Matthias Fischer | www.it-visions.de | dotnetautor.de

Windows-Phone und Windows-8-Apps: Shared Code mit C# und XAML

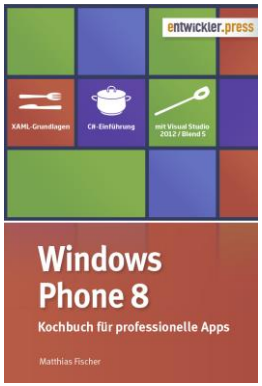
Referentenvorstellung

MobileTech Conference 2014

- Consultant und Trainer
  - .NET-Entwicklung seit 2001
  - ASP.NET, WCF, MVC4, SQL Server 2012
  - WPF, MVVM, Windows Phone 8, Windows 8
- Autor (Auswahl)
  - Carl Hanser Verlag, Addison-Wesley, Wrox,
  - Windows.Developer
- Projekte / Apps
  - OSMLogger, BatchUploader, MobileTech Conference usw.

  - Material zu diesem Workshop : http://bit.ly/mfslides
  - Mehr Informationen: www.dotnetautor.de
  - WP 8 Seminare : dotnetautor.de/training/wp8
  - Kontakt : matthias@dotnetautor.de

NOKIA Developer Certified Trainer



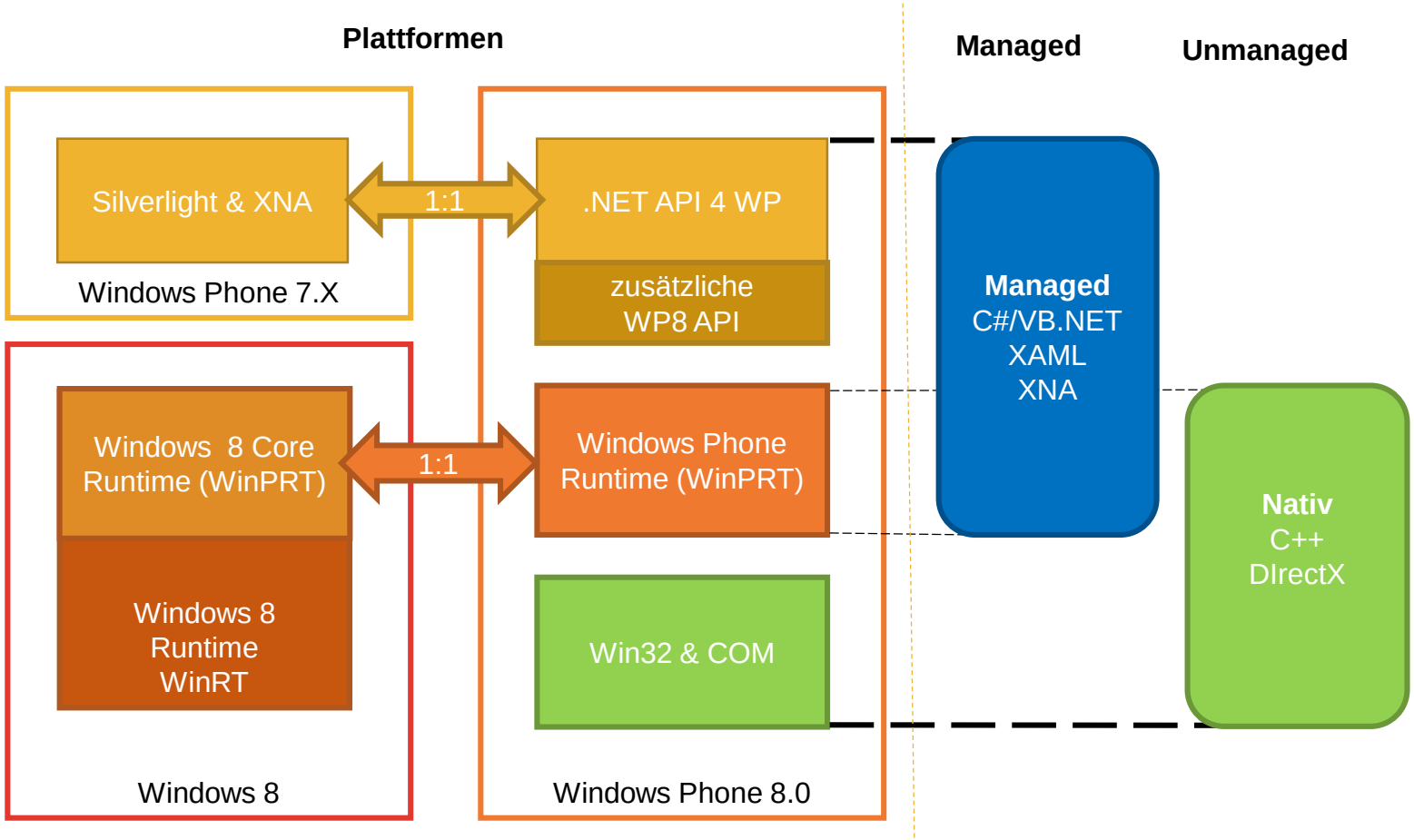
Matthias Fischer

NOKIA Developer Champion

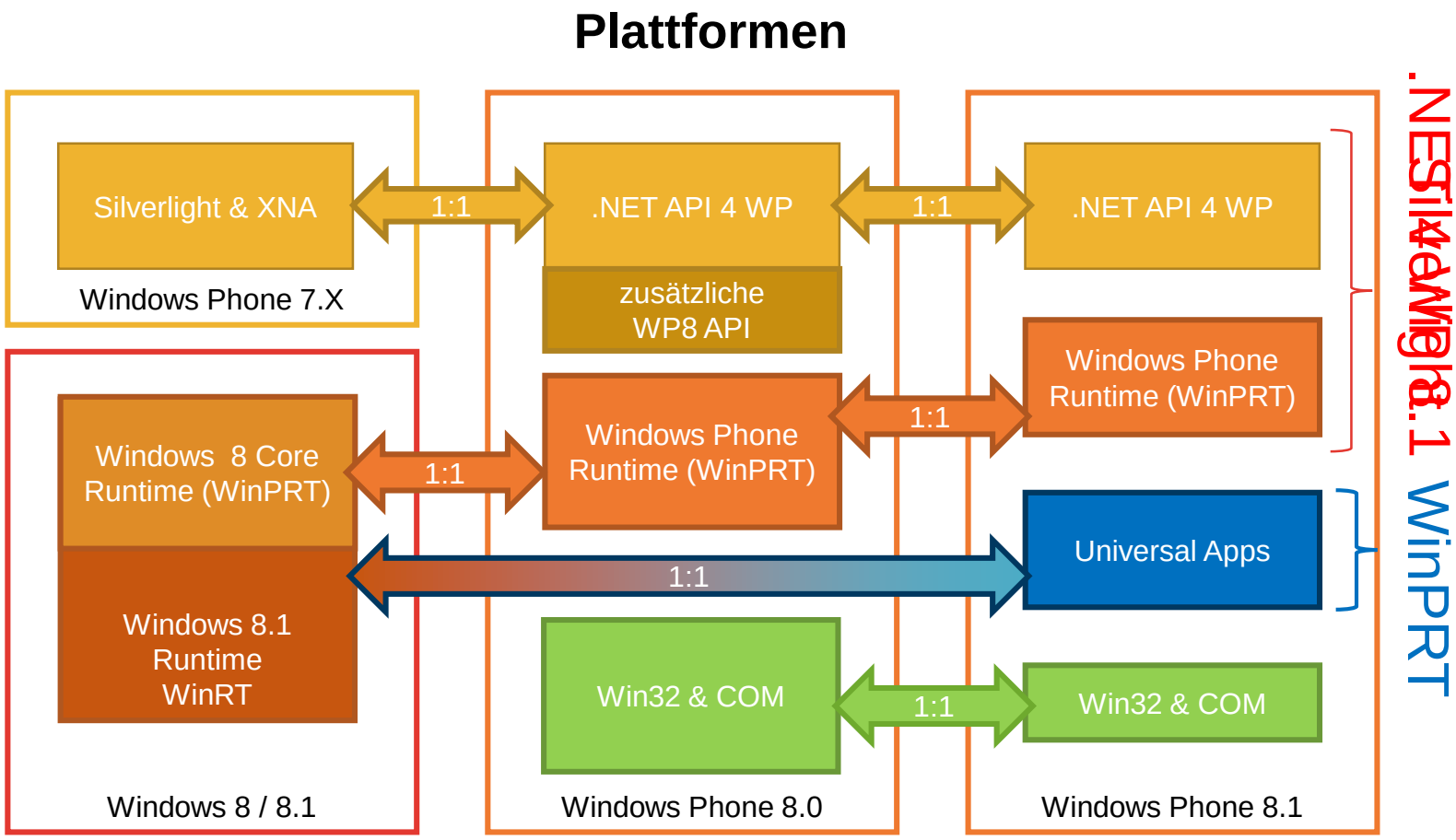
Die Windows 8 Plattform

Shared WinRT Core

