Code Dokumentation

Thomas Reinwart

1. Code Dokumentation generieren

Zum Thema Codedokumentation gibt es immer wieder Diskussionen zwischen Entwicklern, Entwicklungs- und Projektleitern. Dabei ist dies ein unternehmenswichtiger Bereich.

Einerseits sollen die Projekte möglichst schnell und kostengünstig umgesetzt werden, andererseits wird aus Quick und Dirty Prototypen nach der Kundenaquisition eine Applikation. Diese wird dann über mehrere Jahre weiterentwickelt, die Entwickler wechseln das Projekt oder das Unternehmen.

Am Ende gibt es eine Applikation, von der die Projektleiter oder das Marketing glauben die Funktionen noch zu kennen, technisch im Code sieht es aber anders aus als am Marketing Prospekt bzw. im ursprünglichen Konzept. Die Änderungen werden oft in den Konzepten nicht nachgetragen, somit läuft beides immer weiter auseinander.

An der Applikation ist weiterhin Support für den Kunden zu leisten ist. Dann stellt sich die Frage, wer kennt sich da aus, ist das Problem *„work as designed"* (damals so von jemanden mündlich am Telefon definiert) oder ist es doch ein Fehler der Umsetzung selber. Ist es möglicherweise ein Bedienungsfehler des Kunden, kann man den Aufwand nun verrechnen? Der derzeit involvierte Entwickler steht jedenfalls vor einem undokumentierten oder unzureichend dokumentierten Sourcecode, eventuell nicht mal der eigene, der Aufwand sich hier einzulesen ist hoch, die entstehenden Kosten ebenfalls.

Code-Doku sollte also bereits während der Entwicklung gemacht werden, einheitlich in einer Sprache und im Stil der .net Code Doku in XML. Der Aufwand, seinen aktuellen Gedankengang in 2 Sätzen im Code zu dokumentieren, ist der geringste Aufwand, im Vergleich dazu was in Summe für ein Aufwand entstehen kann. Was, wie tief dokumentiert wird, sollte pro Projekt bzw. als Unternehmensrichtlinie schriftlich festgehalten werden, damit sich alle beteiligten Entwickler auch daran halten können.

Eine Sourcecode-Dokumentation ist nun die Voraussetzung, um daraus eine automatisierte Erstellung einer Library bereitzustellen. Eine Automatisierung erhält man durch einen Build-Prozess, etwa durch **nant** (Open Source) oder mittels **Microsoft Team Foundation Server**.

Die Code-Dokumentation dient bereits während der Entwicklung als aktuelles Nachschlagewerk, nun erübrigt sich auch so mache F&A im Team, wo nun welche Funktion zu finden ist, was sich geändert hat usw. Diese Doku kann auch für externe Mitarbeiter oder auch als Produkt-API zur Verfügung gestellt werden.

Weiters werden nicht dokumentierte Stellen im Code bei der Codedocu Generierung durch Dritte schnell sichtbar, also jene, die keinen Einblick in den Sourcecode haben. Alle Stellen werden nämlich mit roten Text Missing etc. dargestellt. Welche Bereiche nun als Codedoku im Unternehmen gelten, ob nur die Public Methode, oder alle, ist bei der Konfiguration in Sandcastle (im `shfb`-File) zu definieren. Somit ist die Kontrolle gegeben, das hier die *Coding*-Richtlinie auch eingehalten wird.

