Stopwatch – Messung von Zeit von bestimmten Algorithmen oder Operationen

Manchmal möchte man wissen, wie lange mein Computer braucht um einen bestimmten Code auszuführen. Hierzu könnte man natürlich DateTime.Now für die Operationen verwenden. Allerdings ist dieser Wert ungenau. Einen Genaueren Wert erhalten wir mit dem Stopwatch. Dieser funktioniert aus recht simpel:
[crayon-676a0779813b5355512109/]
[crayon-676a0779813bd670892270/]

C# Vergleichsoperatoren ? : und ??

Oft findet man im Code ewig lange if – else Verzweigungen, die zuviel Platz einnehmen und irgendwann die Übersicht rauben.

Die beiden Operatoren ? : und ?? sehen merkwürdig aus, sind aber in der Handhabung sehr praktikabel.

Der ? : prüft ob der Wert true ist. Falls ja, zählt der linke Wert, wenn nicht, zählt der Wert rechts davon.
[crayon-676a0779819d3274308639/]
Der ?? Operator prüft, ob der Wert null ist. Bei nein, zählt der linke Wert, bei ja der rechte.
[crayon-676a0779819d9064508366/]

Das ganze in lang könnte so aussehen:

[crayon-676a0779819dc219698569/]
[crayon-676a0779819e6473364941/]
Ab C# 6 soll noch das Feature hinzufügen, dass auch das „Elternobjekt“ geprüft werden kann, ob dieser nicht null ist.
[crayon-676a0779819e7219838065/]

TSQL Mod – Eine DLL für schnelle SQL Server arbeiten

TSQLmod Download

Klasse db (Connection String)

benötigt in erster Linie die SQL Instanz. Nachdem das Objekt erfolgreich initialisiert wurde, wird auch gleichzeitig die Verbindung aufgemacht und die folgenden Methoden können genutzt werden.

LookUP(…)

gibt aus einem SQL Query den ersten Treffer der angegebenen Spalte als String wieder. Ideal um einen Wert aus der Datenbank auszulesen. Möglich ist es entweder die Spaltennummer oder den Spaltennamen anzugeben.

getRowList(…)

gibt eine List<string> oder generische List<T> von der angegebenen Spalte zurück. Man erhält quasi aus dem Select eine gewünschte Spalte

getRowStringBuilder(…)

ähnlich wie die getRowList(…) ist der Rückgabewert aber ein Stringbuilder, in welchen alle Zeilen einer selektierten Spalte enthalten sind.

getDynamicList(…)

erfordert eine Klasse welche dieselben Datentypen und Bezeichnung hat wie das SQL Select. Als Rückgabe erhält man man eine List<meineKlasse>, welche 1:1 so viele Elemente und Spalten hat wie das Sql Query. Das ganze arbeitet nicht mit Reflektionen, sondern nach dem Prinzip von diesem genialen Autor: KLICK

Dazu jeweils ein Beispiel. Ausgehend vom folgenden Select:

```
SELECT TOP 20 [ContactTypeID]
      ,[Name]
      ,[ModifiedDate]
  FROM [AdventureWorks2014].[Person].[ContactType]
```

100 %

Ergebnisse | Meldungen

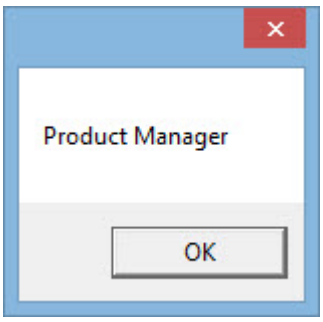
	ContactTypeID	Name	ModifiedDate
1	1	Accounting Manager	2008-04-30 00:00:00.000
2	2	Assistant Sales Agent	2008-04-30 00:00:00.000
3	3	Assistant Sales Representative	2008-04-30 00:00:00.000
4	4	Coordinator Foreign Markets	2008-04-30 00:00:00.000
5	5	Export Administrator	2008-04-30 00:00:00.000
6	6	International Marketing Manager	2008-04-30 00:00:00.000
7	7	Marketing Assistant	2008-04-30 00:00:00.000
8	8	Marketing Manager	2008-04-30 00:00:00.000
9	9	Marketing Representative	2008-04-30 00:00:00.000
10	10	Order Administrator	2008-04-30 00:00:00.000
11	11	Owner	2008-04-30 00:00:00.000
12	12	Owner/Marketing Assistant	2008-04-30 00:00:00.000
13	13	Product Manager	2008-04-30 00:00:00.000
14	14	Purchasing Agent	2008-04-30 00:00:00.000
15	15	Purchasing Manager	2008-04-30 00:00:00.000
16	16	Regional Account Representative	2008-04-30 00:00:00.000
17	17	Sales Agent	2008-04-30 00:00:00.000
18	18	Sales Associate	2008-04-30 00:00:00.000
19	19	Sales Manager	2008-04-30 00:00:00.000
20	20	Sales Representative	2008-04-30 00:00:00.000

einem erstellten Objekt der Klasse db:
[crayon-676a077981bec780247125/]
und ein string mit folgendem select:
[crayon-676a077981bf0133897362/]

