5.– 8. September 2011
in Nürnberg

Herbstcampus

Wissenstransfer
par excellence

Weben statt kleben

Einführung in die Aspektorientierte Programmierung mit PostSharp

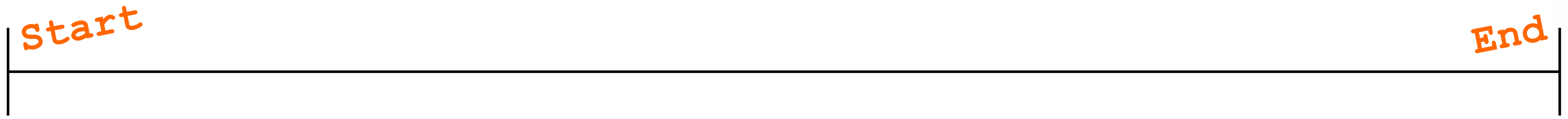
Bernd Hengelein
Siemens AG

Bernd Hengelein

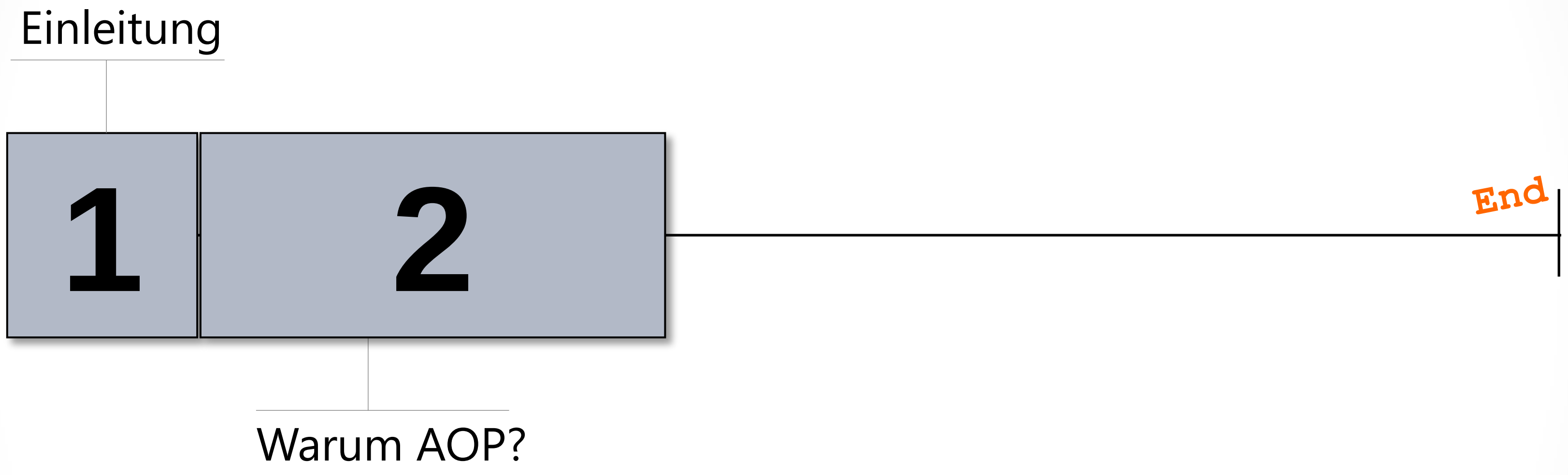
Software Engineer/Architect bei
Siemens Healthcare MR

