عنوان: **قبل از رفع باگ، برای آن تست بنویسید**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۲۰ ۲۱:۵۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: Unit testing

از دقت کردن در نحوه اداره پروژه‌های خوب و بزرگ در سطح دنیا، می‌توان به نکات آموزنده‌ای رسید. برای مثال NHibernate را درنظر بگیرید. این پروژه شاید روز اول کپی مطابق اصل نمونه جاوای آن بوده، اما الان از خیلی از جهات یک سر و گردن از آن بالاتر است. پشتیبانی LINQ را اضافه کرده، خودش Syntax جدیدی را به نام QueryOver ارائه داده و همچنین معادلی را جهت حذف فایل‌های XML به کمک امکانات جدید زبان‌های دات نتی مانند lambda expressions ارائه کرده. خلاصه این تیم، فقط یک کپی کار نیست. پایه رو از یک جایی گرفته اما سبب تحول در آن شده. از اهداف پروژه‌های سورس باز هم همین است: برای هر کاری چرخ را از صفر ابداع نکنید.

اگر به نحوه اداره کلی پروژه NHibernate دقت کنید یک مورد مشهود است:
تمام گزارش‌های باگ بدون Unit test ندید گرفته می‌شوند.
از کلیه بهبودهای ارائه شده (وصله‌ها یا patch ها) بدون Unit test صرفنظر می‌شود.
از کلیه موارد جدید ارائه شده بدون Unit test هم صرفنظر خواهد شد.

بنابراین اگر در issue tracker این تیم رفتید و گفتید: «سلام، اینجا این مشکل هست»، خیالتان راحت باشد که ندید گرفته خواهید شد.

سؤال : چرا این‌ها اینطور رفتار می‌کنند؟!
- وجود Unit test دقیقا مشخص می‌کند که چه قسمت یا قسمت‌هایی به گزارش باگ شما مرتبط هستند. نیازی نیست حتما بتوانید یک خطا را با جملات ساده شرح دهید. این مساله خصوصا در پروژه‌های بین المللی حائز اهمیت است. ضعف زبان انگلیسی همه جا هست. همینقدر که توانسته‌اید برای آن یک Unit test بنویسید که مثلا در این عملیات با این ورودی، نتیجه قرار بوده بشود 10 و مثلا شده 5 یا حتی این Exception صادر شده که باید کنترل شود، یعنی مشکل را کاملا مشخص کرده‌اید.
- وجود Unit tests ، انجام Code review و همچنین Refactoring را تسهیل می‌بخشند. در هر دو حالت یاد شده، هدف تغییر کارکرد سیستم نیست؛ هدف بهبود کیفیت کدهای موجود است. بنابراین دست به یک سری تغییرات زده خواهد شد. اما سؤال اینجا است که از کجا باید مطمئن شد که این تغییرات، سیستم را به هم نریخته‌اند. پروژه‌ی جاری چند سال است که در حال توسعه است. قسمت‌های زیادی به آن اضافه شده. با نبود Unit tests ممکن است بعضی از قسمت‌ها زاید یا احمقانه به نظر برسند.
- بهترین مستندات کدهای تهیه شده، Unit tests آن هستند. برای مثال علاقمند هستید که NHibernate را یاد بگیرید؟ هرچه می‌گردید مثال‌های کمی را در اینترنت در این زمینه پیدا می‌کنید؟ وقت خودتان را تلف نکنید! این پروژه بالای 2000 آزمون واحد دارد. هر کدام از این آزمون‌ها نحوه‌ی بکارگیری قسمت‌های مختلف را به نحوی کاربردی بیان می‌کنند.
- وجود Unit tests از پیدایش مجدد باگ‌ها جلوگیری می‌کنند. اگر آزمون واحدی وجود نداشته باشد، امروز کدی اضافه می‌شود. فردا همین کد توسط عده‌ای دیگر زاید تشخیص داده شده و حذف می‌شود! بنابراین احتمال بروز مجدد این خطا در آینده وجود خواهد داشت. با وجود Unit tests، فلسفه وجودی هر قسمتی از کدهای موجود پروژه دقیقا مشخص می‌شود و در صورت حذف آن، با اجرای آزمون‌های خودکار سریعا می‌توان به کمبودهای حاصل پی‌برد.

نظرات خوانندگان

نویسنده: بهزاد
تاریخ: ۱۳۹۰/۱۰/۲۰ ۲۳:۲۷:۰۱

وحید عزیز، من وبلاگ ت رو دنبال می کنم و کتاب فوق العاده ات درباره Exchange Server را خوانده ام
مدتهاست که به دنبال یک جزوه یا کتاب مختصر برای یاد گرفتن تست هستم، کلیات رو می دونم، ولی به نظرم یک پروژه نمونه که تا حد زیادی واقعی باشه خیلی بیشتر از دهها کتاب قطور می تونه کمک کنه
می تونی در این زمینه منبعی معرفی کنی؟ پلتفرمی که من کار می کنم ASP.NET WebForm چهار لایه است که از NH هم به عنوان لایه دیتابیس استفاده می کنم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۲۱ ۰۰:۳۸:۴۲

- برای NH اگر از الگوی Repository استفاده می‌کنید می‌تونید از SQLite به عنوان ابزار نوشتن آزمون‌های واحد استفاده کنید. SQLite یک مزیت جالبی که دارد این است که امکان تشکیل دیتابیس در حافظه را دارد. این یعنی همان پیشنیاز اصلی نوشتن آزمون‌های واحد: سرعت بالای انجام کار، خارج نشدن از مرزهای سیستم. ضمن اینکه این بانک اطلاعاتی تشکیل شده، یک بانک اطلاعاتی واقعی است اما پس از پایان کار به صورت خودکار نابود می‌شود که برای آزمون‌های واحد بسیار مفید است.
برای ORM های دیگر چون پشتیبانی از سایر بانک‌های اطلاعاتی آن‌ها ضعیف است، روش‌های mocking و غیره مطرح می‌شود (که اینبار دیگر با یک دیتابیس واقعی کار نمی‌شود و سطح کار کمی پایین‌تر است) اما با NH راحت می‌شود از SQLite تشکیل شده در حافظه استفاده کرد. فقط باید تنظیمات اتصال ابتدای برنامه را عوض کرد.
- خوب؛ تا اینجا واژه کلیدی مورد نیاز برای جستجو مشخص شد، مابقی را در اینجا (^) جستجو کنید.

نویسنده: Shima2012
تاریخ: ۱۳۹۰/۱۰/۲۲ ۲۰:۴۸:۳۳

با سلام.
جناب نصیری در حال حاضر برای Unit testing استفاده از MBUnit 3 رو پیشنهاد میدید یا NUnit

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۰/۲۲ ۲۱:۳۲:۰۳

فریم ورک زیاد هست. حتی خود مایکروسافت هم مثلا MSTest رو داره که با VS.NET یکپارچه است. نکته مهم این ابزارها نیستند. مهم نوشتن تست است. مهم این نیست که از SVN استفاده کنید یا از GIT . مهم این است که از یک سورس کنترل استفاده شود.