می‌شود در Application_BeginRequest اطلاعات آدرس ریشه سایت را در یک متغیر استاتیک ذخیره کرد. مقدار آن در Application_End قابل استفاده است:

```
namespace TestApp
{
    public static class App
    {
        public static string SiteRootUrl;
    }

    public class TestApplication : HttpApplication
    {
        protected void Application_BeginRequest(object sender, EventArgs e)
        {
            if (string.IsNullOrWhiteSpace(App.SiteRootUrl))
            {
                App.SiteRootUrl = Request.Url.GetLeftPart(UriPartial.Authority) +
Request.ApplicationPath;
            }
        }

        protected void Application_End()
        {
            // use App.SiteRootUrl
        }
    }
}
```

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۳/۰۱/۰۲ ۱۳:۲

خیلی خیلی ممنون :) ... اگه میشه نسخه‌ی nuget کتابخانه را هم بذارید

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۳/۰۱/۲۵ ۱۴:۳۱

1.برای زنده نگهداشتن سایت میشه از خود همین کتابخانه استفاده کرد؟
2.برای چه تعداد جاب بنظرتون منطقیه که استفاده بشه؟
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۲۵ ۱۶:۴۳

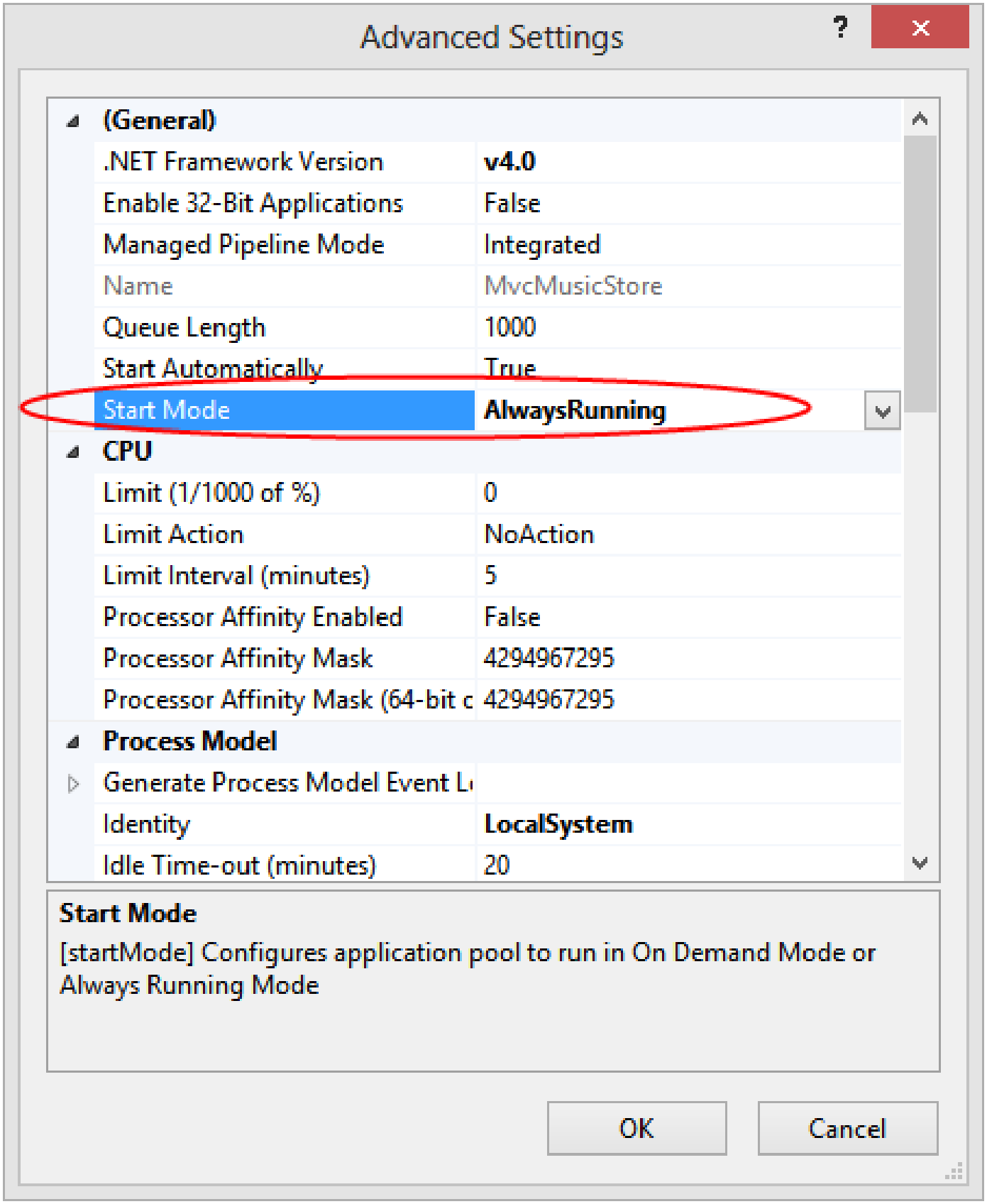
- کمی بالاتر توضیح دادم « فقط دو حالت وجود دارد که برنامه اجرا نشود: ... »
- این مورد فقط بستگی به توان سرور شما دارد.