Plattformen aus Entwicklersicht



Windows Phone 8.1 / Windows 8.1



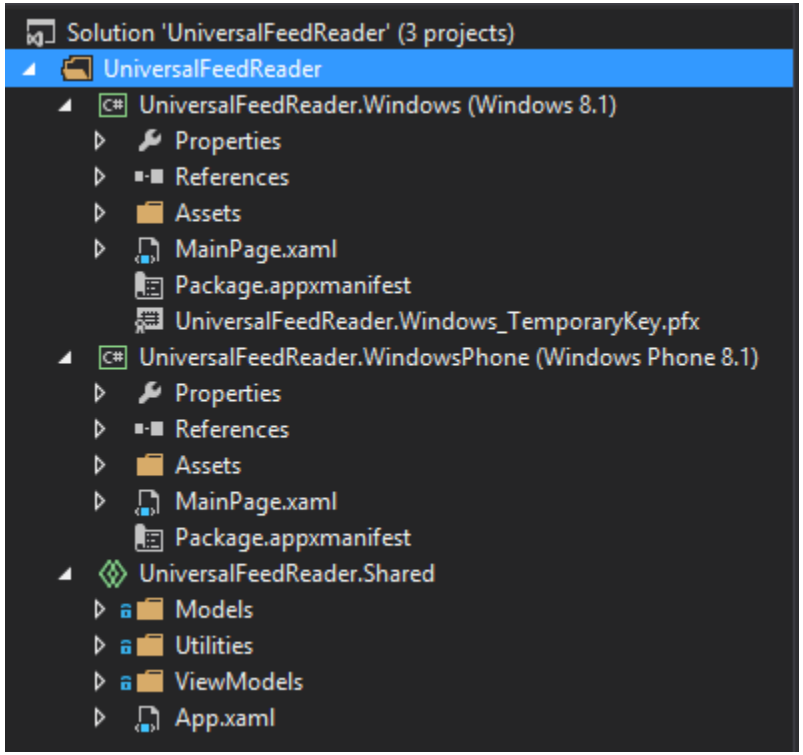
Shared Code

- Ctrl-C, Crtl-V
  - Einfach und schnell, jedoch (fast) unmöglich zu warten
  - Einzige Möglichkeit für XAML Design-Code-Share
- Linked Files
  - Visual Studio 2013 bietet die Möglichkeit Quellcode durch Drag‘n‘Drop + Alt über Projekte zu verlinken
- Linked Files mit #if Blöcken
  - Unter Umständen werden geringe Anpassungen benötigt, diese können mit Hilfe von #if … #else … #endif Blöcken angepasst werden
- Portable Class Library‘s (PCLs)
  - Eingeführt mit Visual Studio 2012, elegante Möglichkeit für mehrere Plattformen gleichzeitig zu entwickeln, ohne die rohen Quelldateien zu sharen
  - Nachteil: kleinste gemeinsame SubSet von Funktionen
- Plattform Adapter Abstraktion
  - Erzeugen einer abstrakten Klasse
  - Implementieren auf der jeweiligen Plattform

