ERGOSIGN

WPF UI DEVELOPMENT UNCHAINED

David C. Thömmes
Software Engineer
Lead Software Engineering Standards
thoemmes@ergosign.de

Ergosign GmbH

www.davidchristian.de

André Lanninger
UI Developer

lanninger@ergosign.de

Ergosign GmbH

UI DEVELOPMENT?

Overview | Process CW | Process TBG | Process Alpha | Hydraulic | Oil Tanks | Settings

Param 1	220 c	Length	55mm	Length 2	567mm	#1	Level #1 / Level #2 / Level #3 / Level #4 / Stop / Open Setting
Heap:	-99	Tailout	8	Tailout 2	83		
ID:	#1235	Slab	13:45	Slab 2	---		
Volt:	12v	Time	NaN	Time 2	NaN		

Coax

--

Function Point

Retriv

Action 1 | Action 2 | Action 3

90 °C

Monday

10/10/2012 - 13:45

Overview Gate 1

Quad 656

Fast 24536.657567

GHF MODE

Net 12324

Kmu

Major 1 Gate Overview

Param 1	86	Length	3452	Length 2	976	Major	Major #1 / Major #2 / Major #3 / Major #4
Heap:	2234	Tailout	8523	Tailout 2	324		
ID:	2321	Slab	#	Slab 2	1234		
Volt:	534	Time	def.error	Time 2	245		

THC System

Eloo 55

Running 34

Error 12

Remaining 334

Main

Temp Surro Gate

Torrent 6

Floated 27

Released 30

Heat 4

Temp Surro Gate

Clean up

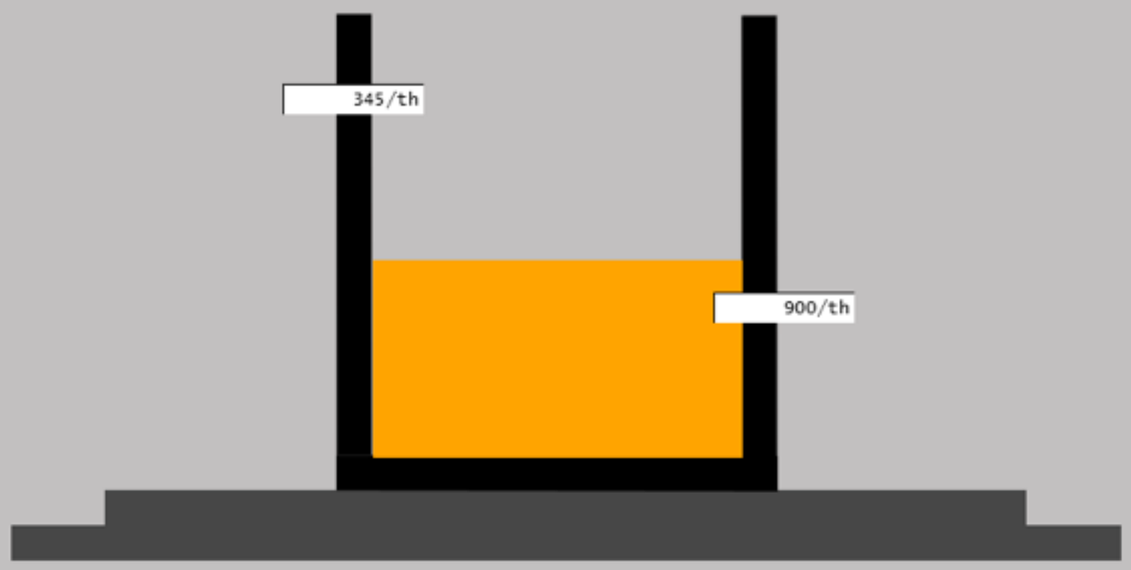
Inform

Help

Rollout

Swap

ABC Test



Param 1	86	Length	3452	Length 2	976	Major	Major #1 / Major #2 / Major #3 / Major #4
Heap:	2234	Tailout	8523	Tailout 2	324		
ID:	2321	Slab	#	Slab 2	1234		
Volt:	534	Time	def.error	Time 2	245		

Weight Gater

Netto 1.5t

Brutto 1.7t

Burst 0.001mg

Swap -

State

- Safe is ok
- Error Container #1
- Alert is ok
- Maincontainer
- Route is ok
- Hydro is ok
- Giga is ok
- Mega is ok
- Herz is neutral
- CNN is ok
- BBCx is ok
- Mount error

```
10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx                ACK
10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx                ACK
10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx                ACK
10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx                ACK
10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx                ACK
10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx                ACK
10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx                ACK
10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx                ACK
```

Quit | Condor Mode | FCK free | UX Mode | Torrent Car

Overview | Process CW | Process TBG | Process Alpha | Hydraulic | Oil Tanks | Settings

Param 1	220 c	Length	55mm	Length 2	567mm		
Heap:	-99	Tailout	8	Tailout 2	83		
ID:	#1235	Slab	13:45	Slab 2	---		
Volt:	12v	Time	NaN	Time 2	NaN		

#1

Level #1 | Level #2 | Level #3 | Level #4 | Stop

Open Setting

Coax

--

Function Point

Retriv

Action 1 | Action 2 | Action 3

90 °C

Monday

10/10/2012 - 13:45

Overview Gate 1

Quad 656

Fast 24536.657567

GHF MODE

Net 12324

Kmu

Major 1 Gate Overview

Param 1	86	Length	3452	Length 2	976		
Heap:	2234	Tailout	8523	Tailout 2	324		
ID:	2321	Slab	#	Slab 2	1234		
Volt:	534	Time	def.error	Time 2	245		

