11/28/2012

# Design-Patter: Model-View-ViewModel (MVVM)

Lars Woitzik

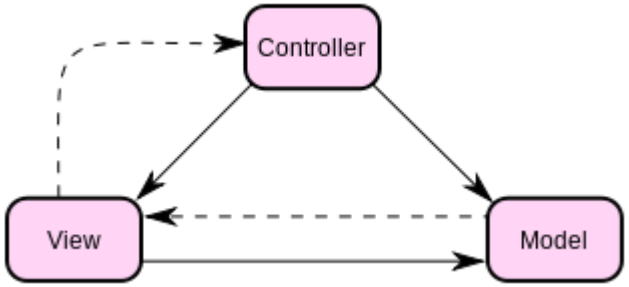
dkfz.
GERMAN
CANCER RESEARCH CENTER
IN THE HELMHOLTZ ASSOCIATION

# History

- Originated from Microsoft as an specialization of the presentation model design pattern.
- Conceived to support WPF and Silverlight
- Now being more broadly applied in other technology domains

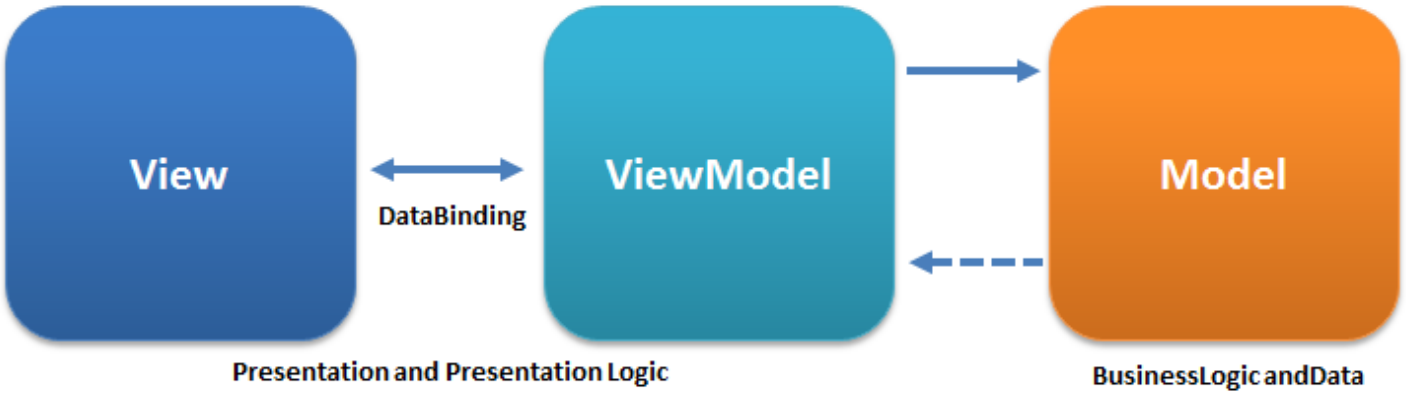
# Basic knowledge

- Model View ViewModel (MVVM) is an architectural pattern
- Largely based on the model–view–controller pattern (MVC)