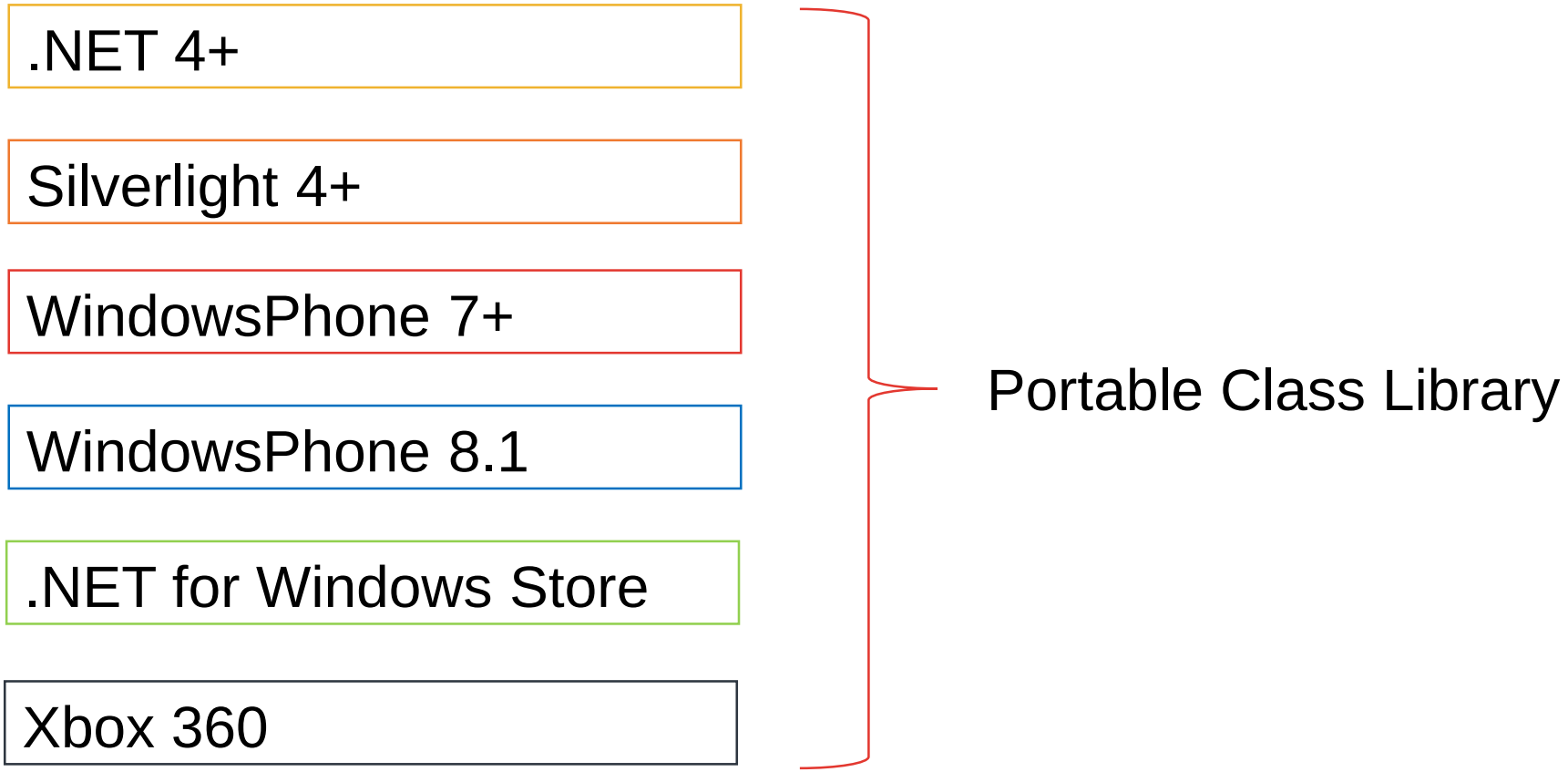
Universal App Projekt

MobileTech Conference 2014

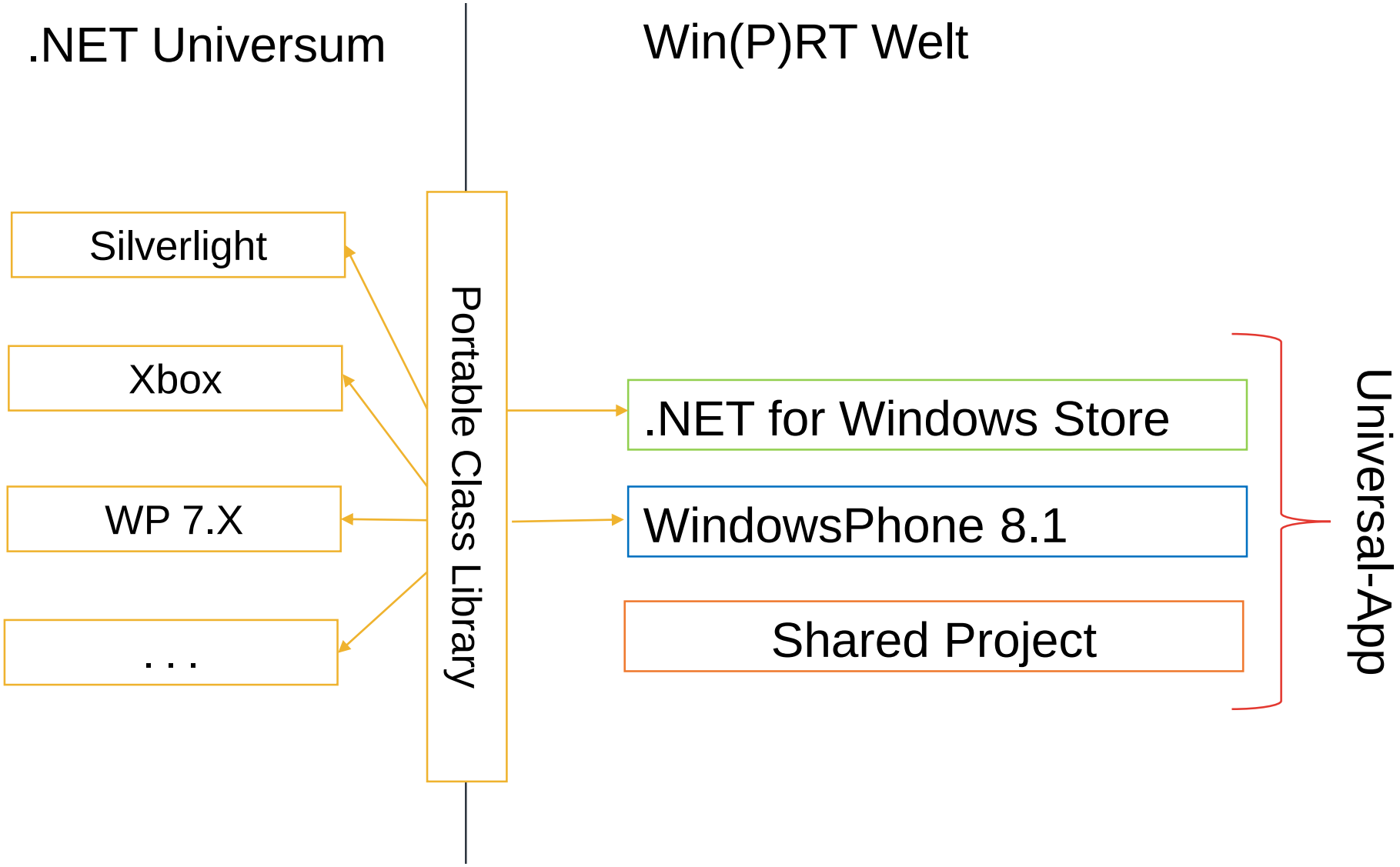
- Besteht aus drei Projekten
  - Zwei plattformabhänigen Projekten (Windows 8.1 und Windows Phon 8.1)
  - Ein Projekt mit den gemeinsamen Quellen und Ressourcen
- Erstellt je ein unabhängiges AppPaket je Plattform
  - Kein Universal-Binary Konzept