Major

Major #1 | Major #2 | Major #3 | Major #4

THC System

Eloo 55

Running 34

Error 12

Remaining 334

Main

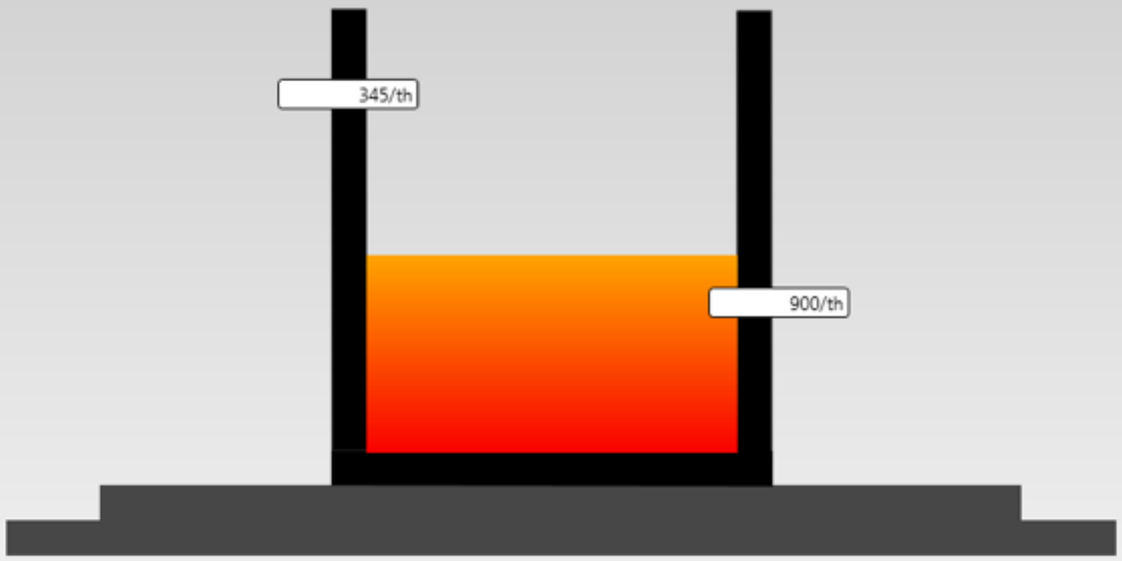
Temp Surro Gate

Torrent 6

Floated 27

Released 30

Heat 4



Weight Gater

Netto 1.5t

Brutto 1.7t

Burst 0.001mg

Swap -

Temp Surro Gate

Clean up

Inform

Help

Rollout

Swap

ABC Test

State

- Safe is ok
- Error Container #1
- Alert is ok
- Maincontainer
- Route is ok
- Hydro is ok
- Giga is ok
- Mega is ok
- Herz is neutral
- CNN is ok
- BBCx is ok
- Mount error

Param 1	86	Length	3452	Length 2	976		
Heap:	2234	Tailout	8523	Tailout 2	324		
ID:	2321	Slab	#	Slab 2	1234		
Volt:	534	Time	def.error	Time 2	245		

Major

Major #1 | Major #2 | Major #3 | Major #4

10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx ACK

10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx ACK

10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx ACK

10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx ACK

10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx ACK

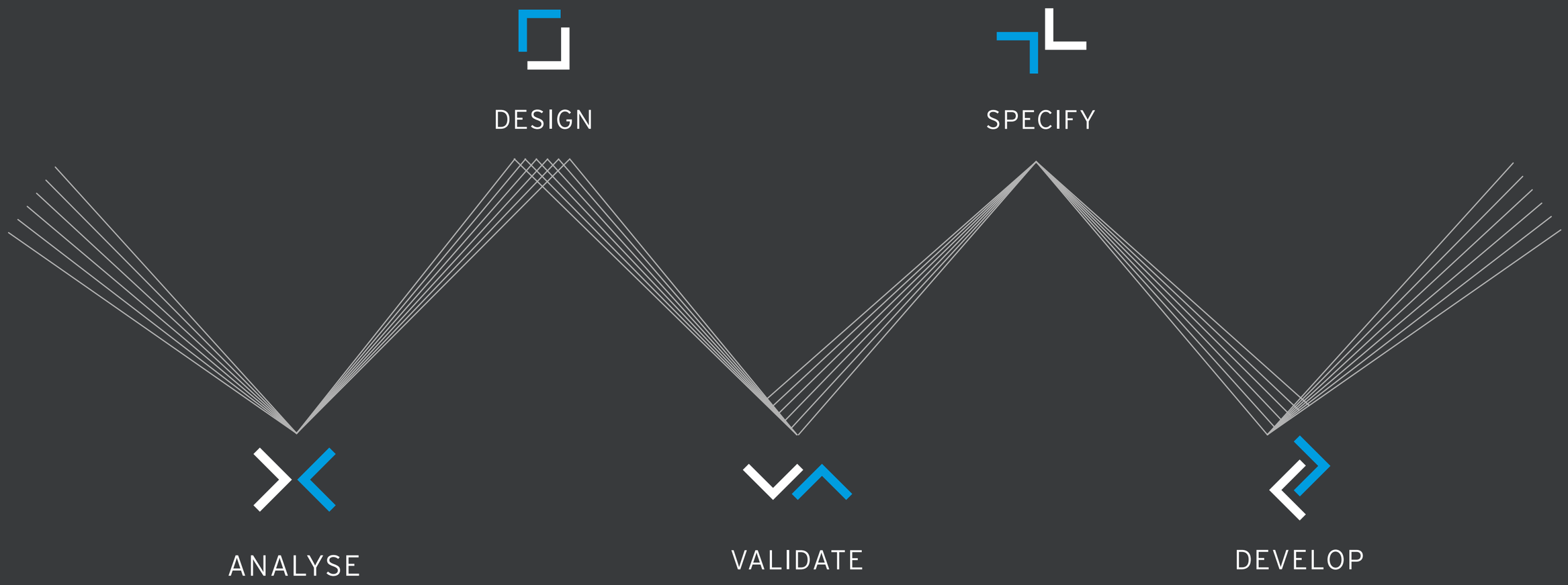
10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx ACK

10/10/2012 - 13:45 :: Shift Log :: Performance Filter - Down Time up to 6c net frame fps:80 87 64 ach gx ACK

