عنوان: **اصول برنامه نویسی موازی درNET. نسخه 4 بخش اول - 1**
نویسنده: اشکان 56
تاریخ: ۱۳۹۱/۰۳/۳۱ ۱۲:۷
آدرس: www.dotnettips.info
برچسب‌ها: C#, Task Programming Language, Threading, Parallel Extensions

بدون هیچ مطلب اضافی به سراغ اولین مثال می‌رویم. قطعه کد زیر را در نظر بگیرید :

```
using System;
using System.Threading.Tasks;

namespace Listing_01 {

class Listing_01 {

static void Main(string[] args) {
  Task.Factory.StartNew(() => {
         Console.WriteLine("Hello World");
  });

  // wait for input before exiting
  Console.WriteLine("Main method complete. Press enter to finish.");
  Console.ReadLine();
 }
}
```

در کد بالا کلاس Task نقش اصلی را بازی می‌کند.این کلاس قلب کتابخانه برنامه نویسی Task یا Task Programming Library می‌باشد.

در این بخش با موارد زیر در مورد Taskها آشنا می‌شویم:

- ایجاد و به کار انداختن انواع مختلف Task ها.
- کنسل کردن Task ها.
- منتظر شدن برای پایان یک Task.
- دریافت خروجی یا نتیجه از یک Task پایان یافته.
- مدیریت خطا در طول انجام یک Task

خب بهتر است به شرح کد بالا بپردازیم:

رای استفاده از کلاس Task باید فضای نام System.Threading.Tasks را بصورت ریر مورد استفاده قرار دهیم.

```
using System.Threading.Tasks;
```

این فضای نام نقش بسیار مهمی در برنامه نویسی Taskها دارد . فضای نام بعدی معروف است : System.Threading . اگر با برنامه نویسی تریدها بروش مرسوم وکلاسیک آشنایی دارید قطعاً با این فضای نام آشنایی دارید. اگر بخواهیم با چندین Task بطور همزمان کار کنیم به این فضای نام نیاز مبرم داریم. پس :

```
using System.Threading;
```

خب رسیدیم به بخش مهم برنامه :

```
Task.Factory.StartNew(() => {
   Console.WriteLine("Hello World");
});
```

متد استاتیک Task.Factory.StartNew یک Task جدید را ایجاد و شروع می‌کند که متن Hello Word را در خروجی کنسول نمایش می‌دهد. این روش ساده‌ترین راه برای ایجاد و شروع یک Task است.

در بخش‌های بعدی چگونگی ایجاد Taskهای پیچیده‌تر را بررسی خواهیم کرد . خروجی برنامه بالا بصورت زیر خواهد بود:

```
Main method complete. Press enter to finish.

Hello World
```

روشهای مختلف ایجاد یک Task ساده :

- ایجاد کلاس Task با استفاده از یک متد دارای نام که در داخل یک کلاس Action صدا زده می‌شود. مثال :

```
Task task1 = new Task(new Action(printMessage));
```

استفاده از یک delegate ناشناس (بدون نام). مثال :

```
Task task2 = new Task(delegate {
   printMessage();
});
```

- استفاده از یک عبارت لامبدا و یک متد دارای نام . مثال :

```
Task task3 = new Task(() => printMessage());
```

- استفاده از یک عبارت لامبدا و یک متد ناشناس (بدون نام). مثال :

```
Task task4 = new Task(() => {
   printMessage();
});
```

قطعه کد زیر مثال خوبی برای چهار روشی که در بالا شرح دادیم می‌باشد:

```
using System;
using System.Threading.Tasks;

namespace Listing_02 {

class Listing_02 {

static void Main(string[] args) {

   // use an Action delegate and a named method
   Task task1 = new Task(new Action(printMessage));

   // use a anonymous delegate
   Task task2 = new Task(delegate {
   printMessage();
});

  // use a lambda expression and a named method
  Task task3 = new Task(() => printMessage());

  // use a lambda expression and an anonymous method
  Task task4 = new Task(() => {
    printMessage();
  });
  task1.Start();
  task2.Start();
  task3.Start();
  task4.Start();

  // wait for input before exiting
```

```
  Console.WriteLine("Main method complete. Press enter to finish.");
  Console.ReadLine();
 }

 static void printMessage() {
  Console.WriteLine("Hello World");
  }
 }
}
```

خروجی برنامه بالا بصورت زیر است :

```
Main method complete. Press enter to finish.

Hello World

Hello World

Hello World

Hello World
```

نکته 1 : از مند استاتیک Task.Factory.StartNew برای ایجاد Task هایی که رمان اجرای کوتاه دارند استفاده می‌شود.

نکته 2 : اگر یک Taskدر حال اجرا باشد نمی‌توان آنرا دوباره استارت نمود باید برای یک نمونه جدید از آن Task ایجاد نمود و آنرا استارت کرد.

نظرات خوانندگان

نویسنده: رحمت رضایی
تاریخ: ۱۳۹۱/۰۳/۳۱ ۱۷:۵۱

از مند استاتیک Task.Factory.StartNew برای ایجاد Task هایی که زمان اجرای طولانی هم دارند استفاده می‌شود :

```
Task.Factory.StartNew(() =>
            {
                Thread.Sleep(1000);
            }, TaskCreationOptions.LongRunning);
```

نویسنده: ali
تاریخ: ۱۳۹۱/۰۳/۳۱ ۱۸:۳۷

سلام
مرسی از آموزشتون

این روش چه برتری نسبت به شیوه کلاسیک موازی کاری داره ؟
آیا همه امکانات شیوه کلاسیک رو پوشش می‌ده ؟

بی صبرانه منتظر ادامه آموزش هستم.
پیروز باشید.

نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۱/۰۴/۰۱ ۱:۵۹

اگر منظور شما از روش‌های کلاسیک استفاده از Threadهاست باید بدانید که آن روش‌ها برای CPUهای تک هسته ای در نظر گرفته شده بودند. همانطور که می‌دانید در CPUهای تک هسته ای ، CPU تنها قادر به اجرای یک وظیفه در یک واحد زمان می‌باشد. در این CPUها برای اینکه بتوان چندین وظیفه را همراه با هم انجام داد CPU بین کارهای در حال انجام در بازه‌های زمانی مختلف سوییچ میکند و برای ما اینطور به نظر می‌آید که CPU در حال انجام چند وظیفه در یک زمان است.
اما در CPUها چند هسته ای امروزی هر هسته قادر به اجرای یک وظیفه به صورت مجزا می‌باشد و این CPUها برای انجام کارهای همزمان عملکرد بسیار بسیار بهتری نسبت به CPUهای تک هسته ای دارند.
با توجه به این موضوع برای اینکه بتوان از قابلیتهای چند هسته ای CPUهای امروزی استفاده کرد باید برنامه نویسی موازی (Parallel Programming) انجام داد و روش‌های کلاسیک مناسب این کار نمی‌باشند.

نویسنده: محمد
تاریخ: ۱۳۹۱/۰۴/۰۱ ۹:۴۸

ممنون

نویسنده: saleh
تاریخ: ۱۳۹۱/۰۴/۱۷ ۰:۱۲

شما در کد خودتون taskها را قبل از دستور چاپ متن main method ... نوشته و استارت داده بودید ولی در خروجی برعکس این موضوع اتفاق افتاده! میشه درموردش توضیح بدید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۷ ۰:۳۰

توضیحات بیشتر در اینجا