

BASTA! ON TOUR

Rainer Stropek | software architects gmbh

Code Like the Wind

C# IDE in Visual Studio 2010

Abstract (German)

In diesem Workshop lernen Sie Funktionen von Visual Studio 2010 kennen, die Sie beim Schreiben von C#-Code effizienter und produktiver machen. Egal, ob Sie sich in fremden Code einlesen, in großen C# Solutions navigieren, neuen Code entwickeln oder Tests schreiben - in Visual Studio 2010 stecken viele hilfreiche Features, die Ihre Arbeit erleichtern werden. Der Workshop ist ideal für Entwickler, die neu in Visual Studio einsteigen oder jene, die noch nicht die Gelegenheit hatten, alle Feinheiten der Entwicklungsumgebung auszuforschen. Bringen Sie Ihren Laptop mit, Sie können alles, was Sie lernen gleich ausprobieren.

Introduction

- [software architects gmbh](#)
- Rainer Stropek
- Developer, Speaker, Trainer
- MVP for Windows Azure
- rainer@timecockpit.com
-  @rstropek



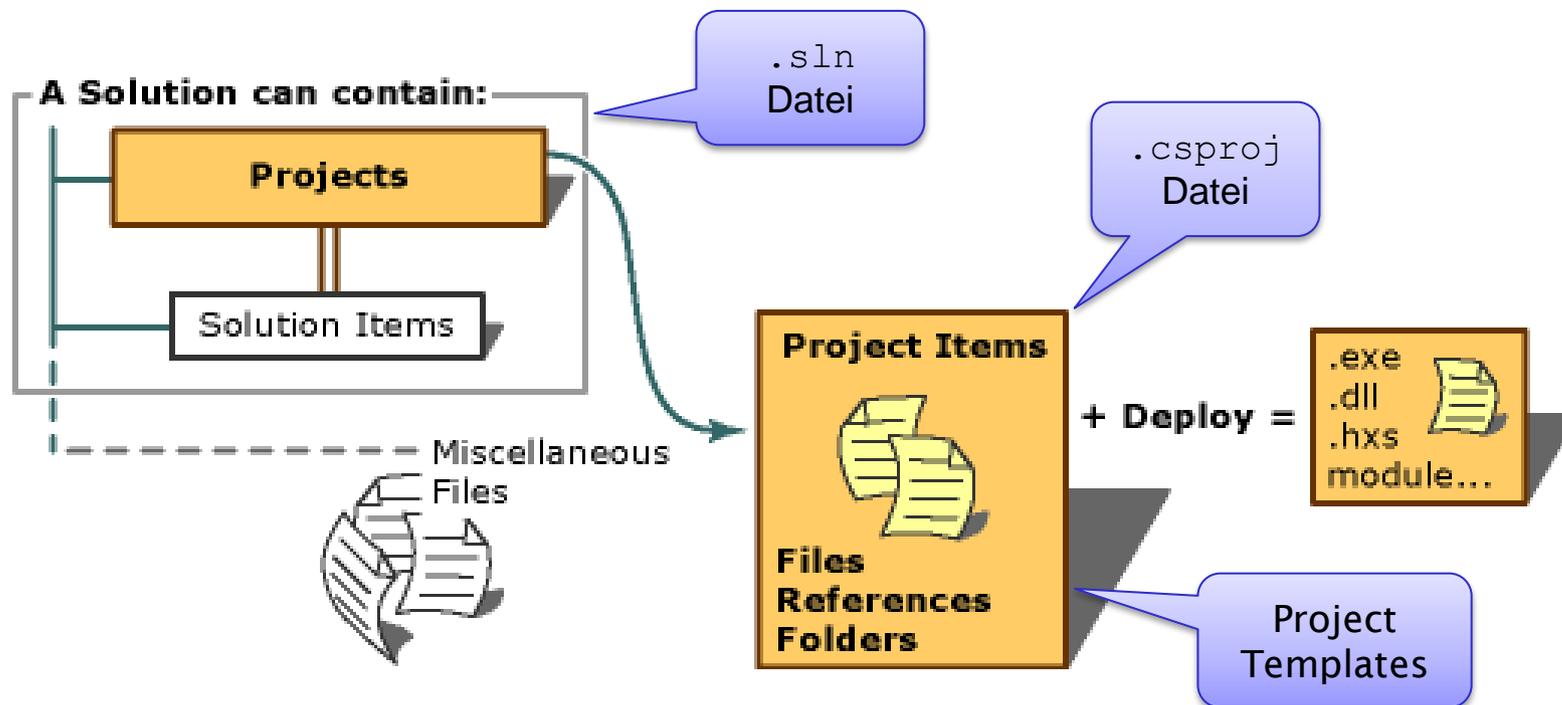
<http://www.timecockpit.com>

<http://www.software-architects.com>

SOLUTIONS UND PROJEKTE

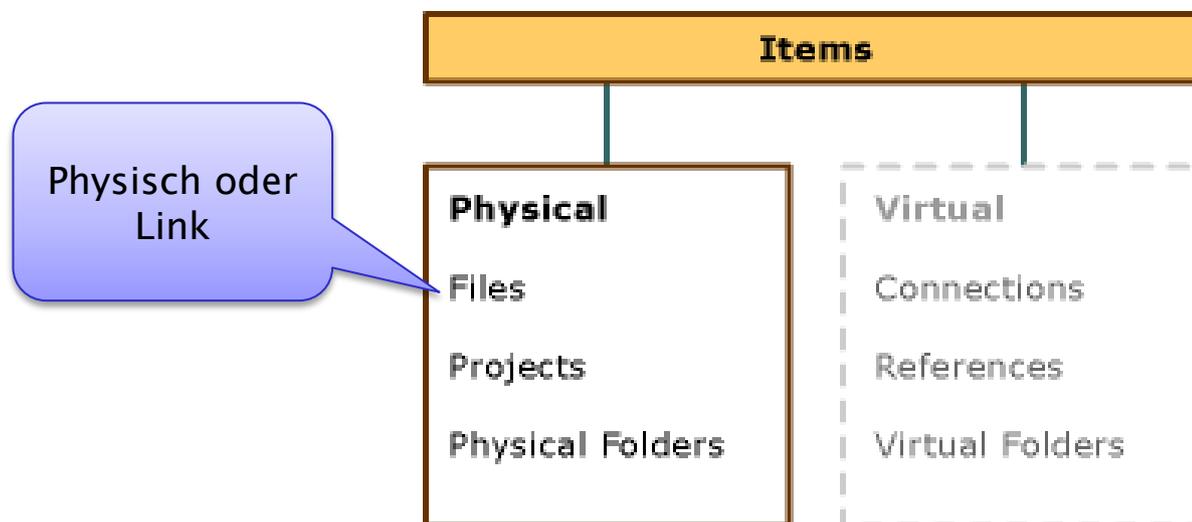
Solutions und Projects

- Solution = Zusammenfassung 1..n Projekten
- Solution = Einheit, die gemeinsam gebaut, konfiguriert und verteilt wird
 - Gilt für einfache Projekte, bei großen Projekten oft anders



Projects

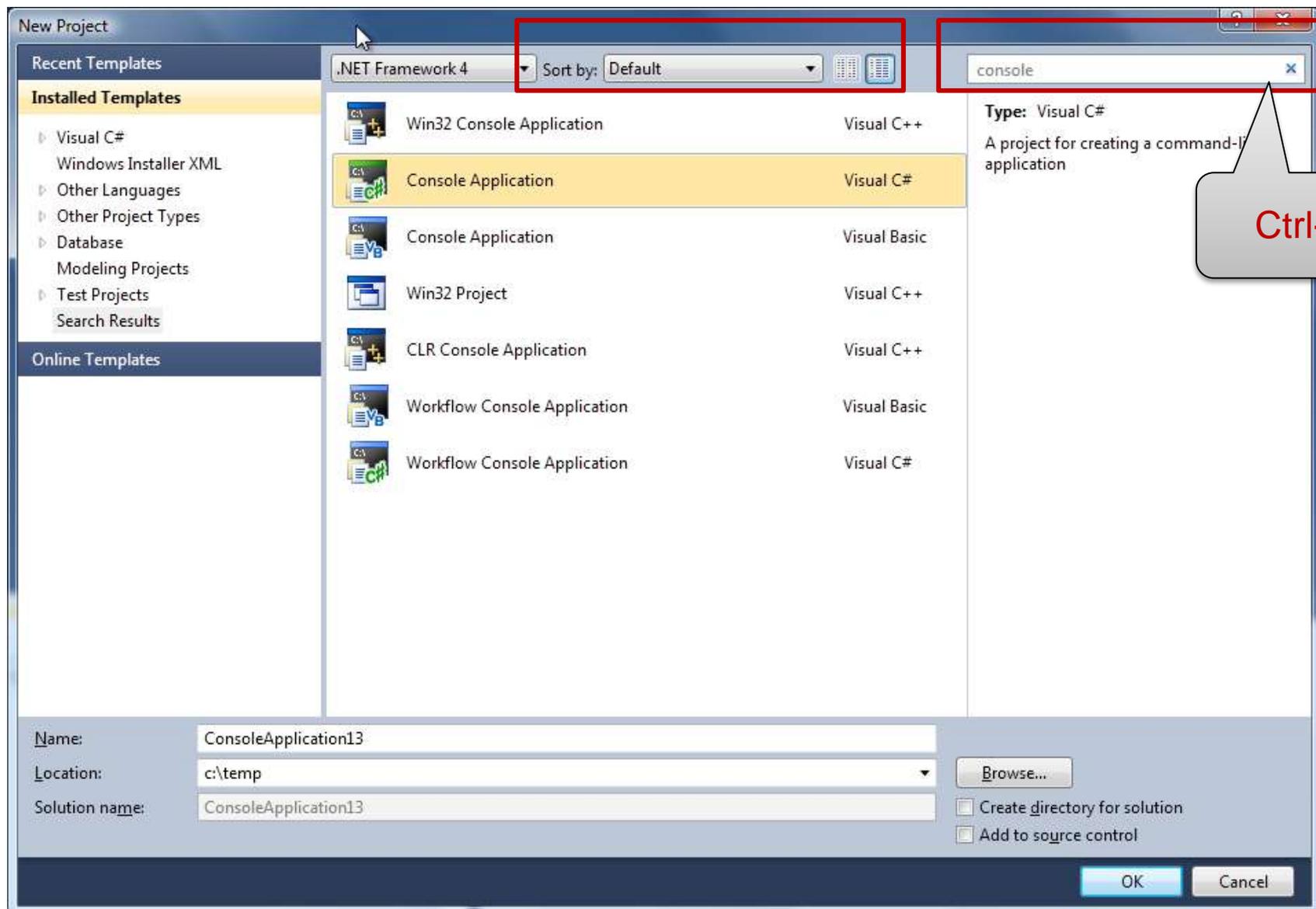
- [Project Templates](#)
- Project Properties
- Bei vielen Projekten in einer Solution: Solution Folders
- Stand-alone Projects: Solutions mit nur einem Projekt
 - Tipp: *General / Projects and Solutions / Options / Always show solution*
- Temporary Projects
 - Tipp: *General / Projects and Solutions / Options / Save new projects when created*



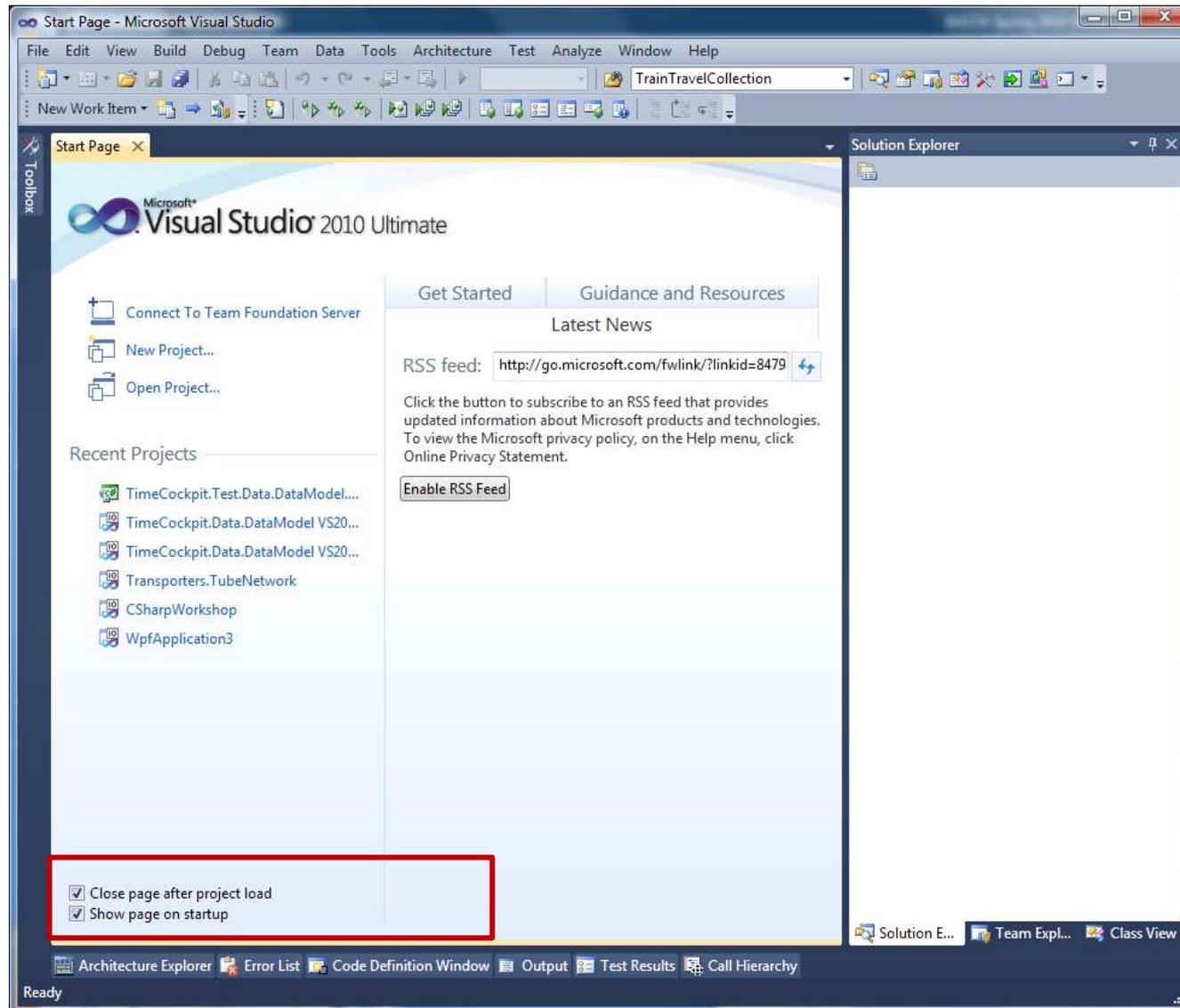
Visual Studio Templates

- Vorlagen für Solutions und Projekte
- Eigene Templates können erstellt werden
 - Teamarbeit
 - Häufig verwendete Projekttypen (z.B. in Beratungsprojekten)
- Starter Kits
 - Spezialform von Templates (technisch gesehen ident)
 - Beispielanwendungen, Lernanwendungen
- Details siehe [MSDN](#)

Verbessertes Project/New Dialog



Verbesserte Startpage



- Interaktion mit Windows Explorer...
- Stand-alone projects, temporary projects...
- Add New...
- Add Existing...
- Multi-Targeting...
- Unload and reload...
- Show all, refresh...