Beispiel für eine Source-Code-Dokumentation in Visual Studio

```
/// <summary>
/// Updates the progress or status information on the last converted image.
/// </summary>
/// <remarks>This function is called through an invoke of _imageInfoDelegate.</remarks>
/// <param name="thumbnail">The thumbnail just handled.</param>
/// <param name="targetFile">The name of the file for the converted image.</param>
/// <param name="toSize">The size for the converted image.</param>
private void DisplayImageInfo (Thumbnail thumbnail, string targetFile, Size toSize)
{
   // some code
```

1.1 Code Dokumentieren mit XML

Durch Eingabe von `///` im Visual Studio über einer Methode im Code wird automatisch ein Dokumentationsgerüst in XML erzeugt. Diese leeren Texte und die Beschreibung der Parameter ist nun einzugeben. (siehe Beispiel im Rahmen)

1.2 Recommended XML Tags for Documentation Comments

C#

http://msdn.microsoft.com/en-us/library/b2s063f7.aspx

Visual Basic .net

http://msdn.microsoft.com/en-us/library/ms172653(VS.80).aspx

1.3 XML Docu im Visual Studio aktivieren

In den Projekteinstellungen muss die XML-Code-Dokumentierung aktiviert werden. *(XML documentation file)* Damit wird beim Kompilieren zu jeder Assembly ein `XML`-File mit der Dokumentation aus dem Sourcecode geschrieben.

2. Ndoc

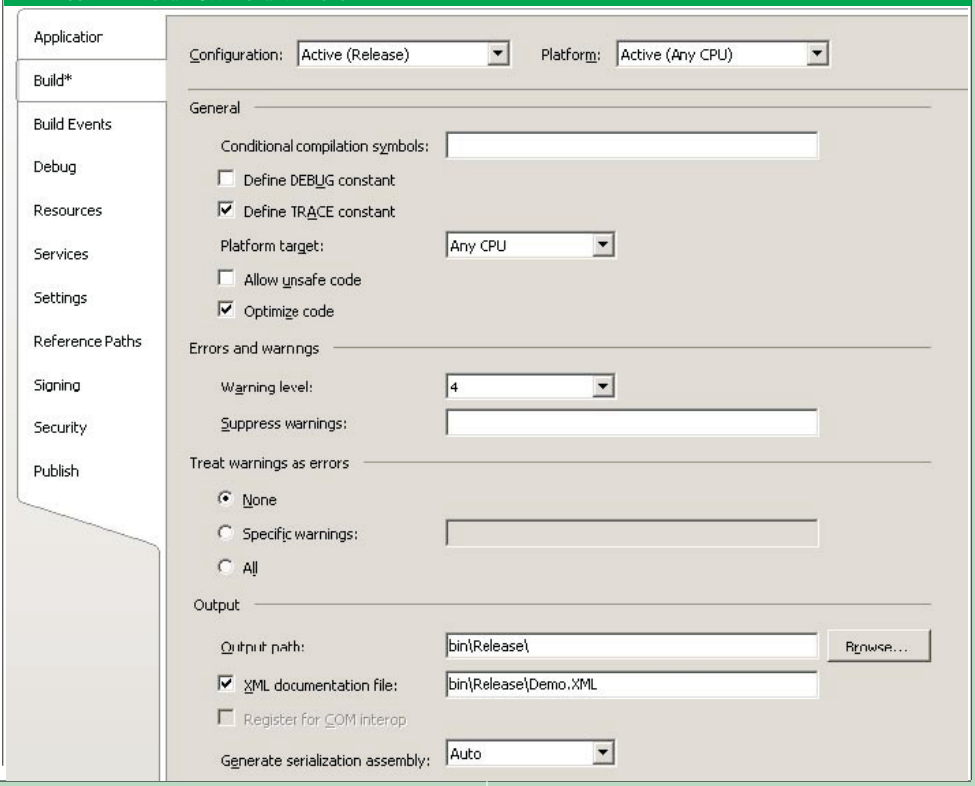
Das bis zur .net-Version 2.0 Tool zur Generierung von Code Dokumetation im MSDN Stil in den Formaten `chm`, `html`, … war **Ndoc**. Assemblies ab den Versionen 3.0 werden jedoch nicht mehr unterstützt. Das Tool wird seit 2005 nicht weiterentwickelt.

