Vorwort

»Die Realisierung einer Anwendung dauert genauso lange, wie sich Zeit dafür nehmen«.

Rolf Wenger

Es begann an einem regnerischen Sonntag vor dreieinhalb Jahren. Die ersten Seiten von Band I des Handbuchs der .NET 4.0-Programmierung waren geschrieben. Gute Ansätze im Text, kleine und funktionierende Beispiele auf dem Rechner, ein zufriedenes und zugleich beglückendes Gefühl beschlich mich, als ich den Rechner ausschaltete und dem weinenden Wetter ins Gesicht sage: »Mir doch Wurst, wenn du da draußen mal wieder regnest«.

Jetzt, 2.500 Seiten und 100 MB Code später, schreibe ich das, was Sie jetzt gerade lesen. Das Glücksgefühl ist um mindestens die gleichen Faktoren grösser, wie der Zuwachs an Seiten und Megabyte an Code. Und trotzdem, da ist etwas, was mich aufwühlt. Was ist es? Wo ist es? Der Gedanke sagt mir: »Du hättest es noch besser machen können, die Beispiele noch mehr aus der Praxis nehmen können und die weggelegten Themen aufgreifen und Bestehendes noch mehr vertiefen können«!

Ja, die innere Stimme hat Recht. Zu oft habe ich schon erlebt, dass beim Gebrauch meiner eigenen Bücher vieles besser sein könnte. Doch was heißt besser machen, und wie geht das? Eigentlich müsste ich doch das »Besser machen« in Perfektion beherrschen – ja, ich müsste der beste Bessermacher auf der Welt sein, denn ich beschäftige mich schließlich den ganzen Tag mit nichts anderem als dem »Besser machen«. Ich unterrichte, erkläre, bearbeite, reviewe, codiere, argumentiere, schreibe, diskutiere und reflektiere immerzu. Es ist stets dasselbe Thema: Entwicklung von Anwendungen mit .NET. Natürlich finden diese Tätigkeiten in unterschiedlichen Formen statt. Mal durch eine Beratung beim Kunden A, mal durch eine Programmierung für ein Projekt beim Kunden B, mal eine Analyse beim Kunden C, dann entspannen am eigenen Projekt und natürlich genauso wichtig, an der Fachhochschule, zusammen mit den Studenten, die von mir lernen wollen, wie man es besser macht.

Da ist sie wieder, die innere Stimme, die sagt: »Du musst es besser machen«. Ja, ich stelle fest, dass ich richtig davon besessen bin, es besser machen zu wollen und … stopp! So geht das nicht. 2.500 Seiten und 100 MB Code. Pro Seite Text eine Stunde Arbeit, drei Jahre sind 3 × 365 Tage sind ca. 1.100 Tage, also 2.500 geteilt durch 1.100 sind – ohh! – das sind ja 2,27 Stunden Arbeit pro Tag ohne den Code gerechnet, und das eben 3 Jahre lang. Wäre ja ein Klax, wenn ich nicht noch einen Nebenjob hätte, die normale Arbeit! Und genau die ist schuld, dass ich es nicht besser machen kann, denn ich finde die Zeit dazu nicht.

Ahaaa! Jetzt habe ich die Erkenntnis: Das »Besser machen« steht in Relation zur verfügbaren Zeit. Je mehr Zeit ich habe, desto besser kann ich meine Arbeit machen. Kann man etwas auch zu gut machen? Wenn diese Frage mit Ja beantwortet werden kann, ist dann der Schluss richtig, dass man für etwas auch zu viel Zeit haben kann? Ich denke schon, denn wenn ich in die Praxis schaue, gibt es irgendwann bei jeder Arbeit einen Punkt, an dem man sich in der Lösungsfindung nicht mehr vorwärts, sondern seitwärts bewegt. Seitwärts heißt für mich, dass ich eine andere Lösung entwickle, die aber nicht besser ist als die vorangehende. Das wäre dann quasi der Punkt, an dem man eben keine Zeit investieren sollte, sondern zu neuen Ufern aufbrechen kann.

Die Erfahrung zeigt aber, dass diese Situationen recht selten sind. Ich erlebe mehr die Situation, dass man noch lange wüsste, wie man eine Sache verbessern kann, aber schon lange keine Zeit mehr hat. Haben Sie übrigens auch schon beobachtet, dass in den Projekten der Begriff Zeit das Synonym des Begriffs Geld ist? Ja, Zeit ist Geld. Da bin ich wohl nicht der erste, der diese Erkenntnis hat.