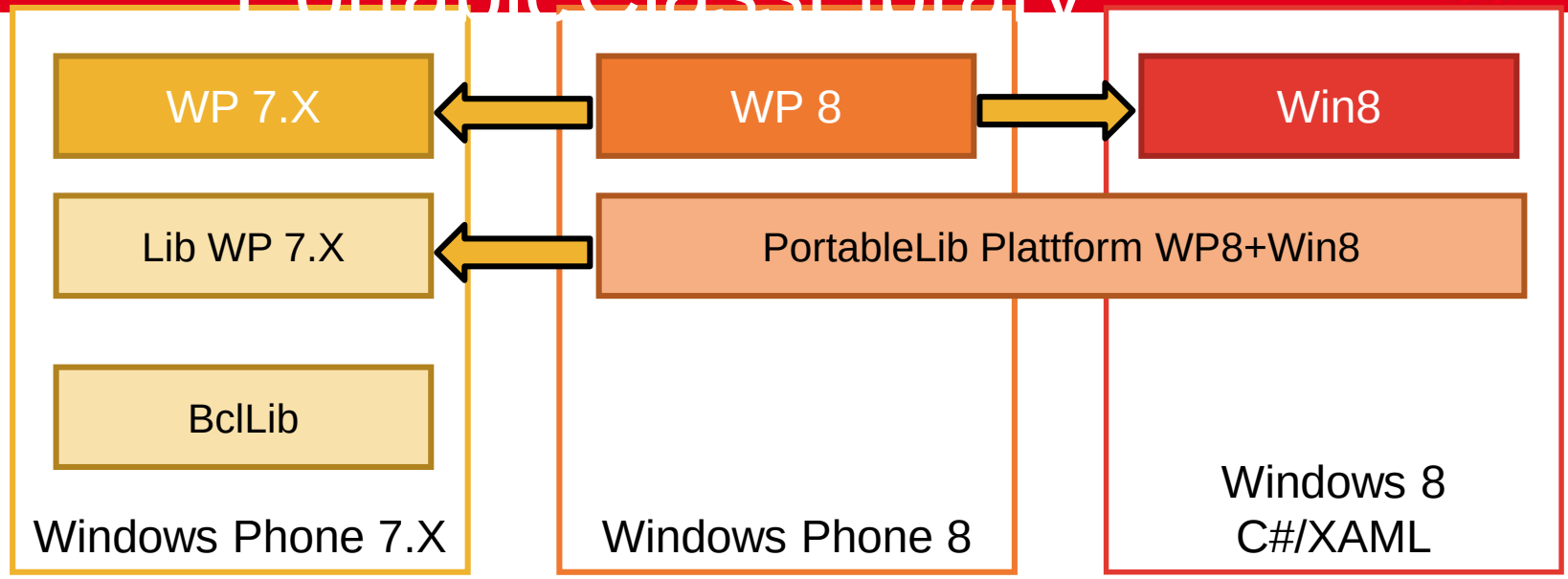
Shared Sources vs. PortableClassLibrary

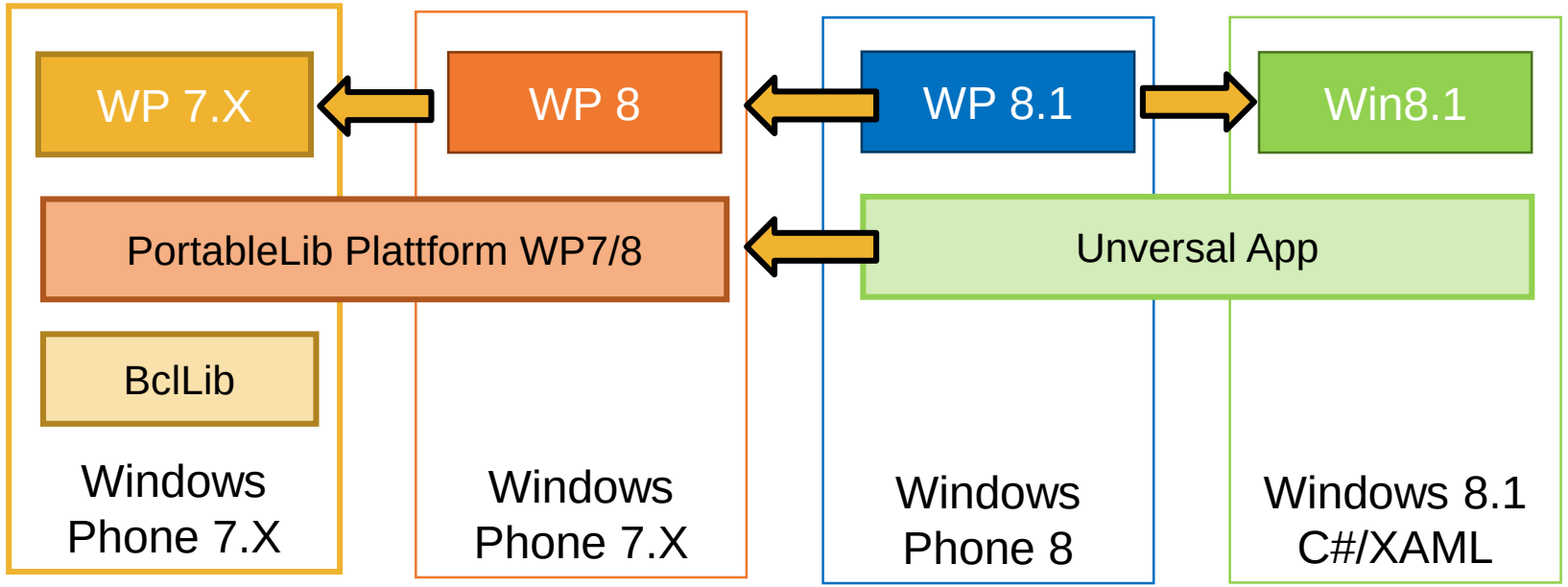


Shared Sources vs. PortableClassLibrary



Shared Sources - PortableClassLibrary





Simple Feed Client

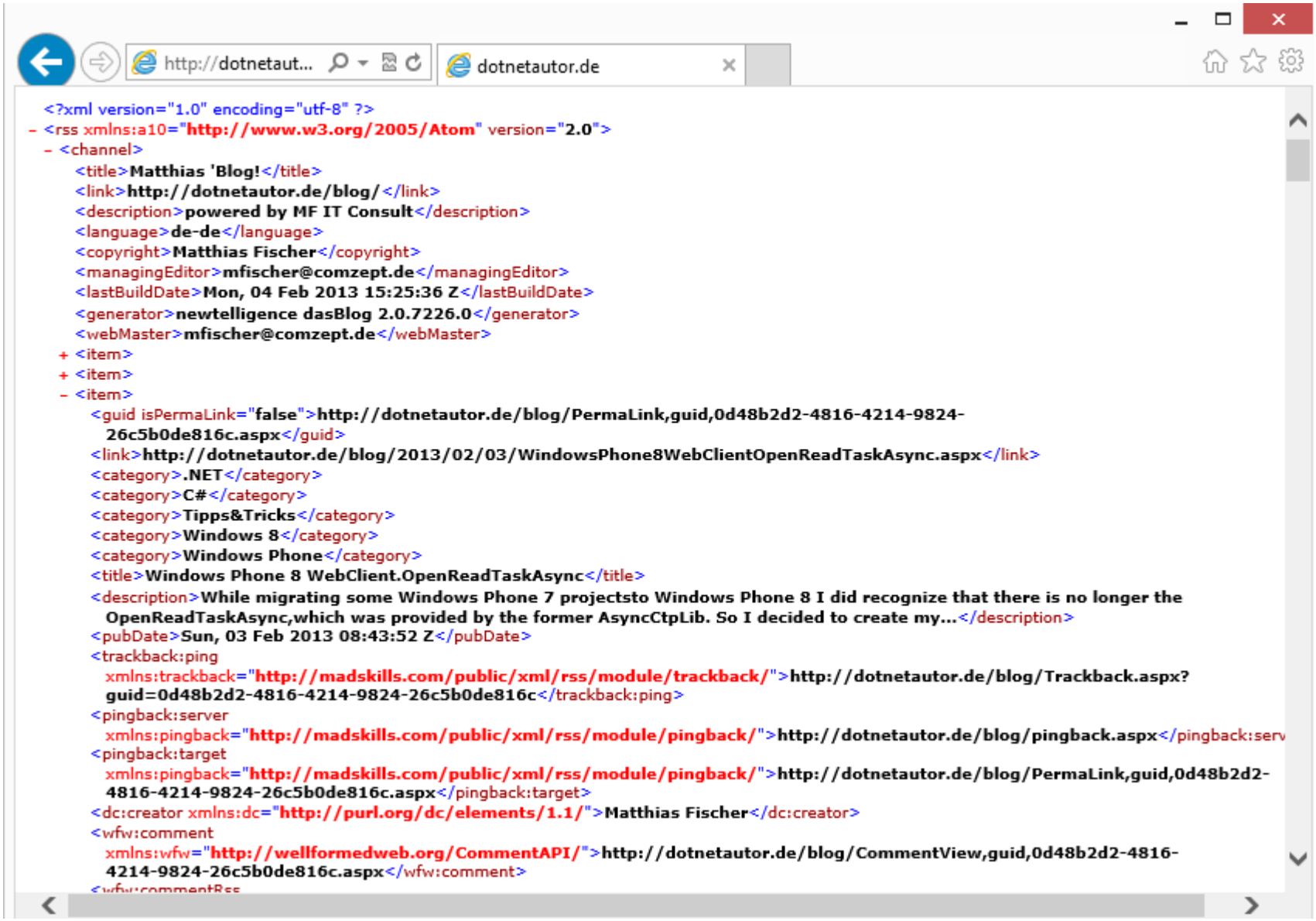
Herunterladen des Feeds

- Verwendung der **WebRequest**-Klasse
  - Zusammensetzen der URL
  - Festlegen des Accept-Headers
- Asynchrones herunterladen der Antwort (await)

```
string result = "";
var request = WebRequest.CreateHttp("http://dotnetautor.de/GetRssFeed");

request.Accept = "application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8";
//request.Headers[HttpRequestHeader.AcceptEncoding] = "gzip";

var response = await request.GetResponseAsync();

using (var stream = response.GetResponseStream()) {
  using (var reader = new StreamReader(stream)) {
     result = await reader.ReadToEndAsync();
  }
}
```