DEMO 1:

SOLUTION EXPLORER

VISUAL STUDIO EDITOR

The Great Escape auf Flickr - Fotosharing! - Windows Internet Explorer

http://www.flickr.com/photos/ukaaa/1102090491/

The Great Escape keyboard

Favorites | Vorgeschlagene Sites | Mehr Add-ons erhalten | The Money Converter - c...

The Great Escape auf Flickr - Fotosharing!

flickr® von YAHOO!

Angemeldet als anonym0511 | (1 neu) | Hilfe | Abmelden

Startseite | Sie | Organisieren | Kontakte | Gruppen | Entdecken | Suchen

The Great Escape

Verbreiten



Hochgeladen am 13. August 2007 von [ukaaa](#)

Fotostream von [ukaaa](#)

1.182 Uploads

durchsuchen

Dieses Foto gehört auch zu:

50 Most Popular (Album)

Sie sind beim ersten Foto.



50 Elemente

durchsuchen

Teil von: [Other](#)

Featured in [EXPLORE](#)

#1 on August 13, 2007!! Thanks everyone!

Dieses Foto hat Notizen. Bewegen Sie den Mauszeiger über das Foto, um sie anzuzeigen.

Visual C# 2008
Keybinding Reference Poster

EDITOR BASICS

Automatic Brace Matching

- Klammern () {} []...
- Strings "" @""
- Regionen #region #endregion
- Präprozessor #if #else #endif
- Genereller Tipp:
 - Visual Studio Color Schemes
 - Z.B. <http://studiostyl.es/>, etc.

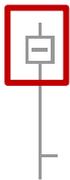
```
var catalog = new AssemblyCatalog(typeof(DisposableObject).Assembly);  
var container = new CompositionContainer(catalog);
```

Code Selection, Copy/Move

- Stream Mode
 - Maus oder **Shift(+Ctrl)+Cursor**
- Column Mode
 - Alt+Maus oder **Shift+Alt+Cursor**
- Cut, Copy, Paste
 - **Ctrl+X, Ctrl+C, Ctrl+V** 
 - Tipp: Clipboard ring (**Ctrl+Shift+V**)
 - Zugriff auf die letzten 20 kopierten Texte
 - Tipp: Ohne Markierung ganze Zeile ausschneiden/kopieren

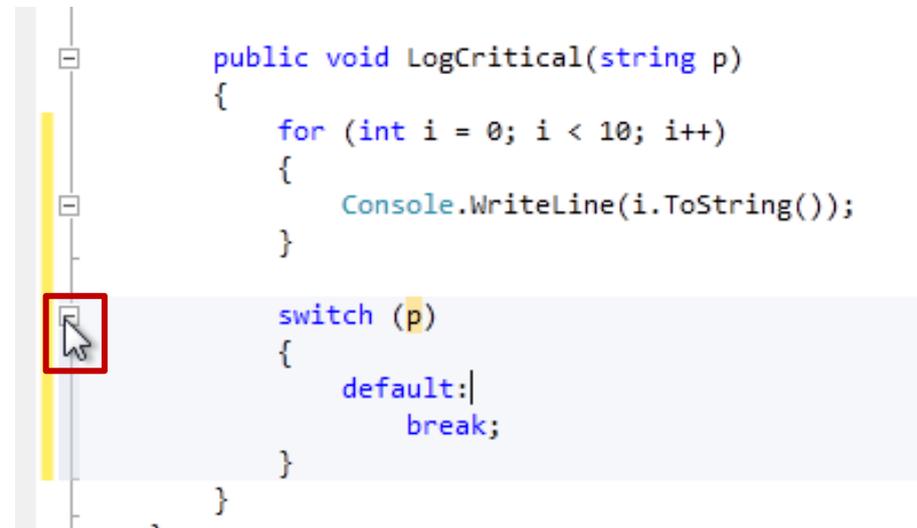
Outlining

- Toggle Outlining (**Ctrl+M, M**)
- Collapse to Definitions (**Ctrl+M, O**)
 - Tipp: #region Code Snippet



```
Sub Main()
    Dim Proj As EnvDTE.Project
End Sub
```

- Neu: Ad Hoc Blocks
 - Markieren des gewünschten Codeblocks
 - *Hide Selection* (**Ctrl+M, H**)
 - → Ad Hoc Block erzeugt



Coding Problem Indicators

```
static void Main(string[] args)
{
    new Program().Run();
}
```

Syntax
Error

```
public string ToString()
{
    return "";
}
```

Warning

```
private void Run()
{
```

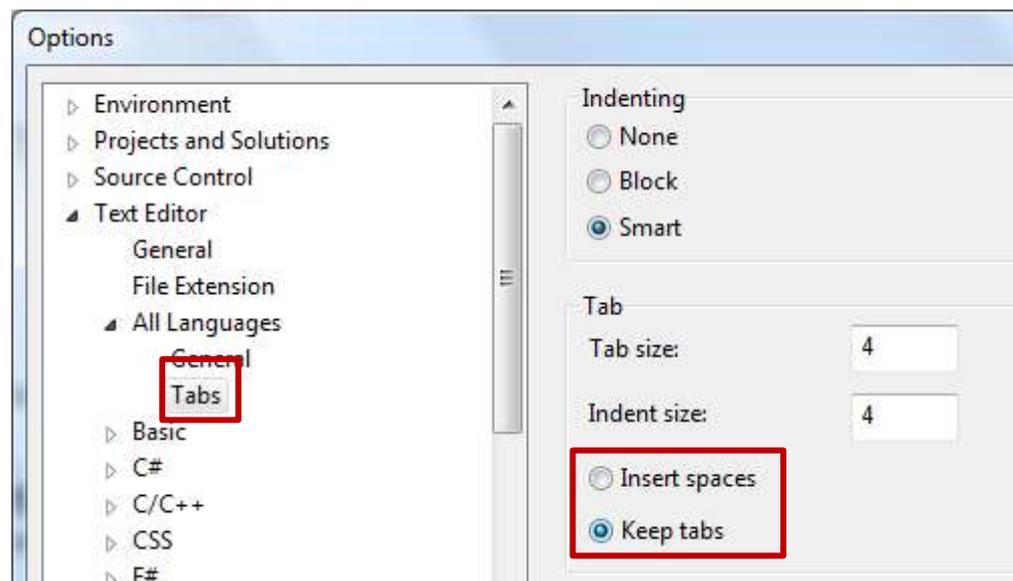
```
    var catalog = new AssemblyCatalog(typeof(DisposableObject).Assembly);
    var container = new CompositionContainer(catalog);
```

```
    container = "Hello world!";
```

Semantic
Error

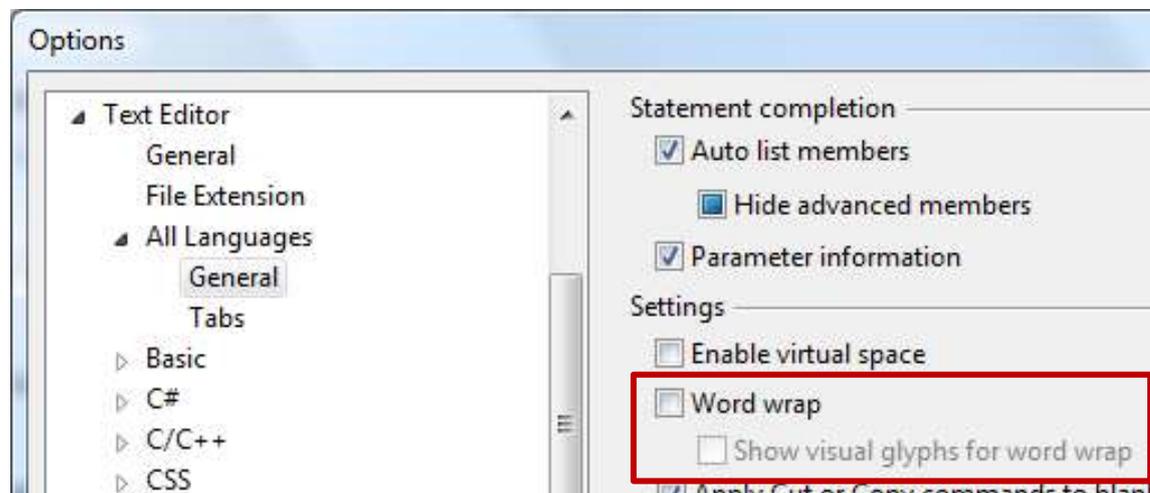
Indenting

- Tabs oder Spaces, das ist hier die Frage...
 - Wenn Tabs einheitliche Settings!
 - StyleCop: Spaces, keine Tabs
 - Entscheidung bleibt ihrem Geschmack überlassen
- Tipp: **Select + Tab**, **Select + Shift + Tab**

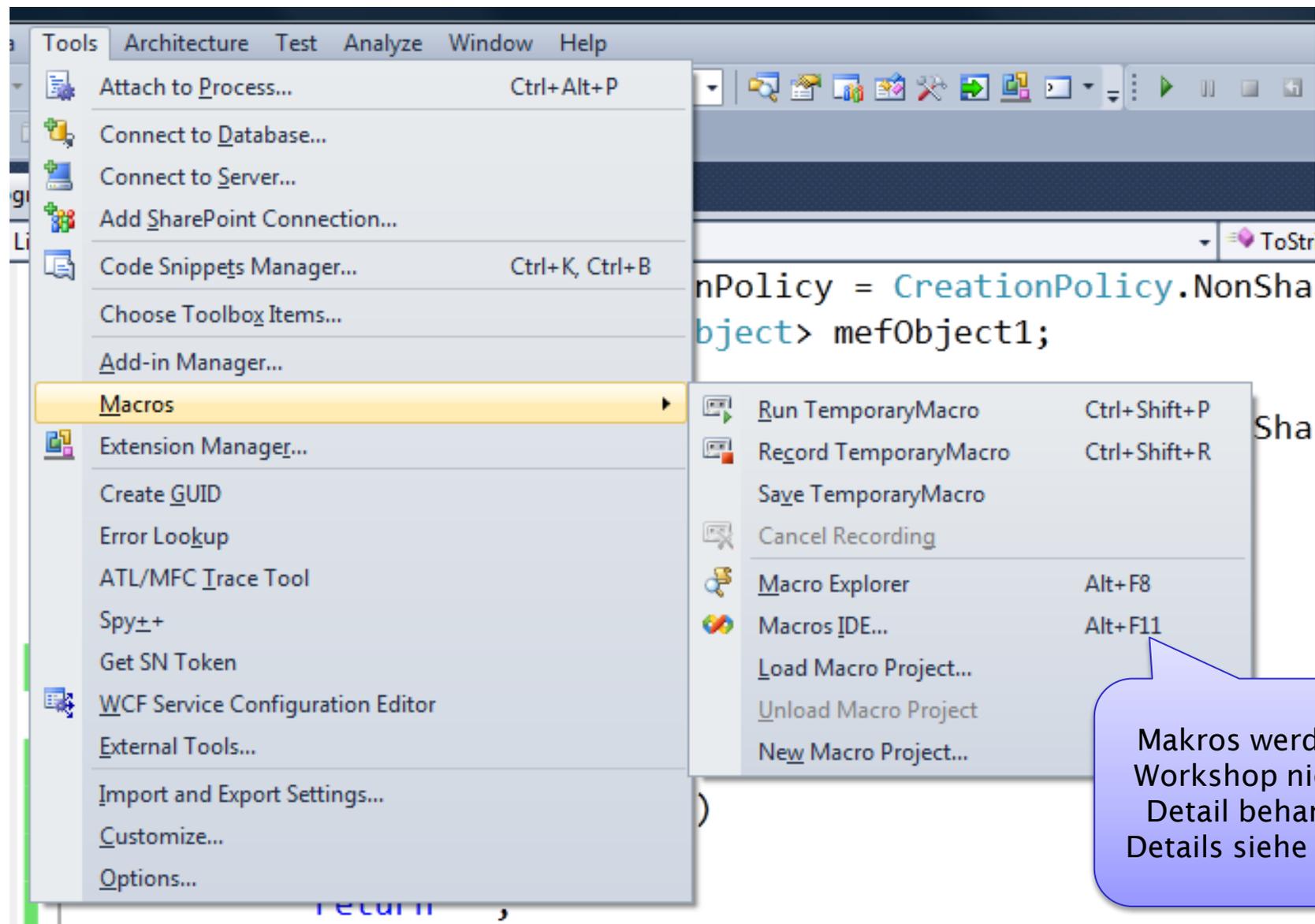


Sonstige Editor-Tipps

- Zooming
 - Zoom in Textfenster mit **Ctrl+Mousewheel**
 - Nicht in Fenstern mit Icons
- Word Wrap



Sonstige Editor-Tipps



The screenshot shows the Visual Studio IDE with the 'Tools' menu open. The 'Macros' option is selected, and its submenu is visible. The submenu includes options like 'Run TemporaryMacro', 'Record TemporaryMacro', 'Save TemporaryMacro', 'Cancel Recording', 'Macro Explorer', 'Macros IDE...', 'Load Macro Project...', 'Unload Macro Project', and 'New Macro Project...'. The background shows a code editor with a snippet of C# code: `nPolicy = CreationPolicy.NonSha` and `bject> mefObject1;`. A blue callout bubble in the bottom right corner contains the text: 'Makros werden im Workshop nicht im Detail behandelt. Details siehe [MSDN](#).'