Quit | Condor Mode | FCK free | UX Mode | Torrent Car

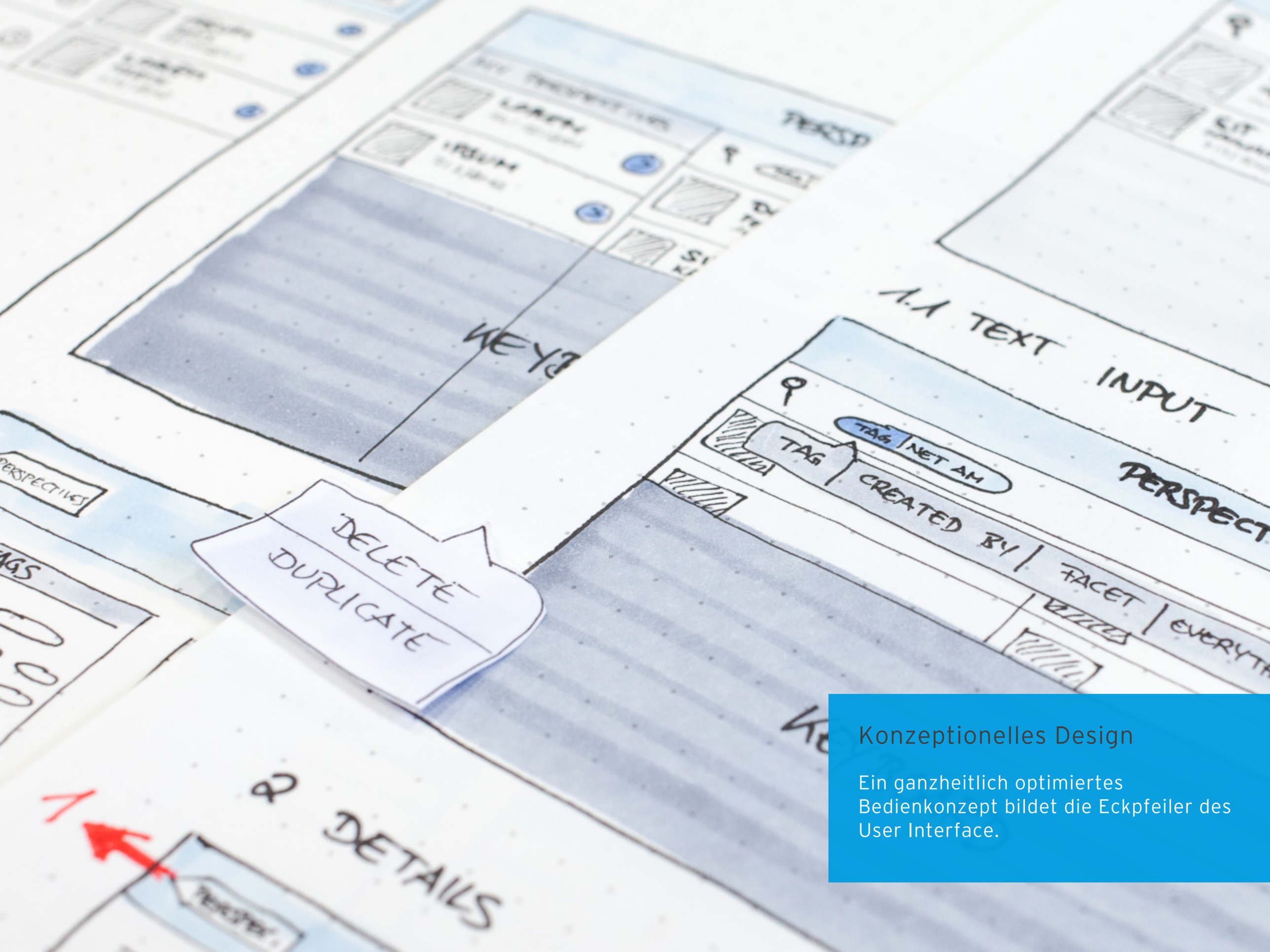
> **UI DEVELOPMENT?** USER CENTERED DESIGN

> **UI DEVELOPMENT?** USER CENTERED DESIGN



Kontextuelle Analyse

Beobachtungen und semistrukturierte Interviews bei Endanwendern vor Ort.



Konzeptionelles Design

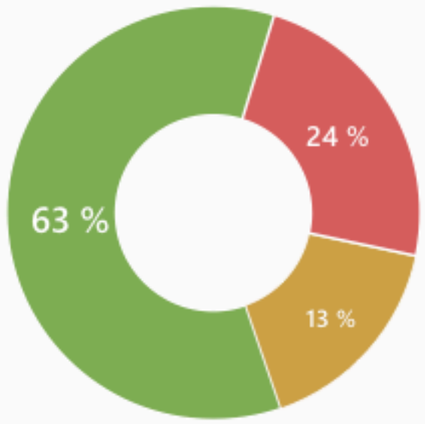
Ein ganzheitlich optimiertes Bedienkonzept bildet die Eckpfeiler des User Interface.

Start | Dashboard | Jobs | Clients | Benutzer | 3

Systemübersicht | Office Rollout | Seite 3 | Verwalten

Aktivität

Jobs | Clients | Letzte 24 Stunden

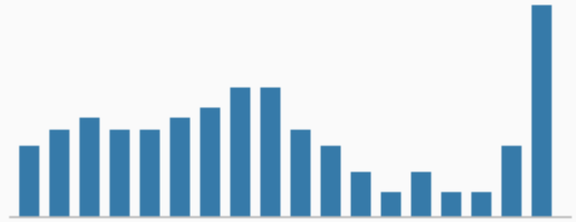


Jobs

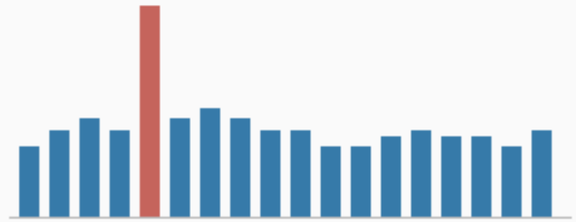
Letzte 24 Stunden | 57 Jobs

Bezeichnung	Zustand	alle	fertig	fehlerhaft
.NET Framework 2.0	97%	120	109	11
7-zip 4.65	95%	74	70	4
Avira Antivir 10.0	81%	365	354	11
Basic PC Setup	78%	118	98	20
BGInfo	74%	37	31	6
Deinst: OpenOffice	50%	88	70	18
Excel Viewer 2003	44%	81	66	15
Filezilla 3.2.3.1	44%	111	97	14
INST: Apple iTunes 10.6.1-x86	31%	65	62	3
ImgBurn 2.4.3.0	30%	21	12	9
INST: Adobe Reader	23%	75	73	2
INST: Apple iTunes 10.6.1-x86	10%	63	43	20

Energieverbrauch



Verbrauch letzte Woche: 1.573 kWh



Durchschnittl. Wochenverbrauch: 1.300 kWh

Wiederkehrende Jobs

Bezeichnung	Fälligkeit / Fortschritt
BGInfo	81%
Avira Antivir 10.0	73%
Basic PC Setup	21%
ImgBurn 2.4.3.0	in 3 Tagen
OpenOffice	in 8 Tagen
.NET Framework 2.0	in 10 Tagen
7-zip 4.65	in 2 Wochen
Excel Viewer 2003	in 2 Wochen
Filezilla 3.2.3.1	in 4 Wochen
INST: Adobe Reader	in 6 Wochen