Irgendwie verwirrend das Ganze, finden Sie nicht? Ich kann den Gedanken drehen und wenden wie ich will, aber ich komme immer wieder auf dasselbe Resultat. Eine Lösung dauert immer genau so lange, wie ich mir Zeit dafür nehme. Sicher, die Lösungen, die mit einem Aufwand von wenigen Tagen erzielt werden, sind nicht vergleichbar mit den Lösungen, die Monate verschlungen haben. Aber wenn es in wenigen Tagen eine Lösung gibt, wieso soll ich dann Monate einsetzen?

Zurück zu meinem Buch. Wie kann ich das besser machen? Ich denke nicht, dass die Buchreihe besser würde, wenn ich mehr Zeit investieren würde. Sicher, der eine oder andere Fehler wäre nicht vorhanden, aber ich weiß nicht, ob das die Essenz wirklich steigern würde. Ich glaube, ich lasse jetzt das Glücksgefühl ob des Erreichten, das Gefühl der Rastlosigkeit, überschwappen – zumindest für einen Moment.

In die Erstellung dieses Werks, und damit meine ich wieder alle drei Bände, habe ich viel Zeit investiert. Ich habe auch Zeit von anderen Personen verbraucht. Sei es durch Diskussionen oder durch Erklärungen meinerseits, die sich dann später vielleicht sogar als falsch herausgestellt haben. Ich bedanke mich bei allen Teilnehmern, die im Verlauf eines Kurses zum Teil bewusst und zum Teil unbewusst dazu beigetragen haben, dass diese Buchreihe so entstanden ist, wie sie sich eben jetzt präsentiert. Ein großer Dank gebührt auch den Mitarbeitern, Kollegen, Vorgesetzten und allen Beteiligten, die meine Ideen helfen mitzutragen und sich anstrengen, gemeinsam mit mir die bessere Lösung zu finden.

Ein besonderer Dank gilt allerdings Herrn Florian Helmchen, der mich von Seiten des Verlags betreut hat, und mit Geduld alle Verzögerungen hingenommen hat. Ebenfalls danke ich Herrn Uwe Thiemann, der als Fachlektor dabei half, meine sich manchmal überstürzenden Gedanken zu ordnen und mein Schweizerdeutsch in ein verständlicheres Hochdeutsch zu übersetzen. Frau Silja Brands hat mit viel Sachverstand, Voraussicht, persönlichem Einsatz und großer Geduld ermöglicht, dass die zeitlichen Limits für die Fertigstellung nicht gar zu arg ins Hintertreffen geraten sind. Herzlichen Dank auch dafür.

Der größte Dank allerdings gehört nach wie vor meiner Familie, allen voran meiner Frau Irene. Sie hat mit einer unendlichen Geduld und Großzügigkeit alle Abwesenheiten und die langen einsamen Abende ertragen, an denen ich gelernt habe, wie man es besser macht.

Rolf Wenger im August 2012

Einleitung

Das Handbuch der .NET 4.0-Programmierung gliedert sich in drei Bände.

- Band 1 beschäftigt sich mit den grundlegenden Techniken von .NET. Die in diesem Band angesprochenen Elemente werden wohl die meisten auf die eine oder andere Weise in einem konkreten Projekt anwenden. Ein Hauptbestandteil des Handbuchs bildet die Programmiersprache C# und LINQ (Language Integrated Query). Die beiden Elemente bilden den Leim zwischen den einzelnen Bausteinen der BCL (Base Class Library), die in diesem Band vorgestellt wird.
- Band 2 beschäftigt sich mit den Techniken und der Gestaltung von Benutzerschnittstellen unserer Anwendungen. Dabei werden die Techniken Windows Forms, Windows Presentation Foundation, ASP.NET und Silverlight erklärt und gegenübergestellt. Viele Details in den Erklärungen machen auch diesen Band zum Nachschlagewerk.
- Band 3 widmet sich der Herstellung von verteilten Anwendungen. Entsprechend befindet sich die Windows Communication Foundation (WCF) im Zentrum des Bands. Ergänzt wird dieser Band mit Überlegungen zur Architektur verteilter Anwendungen und allgemein zum Testen von Anwendungen mit Visual Studio.

Zielpublikum dieses Buchs

Als Generalist werden Sie über .NET und seine Anwendung wissen wollen, wo und mit welchem Aufwand diese neuen Technologien einsetzbar sind. Dieser Band gibt Ihnen einen breiten Einblick in die aktuellen grundlegenden Techniken von .NET.

Als Architekt von Lösungen wird Sie interessieren, welche Techniken für welche Lösungsansätze .NET zu bieten hat. Fragen zu Persistenz, Aufbau von Clientsoftware und technischen Implementierungen von Software beantwortet dieses Buch genauso, wie die Fragen um die Wahl der richtigen Servertechnik.