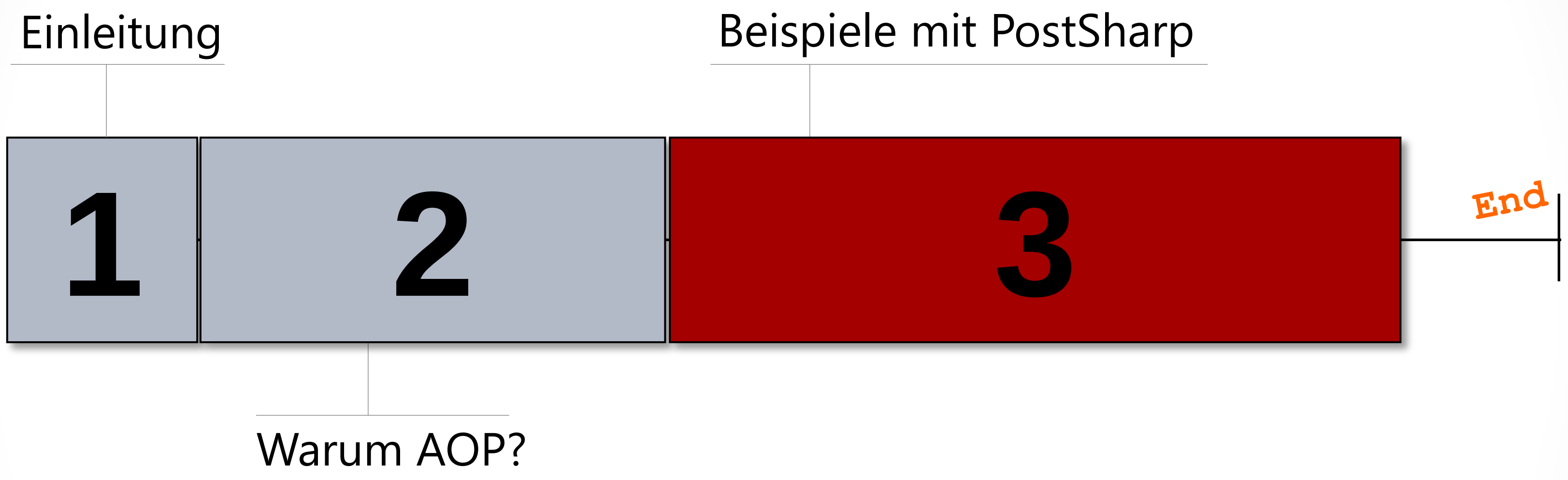
Co-Lead der .NET User Group Franken

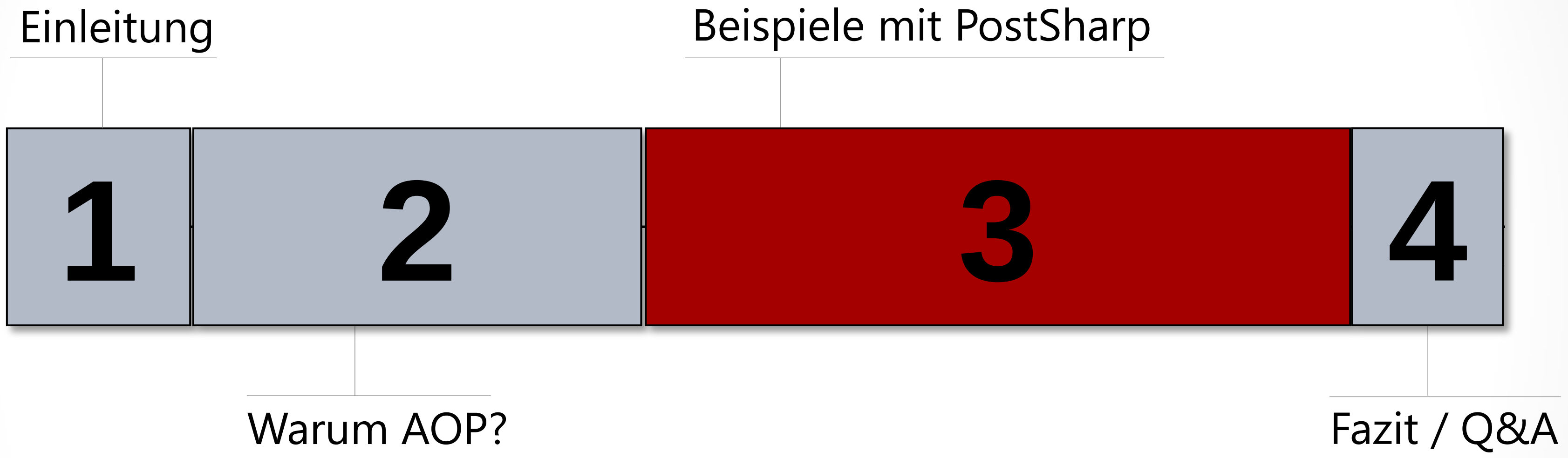
http://berndhengelein.de
http://www.dotnet-day-franken.de/
http://dodnedder.de/


Start
End

Einleitung
1
End

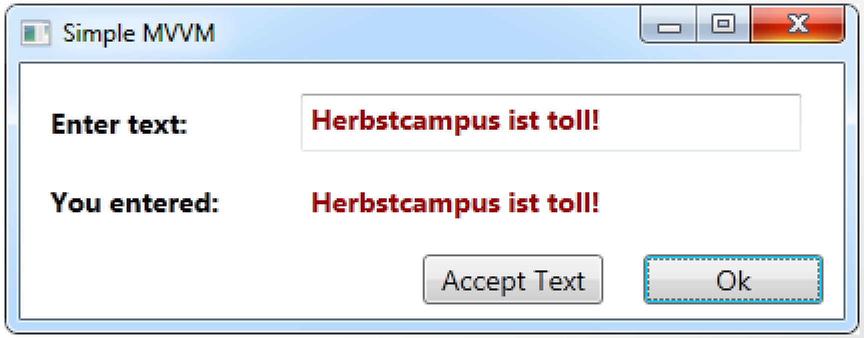

Einleitung
1
2
End
Warum AOP?


Einleitung
Beispiele mit PostSharp
1
2
3
End
Warum AOP?


Einleitung
Beispiele mit PostSharp
1
2
3
4
Warum AOP?
Fazit / Q&A

```
public class MainViewModel : INotifyPropertyChanged
{
    private ICommand _acceptTextCommand;
    private ICommand _okCommand;
    private string _enteredText;

    public string EnteredText
    {
        get { return _enteredText; }
        set
        {
            _enteredText = value;
            RaisePropertyChanged("EnteredText");
        }
    }

    public ICommand AcceptTextCommand
    {
        get
        {
            if (_acceptTextCommand == null)
            {
                _acceptTextCommand =
                    new DelegateCommand(OnAcceptText);
            }
            return _acceptTextCommand;
        }
    }

    public ICommand OkCommand
    {
        get
        {
            if (_okCommand == null)
            {
                _okCommand =
                    new DelegateCommand(OnOk);
            }
            return _okCommand;
        }
    }
```

```
private void OnAcceptText()
{
    // do sth.
}
private void OnOk()
{
    // do sth
}
public event PropertyChangedEventHandler
             PropertyChanged;


private void RaisePropertyChanged(string propertyName)
{
    PropertyChangedEventHandler propertyChanged =
            PropertyChanged;
    if (propertyChanged != null)
    {
        propertyChanged(this,
            new PropertyChangedEventArgs(propertyName));
    }
}
}
```