Managed Software

Bezeichnung	Inst.-Status	Freigabe	Veröffentlicht	Σ
▼ Adobe Flash Player				
▼ Adobe Flash Player 10.3.6x-x86				
Adobe Flash Player 10.3.6.181.014.x-x86	91%	⊘	14.05.2012	
Adobe Flash Player 10.3.6.181.022.x-x86	74%	⊘	22.05.2012	
Adobe Flash Player 10.3.6.181.026.x-x86	60%	💡	26.06.2012	
▸ Adobe Reader	50%	✓ 1 💡 4		1
▸ Adobe Shockwave Player	50%	✓ 2 💡 1		
▸ Apple iPhone Configuration Utility	31%	✓ 1		
▸ Apple iTunes	30%	✓ 1 💡 4		
▸ Apple Quicktime	23%	✓ 3 💡 2		
▸ baramundi Management Suite	10%	✓ 1 💡 4		
▸ INST: Rocket Dock	10%	✓ 1 💡 1		
▸ Label				

Beobachtungsliste

Client	Job	Zustand
GER-A-002	7-zip 4.65	91%
GER-A-005	7-zip 4.65	91%
GER-A-007	7-zip 4.65	88%
GER-A-012	7-zip 4.65	59%
GER-A-013	7-zip 4.65	58%
GER-A-019	7-zip 4.65	31%
GER-A-022	7-zip 4.65	0%

Visuelles Design

Nach einem gemeinsamen Stilfindungsprozess wird anhand der erarbeiteten Vorgaben ein ansprechendes Visuelles Design entwickelt.

Start | Dashboard | Jobs | Clients | Benutzer

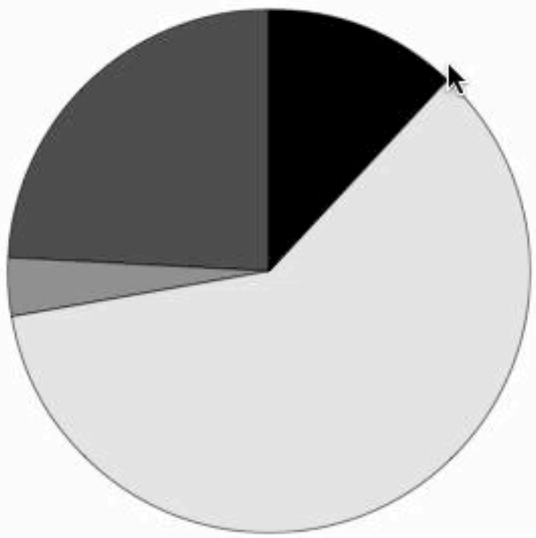
3

Systemübersicht | Office Rollout | Seite 3

Verwalten

Aktivität

Jobs | Clients | Letzte 24 Stunden

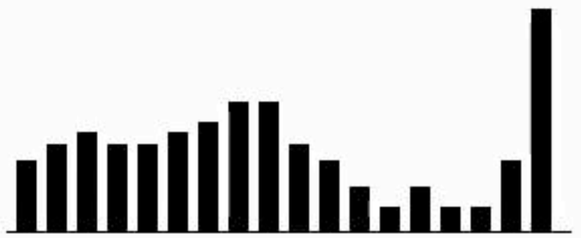


Jobs

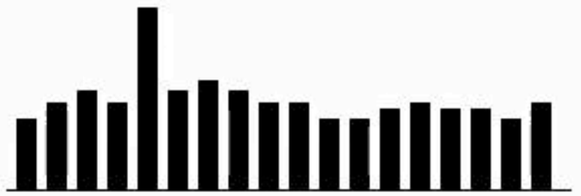
Letzte 24 Stunden | 57 Jobs

Bezeichnung	Zustand	alle	fertig	fehlerhaft
.NET Framework 2.0	97%	120	109	11
7-zip 4.65	95%	74	70	4
Avira Antivir 10.0	81%	365	354	11
Basic PC Setup	78%	118	98	20
BGInfo	74%	37	31	6
Deinst: OpenOffice	50%	88	70	18
Excel Viewer 2003	44%	81	66	15
Filezilla 3.2.3.1	44%	111	97	14
INST: Apple iTunes 10.6.1-x86	31%	65	62	3
ImgBurn 2.4.3.0	30%	21	12	9
INST: Adobe Reader	23%	75	73	2
INST: Apple iTunes 10.6.1-x86	10%	63	43	20
INST: Rocket Dock	10%	17	7	10

Energieverbrauch



Verbrauch letzte Woche: 1.573 kWh



Durchschnitt. Wochenverbrauch: 1.300 kWh

Wiederkehrende Jobs

Bezeichnung	Fälligkeit / Fortschritt
BGInfo	81%
Avira Antivir 10.0	73%
Basic PC Setup	21%
ImgBurn 2.4.3.0	in 3 Tagen
OpenOffice	in 8 Tagen
.NET Framework 2.0	in 10 Tagen
7-zip 4.65	in 2 Wochen
Excel Viewer 2003	in 2 Wochen
Filezilla 3.2.3.1	in 4 Wochen
INST: Adobe Reader	in 6 Wochen

Managed Software

Bezeichnung	Inst.-Status	Freigabe	Veröffentlicht	Σ
▾ Adobe Flash Player				
▾ Adobe Flash Player 10.3.6x-x86				
Adobe Flash Player 10.3.6.181.014.x-x86	91%	⊘	14.05.2012	
Adobe Flash Player 10.3.6.181.022.x-x86	74%	⊘	22.05.2012	
Adobe Flash Player 10.3.6.181.026.x-x86	60%	💡	26.06.2012	
▸ Adobe Reader	50%	✓ 1 💡 4		
▸ Adobe Shockwave Player	50%	✓ 2 💡 1		
▸ Apple iPhone Configuration Utility	31%	✓ 1		
▸ Apple iTunes	30%	✓ 1 💡 4		
▸ Apple Quicktime	23%	✓ 3 💡 2		
▸ baramundi Management Suite	10%	✓ 1 💡 4		
▸ INST: Rocket Dock	10%	✓ 1 💡 1		
▸ Label				
▸ Label				

Beobachtungsliste

Client	Job	Zustand
GER-A-002	7-zip 4.65	91%
GER-A-005	7-zip 4.65	91%
GER-A-007	7-zip 4.65	88%
GER-A-012	7-zip 4.65	59%
GER-A-013	7-zip 4.65	58%