Tools Architecture Test Analyze Window Help

- Attach to Process... Ctrl+Alt+P
- Connect to Database...
- Connect to Server...
- Add SharePoint Connection...
- Code Snippets Manager... Ctrl+K, Ctrl+B
- Choose Toolbox Items...
- Add-in Manager...
- Macros**
- Extension Manager...
- Create GUID
- Error Lookup
- ATL/MFC Trace Tool
- Spy±+
- Get SN Token
- WCF Service Configuration Editor
- External Tools...
- Import and Export Settings...
- Customize...
- Options...

Run TemporaryMacro Ctrl+Shift+P

Record TemporaryMacro Ctrl+Shift+R

Save TemporaryMacro

Cancel Recording

Macro Explorer Alt+F8

Macros IDE... Alt+F11

Load Macro Project...

Unload Macro Project

New Macro Project...

nPolicy = CreationPolicy.NonSha

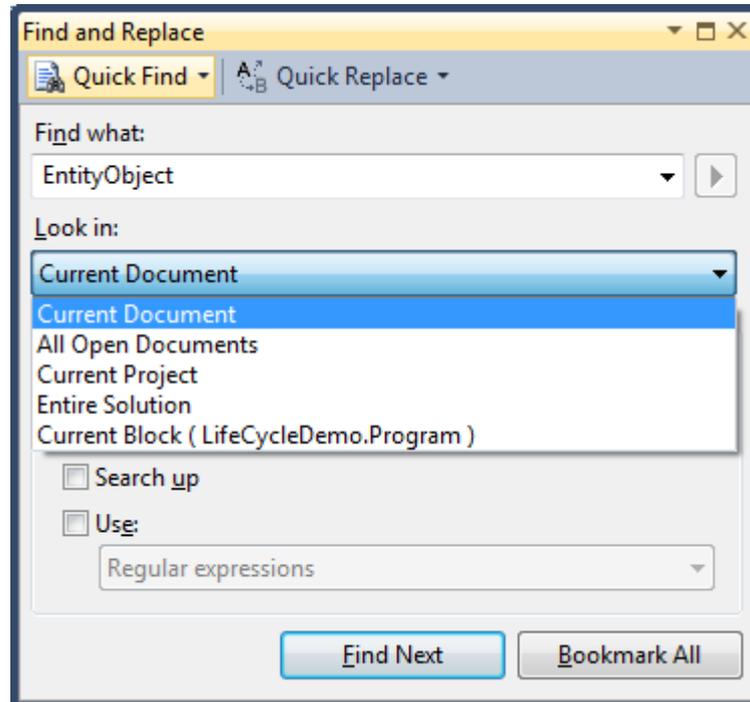
bject> mefObject1;

Sha

Makros werden im Workshop nicht im Detail behandelt. Details siehe [MSDN](#).

NAVIGIEREN

Suchen und Ersetzen (1/3)



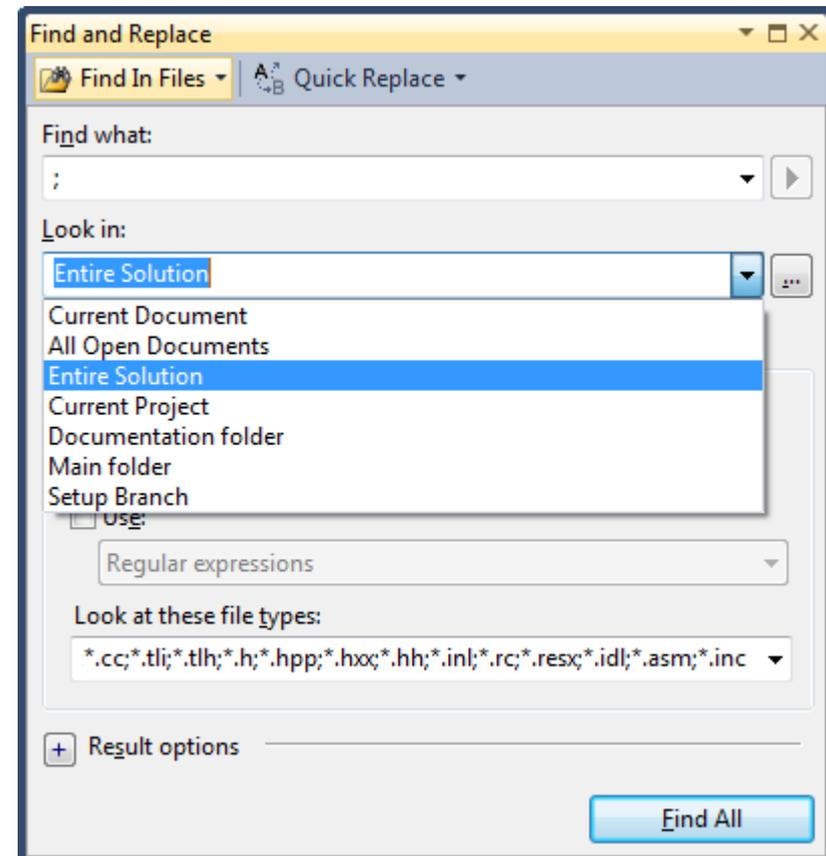
Quick Find

Ctrl+F

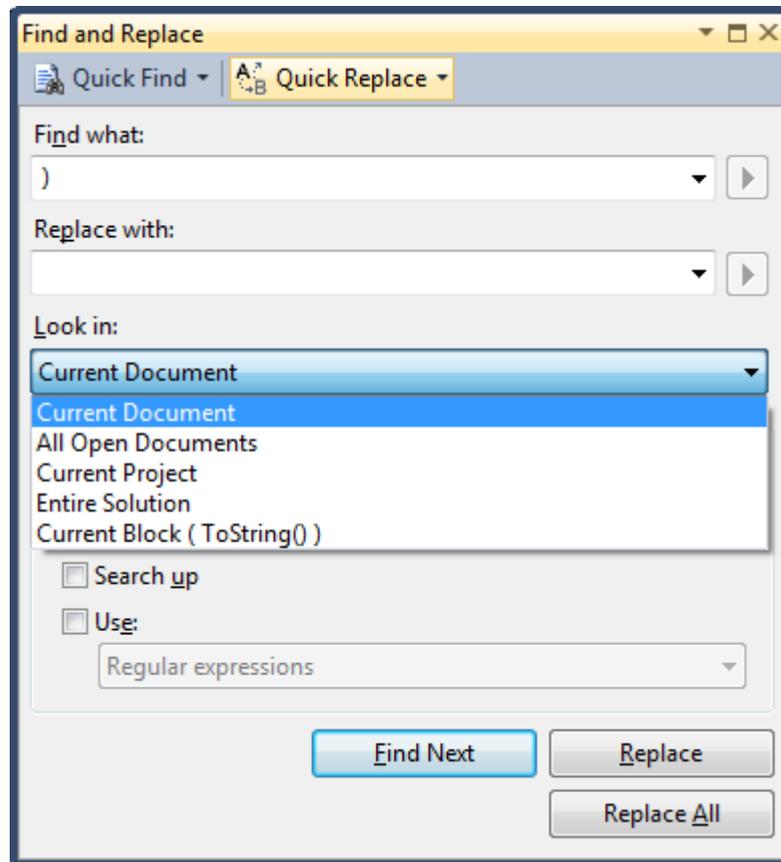


Find in Files

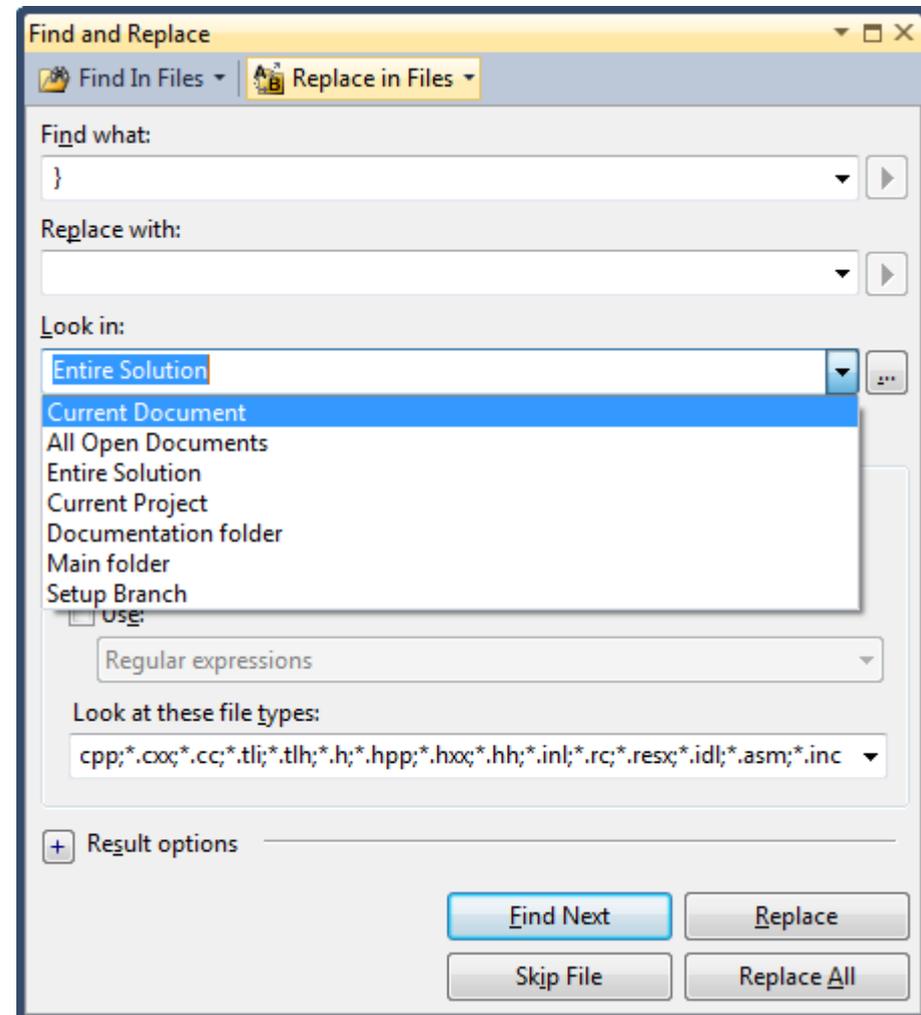
Ctrl+Shift+F



Suchen und Ersetzen (2/3)



Quick Replace
Ctrl+H



Replace in Files
Ctrl+Shift+H

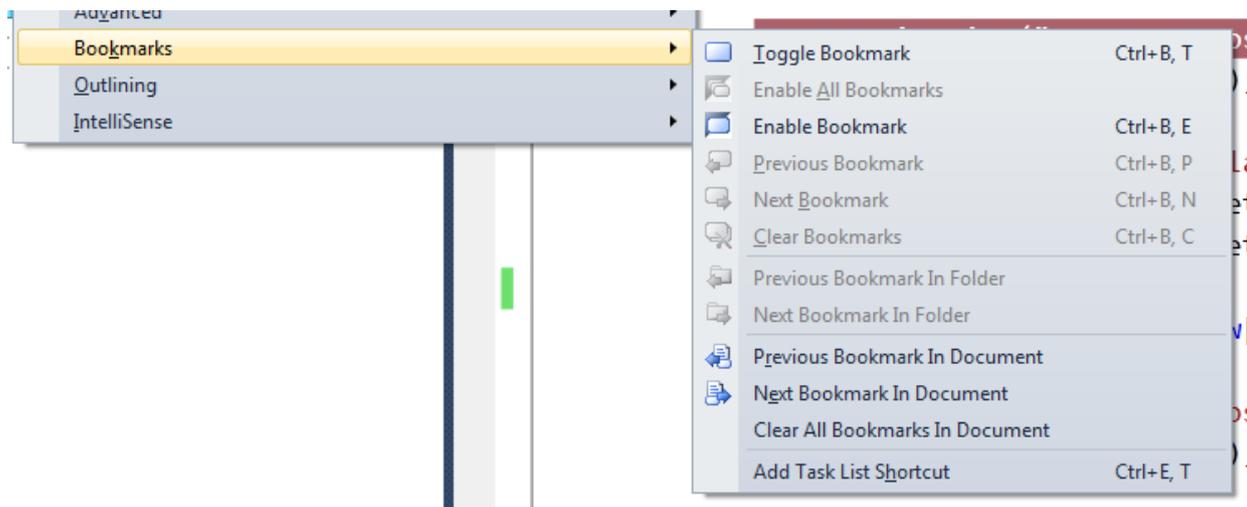


Suchen und Ersetzen (3/3)

- Wildcards (Auszug)
 - **?** → ein Zeichen
 - **#** → eine Ziffer
 - ***** → 0..n Zeichen
- Regular Expressions
 - Sehr mächtig und umfangreich
 - Details siehe [MSDN](#)
 - Tipp: [Ultrapico Espresso](#)