Als Programmierer sind Sie interessiert zu erfahren, wie Sie eine konkrete Technik anwenden oder umsetzen. Sie finden im vorliegenden Werk zahlreiche reich kommentierte Codebeispiele, Ideen für Lösungsansätze der Programmierung und ein sehr kompaktes Nachschlagewerk zur Programmierung von .NET.

Der Nutzen dieser Buchreihe

Der Nutzen der Buchreihe liegt in der Kompaktheit und zugleich der Tiefe des Stoffs. Wieso finde ich drei Bände und über 2500 Seiten kompakt? Diese Buchreihe deckt die .NET-Technologie in einer nur selten anzutreffenden Breite ab, und dies in nur drei Bänden. Die Beschreibungen beschränken sich nicht nur auf die theoretischen Betrachtungen, sondern liefern viele Praxishinweise und konkrete Umsetzungen mit Beispielcode für die Programmierung bis hin zur Systemkonfiguration.

Wenn Sie das Handbuch an Ihrem Arbeitsplatz zur Hand haben, werden Sie es bald als Nachschlagewerk in der Praxis benutzen, denn das Handbuch ist themenorientiert aufgebaut und definiert dazu die tagtäglichen und viele auch nicht alltägliche Handgriffe, Tipps und Tricks. Prüfen Sie sich selbst:

- Welches ist die ideale Klasse für die Verarbeitung einer Auflistung von Objekten?
- Wie setze ich eine Vererbung in eine relationale Datenbank um?
- Wie konfiguriere ich einen Webserver und die Anwendung für die Nutzung der formularbasierten Authentifizierung?
- Welches ist die schnellste Client/Server-Technik?
- Mit welcher Genauigkeit arbeitet die Klasse `Timer`?
- Wie kombiniere ich unterschiedliche `HierarchicalData`-Templates von unterschiedlichen Typen im gleichen `TreeView`-Steuerelement?
- Wie erstelle ich eine serverseitige asynchrone Verarbeitung?
- Wie programmiere ich eine Ablaufverfolgung?
- Kann eine generische Klasse von einer anderen generischen Klasse erben?
- Wie erstelle ich einen Webdienst?

Diese Fragen und noch viele mehr beantwortet das Handbuch der .NET 4.0/4.5-Programmierung in drei Bänden. Dies geschieht mithilfe von Beschreibungen, über eintausend Abbildungen und zahlreichen Tabellen und Codebeispielen.

Mit der neu verfügbaren Version als PDF nutzen Sie das Handbuch am Arbeitsplatz noch effizienter, weil Sie nach Textstellen suchen und Code kopieren können, ohne dass Sie ihn im Beispielcode suchen müssen.

Voraussetzungen für den dritten Band

Dieser Band setzt umfassende Kenntnisse der .NET-Bibliothek (BCL) und der Programmierung von Benutzeroberflächen voraus. Die Basis des Wissens bilden die ersten beiden Bände dieser Buchreihe, wie sie zu Beginn der Einleitung erwähnt wurden.

Erklärungen zu den Benutzeroberflächen oder der BCL erfolgen in diesem Buch nur dann, wenn sie in den anderen Bänden nicht vorhanden sind. Andernfalls sind Querverweise vorhanden.

Der Benutzer, die Benutzer, die Benutzerin

In der ganzen Buchreihe verstehe ich den Begriff »Benutzer« im Sinne von »Benutzer und Benutzerinnen«. Ich habe mich für die Kurzform der beiden Nennungen entschieden, weil die Texte so einfacher und flüssiger zu lesen sind.

Die Codebeispiele

Das Handbuch der .NET 4.0/4.5-Programmierung umfasst viele Codebeispiele. Die Beispiele sind nach Kapitel und Themengebiet geordnet. Die Organisation der Beispiele ist der Kapitelstruktur nachempfunden, und somit sicher für alle einfach verständlich. Um Ihnen die Referenzierung zwischen dem Buch und dem Beispielcode zusätzlich zu erleichtern, ist beim abgedruckten Code in der Listingunterschrift das Beispiel direkt als Textreferenz definiert, oder in den Abschnitten sind die Beispiel direkt referenziert.