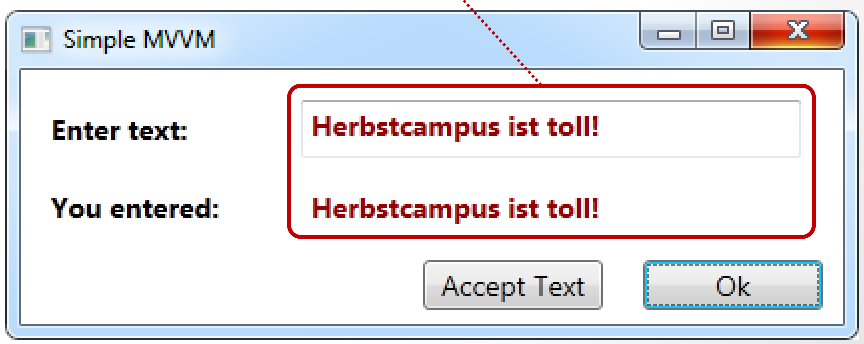

Simple MVVM

Enter text: Herbstcampus ist toll!

You entered: Herbstcampus ist toll!

```
public class MainViewModel : INotifyPropertyChanged
{
    private ICommand _acceptTextCommand;
    private ICommand _okCommand;
    private string _enteredText;

    public string EnteredText
    {
        get { return _enteredText; }
        set
        {
            _enteredText = value;
            RaisePropertyChanged("EnteredText");
        }
    }

    public ICommand AcceptTextCommand
    {
        get
        {
            if (_acceptTextCommand == null)
            {
                _acceptTextCommand =
                    new DelegateCommand(OnAcceptText);
            }
            return _acceptTextCommand;
        }
    }

    public ICommand OkCommand
    {
        get
        {
            if (_okCommand == null)
            {
                _okCommand =
                    new DelegateCommand(OnOk);
            }
            return _okCommand;
        }
    }
```

```
private void OnAcceptText()
{
    // do sth.
}
private void OnOk()
{
    // do sth
}
public event PropertyChangedEventHandler
             PropertyChanged;

private void RaisePropertyChanged(string propertyName)
{
    PropertyChangedEventHandler propertyChanged =
           PropertyChanged;
    if (propertyChanged != null)
    {
        propertyChanged(this,
            new PropertyChangedEventArgs(propertyName));
    }
}
}
```

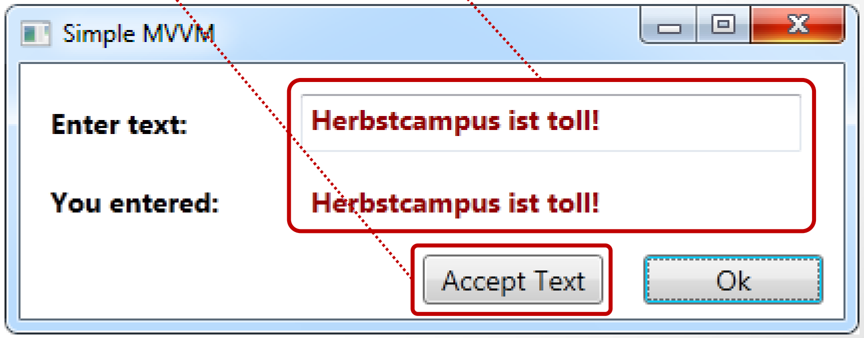


Herbstcampus ist toll!

Herbstcampus ist toll!

```
public class MainViewModel : INotifyPropertyChanged
{
    private ICommand _acceptTextCommand;
    private ICommand _okCommand;
    private string _enteredText;

    public string EnteredText
    {
        get { return _enteredText; }
        set
        {
            _enteredText = value;
            RaisePropertyChanged("EnteredText");
        }
    }

    public ICommand AcceptTextCommand
    {
        get
        {
            if (_acceptTextCommand == null)
            {
                _acceptTextCommand =
                    new DelegateCommand(OnAcceptText);
            }
            return _acceptTextCommand;
        }
    }

    public ICommand OkCommand
    {
        get
        {
            if (_okCommand == null)
            {
                _okCommand =
                    new DelegateCommand(OnOk);
            }
            return _okCommand;
        }
    }
```

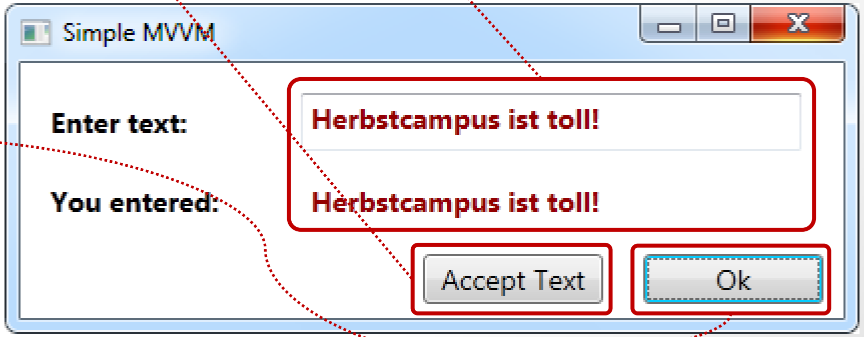
```
private void OnAcceptText()
{
    // do sth.
}
private void OnOk()
{
    // do sth
}
public event PropertyChangedEventHandler
             PropertyChanged;


private void RaisePropertyChanged(string propertyName)
{
    PropertyChangedEventHandler propertyChanged =
           PropertyChanged;
    if (propertyChanged != null)
    {
        propertyChanged(this,
            new PropertyChangedEventArgs(propertyName));
    }
}
}
```



Enter text: Herbstcampus ist toll!