Prototyping

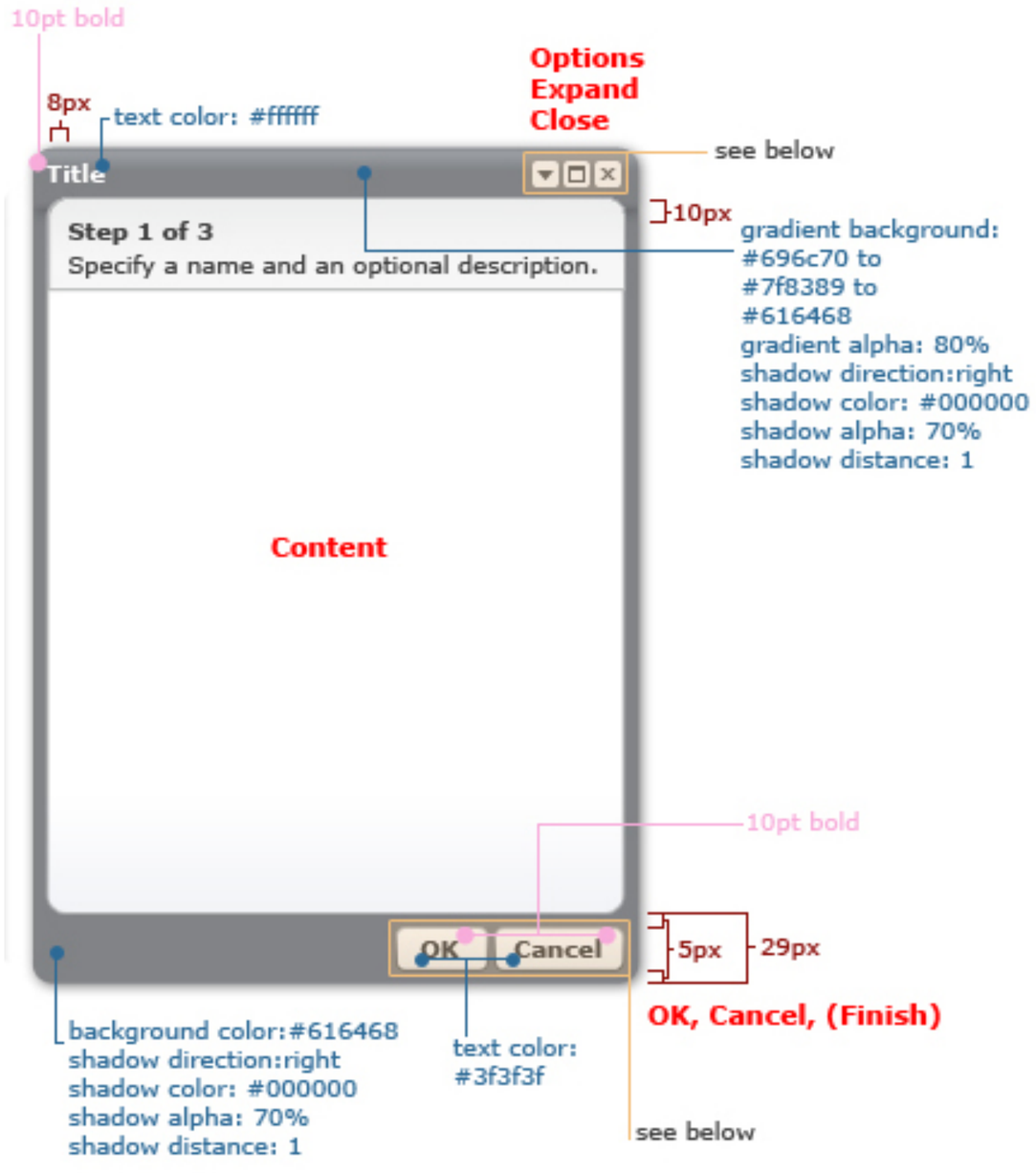
Prototypen sind in verschiedenen Phasen der Gestaltung von Bedeutung, beispielsweise im Rahmen von User- und Task-Analyse, Design, Usability Testing oder zur unterstützenden Dokumentation bei der Spezifikation.

Usability Testing

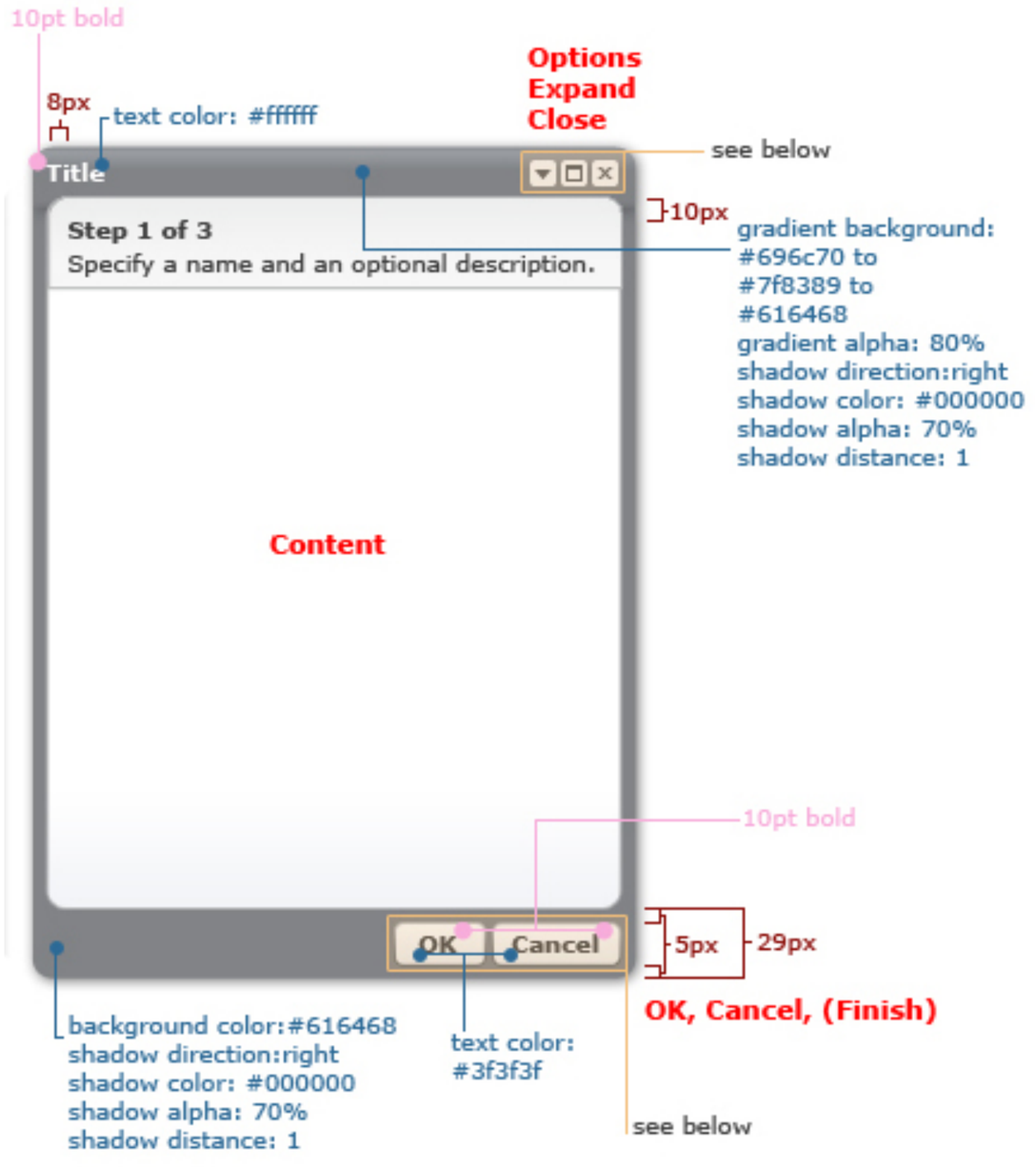
Ein Usability Test mit repräsentativen Endanwendern liefert entscheidende Hinweise zur Optimierung eines User Interface.