LookUp(…)

[crayon-676a077981bf2523424943/]

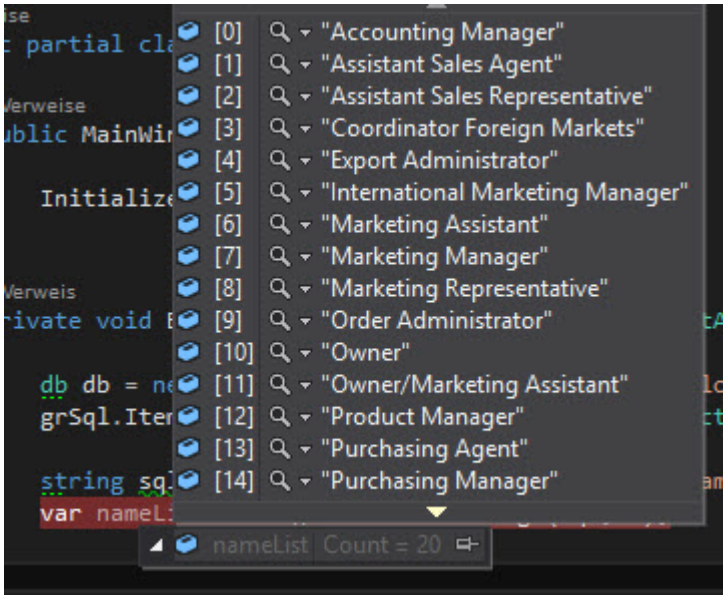
Antwort:



getRowList(…)

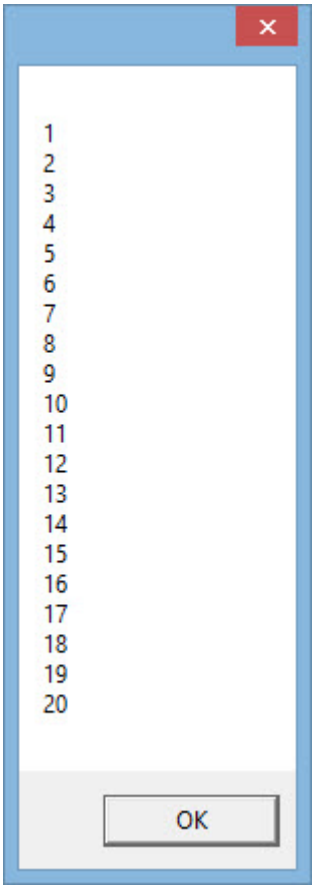
[crayon-676a077981bf4148079562/]

Antwort:



getRowStringBuilder(…)

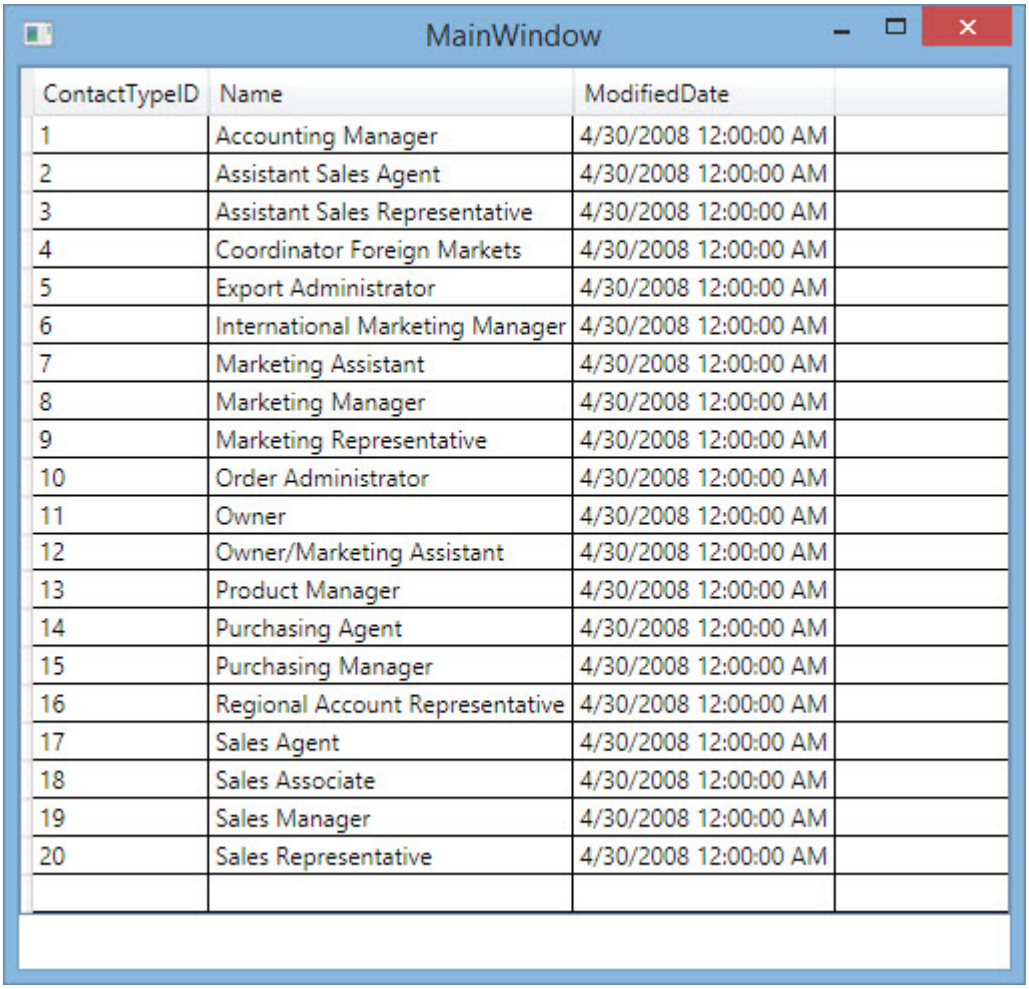
[crayon-676a077981bf5392975971/]