You entered: Herbstcampus ist toll!

```
public class MainViewModel : INotifyPropertyChanged
{
    private ICommand _acceptTextCommand;
    private ICommand _okCommand;
    private string _enteredText;

    public string EnteredText
    {
        get { return _enteredText; }
        set
        {
            _enteredText = value;
            RaisePropertyChanged("EnteredText");
        }
    }

    public ICommand AcceptTextCommand
    {
        get
        {
            if (_acceptTextCommand == null)
            {
                _acceptTextCommand =
                    new DelegateCommand(OnAcceptText);
            }
            return _acceptTextCommand;
        }
    }

    public ICommand OkCommand
    {
        get
        {
            if (_okCommand == null)
            {
                _okCommand =
                    new DelegateCommand(OnOk);
            }
            return _okCommand;
        }
    }

private void OnAcceptText()
{
    // do sth.
}
private void OnOk()
{
    // do sth
}
public event PropertyChangedEventHandler
             PropertyChanged;

private void RaisePropertyChanged(string propertyName)
{
    PropertyChangedEventHandler propertyChanged =
            PropertyChanged;
    if (propertyChanged != null)
    {
        propertyChanged(this,
            new PropertyChangedEventArgs(propertyName));
    }
}
}
```



```
public class MainViewModel : INotifyPropertyChanged
{
    private ICommand _acceptTextCommand;
    private ICommand _okCommand;
    private string _enteredText;

    public string EnteredText
    {
        get { return _enteredText; }
        set
        {
            _enteredText = value;
            RaisePropertyChanged("EnteredText");
        }
    }
```

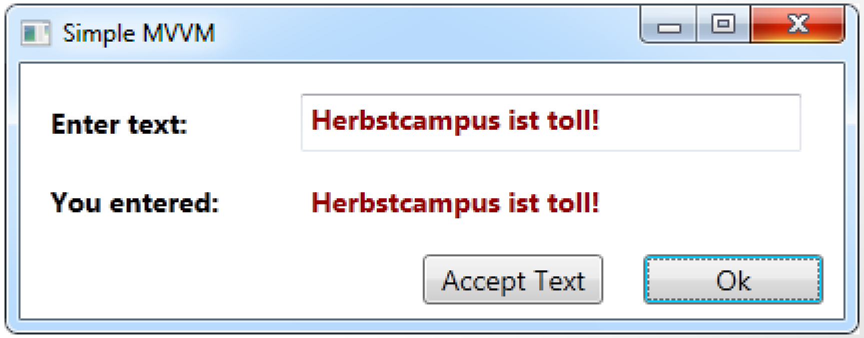
PropertyChanged Event feuern

```
    public ICommand AcceptTextCommand
    {
        get
        {
            if (_acceptTextCommand == null)
            {
                _acceptTextCommand
                    new Delega                  t);
            }
            return _ac
        }
    }

    pub

              (_okCommand == null)
            {
                _okCommand =
                    new DelegateCommand(OnOk);
            }
            return _okCommand;
        }
    }
```

Commands für Databinding erzeugen

```
private void OnAcceptText()
{
    // do sth.
}
private void OnOk()
{
    // do sth
}
public event PropertyChangedEventHandler
             PropertyChanged;

private void RaisePropertyChanged(string prope    Name)
{
    PropertyChangedEventHandl
           PropertyCha
    if (prope
    {
                        s,
                opertyChangedEventArgs(propertyName));
    }
}
}
```

INotifyPropertyChanged Implementierung



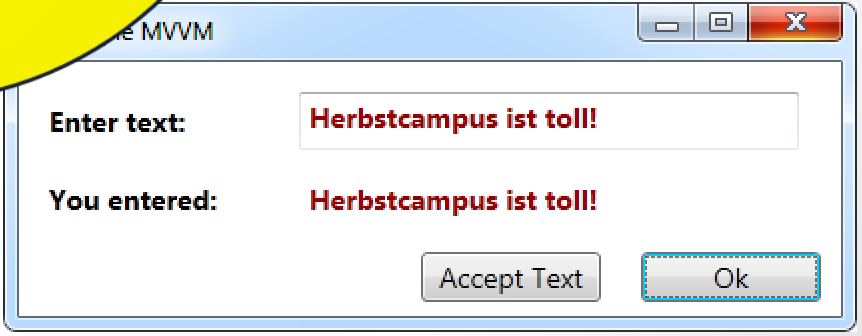
Enter text: Herbstcampus ist toll!

You entered: Herbstcampus ist toll!

HELP

```
public class MainViewModel : INotifyPropertyChanged
{
    private ICommand _acceptTextCommand;
    private ICommand _okCommand;
    private string _enteredText;

    public string EnteredText
    {
        get { return _enteredText; }
        set
        {
            _enteredText = value;
            RaisePropertyChanged("EnteredText");
        }
    }

    public ICommand AcceptTextCommand
    {
        get
        {
            if (_acceptTextCommand
            {
                _acceptTextComman
                    new Delegate
            }
            return _acceptTextCom
        }
    }

    public ICommand OkCommand
    {
        get
        {
            if (_okCommand == null)
            {
                _okCommand =
                    new DelegateCommand(OnOk);
            }
            return _okCommand;
        }
    }
```

```
private void OnAcceptText()
{
    // do sth.
}
private void OnOk()
{
    // do sth
}
public event PropertyChangedEventHandler
             PropertyChanged;

         sePropertyChanged(string propertyName)

         EventHandler propertyChanged =
         hanged;
         ged != null)

         ged(this,
         ertyChangedEventArgs(propertyName));
```



Enter text: Herbstcampus ist toll!