SPECIFY

Spezifikation

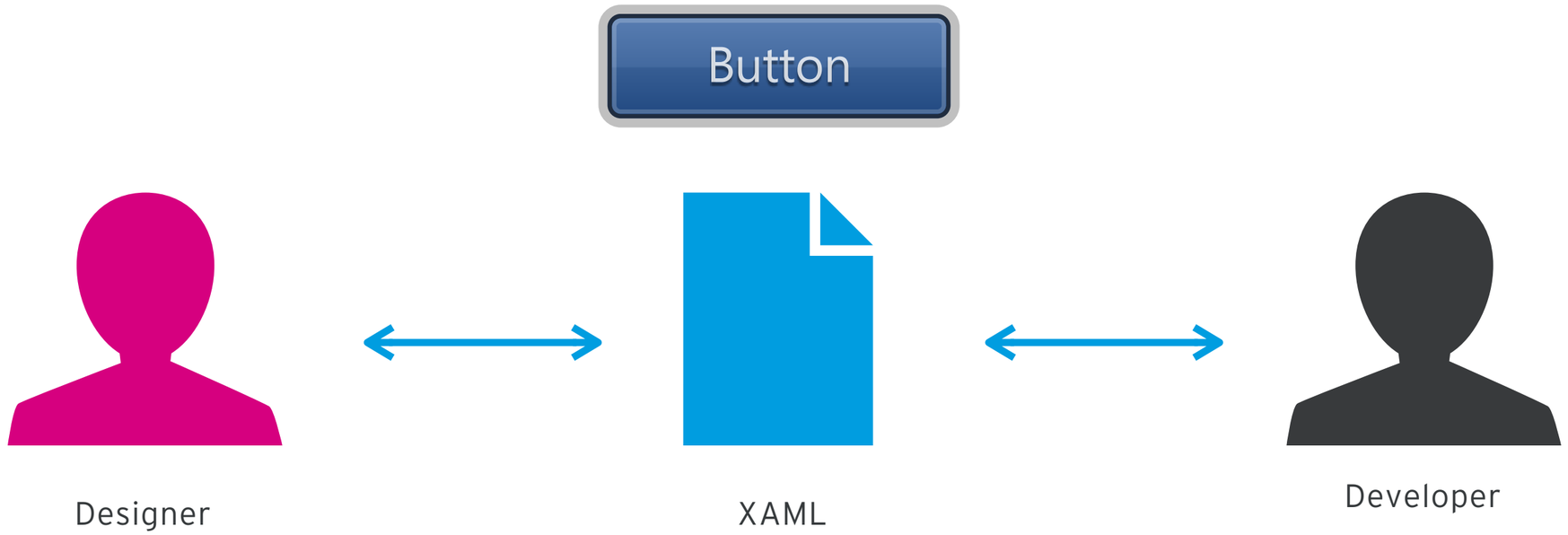


Spezifikation

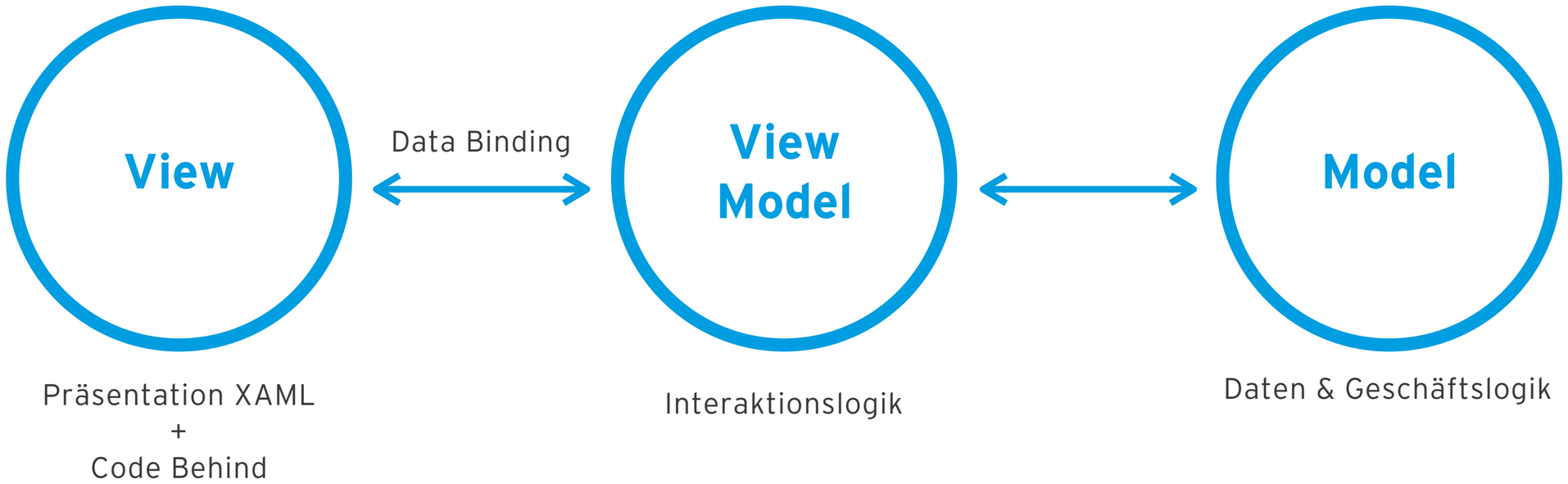


wtf? :-D

Reminder - Deklarative Programmierung mit XAML, Styling, Templates...