Antwort: 

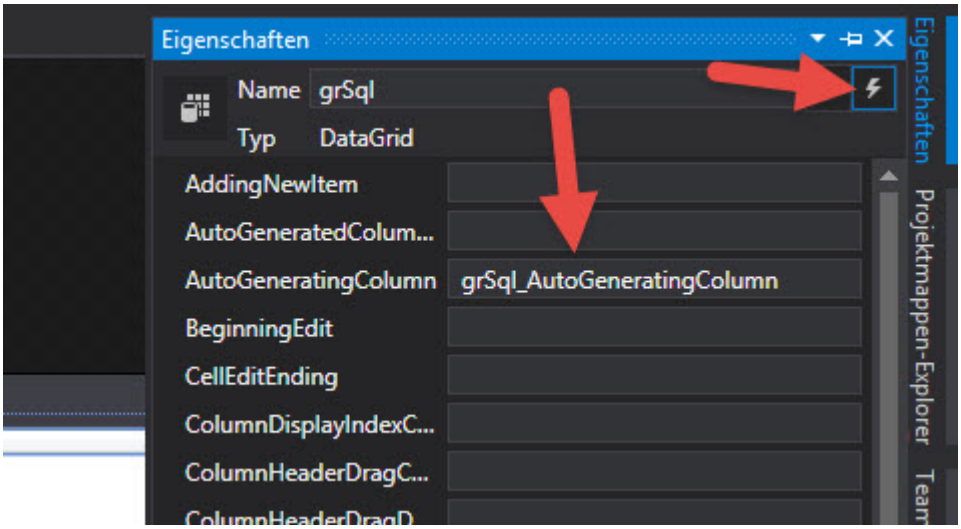
getDynamicList(…)

Wie oben bereits erwähnt, ist hierfür eine Klasse mit Propertys notwendig. Diese kann man ganz einfach auch mit den Methoden aus CreateClass – Klasse erstellen. Dazu weiter unten.
[crayon-676a077981bf7645406799/]
dann kann man so eine dynamische Liste ganz einfach erstellen:
[crayon-676a077981bfd900359687/]

diese Liste kann man nun z.B. einem Datagrid aus WPF zuordnen:
[crayon-676a077981bfe955035393/]
[crayon-676a077981bff322453833/]
Das ganze sieht dann so aus:



wem das Datumsformat stört, der kann dem Ereignis AutogeneratingColumn aus dem Datagrid eine Änderung des Datumsformates durchführen:



folgendes soll nun passieren, wenn das Ereignis eintrifft:

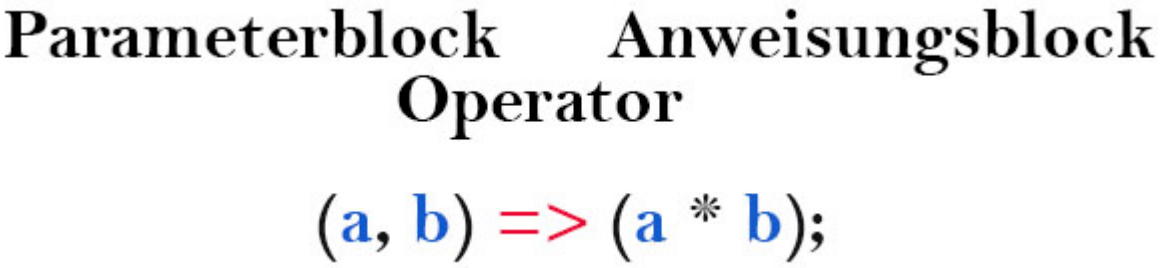
[crayon-676a077981c00492272070/]
Nun sieht das ganze so aus:

ContactTypeID	Name	ModifiedDate
1	Accounting Manager	30.04.2008
2	Assistant Sales Agent	30.04.2008
3	Assistant Sales Representative	30.04.2008
4	Coordinator Foreign Markets	30.04.2008
5	Export Administrator	30.04.2008
6	International Marketing Manager	30.04.2008
7	Marketing Assistant	30.04.2008
8	Marketing Manager	30.04.2008
9	Marketing Representative	30.04.2008
10	Order Administrator	30.04.2008
11	Owner	30.04.2008
12	Owner/Marketing Assistant	30.04.2008
13	Product Manager	30.04.2008
14	Purchasing Agent	30.04.2008
15	Purchasing Manager	30.04.2008
16	Regional Account Representative	30.04.2008
17	Sales Agent	30.04.2008
18	Sales Associate	30.04.2008
19	Sales Manager	30.04.2008
20	Sales Representative	30.04.2008

Lambda Ausdrücke oder anonyme Methoden

Lambda Ausdrücke werden in Kombination mit Delegaten dazu verwendet um schnelle Anweisungen auszuführen und sich dabei jede Menge Code zu ersparen.

Ohne Lambda und anonymer Methode für ein Delegatenaufruf in etwa so aussehen:



Neue Eigenschaft vom Typ delegate erstellen. (Rückgabewert ist ein int, mit 2 in Parameter a und b)
[crayon-676a077981e4a489966339/]
Dann müssen wir eine Methode haben, die ebenfalls denselben Rückgabewert und Parameter besitzt (Anzahl und Typ sind entscheidend)
[crayon-676a077981e4f198648603/]
Nun könnte man ein Objekt vom den oben erstellten MyDelegate erstellen und dem die Methode addiere zuweisen:
[crayon-676a077981e51374719029/]
Durch den Aufruf des Objektes dlg zeigt der Delegat auf die Methode addiere, führt diese aus und gibt den Rückgabewert zurück:
[crayon-676a077981e53031793630/]

_____

Dies ist in der Tat etwas umständlich. Vor allem benötigt man immer ein Methode, die man dem Delegaten immer zuweisen muss.

schneller geht es mit den Lambda Ausdrücken.:
[crayon-676a077981e55054291295/]
Die delegaten delegat1 bis delegat 5 tun immer dasselbe, nur wird der Code dadurch sehr kompakter und wenn man delegaten erstmal verstanden hat, wird es auch übersichtlicher.

Hat man einen Delegaten ohne Rückgabewert, würde der Lambdaausdruck mit Klammer auf- Klammer zu sein:
[crayon-676a077981e57205940010/]

Wegen den => Operator werden die Lampdaausdrücke oft mit Linq

verwechselt. Das eine hat mit dem anderen aber nichts zu tun

XAML Assembly Version an window Title binden

Um die Assembly Version irgendwo im Programm optisch darzustellen, kann man diese z.B. im Titelbereich des windows anzeigen lassen.

1. in app.xaml.cs folgende Eigenschaft hinzufügen:
[crayon-676a07798211d459199682/]
2. Der XAML Code zum binden sieht dann so aus
[crayon-676a07798212159862628 3/]

Vom Projekt bis Fertigstellung – Teil 1: Projektbeginn, Programmablaufplan (PAP)

Neben der eigentlichen Arbeit eines Entwicklers am Programmcode, sind vor allem für größere Projekte Diagramme,

Notizen, und Pläne von sehr hoher Bedeutung und werden nur zu gerne unterschätzt. Vernachlässigt man dieses, verliert den Durchblick und am Ende die Motivation überhaupt weiter zu machen. Was ist also wichtig, wenn man ein Projekt plant?

1. Ziele setzen

Wichtig ist, dass man zunächst nicht alle gewünschten Features auf einmal versucht einzubauen, sondern dies bewusst trennt. Auf der einen Seite habe ich die Grundfunktionen bzw. das eigentliche Programm und auf der anderen Seite der Wichtigkeit nach sortierte Features, die ich danach implementieren werde. Beides schreibt man am besten auf ein Blatt Papier, trennt sie und legt sich den Basisfunktionszettel gut Sichtbar auf dem Schreibtisch.

2. Welche Technologien? Welche Themen?

Welche Programmiersprache und Datenbank zum Einsatz kommt ist eher die leichtere Frage, die geklärt werden muss. Wichtiger ist, dass man sich dann aber mit den Themen, die auf einen zukommen genau informiert. Möchte man einen Server programmieren, so schaut man sich die Protokolle genau an und lernt erstmal den theoretischen Teil.

3. Programmablaufplan (PAP)

Wieder mit Zettel und Stift bewaffnet kann man nun einen ungefähren Ablauf des Programms erstellen. Wichtig ist, dass man sich hier eben nur auf die Basisfunktionen beschränkt. Der Rest kommt, wenn alles fertig ist. Hier gibt es einen einfachen Einstieg in PAP -> http://www.mrknowing.com/2014/03/13/wie-erstelle-ich-ein-pap-programmablaufplan/. Vielleicht ist es nicht wichtig die einzelnen Formen genau zu kennen, doch ist es sehr Hilfreich, sich im groben aufzuschreiben, was genau passieren soll, wenn man dies oder jenes drückt / einstellt. Eben die if / else Verzweigungen sollten gründlich überdacht werden.

