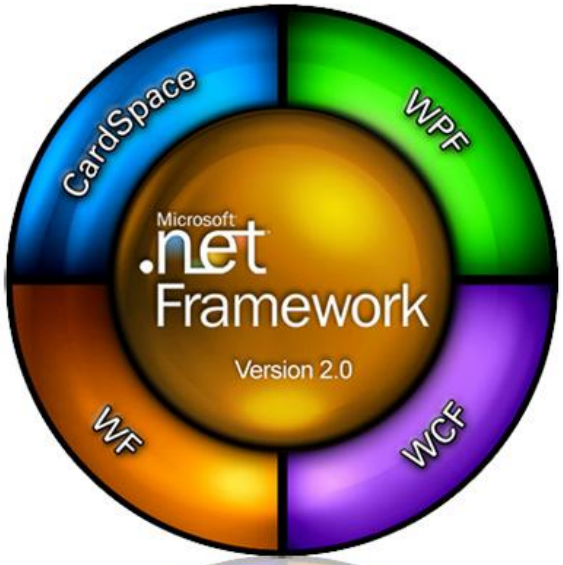

C# Programmierung

Eine Einführung in das .NET Framework

Vorstellung Kursleiter

- **Florian Rappl**
- 9 Jahre Erfahrung mit C#
- Ausbildung zum Fachinformatiker bei Siemens
- Breites Spektrum an Technologien
- Früher Line Of Business und Web Applikationen
- Heute High Performance Computing und MVC 3

Formalitäten

- Vorlesung immer 9-12
- Kleine Pause in Mitte der VL
- Übung ab 13:00 in CIP-Pool Physik
- Schein am Ende durch Abschlussprojekt
- Bevorzugte Durchführung als Zweiergruppe
- VisualStudio für Zuhause über MSDNAA

Randbedingungen

- Kurs ist interaktiv gedacht
- Kursskript mit Übungsaufgaben (Tag 1-8)
- Fragen, Wünsche und Anmerkungen während der Vorlesung erwünscht
- Kursziele: C# **kennen** und *schätzen* lernen
- *Erfahrungen* der Teilnehmer (C/C++, Java, …) und
- *Erwartungen* der Teilnehmer an den Kurs

Zeitplan diese Woche

Mo	Di	Mi	Do	Fr
Wdh C VisualStudio Datentypen Funktionen Kontrolle Rückgabe Operatoren ToString()	Datentypen Arrays Operatoren Parameter Überladen Heap / Stack Klassen (un)Boxing	Statisches Modifizierer Eigenschaften Polymorphie Baupläne Konstruktor WinForms Sealed, Partial	Events Delegates Operatoren Exceptions Namespaces Bibliotheken .NET Klassen Serialisierung	Mehr .NET Designer ToolBox Dialoge Application Systeminfos Controls Bitflags

Mo

Wdh C
VisualStudio
Datentypen
Funktionen
Kontrolle
Rückgabe
Operatoren
ToString()

Tag 1

Grundlagen

Die Gemeinsamkeiten mit C, Einordnung von C# und unsere ersten kleinen (Konsolen-) Programme

Einordnung von Begriffen

- Visual Studio ist unsere *Entwicklungsumgebung* (IDE)
- .NET ist das Framework, Sammlung von *Bibliotheken*
- C# ist die Wahl unserer *Programmiersprache*
- C# bedeutet: **C** Syntax, Java **OOP** und darüber hinaus

Möglichkeiten mit C#

- Sony: PS Vita, Xperia Play, Tablet S, Tablet P
- iOS mit MonoTouch in nativen Code (Cocoa)
- Android mit MonoDroid und den Android APIs
- WindowsPhone 7, Windows Mobile 6.x / CE
- Mac OSX und Linux mit Mono (und Cocoa# oder Gtk#)
- Unity3D (iOS, Android, Wii, PS3, XBOX, PC, …)
- XNA (Windows, XBOX, WindowsPhone 7)
- Windows / Windows 8 (Metro) und Windows Server
- Webentwicklung mit ASP.NET oder ASP.NET MVC
- und noch vieles mehr…