نویسنده: fss
تاریخ: ۱۳۹۳/۰۱/۲۸ ۹:۶

در حالتی که "کل سرور ری استارت شود (نه فقط برنامه) " ، بعد از بالا آمدن سرور و شروع به کار IIS، تابع Application_Start کار شروع برنامه را خود به خود انجام نمی‌دهد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۲۸ ۹:۱۴

نه تا زمانیکه اولین درخواستی به برنامه برسد.
کل سرور ری استارت شده. IIS برنامه را فقط زمانی مجددا بارگذاری می‌کند که درخواست نمایش یکی از قسمت‌های سایت به آن ارسال شود.
البته IISهای جدید قابلیت Auto-Start هم دارند؛ ولی باید در تنظیمات Application pool برنامه انتخاب شود:



همچنین Application Initialization Module نیز برای اجرا خودکار برنامه پس از ری‌استارت سرور طراحی شده .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۱۷ ۹:۶

یک نکته‌ی تکمیلی

در دات نت *4.5.2* ، متدی به نام HostingEnvironment.QueueBackgroundWorkItem اضافه شده‌است تا درخواست اجرای کارهای پس زمینه در ASP.NET به سادگی و همچنین با اطمینان بیشتری قابل انجام باشد.

نویسنده: مهرداد
تاریخ: ۱۳۹۳/۰۲/۱۷ ۱۳:۴۰

یعنی با استفاده از این متد جدیدی که در دات نت 4.5.2 اضافه شده ، دیگه نیازی به Quartz.net و یا DNT scheduler نیست ؟ و میتوان برای کارهایی که نیاز به زمانبندی دارند از این متد استفاده کرد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۱۷ ۱۳:۴۲

فقط قسمت مدیریت تردهای آن با یک سطر QueueBackgroundWorkItem جایگزین می‌شود. مابقی قسمت‌های آن (مانند اینکه یک WorkItem چه زمانی در Queue قرار گیرد) تفاوتی نمی‌کند و مانند قبل است.

نویسنده: مهرداد
تاریخ: ۱۳۹۳/۰۲/۱۷ ۱۴:۱۰

منظور شما این خط هست ؟

```
ScheduledTasksCoordinator.Current.AddScheduledTasks(
                new SendEmailsTask(),
                new DoBackupTask());
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۱۷ ۱۴:۲۳

این‌ها باقی خواهند ماند. قسمت new Thread و مدیریت آن جایگزین می‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۹ ۹:۲۵

یک نکته

مثالی از نحوه‌ی استفاده از متد جدید HostingEnvironment.QueueBackgroundWorkItem
+ یک مثال رسمی

نویسنده: Aria
تاریخ: ۱۳۹۳/۰۳/۱۹ ۲۳:۱۰

با سلام
من روی یه پروژه مالی دارم با ASP WebForm کار میکنم و میخواستم در اول هر ماه یک سری دستورات ویرایش یا update را که به صورت رکورد در یک جدول ذخیره می‌کنم انجام بشن. سوالم اینه که برای این که مجموعه دستورات من انجام بشن باید برنامه جتما Run باشه و چه جوری میشه اونو به صورت پس زمینه در حال اجرا نگه داشت.
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۲۰ ۰:۱۹

- به اندازه کافی در نظرات این بحث در مورد زنده نگه داشتن یک برنامه ASP.NET بحث شده. کمی وقت بگذارید و آن‌ها را مطالعه کنید.
+ اگر برنامه مالی است، احتمالا دسترسی کاملی به سرور و همچنین SQL Server (اگر با آن کار می‌کنید) دارید. در این حالت برای به روز رسانی زمانبندی شده‌ی چند رکورد شاید بهتر باشد از سرویس معروف و همیشه در حال اجرای SQL Server agent استفاده کنید. در اینجا نیز می‌شود یک job را که متشکل از دستورات T-SQL است، در فواصل زمانی مشخصی اجرا کرد.

نویسنده: میثم
تاریخ: ۱۳۹۳/۰۳/۲۵ ۱۳:۴۸

سلام.
منظورتون فقط همین یک خط توی متد Start کلاس ScheduledTasksCoordinator بود که تغییر میکنه؟
یعنی فقط متد Start به شکل زیر بازنویسی میشه دیگه؟