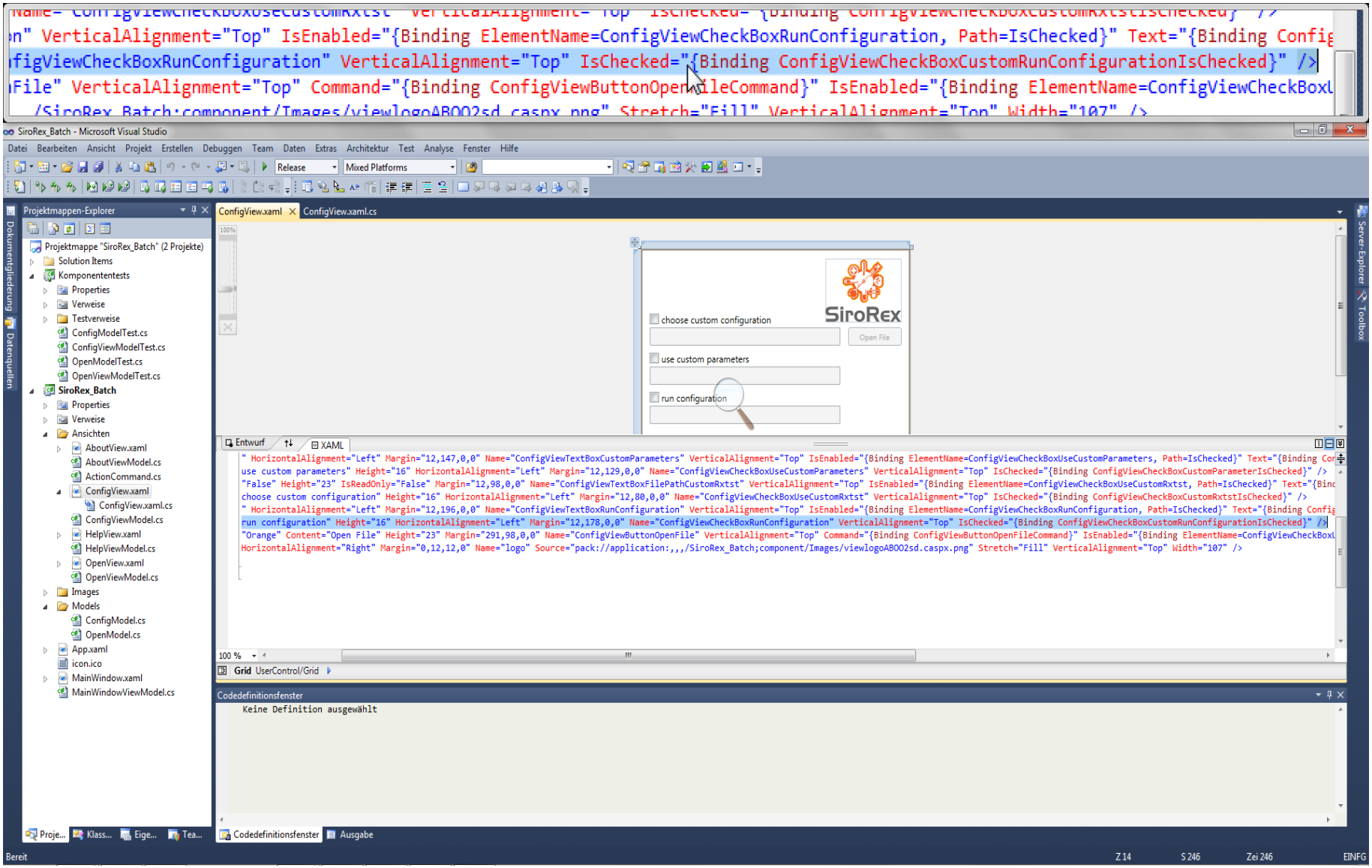

- MVVM is targeted at modern UI development platforms which support Event-driven programming