Wiederholung C

- Typisches „Hallo Welt!“ Programm in C

```
#include <stdio.h> /* Ein Kommentar */
int main() {
   printf("Hallo Welt!\n");
   return 0;
}
```

- Immer mit Prototypen arbeiten (Header-Dateien)
- Keine Klassen (C++: vorhanden aber kein muss!)
- Nur sehr wenige Datentypen (int, char, float, …)

Übergang zu C#

- Typisches „Hallo Welt!“ Programm in **C#**

```
using System; // Einzeilenkommentar (von C++)
class ExampleClass { /* Kommentar (von C) */
    static int Main() {
        Console.WriteLine("Hallo welt!");
        return 0;
    }
}
```

- Kein Arbeiten mit Prototypen notwendig
- Es muss mit Klassen gearbeitet werden (**OOP**!)
- Alles strikt in Namensräume eingeteilt (using)

Beispiel

01 - Hallo Welt

Analyse des Programms

- Möglichkeiten von VisualStudio
- Neue Kommentarart – „///“ = Dokumentation
- Im Visual Studio wird alles über Projekte gesteuert
- Möglichkeiten des Debugging-Vorgangs
- Handhabung von Breakpoints (Kurz: *F5*, *F10*, *F11*)

Namensräume

- Was sind Namensräume und wozu sind diese gut?
- Durch Namensräume erhalten wir Trennung und Ordnung der enthaltenen Objekte...
- Analogie: Landkarte
- Müssen Land für Stadt nennen

- Daher klare und konfliktfreie Aussage – ex- oder implizit

Anschnallen – und Los!

- Native Datentypen in C#

  `char, short, int, long, float, double, decimal, string, bool,…, object`

- Beschränken uns vorerst auf diese
- Konsolenausgabe über statische Konsolenklasse:

  Console --- z.B. `Console.Write(), Console.Read(), …`

- Zugriff auf Methoden einer Klasse mit „.“
- Operatoren wie in C: `+,-,*,/,%,!,<,>,==,<<,>>,…`

Beispiel

02 - Basics

Volle Kontrolle

- Im Prinzip alles analog zu C
- Betrachten wir z.B. Verzweigungen

```
if(bedingung)
{
    ...
}
else
{
    ...
}
```

sog. Yoda-Condition

```
if (5 == count)
```

- Bedingung muss hierbei immer ein Boolscher Wert sein

Kontrollstrukturen (1)

- Kopfgesteuerte Schleife mit

```
while(bedingung)
{
    ...
}
```

- Fußgesteuerte Schleife durch

```
do
{
    ...
}
while(bedingung);
```

Kontrollstrukturen (2)

- Zählerschleife mit

```
for(davor; bedingung; danach)
{
    ...
}
```

- Mit switch-case können viele Bed. abgefragt werden

```
switch(Variable)
{
    case Wert:
        ...
        break;
}
```

Beispiel

03 - Kontrollstrukturen

C# Feature - foreach

- Problem: Ein Array in **C** – uns interessiert der Wert

```
int i;
for(i = 0; i < sizeof(array) / sizeof(*array); i++)
        printf("Wert: %d\n", array[i]);
```

- Lösung in **C#** könnte so oder so sein:

```
for(int i = 0; i < array.Length; i++) // lange !
        Console.WriteLine("Wert: " + array[i]);
foreach(int value in array) // kürzer !
        Console.WriteLine(" Wert: " + value);
```

Beispiel

04 - Foreach

Beispielprogramm

- Ziel: Währungsumrechnung von Geld
- Benötigt: Umrechnungsfaktor und Geldmenge
- Feature: Typsichere Eingabe (was bedeutet `out`?)

- **Neues**: Ausnutzen von `ToString()` – Kleiner Vorgeschmack auf OOP!