Reminder - Model View ViewModel - Klare Trennung Visualisierung & Logik



Probleme

- Komplexität und Umfang der Dokumente (Style Guide, etc.)
- Interpretationsfreiräume des Software Engineers
- Know-how zur Realisierung fehlt
- Allgemein UI Development unterbewertet

UI Developer als Schlüsselelement

Lösung

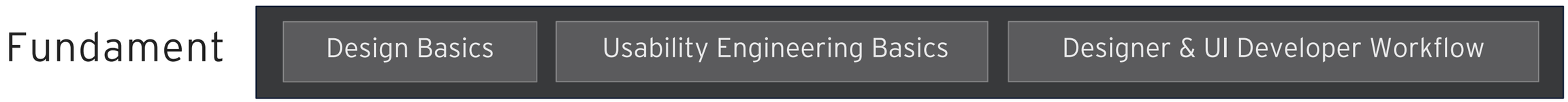
- Rolle „UI Developer“ mit eigener Wissensbasis forcieren

Fokus UI Development

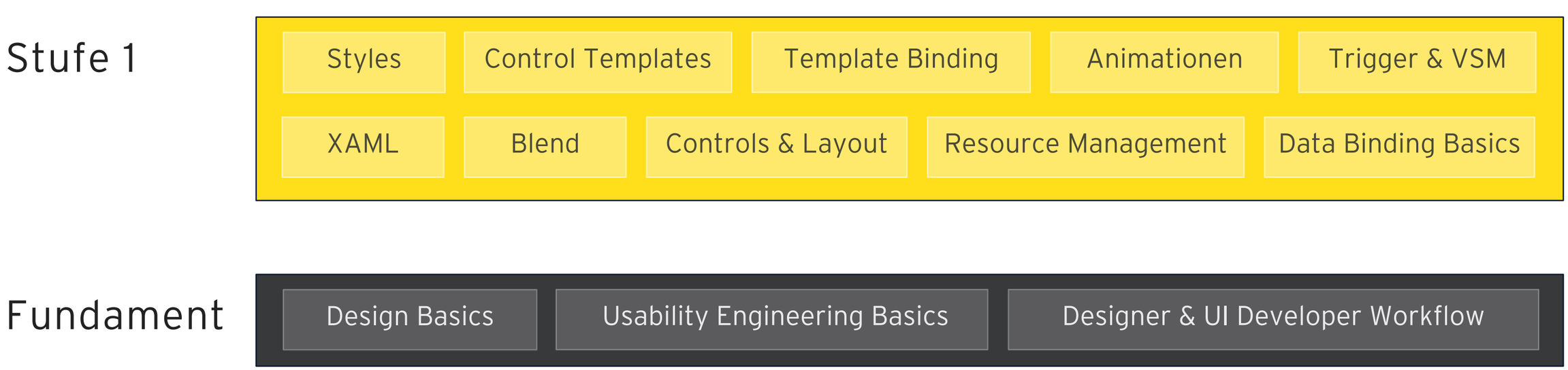
- Zwischen UX Designer und Software Engineer aktiv angesiedelt
- Schicht Präsentation und Interaktion
- Styling, Templating, Layouting, Views, ViewModels, Custom Control Development…

Stufenmodell

Stufenmodell

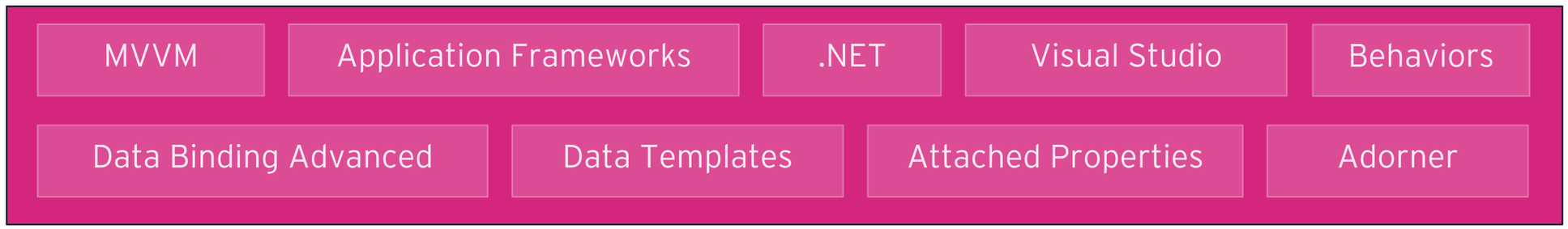


Stufenmodell

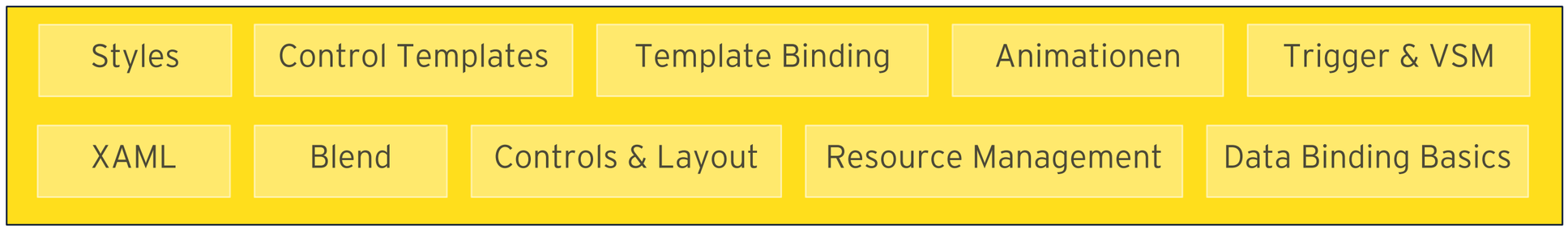


Stufenmodell

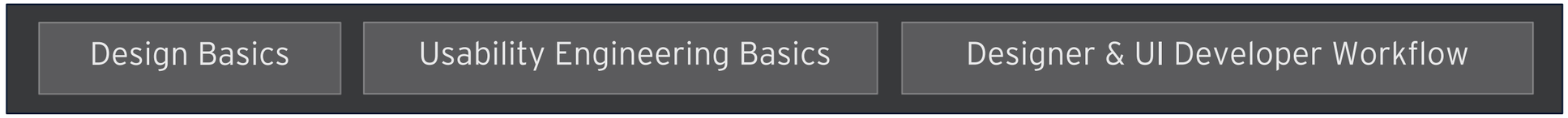
Stufe 2



Stufe 1



Fundament



Stufenmodell

Stufe 3

- Custom Control Development
- Custom Layout Panels
- Miscellaneous

Stufe 2

- MVVM
- Application Frameworks
- .NET
- Visual Studio
- Behaviors
- Data Binding Advanced
- Data Templates
- Attached Properties
- Adorner