```
public void Start()
        {
            _timer.OnTimerCallback = () =>
            {
                var now = DateTime.UtcNow;
                var taskToRun = _tasks.Where(x => !x.IsRunning && x.RunAt(now)).OrderBy(x =>
x.Order).ToList();
                if (_isShuttingDown || !taskToRun.Any())
                    return;

                HostingEnvironment.QueueBackgroundWorkItem(x => taskAction(taskToRun));
            };
        }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۲۵ ۱۵:۱

- بله. قسمت‌های HostingEnvironment.RegisterObject و IRegisteredObject آن هم باید حذف شوند چون در QueueBackgroundWorkItem وجود دارند و یک CancellationToken را تنظیم می‌کند.
+ زمانیکه از DNTScheduler استفاده می‌کنید، عملا نیازی به QueueBackgroundWorkItem ندارید. چون نکته‌ی HostingEnvironment.RegisterObject و IRegisteredObject در آن لحاظ شده. این نکته که خاموش شدن IIS را گزارش می‌کند، چند سال قبل، توسط یکی از اعضای قبلی تیم ASP.NET منتشر شده بود . دقیقا از همین نکته در QueueBackgroundWorkItem استفاده شده.

به صورت خلاصه، DNTScheduler با دات نت 4 به بعد سازگار است و نکات QueueBackgroundWorkItem دات نت 4.5.2 را به صورت توکار پیاده سازی کرده‌است.

نویسنده: حسین
تاریخ: ۱۳۹۳/۰۴/۱۷ ۱۱:۲

سلام. اگر بخواهیم یک کار نسبتاً زمانبر که IO هم هست را توسط این کتابخانه در فواصل زمانی معین اجرا کنیم، میشه از async و await استفاده کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۷ ۱۱:۲۰

بله. از روش و کتابخانه معرفی شده در مطلب « استفاده از async و await در برنامه‌های کنسول و سرویس‌های ویندوز NT » استفاده کنید.

نویسنده: حمید حسین وند
تاریخ: ۱۳۹۳/۰۵/۰۹ ۱۰:۱۹

سلام
من برای این سری کارها از ویندوز سرویس استفاده می‌کنم. مثلا ویندوز سرویس من از ساعت 8 صبح شروع به کار می‌کنه و رویدادهایی مثل سالروز تولد رو با استفاده از پیامک به کاربران پیام تبریک ارسال می‌کنه.
مهمترین عاملی که باعث شد من از ویندوز سرویس استفاده کنم اجرای مداوم وهمیشگی بدون ارسال درخواست به وب سایت من بود. ولی فکر می‌کنم این کتابخانه شما هم مثل ویندوز سرویس عمل می‌کنه و خودش همیشه در حال اجراست.
حالا به نظرتون آیا از ویندوز سرویس استفاده کنم بهتره و یا اینکه از این کتابخانه استفاده کنم؟
ممنون

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۵/۰۹ ۱۱:۳

فرض کن داری از یک هاست اشتراکی استفاده می‌کنی. دسترسی ادمین هم روی سرور نداری برای اجرا سرویس ویندوز یا فرض کن در یک سازمان بهت گفتن ما فقط اجازه می‌دیم فایل‌های سایتت رو روی سرور کپی کنی. دسترسی بیشتری بهت نمی‌دیم. اون وقت چکار می‌کنی؟

نویسنده: Mohammad
تاریخ: ۱۳۹۳/۰۵/۱۵ ۲:۵۶

سلام. ممنون.