Weitere Suchfunktionen

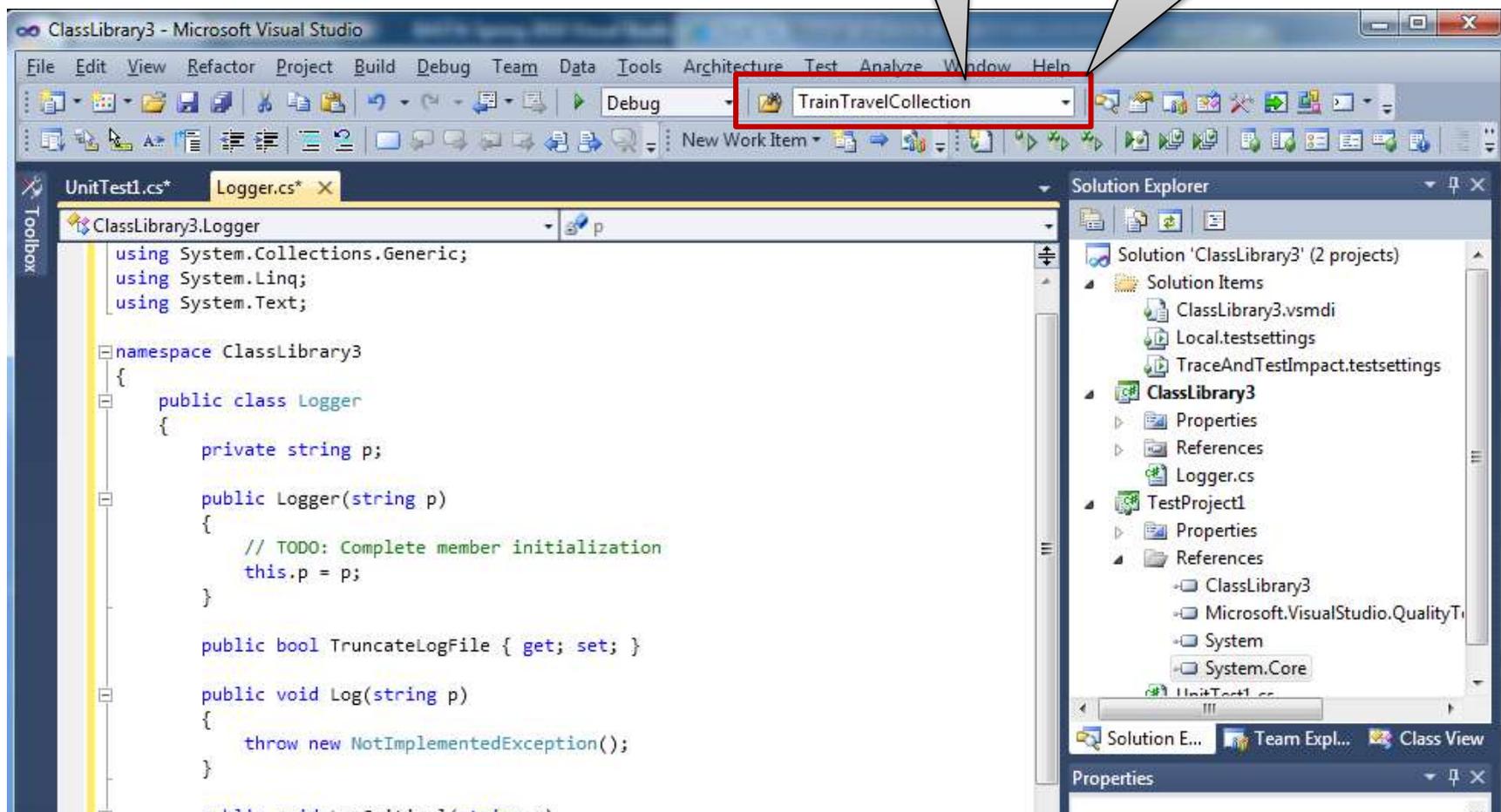
- Incremental search (**Ctrl+I**)
- Bei *Quick Find* und *Incremental Search* zum nächsten Treffer der Suche (**F3**)
- *Go to definition* (**F12**)
 - Geht auch ohne Sourcecodezugriff!
- Find/Command Box (siehe nächste Seite)
- Bookmarks



Find/Command Box (1/2)

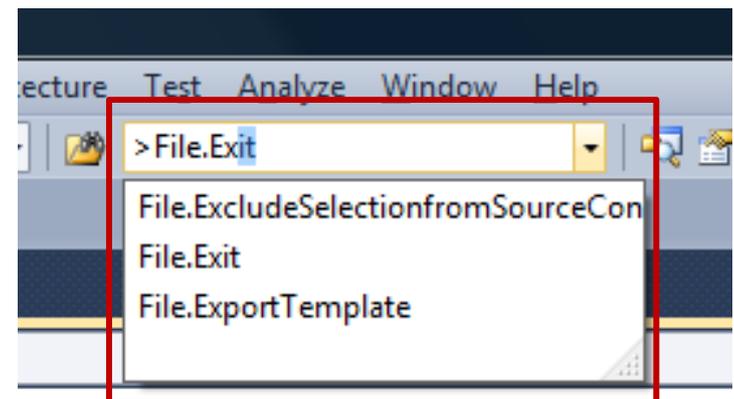
Ctrl+#

Commands mit „>“



Find/Command Box (2/2)

- Text, Enter → Suchen
- Text, F1 → Hilfe durchsuchen
- Zahl, Ctrl+G → Gehe zu Zeile
- Commands mit „>“
 - Command completion
 - Liste aller Commands
siehe [MSDN](#)



Navigate To (1/3)

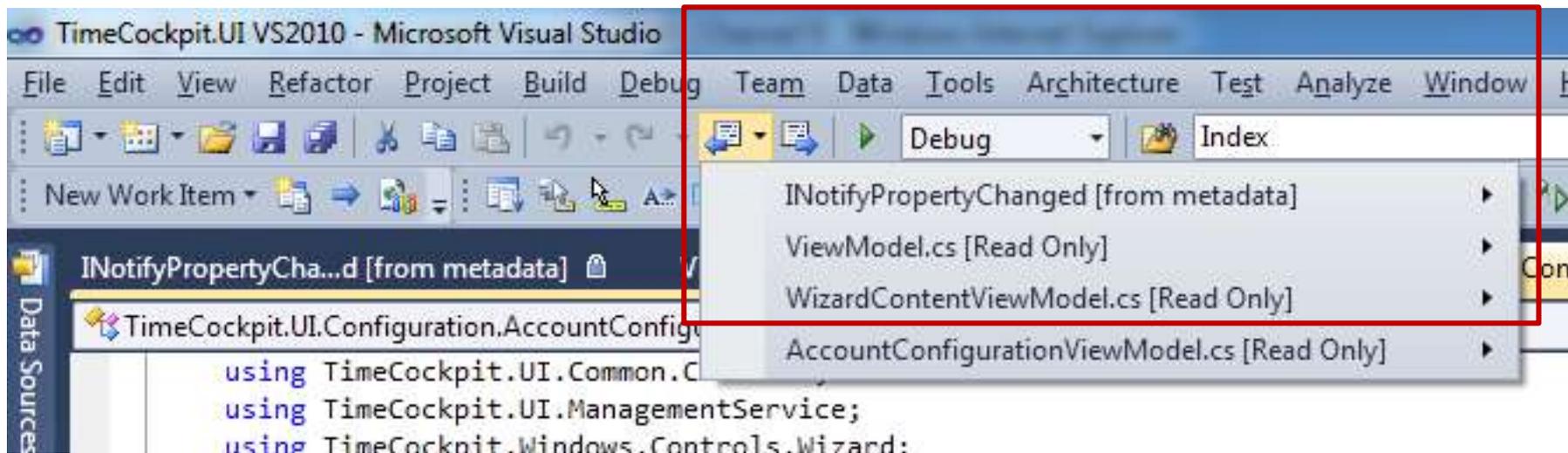
- Verbesserte Suchmöglichkeit
 - IMHO besser als *Object Browser* (Ctrl+W, J)
 - Sucht auch nach Dateinamen ☺ (z.B. DBQ findet DbClientQuery.cs)
 - CamelCaseSuche (z.B. MAN findet *MarkAsNew*)
- Edit, Navigate To (Ctrl+,)
- Tipps
 - Alles kleingeschrieben → case insensitive
 - Groß- und Kleinbuchstaben → case sensitive
 - Leertaste = And-Verknüpfung

Navigate To (2/3)

- Wann ist *Find Symbol* (**Alt+F12**) besser?
 - Search Scope kann festgelegt werden
 - Findet auch Verwendung, nicht nur Definition
 - Kann Komponenten ohne Sourcecode durchsuchen (z.B. Suche nach *File.Open*)
- Wann ist *Find* besser?
 - *Quick Find* (**Ctrl+F**) vs. *Find In Files* (**Ctrl+Shift+F**)
 - Tipp: *Quick Replace* (**Ctrl+H**) vs. *Replace In Files* (**Ctrl+Shift+H**)
 - Regular Expressions

Navigate To (3/3)

- Tipp: **F8**, um in Listen zum nächsten Element zu kommen (*go to next location*)
 - Build Errors
 - Find Results
 - Etc.
- Tipp: **Ctrl+Minus**, um zu zuletzt angesehenen Sourcecodezeile zurück zu springen (*navigate backward*)



Call Hierarchy (1/2)

- Zeigt...
 - ...Aufrufe von/in ausgewähltem Member
 - ...Implementierungen eines Interface
 - ...Implementierungen eines virtuellen oder abstrakten Members
- „*Find all references* (Ctrl+K, R) on steroids“
 - Kontextmenü auf Member, View Call History
 - Ctrl+K, T

Call Hierarchy (1/2)

- Verbesserungen gegenüber *Find all references*
 - Mehrstufig (nicht mehr ein *Find all references* nach dem anderen)
 - Scope kann eingeschränkt werden
 - Deferred execution
 - Richtigere Ergebnisse (vgl. *OnPropertyChanged*-Beispiel)
- Einschränkungen
 - Verwendung außerhalb von C# Code (z.B. XAML)

Code Definition Window

- *View, Code Definition Window* (Ctrl+W, D)
- Zeigt die Definition eines Symbols auf Grundlage von
 - Sourcecode oder
 - binären referenzierten Assemblies
- Reagiert auf
 - Cursorposition
 - Aktuelle Auswahl in *Class View*, *Object Browser* oder *Call Browser*

Reference Highlighting

- Alle Referenzen auf ein Symbol werden hervorgehoben
 - Referenzen = alles, was *Find All References* (**Ctrl+K, R**) finden würde
- **Ctrl+Shift+↓** und **Ctrl+Shift+↑** zum Springen zwischen den Referenzen
- Kann in den Optionen ein- und ausgeschaltet werden
- Farben können in den Optionen verändert werden

Task List

```
// HACK: This is a hack!
```

```
Trace.WriteLine("Accessing lazy object again");
this.mefObject1.Value.DoSomething();
this.mefObject2.Value.DoSomething();
```

```
// TODO: Add some tesing here
```

```
}
```

146 %

Task List - 2 tasks

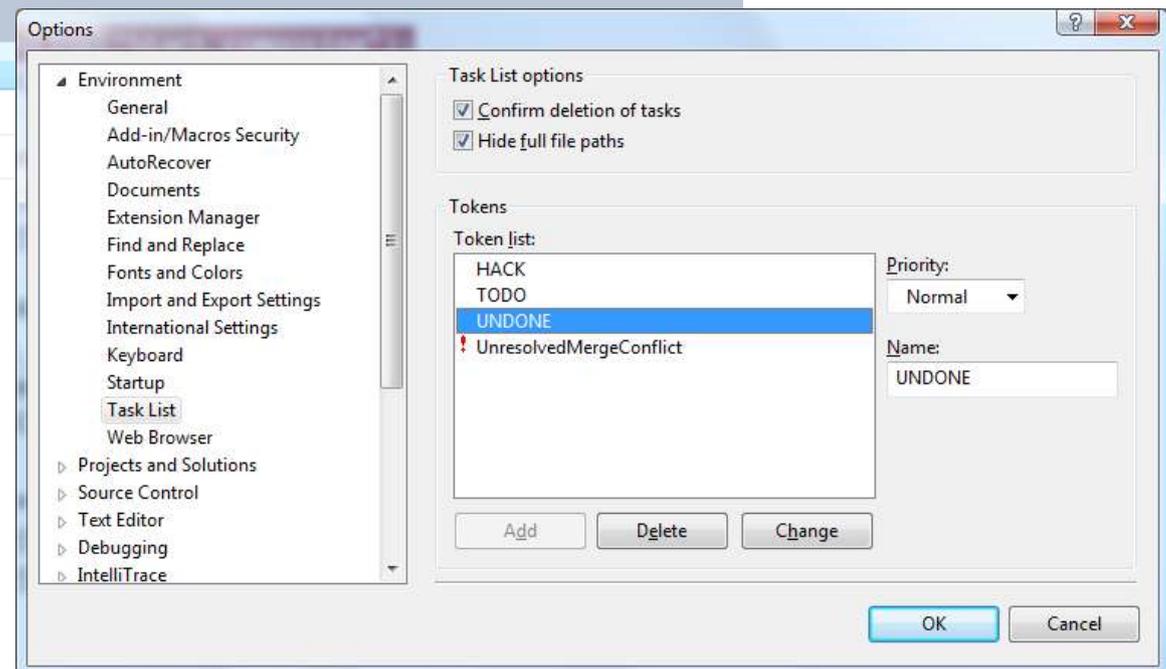
Comments

! Description

TODO: Add some tesing here

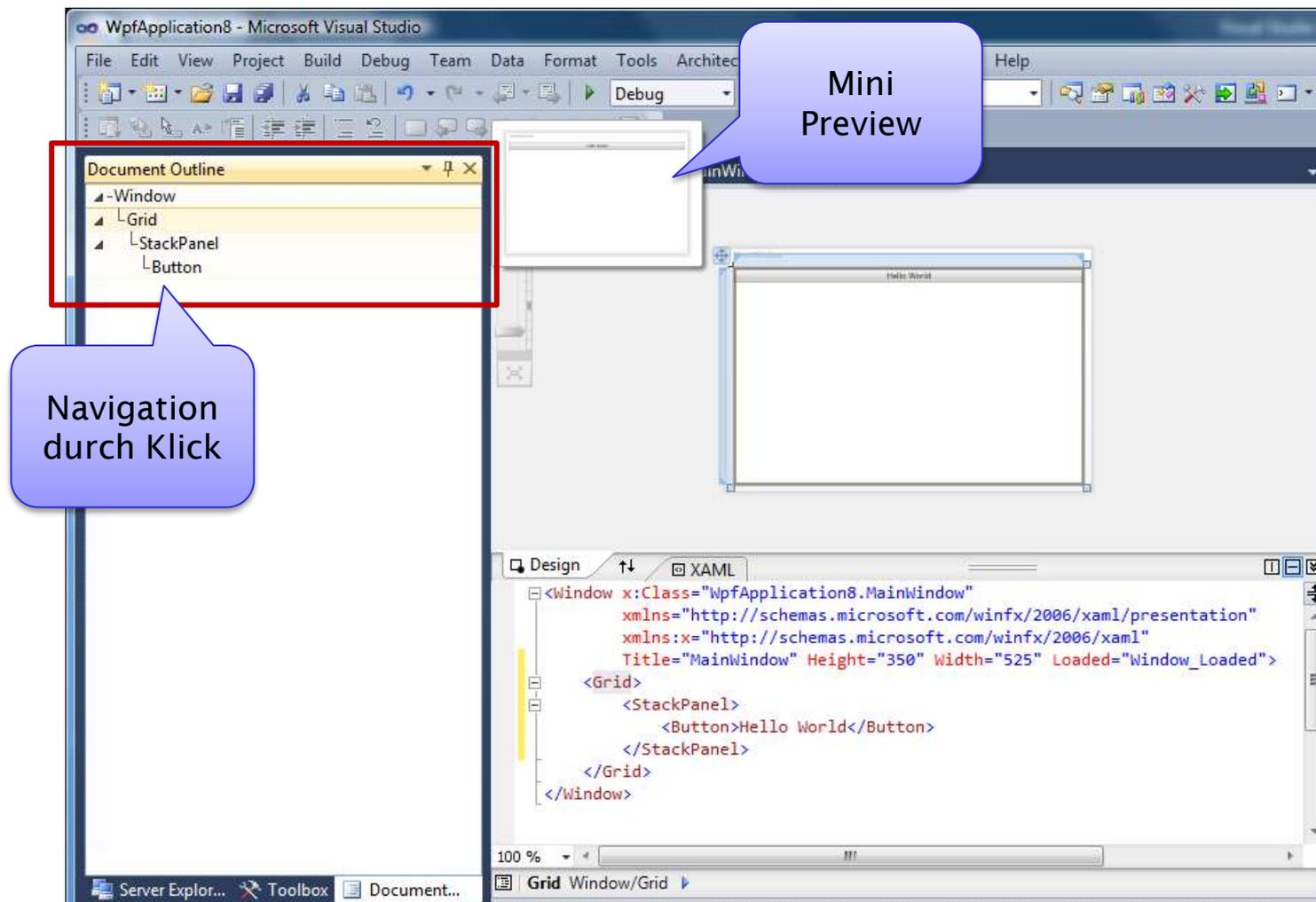
HACK: This is a hack!