You entered: Herbstcampus ist toll!

```
private void OnAcceptText()
{







                // do sth.











}
```

```csharp
private void OnAcceptText()
{
    Trace.TraceInformation("Entering OnAcceptText");







                // do sth.










    Trace.TraceInformation("Leaving OnAcceptText");
}
```

```
private void OnAcceptText()
{
    Trace.TraceInformation("Entering OnAcceptText");

    ThreadPool.QueueUserWorkItem(
        delegate
        {



                // do sth.











        });
    Trace.TraceInformation("Leaving OnAcceptText");
}
```

```
private void OnAcceptText()
{
    Trace.TraceInformation("Entering OnAcceptText");

    ThreadPool.QueueUserWorkItem(
        delegate
        {


                // do sth.
                _dialogService.ShowDialog("Demodialog", "Press ok to continue.");







        });
    Trace.TraceInformation("Leaving OnAcceptText");
}
```

```csharp
private void OnAcceptText()
{
    Trace.TraceInformation("Entering OnAcceptText");
    var dispatcher = Dispatcher.CurrentDispatcher;
    ThreadPool.QueueUserWorkItem(
        delegate
        {


                // do sth.
                dispatcher.BeginInvoke(
                    new Action(() => _dialogService.ShowDialog("Demodialog",
                                                                "Press ok to continue.")));






        });
    Trace.TraceInformation("Leaving OnAcceptText");
}
```

```csharp
private void OnAcceptText()
{
    Trace.TraceInformation("Entering OnAcceptText");
    var dispatcher = Dispatcher.CurrentDispatcher;
    ThreadPool.QueueUserWorkItem(
        delegate
        {
            try
            {
                // do sth.
                dispatcher.BeginInvoke(
                    new Action(() => _dialogService.ShowDialog("Demodialog",
                                                                "Press ok to continue.")));
            }
            catch (Exception e)
            {
                // make sure to do proper exception handling, e.g. inform calling thread
                Trace.TraceError("Exception caught!");
            }
        });
    Trace.TraceInformation("Leaving OnAcceptText");
}
```

Wo liegt das **Problem**?

```csharp
private void OnAcceptText()
{
    Trace.TraceInformation("Entering OnAcceptText");
    var dispatcher = Dispatcher.CurrentDispatcher;
    ThreadPool.QueueUserWorkItem(
        delegate
        {
            try
            {
                // do sth.
                dispatcher.BeginInvoke(
                    new Action(() => _dialogService.ShowDialog("Demodialog",
                                                           "Press ok.")));
            }
            catch (Exception e)
            {
                // make sure to do proper exception handling, e.g. inform calling thread
                Trace.TraceError("Exception caught!");
            }
        });
    Trace.TraceInformation("Leaving OnAcceptText");
}
```