# Structure

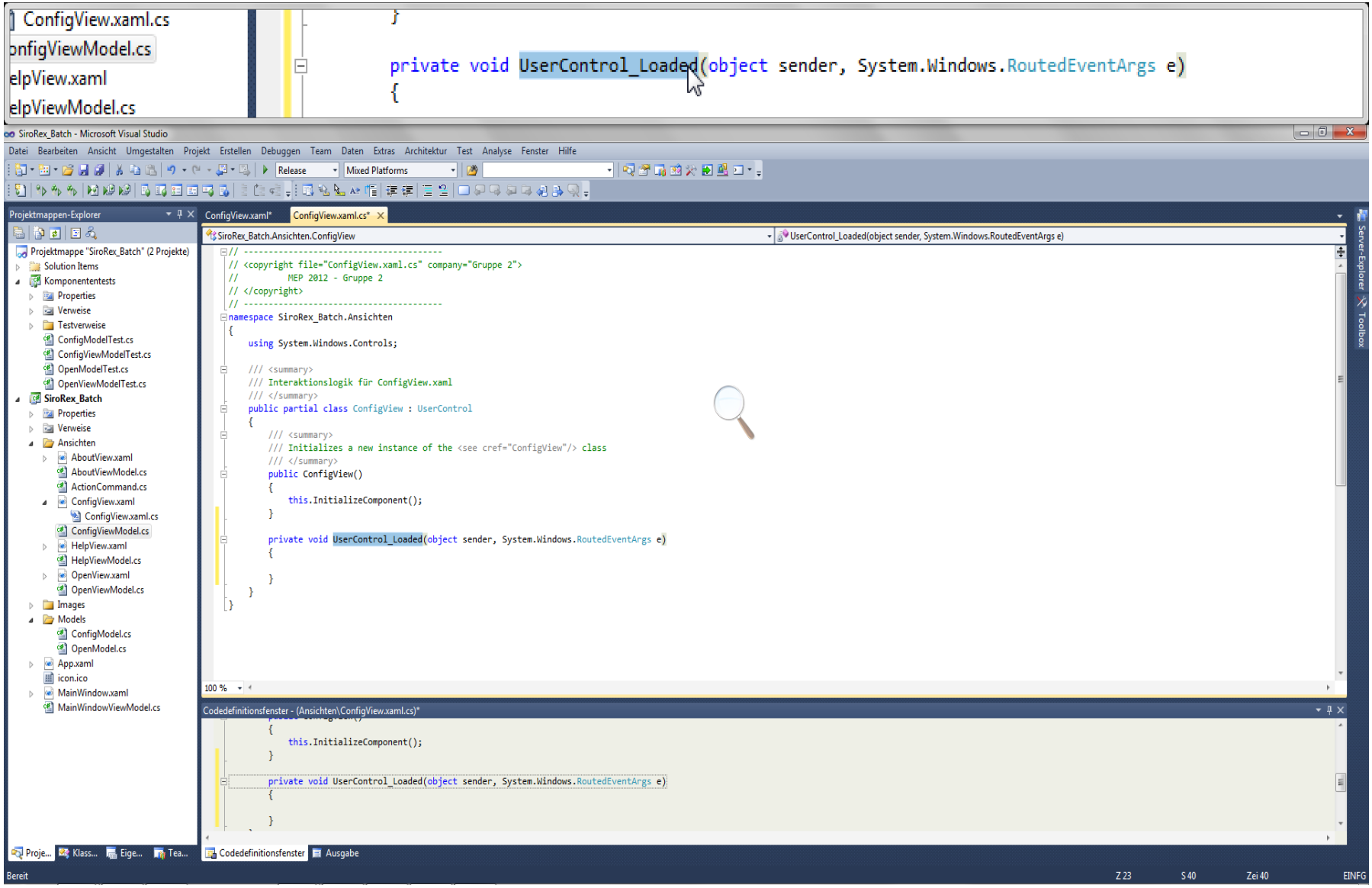



- Model:
  - As in the classic MVC Pattern
- View:
  - As in the classic MVC Pattern
- ViewModel:
  - Abstraction of the view
  - Specialized aspect of a controller
  - Switch Information between View and Model