http://ndoc.sourceforge.net/

3. Sandcastle

Mit Sandcastle kann man – wie mit Ndoc – Codedokumentation erstellten, hier werden alle .net-Versionen unterstützt, hier findet eine kontinuierliche Weiterentwicklung statt. Die Funktionsweise ist gleich wie Ndoc. Je nach Sandcastle-Einstellungen wird eine Dokumentation im MSDN-Stil für unterschiedliche Ziel-

XML-Docu im Visual Studio aktivieren

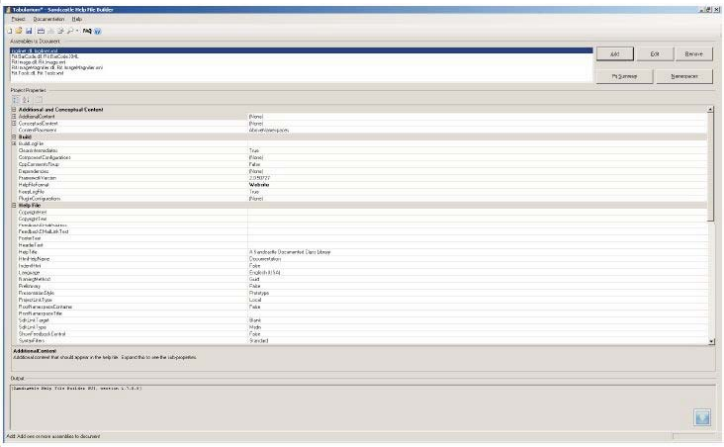


formate erstellt. Zielformate können eine `chm`-Datei oder `html`-Files sein. (oder beides)

Download

http://www.codeplex.com/SHFB

http://www.codeplex.com/Sandcastle



Mit *Add* werden die Assemblies mit zugehöriger XML Datei hinzugefügt, auch die Anhängigkeiten, da Sandcastle auch reflection nutzt.

Einstellungen

Build	
BuildLogFile	
CleanIntermediates	True
ComponentConfigurations	(None)
CppCommentsFixup	False
Dependencies	(None)
FrameworkVersion	2.0.50727
HelpFileFormat	**Website**
KeepLogFile	True
PlugInConfigurations	(None)
Help File	
CopyrightHref	
CopyrightText	
FeedbackEMailAddress	
FeedbackEMailLinkText	
FooterText	
HeaderText	
HelpTitle	A Sandcastle Documented Class Library
HtmlHelpName	Documentation
IndentHtml	False
Language	Englisch (USA)
NamingMethod	Guid
Preliminary	False
PresentationStyle	Prototype
ProjectLinkType	Local
RootNamespaceContainer	False
RootNamespaceTitle	
SdkLinkTarget	Blank
SdkLinkType	Msdn
ShowFeedbackControl	False
SyntaxFilters	Standard

HTML Help 1	
BinaryTOC	True
IncludeFavorites	False
HTML Help 2	
CollectionTocStyle	Hierarchical
HelpAttributes	(None)
HelpFileVersion	1.0.0.0
IncludeStopWordList	True
PlugInNamespaces	ms.vsipcc+, ms.vsexpresscc+
Paths	
HtmlHelp1xCompilerPath	
HtmlHelp2xCompilerPath	
OutputPath	C:\Inetpub\wwwroot\CodeDocu\BarCode
SandcastlePath	
WorkingPath	
Show Missing Tags	
AutoDocumentConstructors	True
ShowMissingNamespaces	True
ShowMissingParams	True
ShowMissingRemarks	False
ShowMissingReturns	True
ShowMissingSummaries	True
ShowMissingTypeParams	True
ShowMissingValues	False

Visibility	
ApiFilter	(None)
DocumentAttributes	False
DocumentExplicitInterfaceImplementations	False
DocumentInheritedFrameworkInternalMembers	False
DocumentInheritedFrameworkMembers	True
DocumentInheritedFrameworkPrivateMembers	False
DocumentInheritedMembers	True
DocumentInternals	False
DocumentPrivateFields	False
DocumentPrivates	False
DocumentProtected	True
DocumentProtectedInternalAsProtected	False
DocumentSealedProtected	True

Mittels GUI (SandcastleBuilderGUI.exe) lassen sich alle Parameter übersichtlich konfigurieren.