Cross-Cutting concerns

Tracing	Exception Handling	Data Binding
Security	Threading	…

Cross-Cutting concerns in Aspekte auslagern

TracingAspect

RunsOnUIThreadAspect

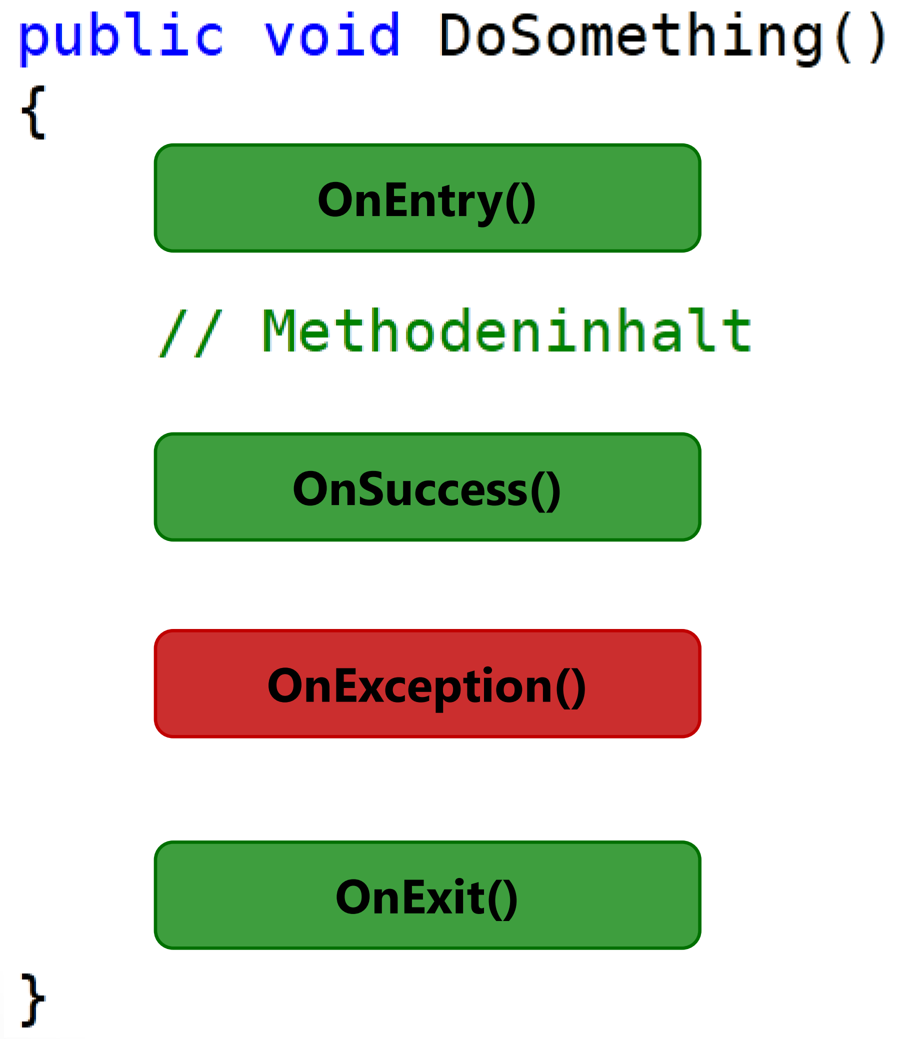
RunAsyncAspect

HandleExceptionAspect

Zeig‘ mir den Code…

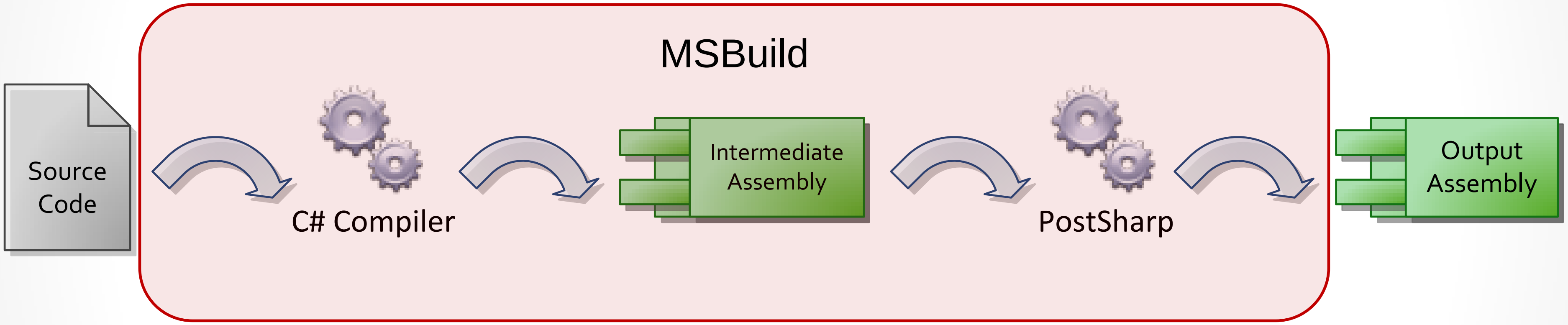
Demo

OnMethodBoundaryAspect



```
public void DoSomething()
{
        OnEntry()

    // Methodeninhalt

        OnSuccess()

        OnException()

        OnExit()
}
```

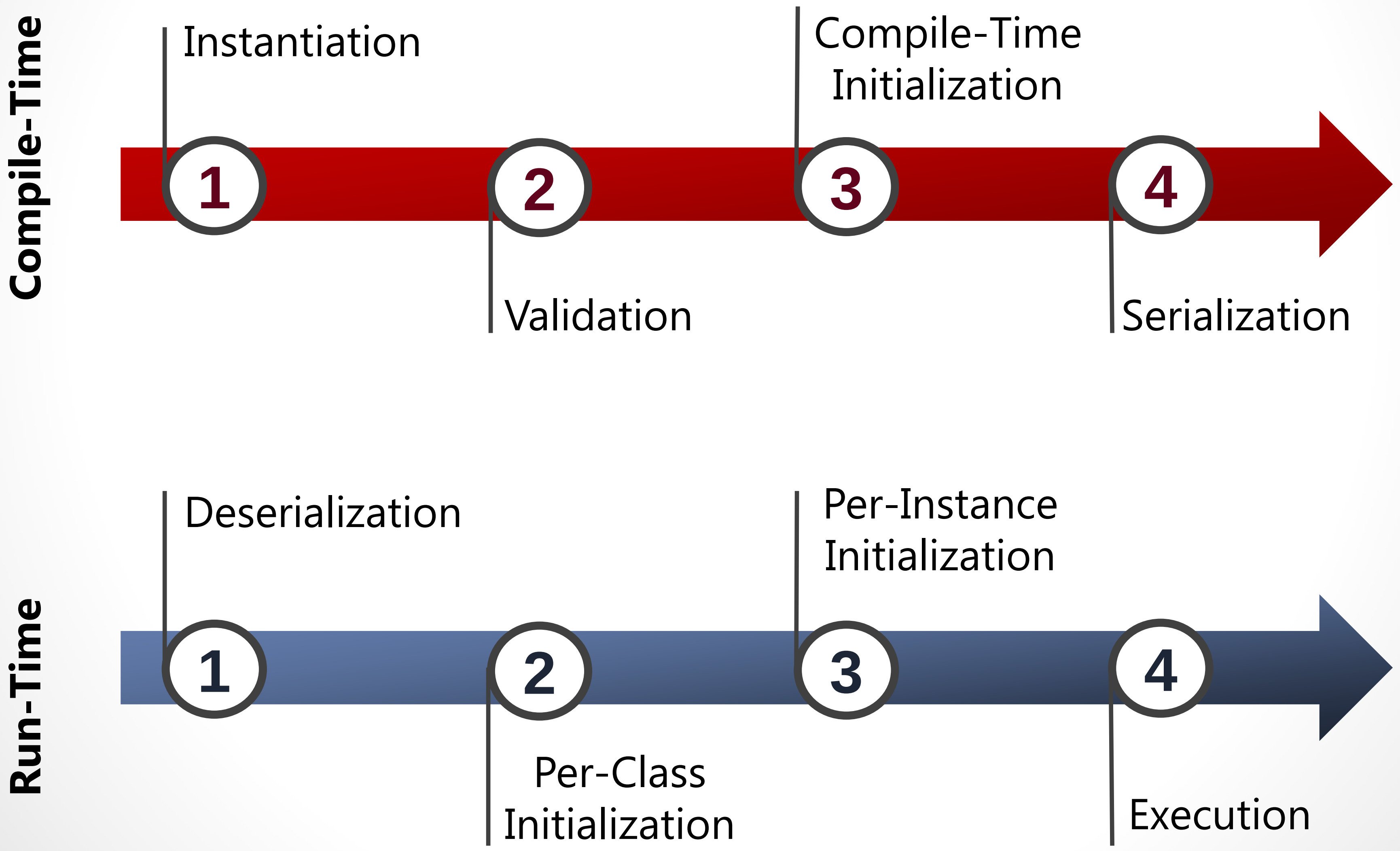
Wie arbeitet PostSharp?