RSS Feed

MobileTech Conference 2014



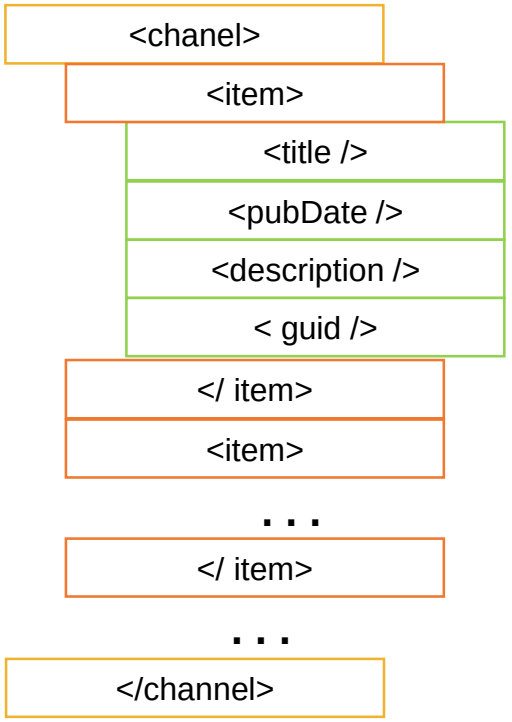

```
<?xml version="1.0" encoding="utf-8" ?>
- <rss xmlns:a10="http://www.w3.org/2005/Atom" version="2.0">
  - <channel>
      <title>Matthias 'Blog!</title>
      <link>http://dotnetautor.de/blog/</link>
      <description>powered by MF IT Consult</description>
      <language>de-de</language>
      <copyright>Matthias Fischer</copyright>
      <managingEditor>mfischer@comzept.de</managingEditor>
      <lastBuildDate>Mon, 04 Feb 2013 15:25:36 Z</lastBuildDate>
      <generator>newtelligence dasBlog 2.0.7226.0</generator>
      <webMaster>mfischer@comzept.de</webMaster>
    + <item>
    + <item>
    - <item>
        <guid isPermaLink="false">http://dotnetautor.de/blog/PermaLink,guid,0d48b2d2-4816-4214-9824-
          26c5b0de816c.aspx</guid>
        <link>http://dotnetautor.de/blog/2013/02/03/WindowsPhone8WebClientOpenReadTaskAsync.aspx</link>
        <category>.NET</category>
        <category>C#</category>
        <category>Tipps&Tricks</category>
        <category>Windows 8</category>
        <category>Windows Phone</category>
        <title>Windows Phone 8 WebClient.OpenReadTaskAsync</title>
        <description>While migrating some Windows Phone 7 projectsto Windows Phone 8 I did recognize that there is no longer the
          OpenReadTaskAsync,which was provided by the former AsyncCtpLib. So I decided to create my...</description>
        <pubDate>Sun, 03 Feb 2013 08:43:52 Z</pubDate>
        <trackback:ping
          xmlns:trackback="http://madskills.com/public/xml/rss/module/trackback/">http://dotnetautor.de/blog/Trackback.aspx?
          guid=0d48b2d2-4816-4214-9824-26c5b0de816c</trackback:ping>
        <pingback:server
          xmlns:pingback="http://madskills.com/public/xml/rss/module/pingback/">http://dotnetautor.de/blog/pingback.aspx</pingback:serv
        <pingback:target
          xmlns:pingback="http://madskills.com/public/xml/rss/module/pingback/">http://dotnetautor.de/blog/PermaLink,guid,0d48b2d2-
          4816-4214-9824-26c5b0de816c.aspx</pingback:target>
        <dc:creator xmlns:dc="http://purl.org/dc/elements/1.1/">Matthias Fischer</dc:creator>
        <wfw:comment
          xmlns:wfw="http://wellformedweb.org/CommentAPI/">http://dotnetautor.de/blog/CommentView,guid,0d48b2d2-4816-
          4214-9824-26c5b0de816c.aspx</wfw:comment>
        <wfw:commentRss
```