Alternative:
#warning

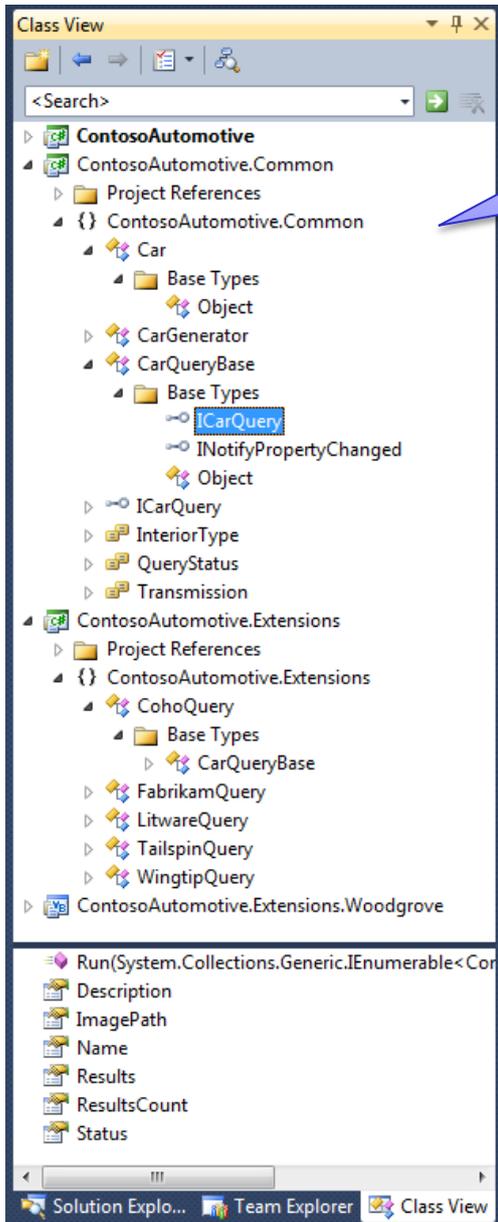


Document Outline Window

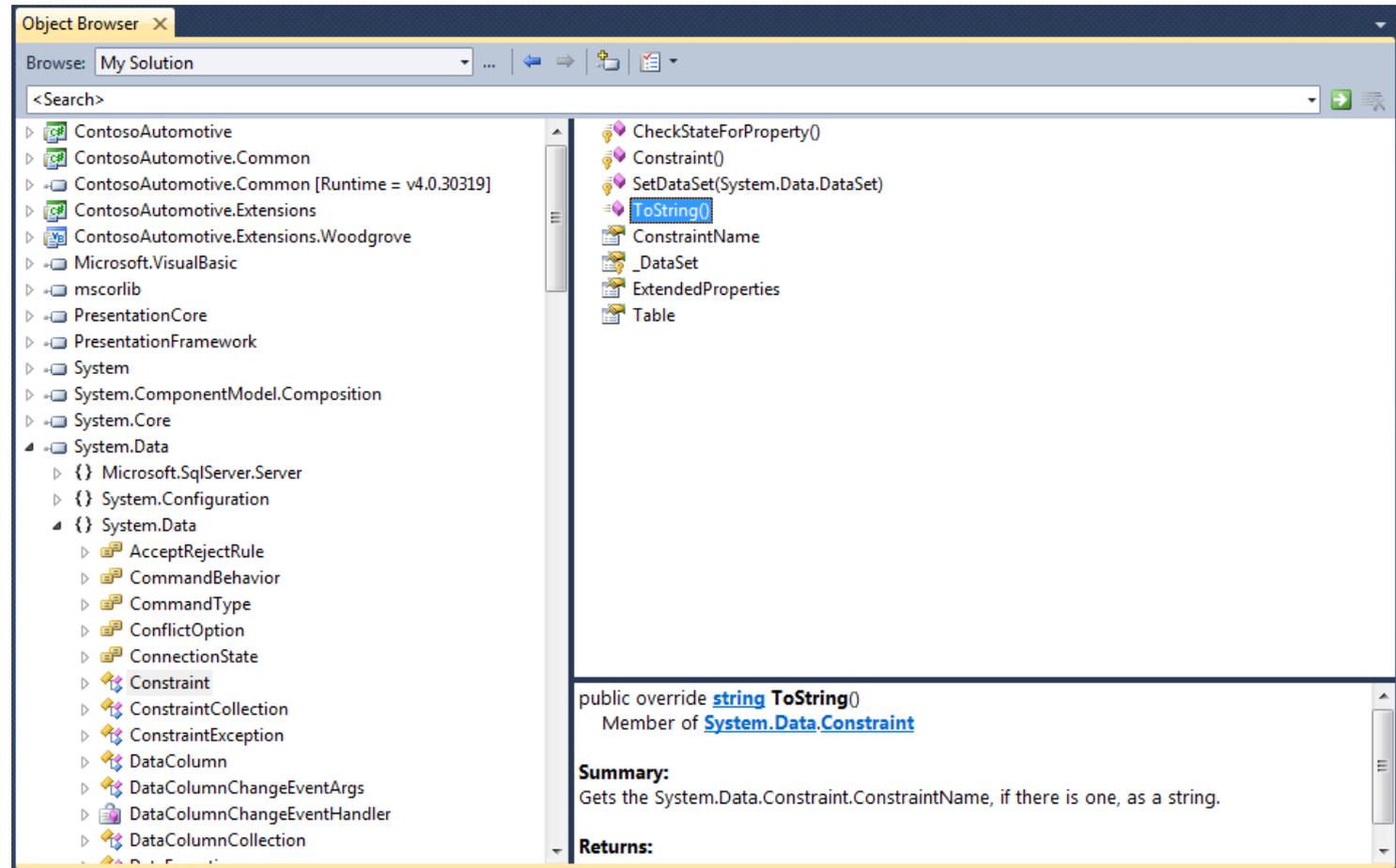
- Für UI Entwicklung



Class View und Object Browser



Tipp: Class Diagrams zum Dokumentieren



Visual Studio Editor, Navigation

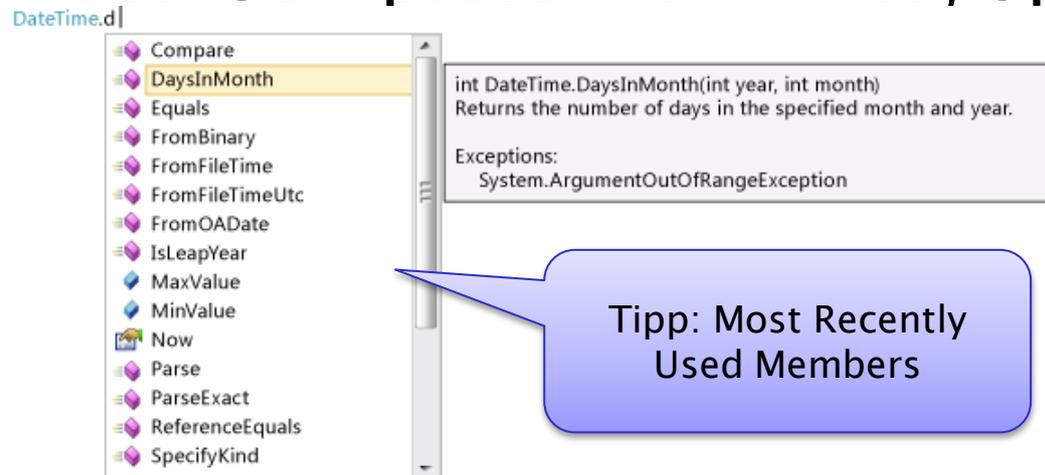
HANDS-ON LAB 1

(15 MINUTES)

CODE GENERIEREN

IntelliSense

- Suche nach Klassen, Methoden, Properties, etc. während man Code schreibt
 - Kein Wechsel zur Hilfe notwendig
- Auto-Complete wenn Tab, Space oder Punkt



- Parameter Info

Console.Write(|

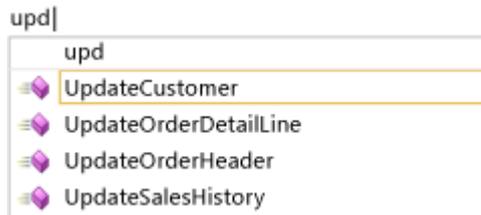
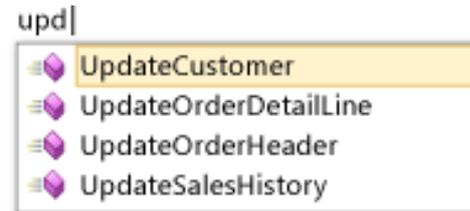
▲ 1 of 18 ▼ void Console.Write (bool value)
value: The value to write.

DateTime.Compare(|

int DateTime.Compare (DateTime t1, DateTime t2)
t1:
The first System.DateTime.

IntelliSense Mode

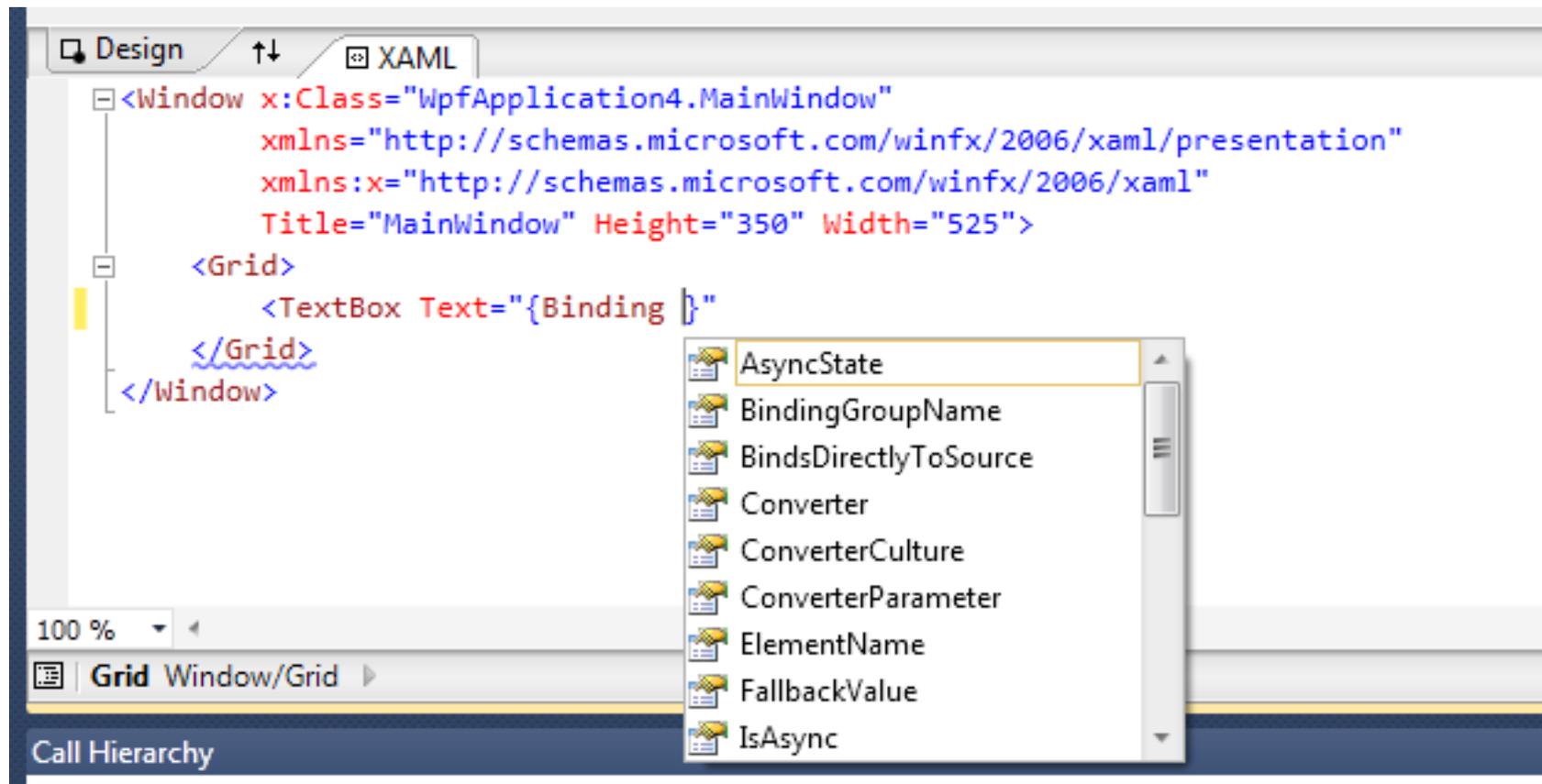
- Modi
 - Completion Mode (wie bisher)
 - Suggestion Mode (für TDD; siehe *Generate From Usage*)



- Umschalten mit **Ctrl+Alt+Space**
- BTW – Wie startet man die Member List manuell?
Ctrl+J
- BTW – Parameterinformationen blendet man mit
Ctrl+Shift+Space ein

IntelliSense in XAML...

...ist endlich da 😊 😊 😊



...und weil wir schon bei XAML sind

WPF Tree Visualizer

Tipp: <http://www.codeproject.com/KB/WPF/WoodstockForWPF.aspx>

The screenshot shows the WPF Visualizer application. On the left, the 'Visual Tree' pane displays a hierarchical tree structure of the visual elements. The root is 'MyButton', which contains a 'Border' containing an 'AdornerDecorator' containing a 'ContentPresenter' containing a 'Grid' containing a 'StackPanel' containing a 'MyButton' (highlighted in green). This 'MyButton' contains a 'Chrome' (ButtonChrome) containing a 'ContentPresenter' containing a 'MyText' (TextBlock). Below the tree is a 'Rendering of MyButton : Button' pane showing a rendered button with the text 'Hello World'.