Methode mit unendlich vielen Parameter mit params

Hat man den Fall, dass man an eine Methode unendlich viele Parameter übergeben möchte oder die Anzahl vielleicht unbekannt ist, so bietet C# die Möglichkeit über params und ein Array unendlich viele Parameter zu hinterlegen. Ein Beispiel:
[crayon-676a0779822db183854445/]
Zu beachten ist, dass der Typ unbedingt ein array [] sein muss.

PHP Variablen

Die Variablen in PHP sind zu mindestens bei der Initialisierung Typen undefiniert. Es ist so, als würde man unter C# ausschließlich mit dem var arbeiten.

Weiter kann die Variable entweder local, global oder static sein. Ist eine Variable außerhalb von Scopes {} definiert, so ist diese global. Kann aber nicht in den Scopes verwendet werden.

Wird eine Variable in den Scopes definiert, so gilt sie als local und kann nur in den Scopes verwendet werden.

Befindet sich eine Variable in den Scopes und ist diese als

static deklariert, so behält sie ihren Wert, auch wenn der Programmcode die Scopes bereits abgeschlossen hat.

[crayon-676a0779824d7713877063/]
Das Resultat wäre so eben 1 2 3. Hätte man hier nicht static genutzt, wäre das Ergebnis immer 1, da die Variable, wie bereits angesprochen gelöscht wird, sobald sie die Scopes verlässt.

Interessant sind auch die anonymen Methoden unter PHP. Man hat die Möglichkeit die Funktion quasi in eine Variable zu stecken. Nimmt man das obere Beispiel, so kann man einfach
[crayon-676a0779824dc310962652/]
dazu verwenden die Funktion zu anonymisieren.

Variablentypen

PHP hat weit weniger Typen als z.B. noch C#.

Während der boolean, integer, string und null gleich sind, haben folgende Typen so ihre Besonderheiten.

Eine numerische Zahl mit Nachkommastellen ist zwar ein **float**, fragt man aber mit
[crayon-676a0779824dd550213216/]
den Variablentypen ab, sagt PHP, dass es ein **double** ist.

Der **array** in PHP darf alle Möglichen Typen enthalten und erfüllt sogar die Funktion einer aus C# bekannten List oder Diktionary mit Key and Value Prinzip.

Betrachtet man das manuel von PHP.net, so findet man auch eigenartige Funktionen um sogar die im array enthaltenen Werte zu summieren, zu mischen usw.

Man kann in PHP Funktionen und sogar Klassen in eine Variable oder als Variable? speichern. Der Typ ist dann ein object.
[crayon-676a0779824df164775147/]
Wie auch in C# kann man jeden anderen Typ in ein object konvertieren, nennt sich hier aber nicht casten… Dies geschieht aber im Prinzip genau so:
[crayon-676a0779824e0269587489/]
Als letzter Type ist die Resource, welche quasi einen Zeiger auf externe Ressourcen zeigt. Diese beinhaltet z.B. das handeln mit Datenbanken, Pdf Dateien usw. Die folgende Liste zeigt einen Auszug davon:

Ressource-List

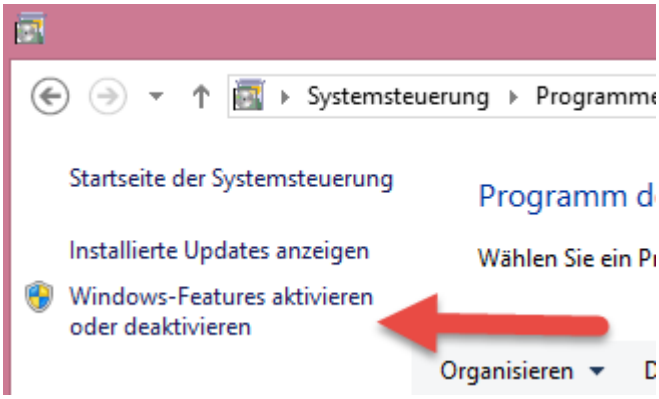
PHP 5.6.8 unter IIS und Windows 8

Mein nächstes Ziel ist die Webkomponente der Programmierung kennen zu lernen. Dazu gehört neben HTML, CSS, JavaScript auch die serverseitige Sprache PHP.