Informationen extrahieren

MobileTech Conference 2014

- Eine Klasse, um die Feed Elemente zu speichern

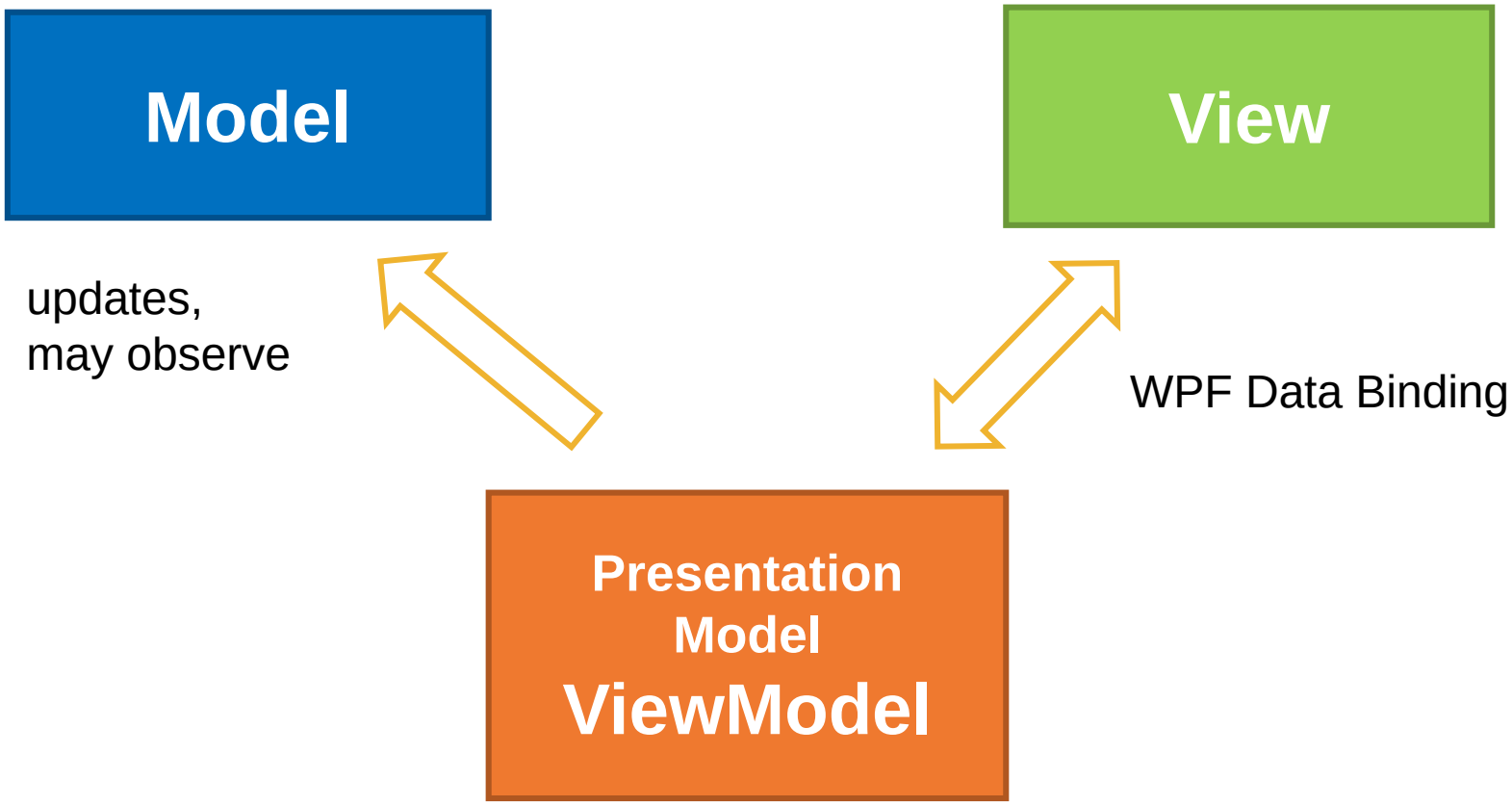
```
public class FeedItem  {

  public string Title { get; set;  }
  public string Description { get; set; }
  public string DatePublished { get; set; }
  public string ArticleURL { get; set;  }

 }
```



- Extrahieren der Items mit Hilfe von Linq2XML

```
var feed = XElement.Parse(result);

var articles = from item in feed.Descendants("item")
               select new FeedItem {
                      Title = item.Element("title").Value,
                      DatePublished = item.Element("pubDate").Value,
                      Description = item.Element("description").Value,
                      ArticleURL = item.Element("guid").Value
                };
```

Presentation Model - MVVM



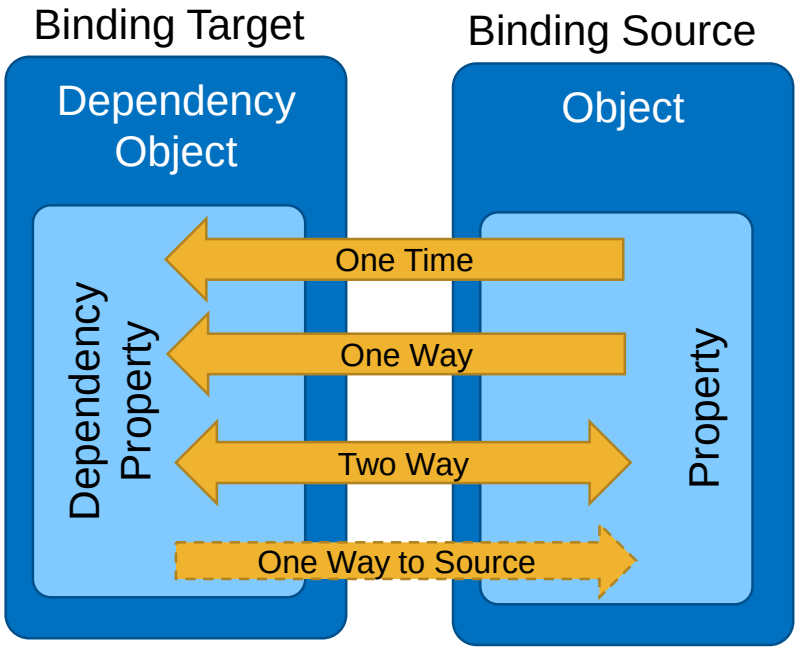
View.DataContext = ViewModel;

http://blogs.msdn.com/b/johngossman/archive/2005/10/08/478683.aspx

Datenbindung

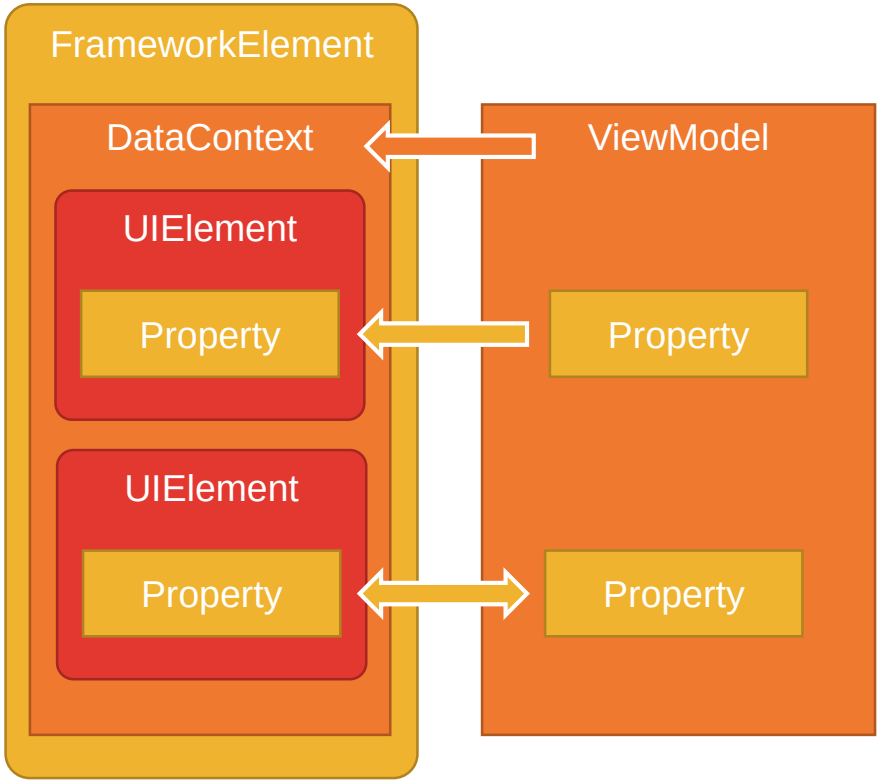
- Datenbindung (data binding) ist ein Mechanismus, um eine lose Kopplung zwischen der UI und der Anwendungslogik herzustellen
- Als Ziel für eine Datenbindung können nur Dependency Objekts verwendet werden.
- Als Quelle kann jedes Objekt verwendet werden. Wenn Änderungen automatisch an das Bindungsziel übermittelt werden sollen, muss
  - entweder die Schnittstelle INotifyPropertyChanged
  - oder ein Dependency Property