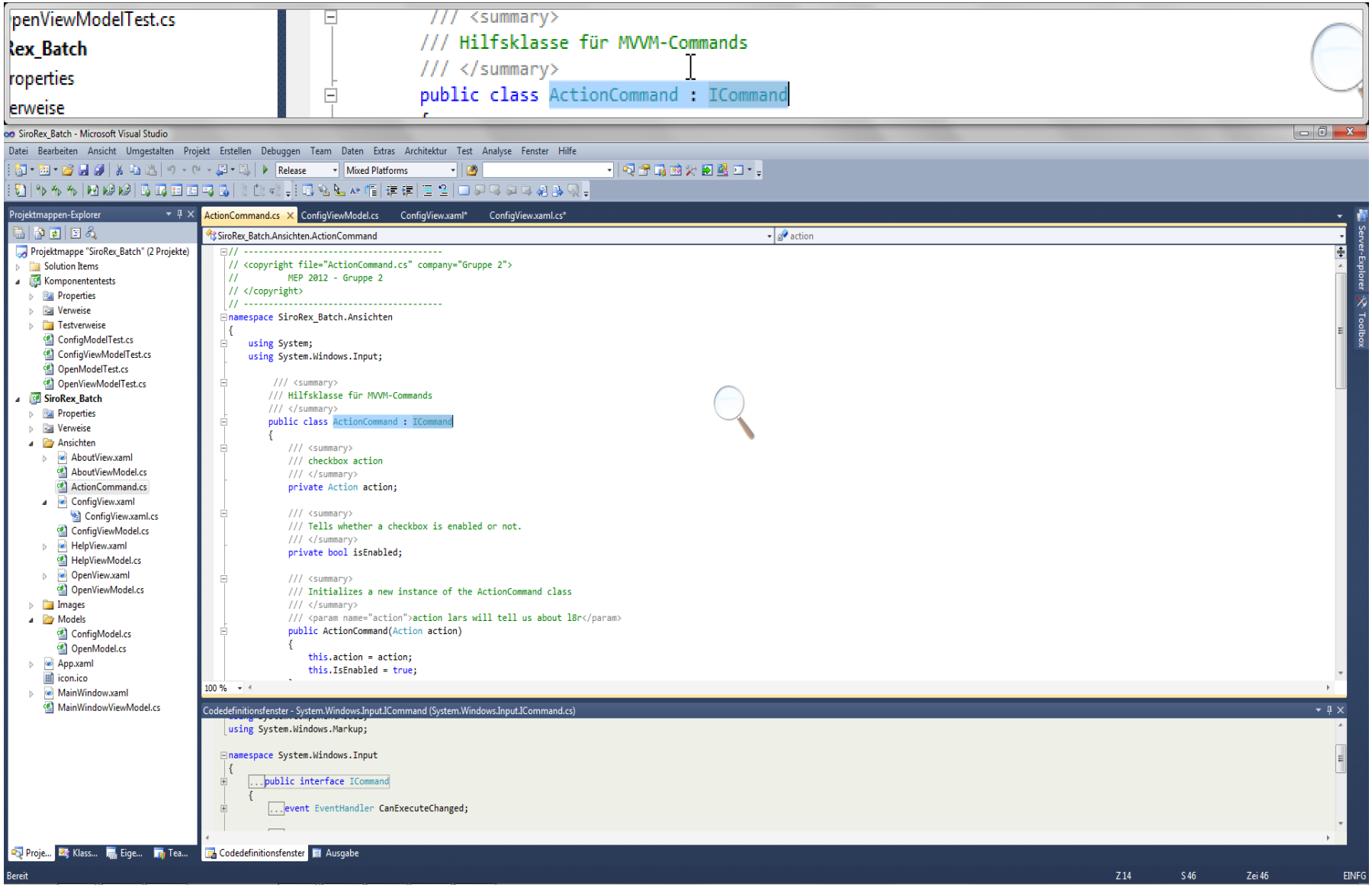
# An example


SiroRex_Batch - Microsoft Visual Studio
Projektmappen-Explorer
ConfigView.xaml
ConfigView.xaml.cs
choose custom configuration
use custom parameters
run configuration
Open File
SiroRex
Entwurf
XAML
Codedefinitionsfenster
Keine Definition ausgewählt
Bereit