Multicasting

Demo

Lebenszeit von Aspekten



Trace Aspekt verbessern

Demo

Welche Möglichkeiten bietet PostSharp?

Veränderungen

- Methoden einfassen
- Methoden unterbrechen (intercept)
- Eigenschaften unterbrechen
- Felder unterbrechen
- Events unterbrechen

Welche Möglichkeiten bietet PostSharp?

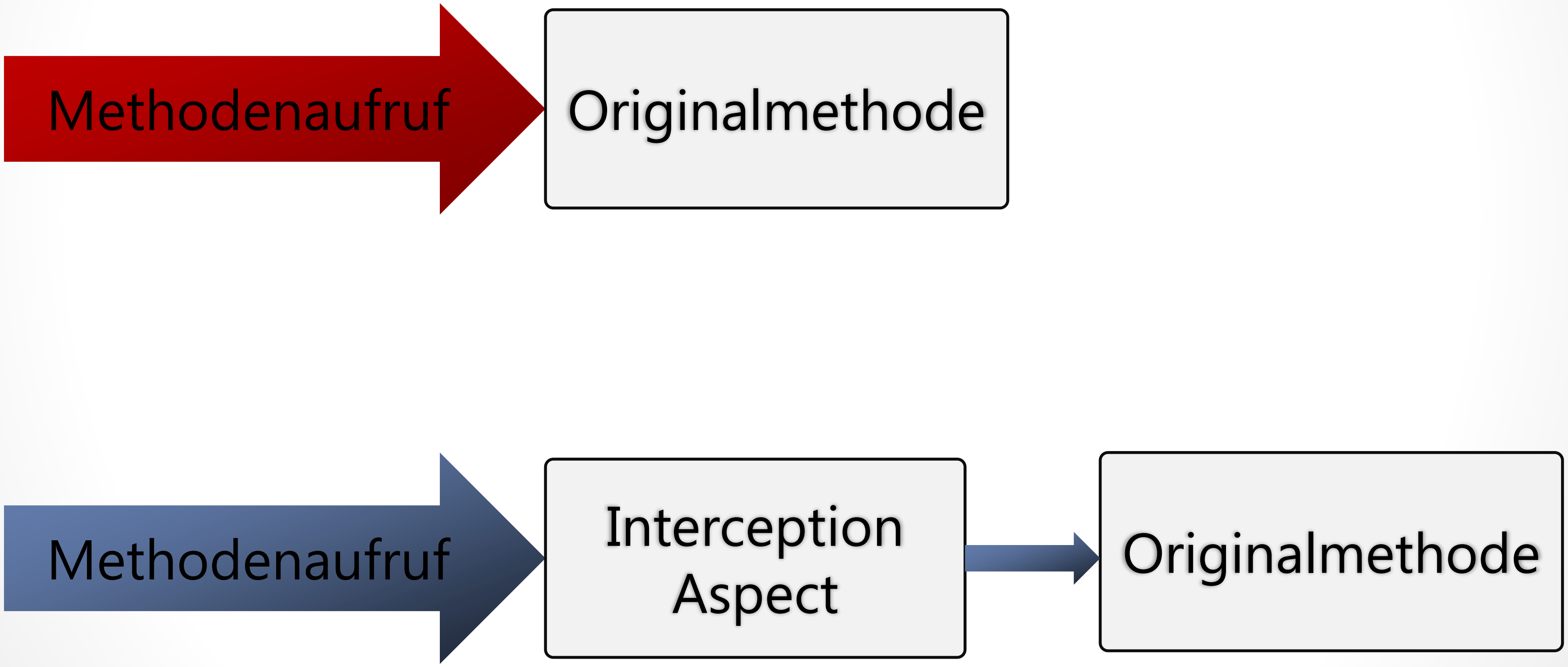
Erweiterung von Klassen

- Interfaces hinzufügen
- Methoden hinzufügen
- Eigenschaften hinzufügen
- Events hinzufügen
- … (siehe PostSharp Dokumentation)

Methoden unterbrechen (Threading)