  verwendet werden.
- Die Bindung kann einseitig, beidseitig oder einmalig erfolgen.
- Die Bindung kann in XAML oder im CodeBehind angelegt werden.



Datenbindung Funktionsweise

- Eine Daten-Klasse, welche als Quelle für die Datenbindung verwendet wird, wird **ViewModel** genannt.
- Die Datenquelle wird festgelegt, indem der Eigenschaft DataContext eines UI-Containers (FrameworkElement) eine Instanz des ViewModels zugewiesen wird.
- Eigenschaften von UI Elementen innerhalb des Containers, sowie Eigenschaften des Containers können nun an Eigenschaften des ViewModels gebunden werden.



Datenbindung im XAML

MobileTech Conference 2014

- ViewModel-Klasse

```
public class MyViewModel {
  public string TextProperty { get; set; }
}
```

- Der Eigenschaft DataContext wird das ViewModel zugewiesen.

```
  MainContainer.DataContext = new MyViewModel {TextProperty = "Hello Word"};
```

- Die Bindung einer Eigenschaft des ViewModels zu einer Eigenschaft des UI-Elements erfolgt in der XAML Datei

```
<Grid Name="MainContainer" >
  <TextBlock Name="ContentText" Text="{Binding TextPropety, Mode=OneWay}"
             TextWrapping="Wrap" Style="{StaticResource PhoneTextTitle3Style}" />
</Grid>
```

µMVVM Framework

- Komponenten für eine MVVM Anwendung
  - DelegateCommand
  - ViewModels (von ViewModelBase abgeleitet)
  - DataBinding und CommandBinding

- Zur Vereinfachung der Erstellung der ViewModel-Klassen
  - die Schnittstelle `INotifyPropertyChanged` in der Klasse ViewModelBase implementieren.

```
public class ViewModelBase : INotifyPropertyChanged {

   public event PropertyChangedEventHandler PropertyChanged;

   protected virtual void OnPropertyChanged(
     [CallerMemberName] string propertyName = null) {

     if (PropertyChanged!= null)
        handler(this, new PropertyChangedEventArgs(propertyName));
   }
```

DelegateCommand

- Die Implementierung für Kommandos mit Hilfe der Schnittstelle ICommand ist immer gleich. Um nicht jedes Mal eine eigene Klasse für jedes Kommando ausprogrammieren zu müssen, kann die folgende Klasse DelegateCommand als allgemeine Implementierung verwendet werden.

```
public class DelegateCommand : ICommand
{
  Func<object, bool> canExecute;
  Action<object> executeAction;

  public DelegateCommand(Action<object> executeAction)
      : this(executeAction, null)
  {
  }

  public DelegateCommand(Action<object> executeAction,
       Func<object, bool> canExecute)
  {
    if (executeAction == null)
    {
      throw new
        ArgumentNullException("executeAction");
    }
    this.executeAction = executeAction;
    this.canExecute = canExecute;
  }

  public bool CanExecute(object parameter) {
    bool result = true;
    if (canExecute != null) {
      result =  canExecute(parameter);
    }
    return result;
  }

  public event EventHandler CanExecuteChanged;
  public void RaiseCanExecuteChanged()
  {
    if (CanExecuteChanged != null) {
      CanExecuteChanged (this,
                  new EventArgs());
    }
  }

  public void Execute(object parameter) {
    this.executeAction(parameter);
  }
}
```

ViewModel


```
public class FeedReaderViewModel : ViewModelBase {
    private ObservableCollection<FeedItem> _feedItems;

    public FeedReaderViewModel() {
      RefreshCommand = new DelegateCommand(async arg => {
        var res =
           await DataManager.Instance.UpdateFeed(
             "http://dotnetautor.de/GetRssFeed");
        FeedItems = new ObservableCollection<FeedItem>(res);
      });
    }

    public ObservableCollection<FeedItem> FeedItems {
      get { return _feedItems; }
      set {
        if (_feedItems == value) return;
        _feedItems = value;
        OnPropertyChanged();
      }
    }

    public DelegateCommand RefreshCommand { get; protected set; }
  }
```

Fragen?

MobileTech Conference 2014

Brauchen Sie Unterstützung bei .NET, Silverlight, Lightswitch, WCF, WPF, ASP.NET, IIS, Windows 8 oder Windows Phone 8 ?

- Beratung bei Einführung, Migration und Betrieb
- (Vor-Ort-)Schulungen, Workshops
- Coaching (Vor-Ort | Telefon | E-Mail | Online-Meeting)
- Support (Vor-Ort | Telefon | E-Mail | Online-Meeting)
- Entwicklung von Prototypen und Lösung

Matthias Fischer IT Consult
http://www.dotnetautor.de.de
Telefon +49 1520 1920 708
matthias@dotnetautor.de

MobileTech Conference 2014

Mobile Development, Marketing & Business

Vielen Dank für Ihre Aufmerksamkeit!

Auszug aus dem Buch:

Windows Phone 8.1 Kochbuch

Erscheinungsdatum: Q4/2014

Autoren: Matthias Fischer, Gordon Breuer