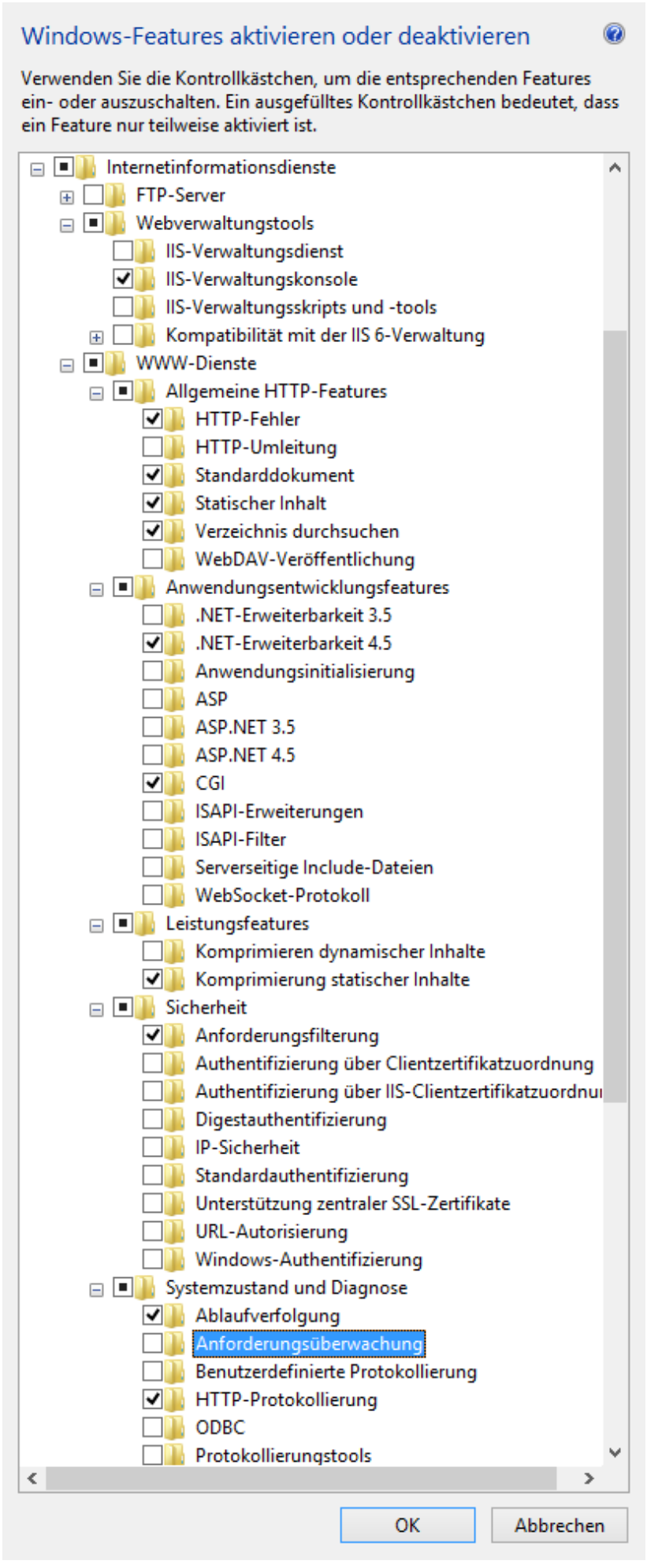
Erstes Ziel ist das Einrichten von PHP auf dem IIS (Man könnte natürlich den einfacheren Weg über XAMP gehen oder den alternativen Weg über den Webserver Apache). Ich wollte das einfach unter IIS machen, weil IIS eigentlich aus einfach von Haus aus auf dem Windows Rechner schlummert und ich so nicht wirklich was installieren brauche. Ok, los gehts.

1. Installieren von IIS

Systemsteurung – Programme – Programme und Features auf



klicken und den Internetinformationsdienste nach folgenden Muster aktivieren.


Windows-Features aktivieren oder deaktivieren
Verwenden Sie die Kontrollkästchen, um die entsprechenden Features ein- oder auszuschalten. Ein ausgefülltes Kontrollkästchen bedeutet, dass ein Feature nur teilweise aktiviert ist.
Internetinformationsdienste
FTP-Server
Webverwaltungstools
IIS-Verwaltungsdienst
IIS-Verwaltungskonsole
IIS-Verwaltungsskripts und -tools
Kompatibilität mit der IIS 6-Verwaltung
WWW-Dienste
Allgemeine HTTP-Features
HTTP-Fehler
HTTP-Umleitung
Standarddokument
Statischer Inhalt
Verzeichnis durchsuchen
WebDAV-Veröffentlichung
Anwendungsentwicklungsfeatures
.NET-Erweiterbarkeit 3.5
.NET-Erweiterbarkeit 4.5
Anwendungsinitialisierung
ASP
ASP.NET 3.5
ASP.NET 4.5
CGI
ISAPI-Erweiterungen
ISAPI-Filter
Serverseitige Include-Dateien
WebSocket-Protokoll
Leistungsfeatures
Komprimieren dynamischer Inhalte
Komprimierung statischer Inhalte
Sicherheit
Anforderungsfilterung
Authentifizierung über Clientzertifikatzuordnung
Authentifizierung über IIS-Clientzertifikatzuordnu
Digestauthentifizierung
IP-Sicherheit
Standardauthentifizierung
Unterstützung zentraler SSL-Zertifikate
URL-Autorisierung
Windows-Authentifizierung
Systemzustand und Diagnose
Ablaufverfolgung
Anforderungsüberwachung
Benutzerdefinierte Protokollierung
HTTP-Protokollierung
ODBC
Protokollierungstools
OK
Abbrechen

Wenn IIS richtig installiert wurde, sind wir für uns unter http://localhost/ erreichbar.