On the right, the 'Properties of MyButton : Button' pane displays a table of properties for the selected button. The table has columns for Name, Value, Source, and Style Value.

Name	Value	Source	Style Value
AcceptsReturn	True	Default	
ActualHeight	21,96	Local	<not set>
ActualWidth	509	Local	<not set>
AllowDrop	False	Default	
AnnotationAlternates	0	Default	
AreAnyTouchesCaptured	False	Default	
AreAnyTouchesCapturedWithin	False	Default	
AreAnyTouchesDirectlyOver	False	Default	
AreAnyTouchesOver	False	Default	
Background	System.Windows	DefaultStyle	System.Windc
BetweenShowDelay	100	Default	
BindingGroup	null	Default	
BitmapEffect	null	Default	
BitmapEffectInput	null	Default	
BitmapScalingMode	Unspecified	Default	
BorderBrush	#FF707070	DefaultStyle	#FF707070
BorderThickness	1,1,1,1	DefaultStyle	1,1,1,1
CacheMode	null	Default	
Capitals	Normal	Default	
CapitalSpacing	False	Default	
CaseSensitiveForms	False	Default	
ClearTypeHint	Auto	Default	
ClickMode	Release	Default	
Clip	null	Default	
ClipToBounds	False	Default	
Command	null	Default	
CommandBindings	System.Windows	Unknown	
CommandParameter	null	Default	

Generate From Usage (1/2)

- Hilfreich bei TDD
- Erreichbar über...
 - ...Maus (Smart Tag = Pain)
 - ...**Ctrl+.** (=Pain Killer)
- Generiert Typ, Field, Property oder Methode
 - Tipp: *Generate New Type* wenn Code in einem anderen Projekt generiert werden soll (typisch bei Testprojekten)

Generate From Usage (2/2)

- `using` hinzufügen
 - Referenz muss im Projekt enthalten sein
 - Problem: Extension Methods
- Abstrakte Basisklassen implementieren
- Interfaces implementieren

Weitere IntelliSense Features

- Eventimplementierungen hinzufügen

```
button1.Click +=|
    new EventHandler(button1_Click); (Press TAB to insert)
button1.Click +=new EventHandler(button1_Click);
    Press TAB to generate handler 'button1_Click' in this class
```

- `override + Space`
 - Implementierung von overrides hinzufügen
- `using`-Statements aufräumen
 - Remove unused usings, sort usings, remove and sort
 - Tipp: Wichtig für StyleCop
- Tipp: [GhostDoc](#)
 - Generiert C# Codedokumentation basierend auf Namenskonventionen

Code Snippets

- Snippets sind vorbereitete Codestücke
 - Reduzieren des Tippaufwands für häufige Muster
 - Präsentationen
- Große Anzahl an vordefinierten Snippets
 - Details siehe [MSDN](#)
- Snippetverwaltung *Tools / Code Snippets Manager*

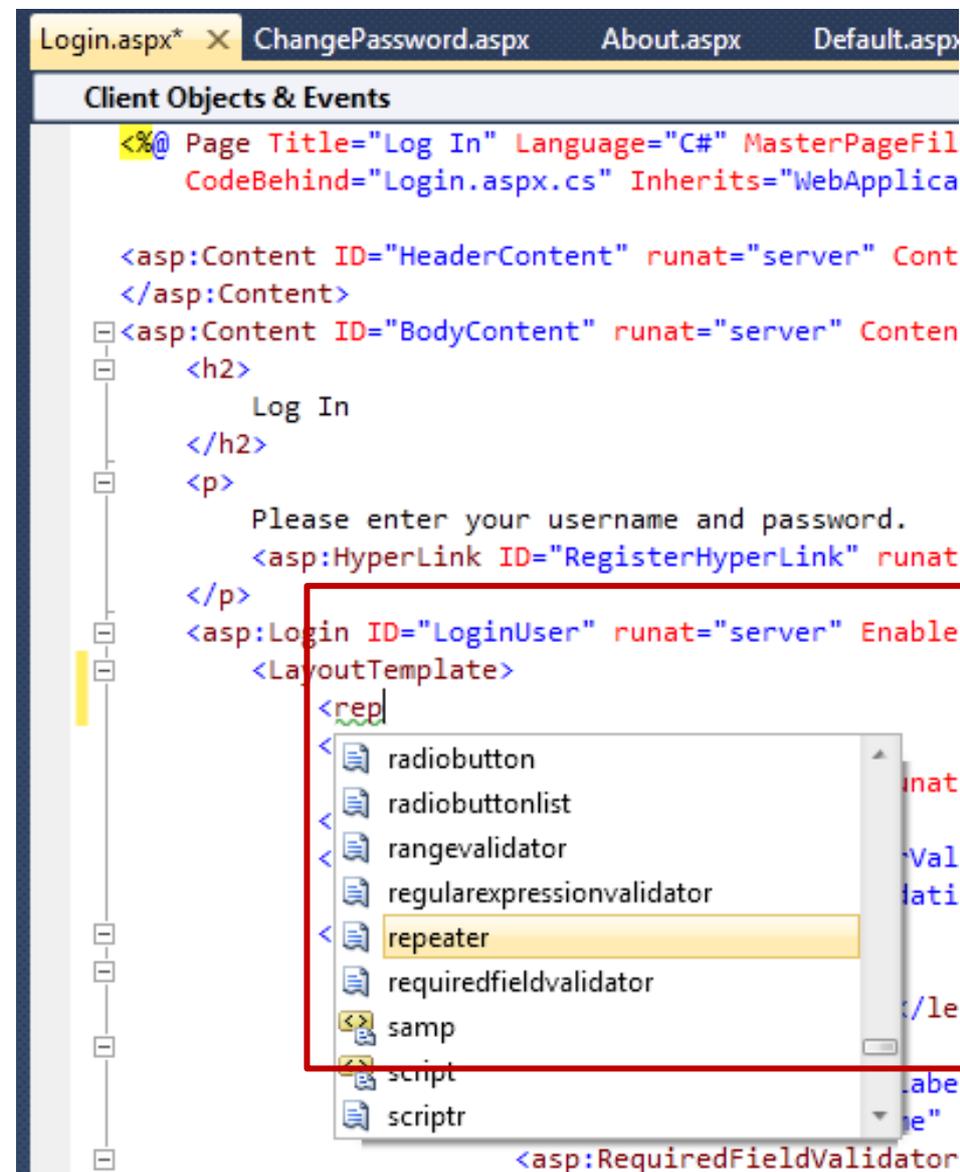
Code Snippets

- Optional
 - Parameter
 - `using` Statements
 - Referenzen
- *Code Snippet UI* (Ctrl+K, X)
- *Code Snippet Manager* (Ctrl+K, B)
- Tipp: [Snippet Designer](#) auf Codeplex

Custom Code Snippets

```
<CodeSnippets xmlns="http://schemas.microsoft.com/VisualStudio/CodeSnippet">
<CodeSnippet>
  <Header><!-- Add Header information here --></Header>
  <Snippet>
    <Declarations>
      <Literal>
        <ID>SqlConnection</ID>
        <ToolTip>Replace with a SQL connection string.</ToolTip>
        <Default>"SQL connection string"</Default>
      </Literal>
      <Object>
        <ID>SqlConnection</ID>
        <Type>System.Data.SqlClient.SqlConnection</Type>
        <ToolTip>Replace with a connection object in your application.</ToolTip>
        <Default>dcConnection</Default>
      </Object>
    </Declarations>
    <Code Language="CSharp">
      <![CDATA[
        daCustomers = new SqlConnection.SqlDataAdapter();
        selectCommand = new SqlConnection.SqlCommand($SqlConnection$);
        daCustomers.SelectCommand = selectCommand;
        daCustomers.SelectCommand.Connection = $SqlConnection$;
      ]]>
    </Code>
  </Snippet>
</CodeSnippet>
</CodeSnippets>
```

Neu: Code Snippets in ASP.NET



IntelliSense

HANDS-ON LAB 2

(10 MINUTES)

FENSTER- UND ANSICHTSVERWALTUNG

Docking (1/2)

- Document Windows
 - Im Document Frame
 - Neu: Auch außerhalb der IDE-Grenzen (auch auf eigenem Monitor)
- Tipp: **Ctrl+Doubleclick** auf Fenstertitel, um das Fenster zur letzten Position zurückzubringen

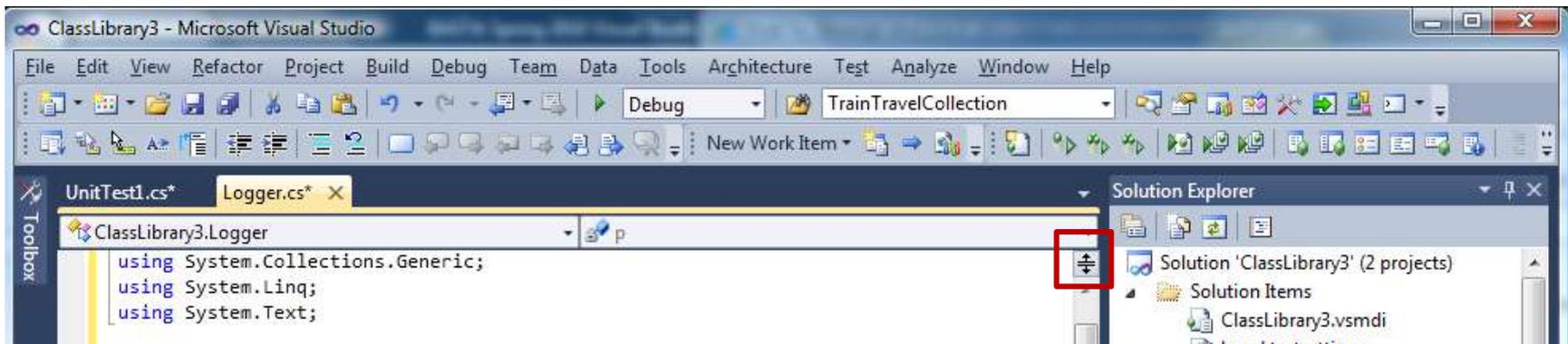
Docking (2/2)

- Tools
 - Wie bisher angedockt am IDE-Rand
 - Neu: Auch im Document Frame
 - Neu: Auch außerhalb der IDE-Grenzen (auch auf eigenem Monitor)

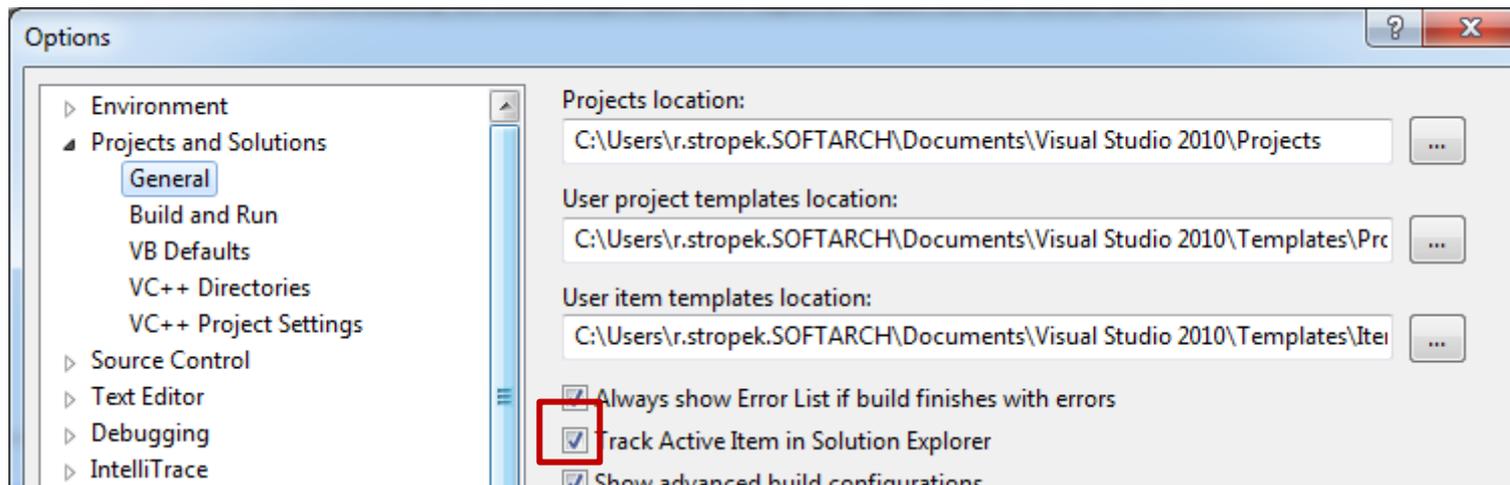


BTW – Kennen Sie den?

