عنوان: **چگونه کدها را مستند سازی کنیم؟**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۴/۰۶/۰۵ ۱۵:۱۰
آدرس: www.dotnettips.info
گروه‌ها: Design patterns, Refactoring

یکی از مهمترین مسائل، به خصوص در کارهای تیمی یا پروژه‌های اشتراکی، قرار دادن کامنت‌ها یا اصطلاحا مستند نویسی است که بسیاری از برنامه نویسان با اینکه نظریه آن‌را به شدت قبول دارند، ولی از انجام آن سرباز می‌زنند که به دو عامل تنبلی و عدم دانش نحوه‌ی مستند نویسی بر می‌گردد. در این مقاله قصد داریم به سوالات زیر پاسخ دهیم:
چرا به کامنت گذاری یا مستند نویسی نیاز داریم؟

چگونه کامنت بنویسیم؟

انواع کامنت‌ها چیست؟

چه کامنت‌هایی اشتباه هستند؟
همانطور که بیان کردیم، کامنت گذاری یکی از مهم‌ترین کارهایی است که یک برنامه نویس انجام می‌دهد. به خصوص زمانیکه به صورت تیمی کار می‌کنید، این امر مهم‌تر از قبل خود را نشان می‌دهد. بسیاری از برنامه نویسان که بیشتر دلیل آن تنبلی است، از این کار سرباز می‌زنند و ممکن است آن را اتلاف وقت بدانند. ولی با کامنت گذاری فهم و درک کد، در آینده بالاتر می‌رود. در مقاله‌ی تخصصی « *هنر کامنت نویسی* » نوشته‌ی « *برنهارد اسپویدا* » بهانه‌های جالبی از برنامه نویسان را برای سرباز زدن از اینکار، ذکر شده است؛ به عنوان نمونه:
من کدم را به خوبی متوجه می‌شوم.
کد خوب، خودش گویای همه چیز هست.
وقتی برای کامنت نویسی وجود ندارد. باید چسبید به کد.

سوالات زیر نیز دلیل این را که چرا باید کامنت گذاری کرد، مشخص می‌کنند:
شاید امروز معنای یک کد را متوجه شوید، ولی آیا در آینده، مثلا یک سال بعد هم چنین خواهد بود؟

آیا می‌توانید هر سیستمی را که طراحی می‌کنید، به خاطر بسپارید که فعالیت‌هایش را به چه نحوی انجام می‌دهد؟

اگر در یک کار تیمی باشید، شاید شما متوجه کدتان می‌شوید، ولی آیا تضمینی وجود دارد که دیگران هم متوجه روش شما شوند؟

آیا کدی که شما بر روی آن فکر کرده‌اید و در ذهن خود روش انجام آن را ترسیم کرده‌اید، می‌تواند برای برنامه نویسی که کد شما را می‌بیند هم رخ دهد؟

اگر شما به صورت تیمی کاری را انجام دهید و یکی از برنامه نویس‌های شما از تیم جدا شود، چگونه می‌توانید کار او را دنبال کنید؟
اگر برای برنامه نویسی اتفاق یا حادثه‌ای پیش بیاید که دسترسی شما به او ممکن نباشد چه؟

کدی که مستند نشود، یا خوب مستند نشود، در اولین مرحله وقت شما یا هر فردی را که روی این کد کار می‌کند، برای مدتی طولانی می‌گیرد. در مرحله‌ی بعدی احتمالا مجبور هستید، کد را خط به خط دنبال کرده تا تاثیر آن را بر ورودی‌ها و خروجی‌ها ببینید. تمام این‌ها باعث هدر رفتن وقت شما شده و ممکن است این اتفاق برای هر تکه کدی رخ بدهد.