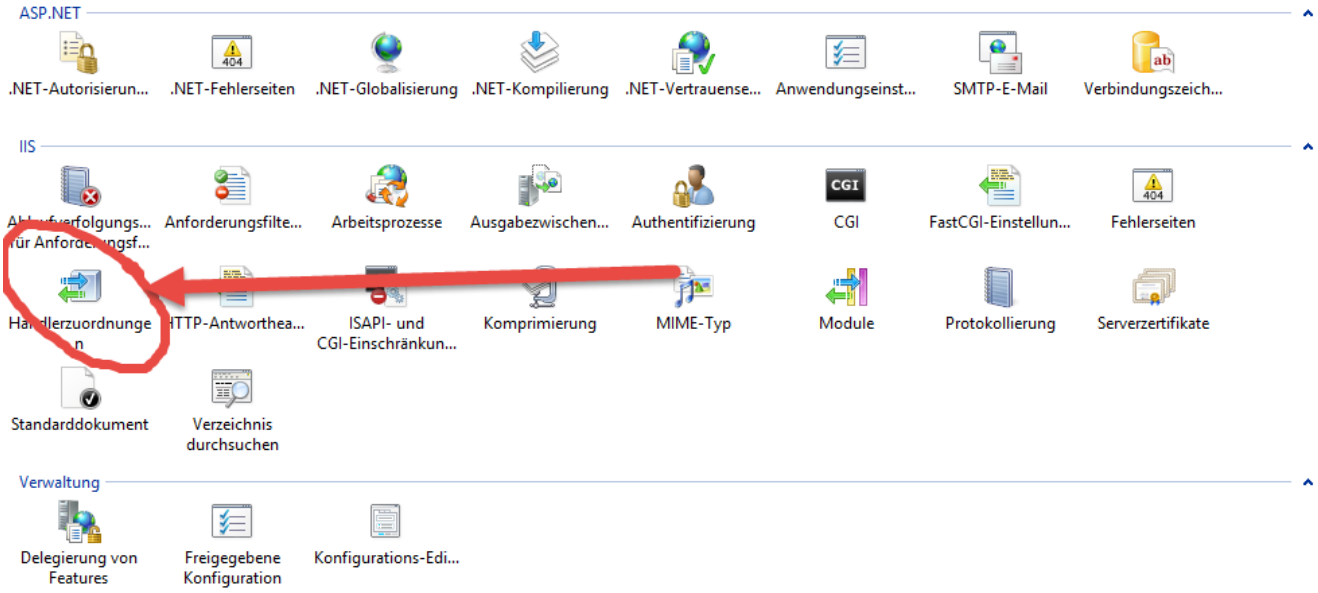
2. PHP installieren

Die Version **VC11 x86 Non Thread Safe** PHP für Windows auf http://windows.php.net/download/ herunterladen, und in C:\Program Files (x86)\PHP\php 5.6.8 entpacken

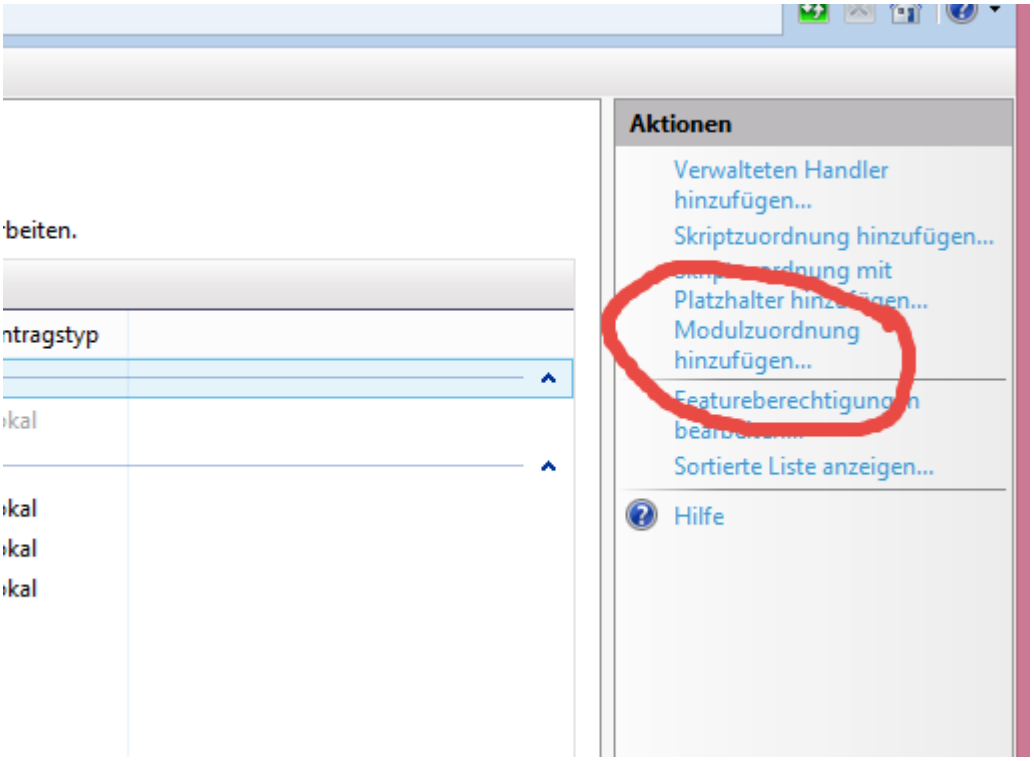
(Wegen den User Rechten muss man die Ordnerstruktur manuell anlegen)