Demo

MethodInterceptionAspect



INotifyPropertyChanged
mit Aspekt implementieren

- Implementieren des Interfaces INPC
- Event `PropertyChanged` definieren
- Methode `OnPropertyChanged` definieren
- Alle Setter von öffentlichen Eigenschaften ändern

INotifyPropertyChanged

Demo

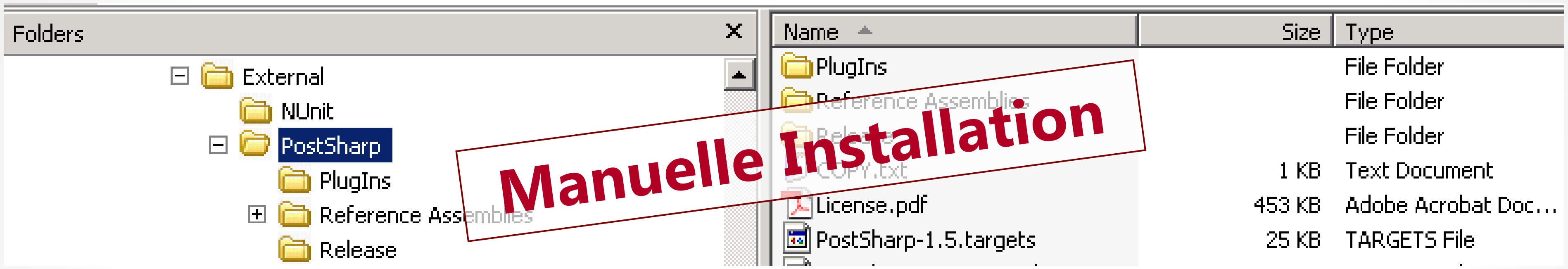
Begriffe: Pointcut, Advice

- Advice: „the what“
  - z.B. Überschreiben von OnEntry ➔ was soll gemacht werden
- Pointcut: „the where“
  - z.B. bei OnMethodBoundaryAspect ➔ alle Methoden, wenn der Aspekt auf der Klasse ist

Einbinden von PostSharp

- Verfügbar ab Version 2.1
- Unterstützt die Visual Studio Extension
- Installation im Source Repository
- Einfache Handhabung

Einbinden von PostSharp



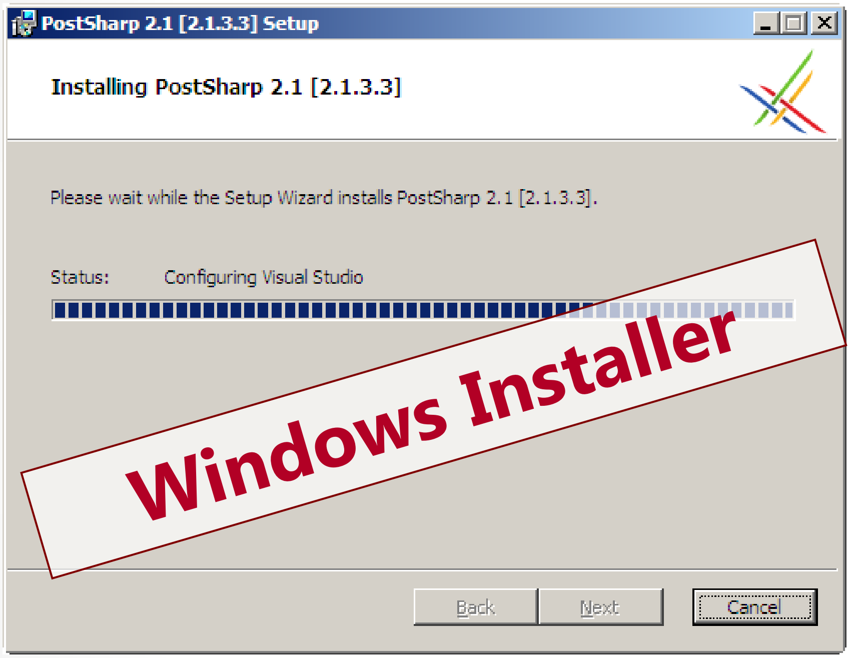
Folders

- External
  - NUnit
  - PostSharp
    - PlugIns
    - Reference Assemblies
    - Release

Name	Size	Type
PlugIns		File Folder
Reference Assemblies		File Folder
Release		File Folder
COPY.txt	1 KB	Text Document
License.pdf	453 KB	Adobe Acrobat Doc...
PostSharp-1.5.targets	25 KB	TARGETS File

- Maximale Eingriffsmöglichkeiten
- Unterstützt die Visual Studio Extension (ab Version 2.1)
- Installation im Source Repository

Einbinden von PostSharp


PostSharp 2.1 [2.1.3.3] Setup

Installing PostSharp 2.1 [2.1.3.3]

Please wait while the Setup Wizard installs PostSharp 2.1 [2.1.3.3].

Status: Configuring Visual Studio

- Sehr einfache Installation
- Unterstützt die Visual Studio Extension (ab 2.0)
- Keine Installation im Source Repository (wird lokal im GAC installiert)

Was bringt's?

Weniger
Lines of Code
schreiben

Weniger
Fehler

Was bringt's?

Mehr Automatisierung

Weniger Fehler

Was bringt's?

Wiederverwendbarkeit

Schneller,
Spart Kosten

Was bringt‘s?

Weniger „Boilerplate“ Code

Weniger Fehler, Bessere Lesbarkeit

Was kostet‘s?

Längere Buildzeiten

Zusätzliches Tool in der Kette

Fragen?

Vielen Dank!

Bernd Hengelein
Siemens AG