سطوح کامنت نویسی بر سه نوع هستند:

Documentary Comments: این مستند سازی در سطح یک سند مثل فایل یا به خصوص یک پروژه رخ می‌دهد که شامل اطلاعات و تاریخچه‌ی آن سند است که این اطلاعات به شرح زیر هستند:

File Name	نام سند
File Number/Version Number	شماره نسخه آن سند
Creation Date	تاریخ ایجاد آن
Last Modification Date	تاریخ آخرین تغییر سند
Author's Name	سازنده‌ی سند
Copyright Notice	اطلاعاتی در مورد کپی رایت سند
Purpose Of Program	هدف کاری برنامه. یک خلاصه از آن چه برنامه انجام می‌دهد.
Change History	لیستی از تغییرات مهمی که در جریان ایجاد آن رخ داده است.
Dependencies	وابستگی‌های سند. بیشتر در سطح پروژه معنا پیدا می‌کند؛ مانند نمونه‌ی آن برای سایت جاری که به صورت عمومی منتشر شده است.
Special Hardware Requirements	سخت افزار مورد نیاز برای اجرای برنامه. حتی قسمتی می‌تواند شامل نیازمندی‌های نرم افزاری هم باشد.

نمونه ای از این مستند سازی برای برنامه ای که به زبان پاسکال نوشته شده است:

```
PCMBOAT5.PAS************************************************************
**
File: PCMBOAT5.PAS
Author: B. Spuida
Date: 1.5.1999
Revision:
1.1
PCM-DAS08 and -16S/12 are supported.
Sorting routine inserted.
Set-files are read in and card as well as
amplification factor are parsed.

1.1.1
Standard deviation is calculated.

1.1.2
Median is output. Modal value is output.

1.1.4
Sign in Set-file is evaluated.
Individual values are no longer output.
(For tests with raw data use PCMRAW.EXE)

To do:
outliers routine to be revised.
Statistics routines need reworking.
Existing Datafile is backed up.

Purpose:
Used for measurement of profiles using the
Water-SP-probes using the amplifier andthe PCM-DAS08-card, values are acquired
with n = 3000. Measurements are taken in 1
second intervals. The values are sorted using
Quicksort and are stacked "raw" as well as after
dismissing the highest and lowest 200 values as
'outliers'.


Requirements:
The Card must have an A/D-converter.
Amplifier and probes must be connected.
```

```
Analog Signal must be present.
CB.CFG must be in the directory specified by the
env-Variable "CBDIREC" or in present directory.
```

در بالا، خصوصیت کپی رایت حذف شده است. دلیل این امر این است که این برنامه برای استفاده در سطح داخلی یک شرکت استفاده می‌شود.

Functional Comments: کامنت نویسی در سطح کاربردی به این معنی نیست که شما اتفاقاتی را که در یک متد یا کلاس یا هر بخشی روی می‌دهد، خط به خط توضیح دهید؛ بلکه چرخه‌ی کاری آن شی را هم توضیح بدهید کفایت می‌کند. این مورد می‌تواند شامل این موارد باشد:
توضیحی در مورد باگ‌های این قسمت
یادداشت گذاری برای دیگر افراد تیم
احتمالاتی که برای بهبود ویژگی‌ها و کارایی کد وجود دارد.

Explanatory Comment: کامنت گذاری توصیفی در سطح کدنویسی رخ می‌دهد و شامل توضیح در مورد کارکرد یک شیء و توضیح کدهای شیء مربوطه می‌گردد. برای قرار دادن کامنت الزامی نیست که کدها را خط به خط توضیح دهید یا اینکه خطوط ساده را هم تشریح کنید؛ بلکه کامنت شما همینقدر که بتواند نحوه‌ی کارکرد هر چند خط کد مرتبط به هم را هم توضیح دهد، کافی است. این توضیح‌ها بیشتر شامل موارد زیر می‌شوند:
کدهای آغازین
کدهای خروجی
توضیح کوتاه از آنچه که این شیء ، متد یا ... انجام می‌دهد.
حلقه‌های طولانی یا پیچیده
کدهای منطقی عجیب و پیچیده
Regular Expression

کدهای آغازین شروع خوبی برای تمرین خواهند بود. به عنوان نمونه اینکه توضیحی در مورد ورودی و خروجی یک متد بدهید که آرگومان‌های ورودی چه چیزهایی هستند و چه کاربری دارند و در آغاز برنامه، برنامه چگونه آماده سازی و اجرا می‌شود. مقادیر پیش فرض چه چیزهایی هستند و پروژه چگونه تنظیم و مقداردهی می‌شود.