Sandcastle GUI setzt auf der Sandcastle *comand runtime* auf, d.h. um das GUI zu nutzen, muss beides installiert werden.

Nach der Bekanntgabe der Assemblies startet Sandcastle GUI auch schon mit der Kompilierung. Dabei wird mit den Default-Einstellungen im Verzeichnis `Help` eine `chm`-Datei erstellt.

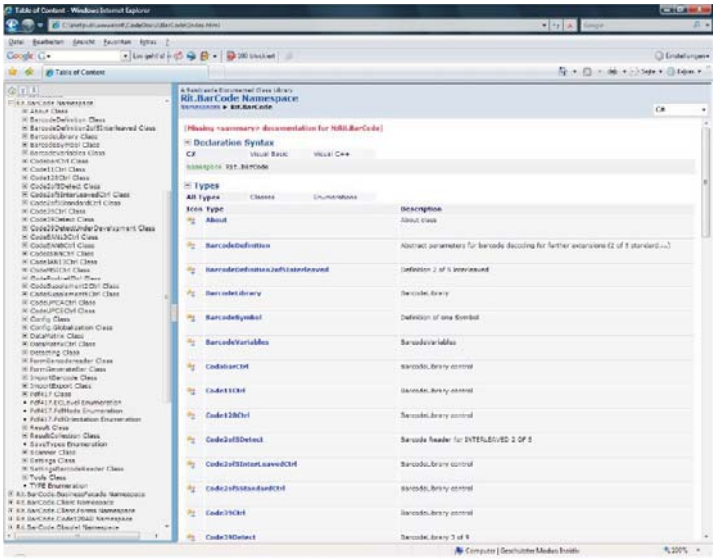
Ausgabeverzeichnis für IIS angeben

Paths	
HtmlHelp1xCompilerPath	
HtmlHelp2xCompilerPath	
OutputPath	C:\Inetpub\wwwroot\CodeDocu\
SandcastlePath	
WorkingPath	

Format angeben

Build	
BuildLogFile	
CleanIntermediates	True
ComponentConfigurations	(None)
CppCommentsFixup	False
Dependencies	(None)
FrameworkVersion	2.0.50727
HelpFileFormat	**Website**
KeepLogFile	True
PlugInConfigurations	(None)

Das fertige Ergebnis als HTML Dokumentation



Die Automatisierung erfolgt mittels Sandcastle *Comand Line Tool*, das man in eine Batchdatei gibt und über den Task Scheduler oder direkt im Build-Prozess aufruft.

```
"C:\Program Files\EWSoftware\Sandcastle Help File Builder\
SandcastleBuilderConsole.exe" $(SolutionDir)Doc\MyTestProject.shfb
```

Die Einstellungen werden in eine Datei mit der Endung `shfb` gespeichert.

Das `shfb`-File gibt man nun in zur Visual Studio Solution hinzu, und auch ins SourceSafe-System.

Bei der Automatisierung mittels Build Prozess werden nun zuerst aus den Quelldateien des SourceSafes Systems die Assemblies erzeugt. Im `shfb` wird auf die erzeugten Assemblies referenziert. Der Build der Codedoku wird im Anschluss gestartet. Somit wird sichergestellt, dass nach jedem Build immer die aktuelle Library der Code-Doku publiziert wird. Den Build lässt man in der Nacht am Server laufen, die Assemblies werden je nach Visual Studio Projekttyp gepublisht, kopiert oder in Setups verpackt. Die generierte Codedoku wird in ein IIS-Verzeichnis abgelegt und steht nun dem Entwicklerteam täglich aktuell zur Verfügung.

Somit steht einer sauberen Code-Dokumentation mit automatischer Generierung nichts mehr im Wege.