- *Go to open file (Ctrl+Alt+Down)*
- *Split Window*



- *Track Active Item in Solution Explorer*



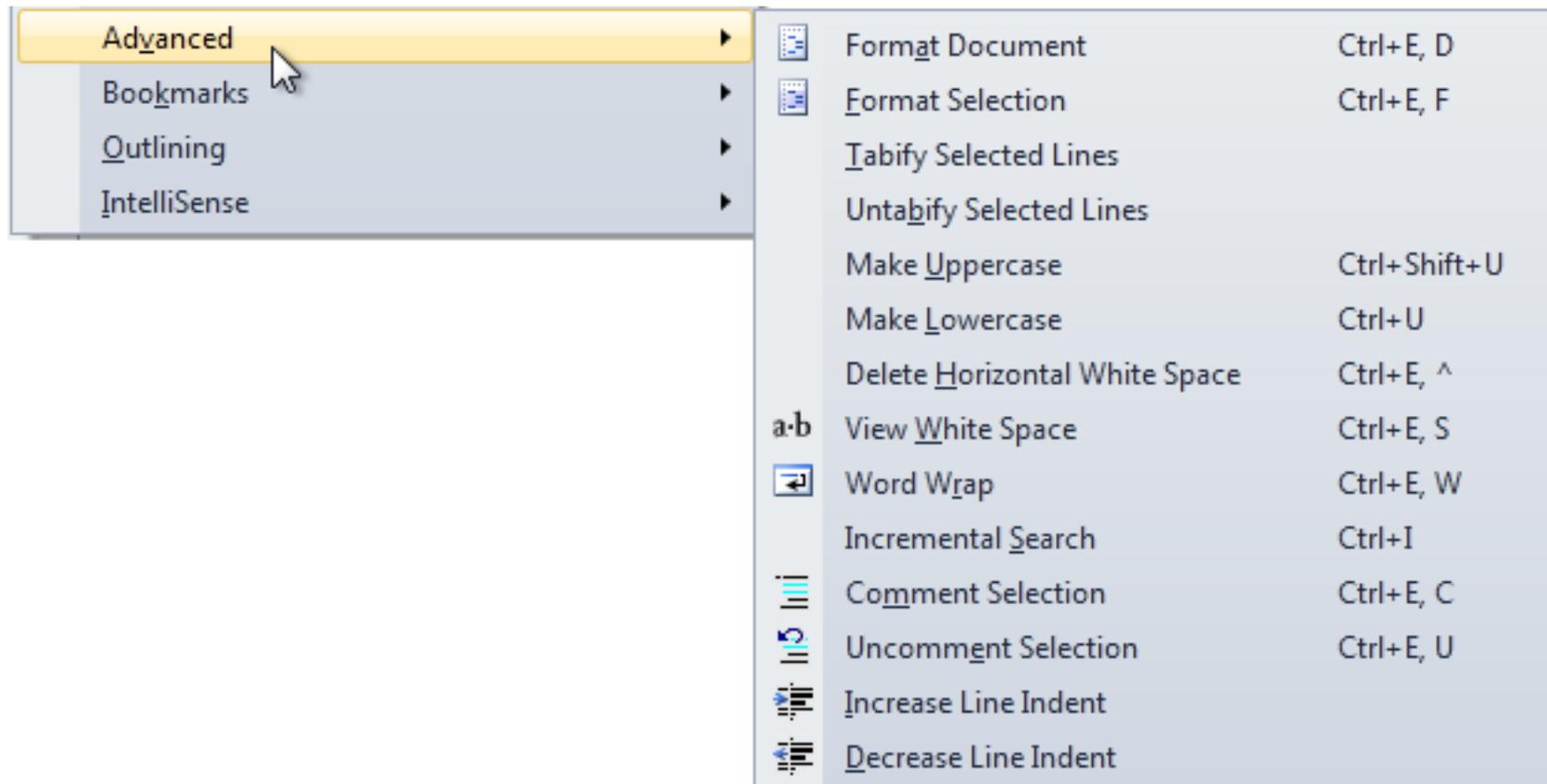
Zoom

- Zoom in Textfenster mit **Ctrl+Mousewheel**
- Nicht in Fenstern mit Icons

Selection

- Wie in früheren Version Boxed Selection mit **Alt+Click&Drag**
- Neu in VS2010
 - Multi-Line Insert
 - Paste
 - Zero-Length Boxes (multi-line insertion point)

BTW – Alles bekannt?



DEBUGGING

Data Tips (1/2)

- Wie bisher im Debugger für Variablen im aktuellen Scope
 - Tipp: Data Tip transparent machen mit **Ctrl**
- Neu:
 - *Pin to source*: Data Tip ist mit Position im Sourcecode verknüpft und scrollt mit
 - Kommentare bei *pinned data tips*

```
};
```

```
var currentTime = startTime;
```

```
foreach (var hop in route)
```

```
{
```

```
    travel.Stops.Add(new StationStop()
```

```
    {
```

```
        Station = stations.FirstOrDefault(s => s.Name == hop.StopName),
```

```
        StopTime = currentTime = currentTime + TimeSpan.FromMinutes(hop.DistanceInMinutes)
```

```
    });
```

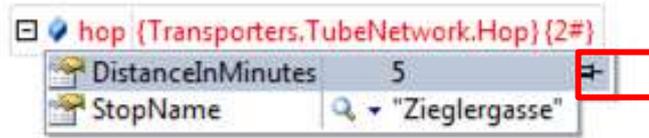
```
}
```

hop	{Transporters.TubeNetwork.Hop}
hop.DistanceInMinutes	5
hop.StopName	"Zieglergasse"

This is strange!

Data Tips (2/2)

- Pinning auch für Subexpressions möglich



- Tipp: Object-IDs

```

var currentTime = startTime;
foreach (var hop in route)
{
    travel.Stops.Add(new StationStop()
    {
        Station = stations.FirstOrDefault(s => s.Name == hop.StopName),
        StopTime = currentTime = currentTime + TimeSpan.FromMinutes(hop.DistanceInMinutes)
    });
}

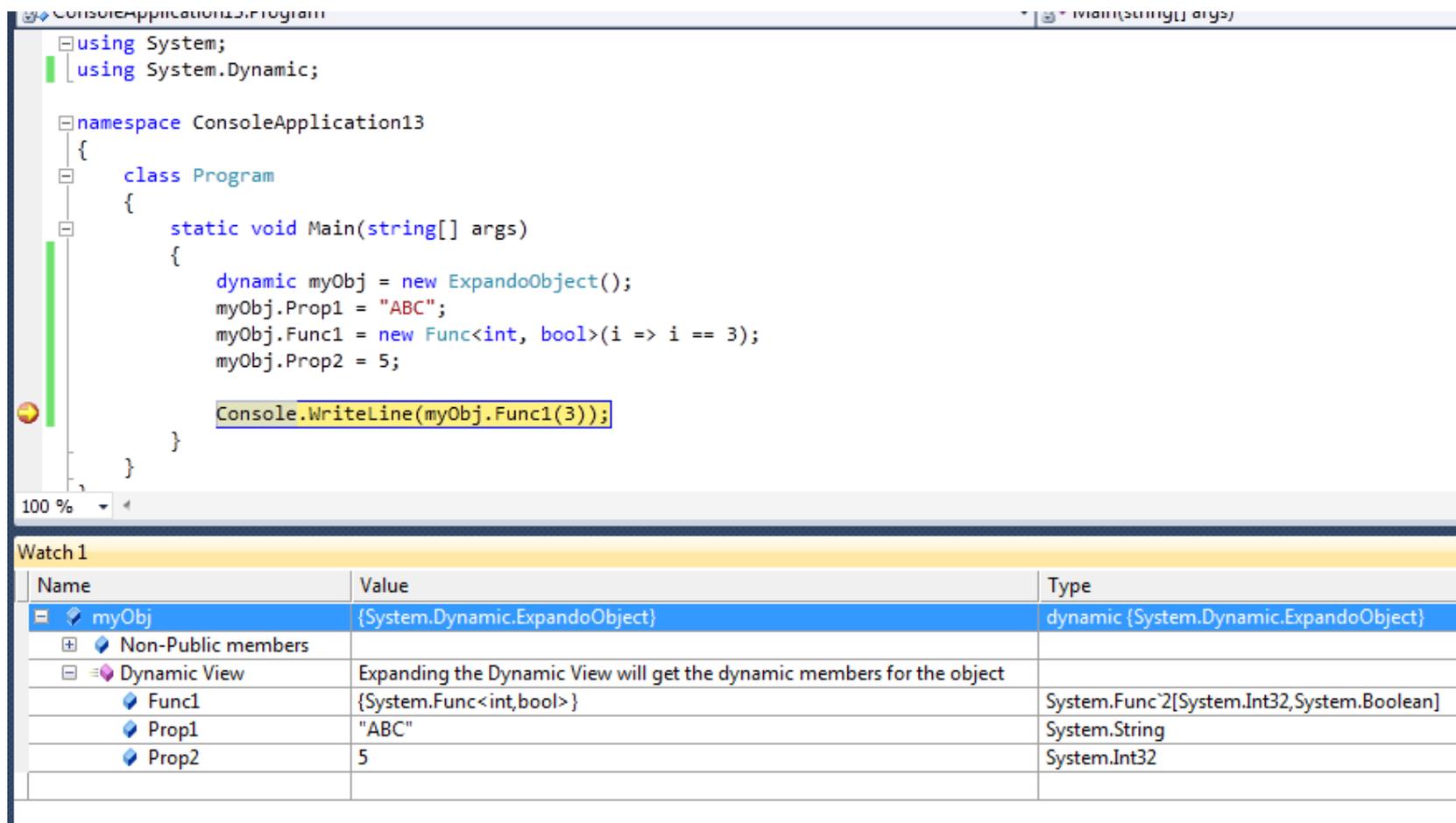
this.TrainTravels.Add(travel);

```

Name	Value	Type
route	{Transporters.TubeNetwork.Hop[4]}	System.Collections.Generic.IEnumerable<Trans
[Transporters.TubeNetwork.Hop[4]]	{Transporters.TubeNetwork.Hop[4]}	Transporters.TubeNetwork.Hop[]
[0]	{Transporters.TubeNetwork.Hop}	Transporters.TubeNetwork.Hop
[1]	{Transporters.TubeNetwork.Hop}[1#]	Transporters.TubeNetwork.Hop
[2]	{Transporters.TubeNetwork.Hop}	Transporters.TubeNetwork.Hop
[3]	{Transporters.TubeNetwork.Hop}	Transporters.TubeNetwork.Hop

Unterstützung für DLR

- `dynamic` Datentyp wird im Debugger speziell unterstützt



The screenshot displays a Visual Studio IDE window with a C# code file named `ConsoleApplication13.Program`. The code defines a `Program` class with a `Main` method. Inside `Main`, a `dynamic` variable `myObj` is created using `ExpandoObject`. It is then populated with properties `Prop1` (value "ABC"), `Prop2` (value 5), and a method `Func1` (a lambda function `i => i == 3`). The `Func1` method is called with the argument 3, and the result is printed to the console.

```
using System;
using System.Dynamic;

namespace ConsoleApplication13
{
    class Program
    {
        static void Main(string[] args)
        {
            dynamic myObj = new ExpandoObject();
            myObj.Prop1 = "ABC";
            myObj.Func1 = new Func<int, bool>(i => i == 3);
            myObj.Prop2 = 5;

            Console.WriteLine(myObj.Func1(3));
        }
    }
}
```

Below the code editor, the Watch window shows the state of `myObj`. The table below summarizes the data shown in the Watch window:

Name	Value	Type
<code>myObj</code>	<code>{System.Dynamic.ExpandoObject}</code>	<code>dynamic {System.Dynamic.ExpandoObject}</code>
Non-Public members		
Dynamic View		
<code>Func1</code>	<code>{System.Func<int,bool>}</code>	<code>System.Func`2[System.Int32,System.Boolean]</code>
<code>Prop1</code>	"ABC"	<code>System.String</code>
<code>Prop2</code>	5	<code>System.Int32</code>

IntelliTrace (1/2)

The screenshot shows the Visual Studio IDE with the following components:

- Code Editor:** Displays the C# code for `MainWindow.xaml.cs`. The code includes a `public partial class MainWindow : Window` with a `MyButton_Click` method that calls `Debugger.Break();`.
- IntelliTrace Window:** Shows a list of events. The selected event is "Debugger: Breakpoint Hit: .ctor, MainWindow.xaml.c", with details: "A breakpoint was hit by the debugger. Thread: Main Thread [5808]. Related views: Locals, Call Stack".
- Locals Window:** Shows the current state of variables. The selected variable is "Breakpoint Hit" with value `this = {WpfApplication5.MainWindow}` and type `WpfApp`.
- Call Stack Window:** Shows the current call stack. The top frame is `WpfApplication5.exe!WpfApplication5.MainWindow..ctor()`.

Callouts point to the following elements:

- Events:** Points to the IntelliTrace event list.
- Locals:** Points to the Locals window.
- Call Stack:** Points to the Call Stack window.

IntelliTrace (2/2)

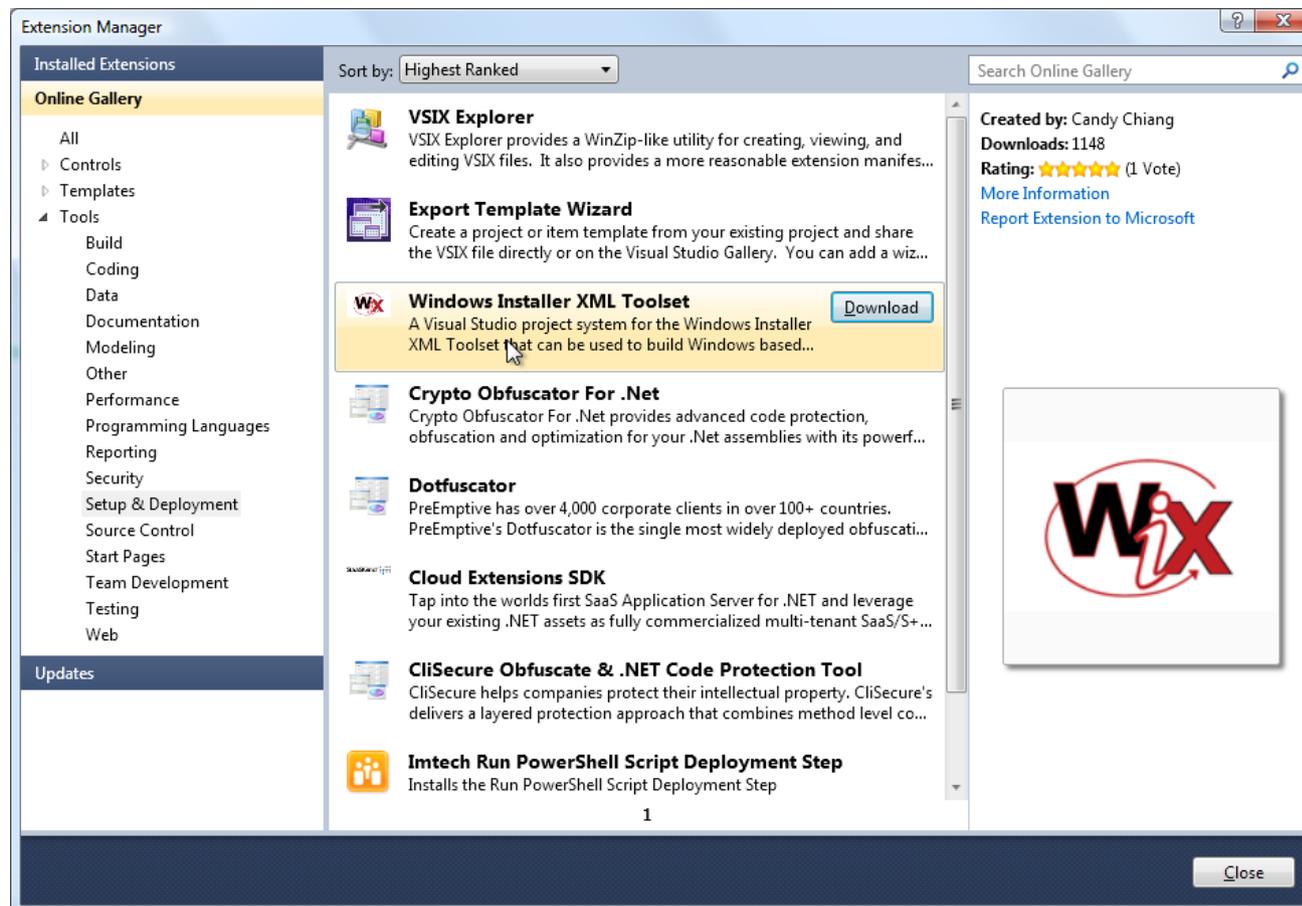
- Aufzeichnen von *Events*
 - Definierbar in *Tools / Options / IntelliTrace*
- Optional auch *Call Informations*
 - Verbraucht mehr Ressourcen
 - Ein/Ausschalten in *Tools / Options / IntelliTrace*