کدهای خروجی هم به همین منوال است. خروجی‌های نرمال و غیرنرمال آن چیست؟ کدهای خطایی که ممکن است برگرداند و ... که باید به درستی توضیح داده شوند.

توضیح اشیاء و متدها و ... شامل مواردی چون: هدف از ایجاد آن، آرگومان هایی که به آن پاس می‌شوند و خروجی که می‌دهد و اینکه قالب یا فرمت آن‌ها چگونه است و چه محدودیت‌هایی در مقادیر قابل انتظار وجود دارند. ذکر محدودیت‌ها، مورد بسیاری مهمی است و دلیل بسیاری از باگ‌ها، عدم توجه یا اطلاع نداشتن از وجود این محدودیت هاست. مثلا محدوده‌ی خاصی برای آرگومان‌های ورودی وجود دارد؟ چه اتفاقی می‌افتد اگر به یک بافر 128 کاراکتری، 1024 کاراکتر را ارسال کنیم؟

کدهای منطقی عجیب ، یکی از حیاتی‌ترین بخش‌های کامنت گذاری برای نگه داری یک برنامه در آینده است. به عنوان نمونه استفاده از عبارات با قاعده، اغلب اوقات باعث سردرگمی کاربران شده است. پس توضیح دادن در مورد این نوع کدها، توصیه زیادی می‌شود. اگر عبارات با قاعده شما طولانی هستند، سعی کنید از هم جدایشان کنید یا خطوط آن را بشکنید و هر خط آن را توضیح دهید.

سیستم کامنت گذاری

هر زبانی از یک سیستم خاص برای کامنت گذاری استفاده می‌کند. به عنوان مثال پرل از سیستم POD (Plain Old Documentation) استفاده می‌کند یا برای Java سیستم JavaDoc یا برای PHP از سیستم PHPDoc (+) که پیاده سازی از JavaDoc می‌باشد استفاده می‌کنند. این سیستم برای سی شارپ استفاده از قالب XML است. کد زیر نمونه‌ای از استفاده از این سیستم

است:

```
// XMLsample.cs
// compile with: /doc:XMLsample.xml
using System;
/// <summary>
/// Class level summary documentation goes here.</summary>
/// <remarks>
/// Longer comments can be associated with a type or member
/// through the remarks tag</remarks>
public class SomeClass
{
    /// <summary>
    /// Store for the name property</summary>
    private string myName = null;

    /// <summary>
    /// The class constructor. </summary>
    public SomeClass()
    {
        // TODO: Add Constructor Logic here
    }

    /// <summary>
    /// Name property </summary>
    /// <value>
    /// A value tag is used to describe the property value</value>
    public string Name
    {
        get
        {
            if (myName == null)
            {
                throw new Exception("Name is null");
            }
            return myName;
        }
    }

    /// <summary>
    /// Description for SomeMethod.</summary>
    /// <param name="s"> Parameter description for s goes here</param>
    /// <seealso cref="String">
    /// You can use the cref attribute on any tag to reference a type or member
    /// and the compiler will check that the reference exists. </seealso>
    public void SomeMethod(string s)
    {
    }
}
```

دستورات سیستم کامنت گذاری سی شارپ

در سایت جاری، دو مقاله زیر اطلاعاتی در رابطه با نحوه‌ی کامنت گذاری ارئه داده‌اند.
- در مقاله « زیباتر کد بنویسیم » چند مورد آن به این موضوع اختصاص دارد.
- مقاله « وادار کردن خود به کامنت نوشتن » گزینه‌ی کامنت گذاری اجباری در ویژوال استودیو را معرفی می‌کند.