# An example


ConfigView.xaml.cs
onfigViewModel.cs
elpView.xaml
elpViewModel.cs
}
private void UserControl_Loaded(object sender, System.Windows.RoutedEventArgs e)
{
SiroRex_Batch - Microsoft Visual Studio
Datei Bearbeiten Ansicht Umgestalten Projekt Erstellen Debuggen Team Daten Extras Architektur Test Analyse Fenster Hilfe
Release
Mixed Platforms
Projektmappen-Explorer
Projektmappe "SiroRex_Batch" (2 Projekte)
Solution Items
Komponententests
Properties
Verweise
Testverweise
ConfigModelTest.cs
ConfigViewModelTest.cs
OpenModelTest.cs
OpenViewModelTest.cs
SiroRex_Batch
Properties
Verweise
Ansichten
AboutView.xaml
AboutViewModel.cs
ActionCommand.cs
ConfigView.xaml
ConfigView.xaml.cs
ConfigViewModel.cs
HelpView.xaml
HelpViewModel.cs
OpenView.xaml
OpenViewModel.cs
Images
Models
ConfigModel.cs
OpenModel.cs
App.xaml
icon.ico
MainWindow.xaml
MainWindowViewModel.cs
ConfigView.xaml*
ConfigView.xaml.cs*
SiroRex_Batch.Ansichten.ConfigView
UserControl_Loaded(object sender, System.Windows.RoutedEventArgs e)
// ------------------------------------------
// <copyright file="ConfigView.xaml.cs" company="Gruppe 2">
// MEP 2012 - Gruppe 2
// </copyright>
// ------------------------------------------
namespace SiroRex_Batch.Ansichten
{
using System.Windows.Controls;
/// <summary>
/// Interaktionslogik für ConfigView.xaml
/// </summary>
public partial class ConfigView : UserControl
{
/// <summary>
/// Initializes a new instance of the <see cref="ConfigView"/> class
/// </summary>
public ConfigView()
{
this.InitializeComponent();
}
private void UserControl_Loaded(object sender, System.Windows.RoutedEventArgs e)
{
}
}
}
100 %
Codedefinitionsfenster - (Ansichten\ConfigView.xaml.cs)*
{
this.InitializeComponent();
}
private void UserControl_Loaded(object sender, System.Windows.RoutedEventArgs e)
{
}
Proje...
Klass...
Eige...
Tea...
Codedefinitionsfenster
Ausgabe
Bereit
Z 23
S 40
Zei 40
EINFG
Server-Explorer
Toolbox

# An example


penViewModelTest.cs
Rex_Batch
roperties
erweise
/// <summary>
/// Hilfsklasse für MVVM-Commands
/// </summary>
public class ActionCommand : ICommand
SiroRex_Batch - Microsoft Visual Studio
Datei Bearbeiten Ansicht Umgestalten Projekt Erstellen Debuggen Team Daten Extras Architektur Test Analyse Fenster Hilfe
Release
Mixed Platforms
Projektmappen-Explorer
Projektmappe "SiroRex_Batch" (2 Projekte)
Solution Items
Komponententests
Properties
Verweise
Testverweise
ConfigModelTest.cs
ConfigViewModelTest.cs
OpenModelTest.cs
OpenViewModelTest.cs
SiroRex_Batch
Properties
Verweise
Ansichten
AboutView.xaml
AboutViewModel.cs
ActionCommand.cs
ConfigView.xaml
ConfigView.xaml.cs
ConfigViewModel.cs
HelpView.xaml
HelpViewModel.cs
OpenView.xaml
OpenViewModel.cs
Images
Models
ConfigModel.cs
OpenModel.cs
App.xaml
icon.ico
MainWindow.xaml
MainWindowViewModel.cs
ActionCommand.cs
ConfigViewModel.cs
ConfigView.xaml*
ConfigView.xaml.cs*
SiroRex_Batch.Ansichten.ActionCommand
action
// ----------------------------------------
// <copyright file="ActionCommand.cs" company="Gruppe 2">
// MEP 2012 - Gruppe 2
// </copyright>
// ----------------------------------------
namespace SiroRex_Batch.Ansichten
{
using System;
using System.Windows.Input;
/// <summary>
/// Hilfsklasse für MVVM-Commands
/// </summary>
public class ActionCommand : ICommand
{
/// <summary>
/// checkbox action
/// </summary>
private Action action;
/// <summary>
/// Tells whether a checkbox is enabled or not.
/// </summary>
private bool isEnabled;
/// <summary>
/// Initializes a new instance of the ActionCommand class
/// </summary>
/// <param name="action">action lars will tell us about l8r</param>
public ActionCommand(Action action)
{
this.action = action;
this.IsEnabled = true;
100 %
Codedefinitionsfenster - System.Windows.Input.ICommand (System.Windows.Input.ICommand.cs)
using System.Windows.Markup;
namespace System.Windows.Input
{
public interface ICommand
{
event EventHandler CanExecuteChanged;
Codedefinitionsfenster
Ausgabe
Bereit
Z 14
S 46
Zei 46
EINFG