TOOLS

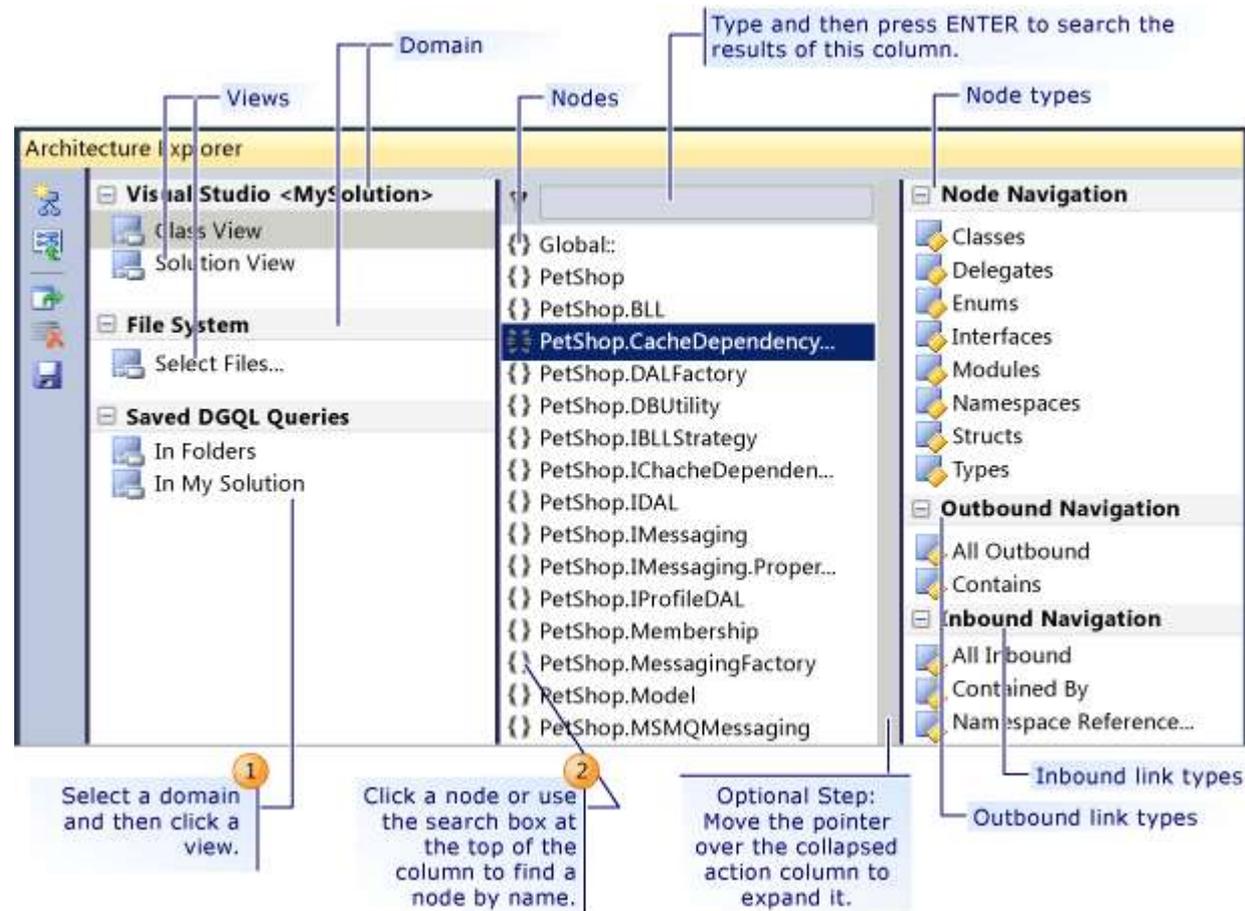
Extension Manager (1/2)

Tools, Extension Manager

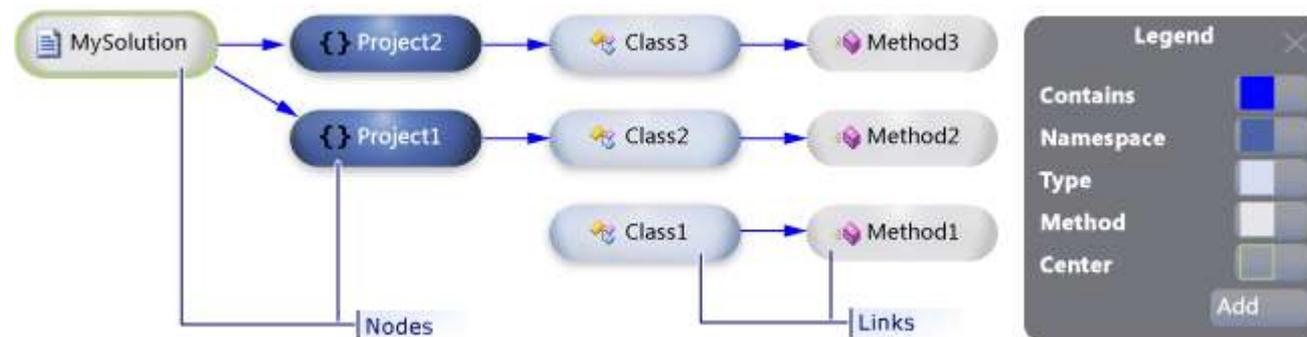
[\(http://visualstudiogallery.msdn.microsoft.com/en-us/\)](http://visualstudiogallery.msdn.microsoft.com/en-us/)



Architecture Explorer

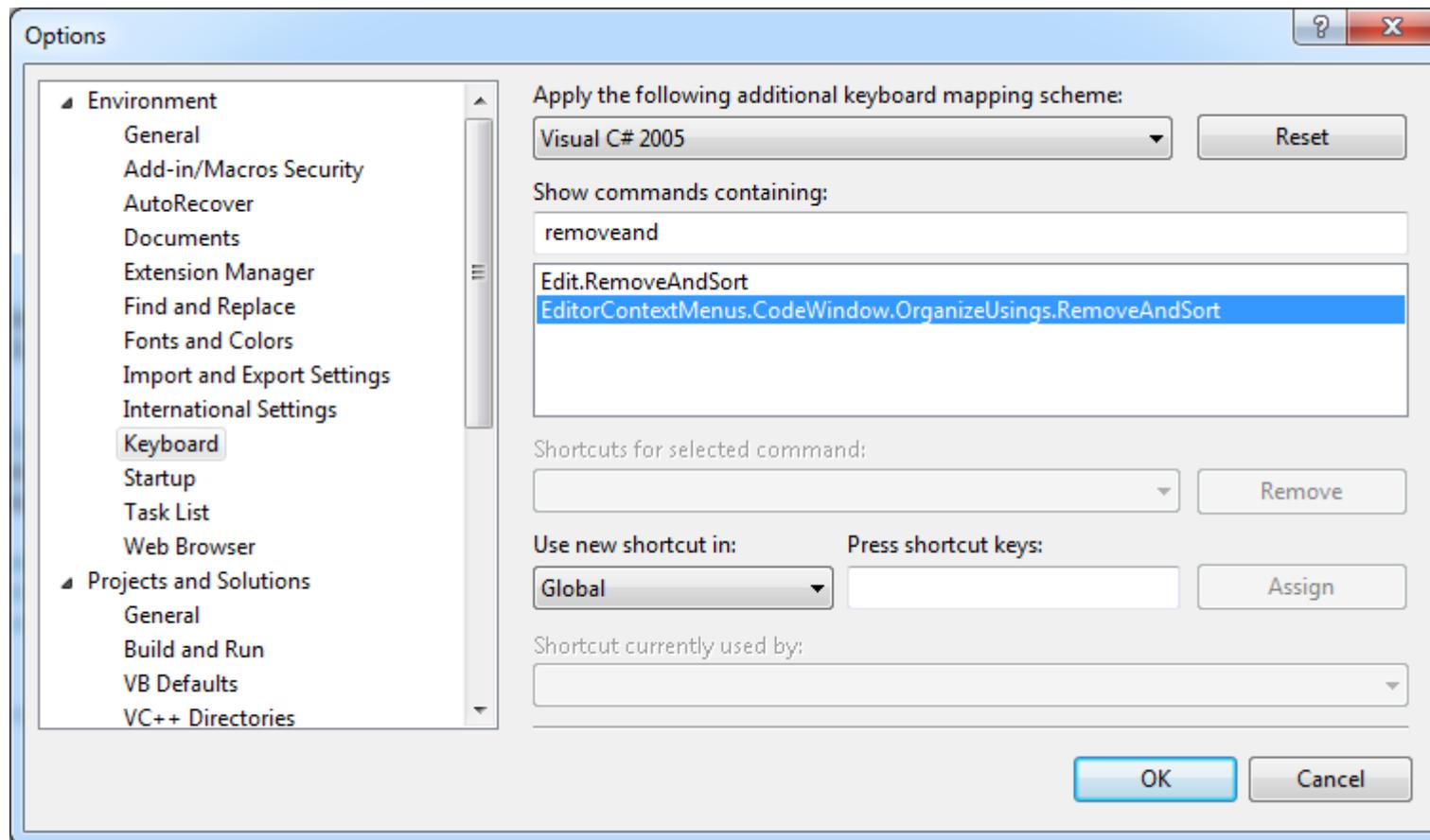


Dependency Graphs



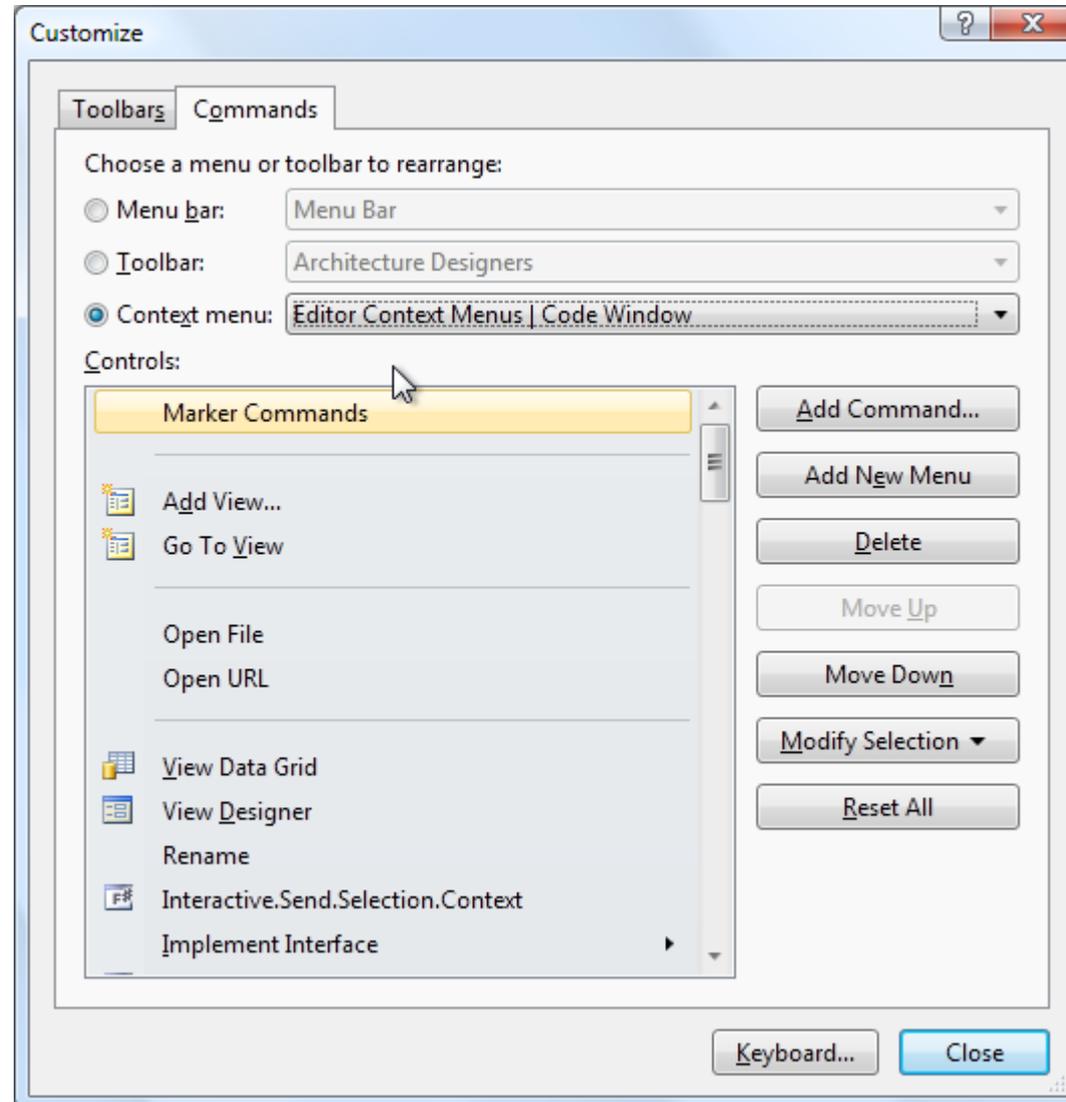
OLDIES BUT GOLDIES

Customize Shortcuts



Beispiel: *Organize Usings*

Customize Context Menu



Read more about help, find the right tools

RESOURCES

Tool Reference

- [Sandcastle](#)
 - Documentation Compiler for Managed Class Libraries
- [GhostDoc](#)
 - Generates documentation based on naming conventions
- [StyleCop](#)
 - Analyzes C# source code to enforce a set of style and consistency rules
- [Sandcastle Help File Builder](#)
 - Provides graphical and command line based tools to build a help file in an automated fashion