3. IIS für PHP einrichten

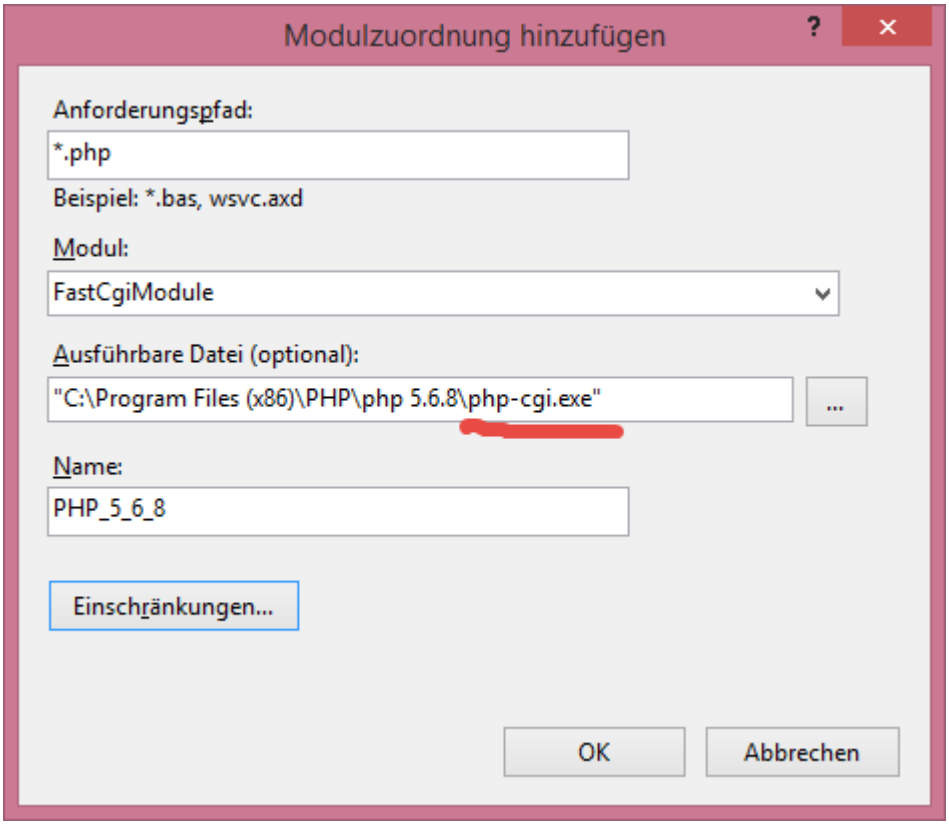
unter Start „IIS“ eintippen und den IIS-Manager wählen.



Dort auf Händlerzuordnungen und



Modelzuordnung hinzufügen…



Fenster folgend ausfüllen. Wichtig, bei der Ausführbaren Datei die **php-chi.exe** auswählen!

Dann auf OK und die Meldung mit **JA** auswählen.

Nun kann man eine info.php Datei mit dem Inhalt

<?php phpinfo() ?> in C:\inetpub\wwwroot speichern und schauen, was passiert wenn wir diese nun unter http://localhost/info.php versuchen zu öffnen.

Werden uns die Werte richtig dargestellt, ist alles richtig installiert. Kann die Seite aber nicht geöffnet werden, fehlt dem höchstwahrscheinlich das C++ Redistributable, welches man hier wiederum herunterladen kann http://www.microsoft.com/en-us/download/details.aspx?id=30679

HttpListenerContext decode/encode Umlaute

Liest man die Url aus dem HttpListenerContext, die Umlaute wie äöü enthält, so sieht das ungefähr so aus:

aus süß wird s%FC%df.

Abhilfe schaft da die Klasse HttpUtility:
[crayon-676a0779826d4508814945/]
möchte man zurück encodieren macht man einfach :
[crayon-676a0779826d7047557628/]