# An example

```
/// <returns>If the CheckedConfiguration is enabled or disabled</returns>
public static bool GetConfigViewCheckBoxCustomRunConfigurationIsChecked()
{
    return configViewCheckBoxCustomRunConfigurationIsChecked;
}
```

SiroRex_Batch - Microsoft Visual Studio

Datei Bearbeiten Ansicht Umgestalten Projekt Erstellen Debuggen Team Daten Extras Architektur Test Analyse Fenster Hilfe

Release | Mixed Platforms

Projektmappen-Explorer

- Projektmappe "SiroRex_Batch" (2 Projekte)
  - Solution Items
  - Komponententests
    - Properties
    - Verweise
    - Testverweise
    - ConfigModelTest.cs
    - ConfigViewModelTest.cs
    - OpenModelTest.cs
    - OpenViewModelTest.cs
  - SiroRex_Batch
    - Properties
    - Verweise
    - Ansichten
      - AboutView.xaml
      - AboutViewModel.cs
      - ActionCommand.cs
      - ConfigView.xaml
        - ConfigView.xaml.cs
      - ConfigViewModel.cs
      - HelpView.xaml
      - HelpViewModel.cs
      - OpenView.xaml
      - OpenViewModel.cs
    - Images
    - Models
      - ConfigModel.cs
      - OpenModel.cs
    - App.xaml
    - icon.ico
    - MainWindow.xaml
    - MainWindowViewModel.cs

ConfigModel.cs | ActionCommand.cs | ConfigViewModel.cs | ConfigView.xaml* | ConfigView.xaml.cs*

SiroRex_Batch.Models.ConfigModel | GetConfigViewCheckBoxCustomRunConfigurationIsChecked()

```
        {
            return configViewCheckBoxCustomRxtstIsChecked;
        }

        /// <summary>
        /// Setter-Methode of the CustomRxstst CheckBox
        /// </summary>
        /// <param name="value">Configured the CheckedConfiguration</param>
        public static void SetConfigViewCheckBoxCustomRxtstIsChecked(bool value)
        {
            configViewCheckBoxCustomRxtstIsChecked = value;
        }

        /// <summary>
        /// Getter-Methode of the CustomRunConfiguration CheckBox
        /// </summary>
        /// <returns>If the CheckedConfiguration is enabled or disabled</returns>
        public static bool GetConfigViewCheckBoxCustomRunConfigurationIsChecked()
        {
            return configViewCheckBoxCustomRunConfigurationIsChecked;
        }

        /// <summary>
        /// Setter-Methode of the CustomRunConfiguration CheckBox
        /// </summary>
        /// <param name="value">Configured the CheckedConfiguration</param>
        public static void SetConfigViewCheckBoxCustomRunConfigurationIsChecked(bool value)
        {
            configViewCheckBoxCustomRunConfigurationIsChecked = value;
        }

        /// <summary>
        /// Getter-Methode for openViewTextBoxFilePathCustomRxtstText
```

100 %

Codedefinitionsfenster - System.Windows.Input.ICommand (System.Windows.Input.ICommand.cs)

```
using System.Windows.Markup;

namespace System.Windows.Input
{
    ...public interface ICommand
    {
        ...event EventHandler CanExecuteChanged;
```

Proje... | Klass... | Eige... | Tea... | Codedefinitionsfenster | Ausgabe

Bereit | Z 97 | S 82 | Zei 82 | EINFG

# Summary

- Model View ViewModel (MVVM) is an architectural pattern
- MVVM facilitates a clear separation of the development of the GUI from the development of the business logic
- For the Implementation:
  - Needs an extra ActionCommand-Class-Implementation
  - The View is replaceable
  - The ViewModel used as a switch between View and Model
  - The Model keeps the values

# Questions