دوستان برای من وقتی به صورت ساعت می‌دم درست کار نمیکنه.مثلا من می‌خوام در ساعت دو و بیست و چهار دقیقه بامداد یه کاری رو انجام بده اینجوری نوشتم:در صورتی اگه بگم دو دقیقه دو دقیقه درست انجام میشه.به نظرتون مشکل از کجاست؟ممنون

```
var now = utcNow.AddHours(3.5);
    return now.Hour == 2 && now.Minute == 24 && now.Second == 1;
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۱۵ ۸:۴۶

مشکل از تنظیم نبودن ساعت سرور است. مقدار DateTime.UtcNow را روی سرور بررسی کنید و با مقدار واقعی تطابق بدید. بعد اختلافش را باید در همینجا اعمال کنید. مثلا اگر پس از بررسی متوجه شدید ساعت سرور یک ساعت عقب هست، now.Hour == 2 می‌شود now.Hour == 1 و امثال این نوع محاسبات.

نویسنده: علی
تاریخ: ۱۳۹۳/۰۶/۱۵ ۱۸:۳۴

ممنون. مفید بود. اگه بخواهیم یه کاری مثلا هر سه روز یکبار انجام بشه باید چه جوری زمان رو تعیین کنیم؟ ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۵ ۱۹:۵۷

```
public override bool RunAt(DateTime utcNow)
        {
            if (this.IsShuttingDown || this.Pause)
                return false;

            var now = utcNow.AddHours(3.5);
            return (now.Day % 3 == 0) && (now.Hour == 0 && now.Minute == 1 && now.Second == 1);
        }
```

نویسنده: مهدی
تاریخ: ۱۳۹۳/۰۶/۱۷ ۲۲:۲۲

یه سوال:اختلاف زمانی ساعت گرینویچ با تهران همیشه 3:30 هست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۷ ۲۳:۸

اگر سرور شما تنظیمات daylight saving time صحیحی داشته باشد، بله. اگر نه، خودتان این یک ساعت تفاوت را 6 ماه یکبار باید محاسبه و اعمال کنید.

نویسنده: آرام
تاریخ: ۱۳۹۳/۰۶/۱۸ ۹:۵۷

سلام.
چرا هیچ ارجاعی به متد Stop در کلاس JobsRunnerTimer وجود نداره؟ برای این کار دلیلی دارید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۸ ۱۰:۱۴

نیازی نیست. چون طول عمر کل این ماژول دقیقا معادل طول عمر برنامه‌ی وب است. خاتمه‌ی آن هم به صورت خودکار با از حافظه خارج کردن AppDomain برنامه توسط IIS انجام می‌شود. تا زمانیکه برنامه در حال اجرا است این ماژول هم به همین ترتیب. هر زمان که IIS تصمیم به خاتمه‌ی برنامه گرفت، نه این ماژول، هیچ ماژول دیگری هم فرصت مقاومت پیدا نمی‌کند و راسا به همراه AppDomain جاری خاتمه می‌یابد.

عنوان: **انجام کارهای پس زمینه در ASP.NET 4.5.2**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۰۲ ۱۰:۲۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Schedule, MVC

دات نت 4.5.2 قابلیت توکاری را به نام در صف قرار دادن یک کار پس زمینه، اضافه کرده‌است که در ادامه خلاصه‌ای از آن‌را مرور خواهیم کرد.

روش متداول ایجاد کارهای پس زمینه

ساده‌ترین روش انجام کارهای پس زمینه در برنامه‌های دات نتی، استفاده از متدهایی هستند که یک ترد جدید را ایجاد می‌کنند مانند Task.Run, Task.Factory.StartNew, Delegate.BeginInvoke, ThreadPool.QueueUserWorkItem و امثال آن. اما ... این روش‌ها در برنامه‌های ASP.NET ایده‌ی خوبی نیستند!
از این جهت که موتور ASP.NET در این حالات اصلا نمی‌داند که شما کار پس زمینه‌ای را ایجاد کرده‌اید. به همین جهت اگر پروسه‌ی برنامه پس از مدتی recycle شود ، تمام کارهای پس زمینه‌ی موجود نیز از بین خواهند رفت.

معرفی HostingEnvironment.QueueBackgroundWorkItem

متد HostingEnvironment.QueueBackgroundWorkItem به دات نت 4.5.2 اضافه شده‌است تا بتوان توسط آن یک کار پس زمینه را توسط موتور ASP.NET شروع کرد و نه مانند قبل، بدون اطلاع آن. البته باید دقت داشت که این کارهای پس زمینه مستقل از هر نوع درخواستی اجرا می‌شوند.
در این حالت چون موتور ASP.NET از وجود کار پس زمینه‌ی آغاز شده مطلع است، در صورت فرا رسیدن زمان recycle شدن برنامه، کل AppDomain را به یکباره نابود نخواهد کرد. البته این مورد فقط به این معنا است که در صورت فرا رسیدن زمان recycle شدن پروسه، با تنظیم یک CancellationToken، اطلاع رسانی خواهد کرد. در این حالت حداکثر 30 ثانیه فرصت خواهید داشت تا کارهای پس زمینه را بدون مشکل خاتمه دهید. اگر کار پس زمینه در این مدت به پایان نرسد، همانند قبل، کل AppDomain نابود خواهد شد.

این متد دو overload دارد و در هر دو حالت، تنظیم خودکار پارامتر CancellationToken توسط ASP.NET، بیانگر آغاز زمان خاتمه‌ی کل برنامه است:

```
public static void QueueBackgroundWorkItem(Action<CancellationToken> workItem);
public static void QueueBackgroundWorkItem(Func<CancellationToken, Task> workItem);
```

در متد اول، یک متد معمولی از نوع void قابل پردازش است. در متد دوم، می‌توان متدهای async Task دار را که قرار است کارهای async را پردازش کنند، معرفی نمود.
علت استفاده از Action و Func در اینجا، امکان تعریف خلاصه و inline یک متد و ارسال پارامتری به آن از طرف برنامه است، بجای تعریف یک اینترفیس جدید، نیاز به پیاده سازی آن اینترفیس و بعد برای مثال ارسال یک مقدار از طرف برنامه به متد Stop آن (بجای تعریف یک اینترفیس تک متدی، از Action و یا Func نیز می‌توان استفاده کرد).

نمونه‌ای از نحوه‌ی فراخوانی این دو overload را در ذیل مشاهده می‌کنید:

```
HostingEnvironment.QueueBackgroundWorkItem(cancellationToken =>
{
        //todo: ...
});

HostingEnvironment.QueueBackgroundWorkItem(async cancellationToken =>
{
        //todo: ...
        await Task.Delay(20000, cancellationToken);
});
```

پشت صحنه‌ی HostingEnvironment.QueueBackgroundWorkItem

روش استاندارد ثبت و معرفی یک کار پس زمینه در ASP.NET، توسط پیاده سازی اینترفیسی به نام IRegisteredObject انجام می‌شود. سپس توسط متد HostingEnvironment.RegisterObject می‌توان این کلاس را به موتور ASP.NET معرفی کرد. در این حالت زمانیکه AppDomain قرار است خاتمه یابد، متد Stop اینترفیس IRegisteredObject کار اطلاع رسانی را انجام می‌دهد.
توسط QueueBackgroundWorkItem دقیقا از همین روش به همراه فراخوانی ThreadPool.QueueUserWorkItemجهت اجرای متد معرفی شده‌ی به آن استفاده می‌شود.
از مکانیزم IRegisteredObject در DNT Scheduler نیز استفاده شده‌است.

پیشنیازها

ابتدا نیاز است به خواص پروژه مراجعه کرده و Target framework را بر روی 5.4.2 قرار داد. اگر به روز رسانی دوم VS 2013 را نصب کرده باشید، این نگارش هم اکنون بر روی سیستم شما فعال است. اگر خیر، امکان دریافت و نصب آن، به صورت جداگانه نیز وجود دارد:

NET Framework 4.5.2.

Developer pack

محدودیت‌های QueueBackgroundWorkItem

- از آن در خارج از یک برنامه‌ی وب ASP.NET نمی‌توان استفاده کرد.
- توسط آن، خاتمه‌ی یک AppDomain تنها به مدت 30 ثانیه به تاخیر می‌افتد؛ تا فرصت داشته باشید کارهای در حال اجرا را با حداقل خسارت به پایان برسانید.
- یک work item، اطلاعاتی را از فراخوان خود دریافت نمی‌کند. به این معنا که مستقل از زمینه‌ی یک درخواست اجرا می‌شود.
- استفاده‌ی از آن الزاما به این معنا نیست که کار درخواستی شما حتما اجرا خواهد شد. زمانیکه که کار خاتمه‌ی AppDomain آغاز می‌شود، فراخوانی‌های QueueBackgroundWorkItem دیگر پردازش نخواهند شد.
- اگر برنامه به مقدار CancellationToken تنظیم شده توسط ASP.NET دقت نکند، جهت پایان یافتن کار در حال اجرا، صبر نخواهد شد.