Dazu ein Beispiel:

```
// Proxy aus dem Verzeichnis holen
IwlbDemoService objProxy = ...;

// Aufruf für den geteilten Zugriff
using (new OperationContextScope((IContextChannel)objProxy)) {
    MessageHeader header = MessageHeader.CreateHeader(
        GwlbContractGlobals.SharedHeaderName,
        GwlbContractGlobals.SharedHeaderNamespace,
        strInstanceId);
    OperationContext.Current.OutgoingMessageHeaders.Add(header);
    CwlbInfoEventArgs objPing = objProxy.Ping(Environment.MachineName);
}
```

Listing 6.9 Band_3\Kapitel_06\LearningWcf\WeroSoft.Samples.LearningWcfClient\MainWindow.xaml.cs

Die Unterschrift im Beispiellisting 6.9 definiert, dass Sie den Originalcode in der Datei *MainWindow.xaml.cs* im Verzeichnis *Band_3\Kapitel_06\LearningWcf\WeroSoft.Samples.LearningWcfClient* finden.

Aus Platzgründen haben wir viele Codebeispiele gekürzt abgedruckt. Dazu gehören:

- Direkte Weglassungen von Code. Diese sind jeweils an der stellvertretenden Codezeile (`Wiedergabe gekürzt`) oder an der Zeichenfolge ... erkennbar.
- Löschung von leeren Ausgabezeilen. Diese Kürzungen sind im abgedruckten Code nicht ersichtlich.
- Geänderte Anordnung von Codeblöcken. Codeblöcke werden in den abgedruckten Beispiel meistens in der einleitenden Zeile geöffnet, und nicht wie im Editor üblich, in einer separaten Zeile (Zeichen {).
- Die XML-Kommentierung von Code ist nicht abgedruckt.

Der Code liegt dem Buch nicht als CD bei. Sie können den Code allerdings bei Microsoft auf der offiziellen Webseite zum Buch von *http://www.microsoft-press.de/support/9783866454408*, von *http://msp.oreilly.de/support/2005/746* oder von meiner eigenen Webseite (*http://www.weroSoft.net*) herunterladen. Auf meiner Webseite finden Sie neben dem Beispielcode wo nötig auch Dokumentationen und weiterführende Angaben.

Ich mache Sie darauf aufmerksam, dass einige Beispiele und Messresultate oder Ausgaben, die im Buch abgedruckt sind, bei der Ausführung auf Ihrem Rechner anders aussehen können. Das wird immer dann der Fall sein, wenn Angaben von der Rechnerphysik oder -logik abhängen.

WICHTIG Die Codebeispiele enthalten meistens keine spezielle Ausnahmebehandlung oder Variantenprüfung. Ferner existieren viele Coderedundanzen zwischen den einzelnen Beispielen. Es ist mir bewusst, dass der Code somit nicht immer praxisorientiert aufgebaut ist. Das ist aber in vollem Bewusstsein geschehen, da der Fokus eines Beispiels sich möglichst klar herausschälen soll. Zu viele Plausibilisierungen und sonstige in der Praxis notwendigen Codierungen würden zu stark von der Darstellung des effektiven Beispiels ablenken.

Ein Wort zu den Bezeichnern

Auch wenn Microsoft die Verwendung von Präfixen nicht mehr empfiehlt, beharre ich persönlich insbesondere bei Code zu Ausbildungszwecken darauf. Gerade der oft herausgestrichene Vorteil der Codetransparenz (alles sieht gleich aus) ist aus meiner Erfahrung vor allem ein Nachteil, denn ich erkenne nicht, was von mir und was von anderen Herstellern in den Code eingebracht wurde. Nun, je nach persönlicher Neigung

sind Sie froh oder stören sich daran, dass ich an den Typpräfixen festhalte. Auf alle Fälle sollte es Ihnen leicht fallen, meinen Code von .NET-Bibliothekscode zu unterscheiden, und gerade das ist in einem Nachschlagewerk meiner Meinung nach wertvoll.

Feedback

Ich will mit diesem Buch ein wertvolles Element für die Ausbildung und die tägliche Arbeit schaffen. Das ist die gute Absicht. Die Realität wird sein, dass ich es fertig bringe, trotz mehrfachen Lesens von mehreren Personen Fehler im Ganzen zu hinterlassen. Sollten Sie einen solchen entdecken und haben Sie gerade fünf Minuten Zeit, schauen Sie auf meiner Korrigenda zum Buch auf der Webseite nach, ob bereits eine Korrektur angemeldet ist. Wenn nicht, können Sie direkt von dort aus den Fehler an mich melden.

URL Korrigenda: *http://www.korrigenda.literatur.weroSoft.net*

URL Feedback: *http://www.feedback.literatur.weroSoft.net*

URL Buchseite: *http://www.literatur.weroSoft.ch*

URL Webseite: *http://www.weroSoft.net*