Stufe 1

- Styles
- Control Templates
- Template Binding
- Animationen
- Trigger & VSM
- XAML
- Blend
- Controls & Layout
- Resource Management
- Data Binding Basics

Fundament

- Design Basics
- Usability Engineering Basics
- Designer & UI Developer Workflow

Zusammenfassung

- UI Development muss als eigene Disziplin innerhalb des Software Engineerings verstanden werden!
- UX Design ist die **Basis** für UI Development
- Ein benutzer-zentrierter Designprozess ist essentiell!

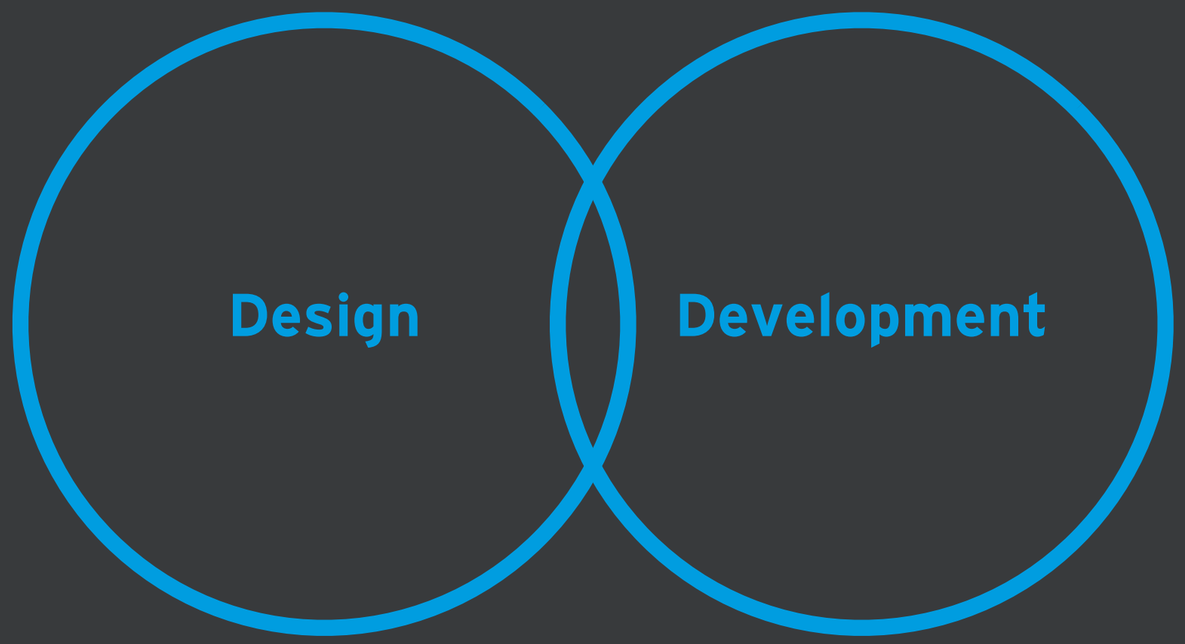
Zusammenfassung

- UI Development muss als eigene Disziplin innerhalb des Software Engineerings verstanden werden!
- UX Design ist die **Basis** für UI Development
- Ein benutzer-zentrierter Designprozess ist essentiell!

User Centered Design
+ Software Engineering
+ UI Development

= Application! ;-)

DESIGN UND DEVELOPMENT IM EINKLANG

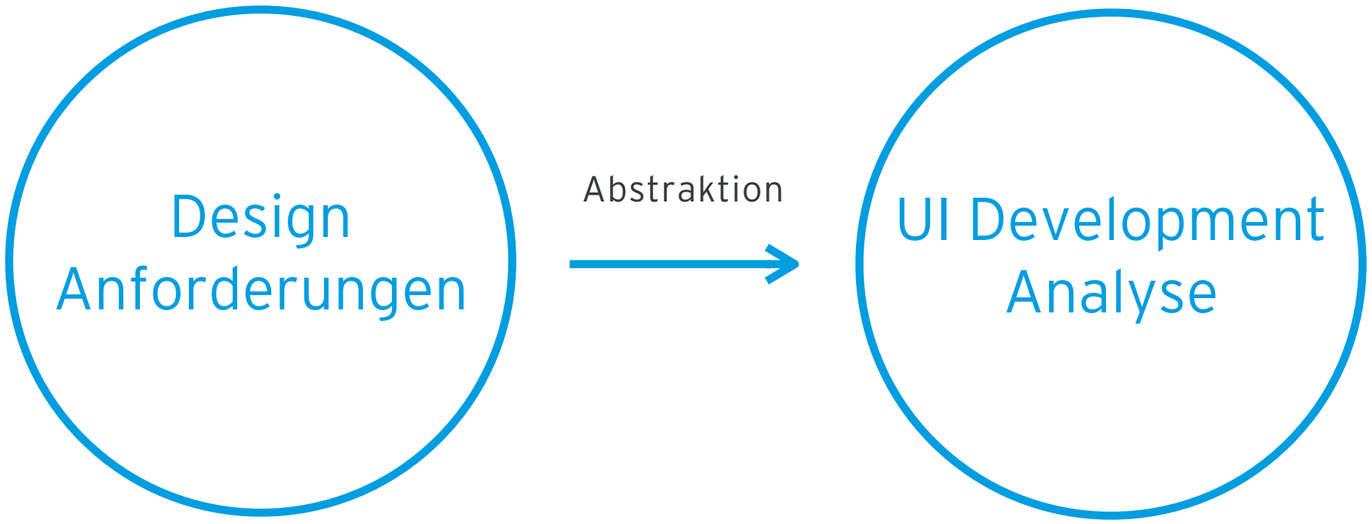


Eigene Prozesse

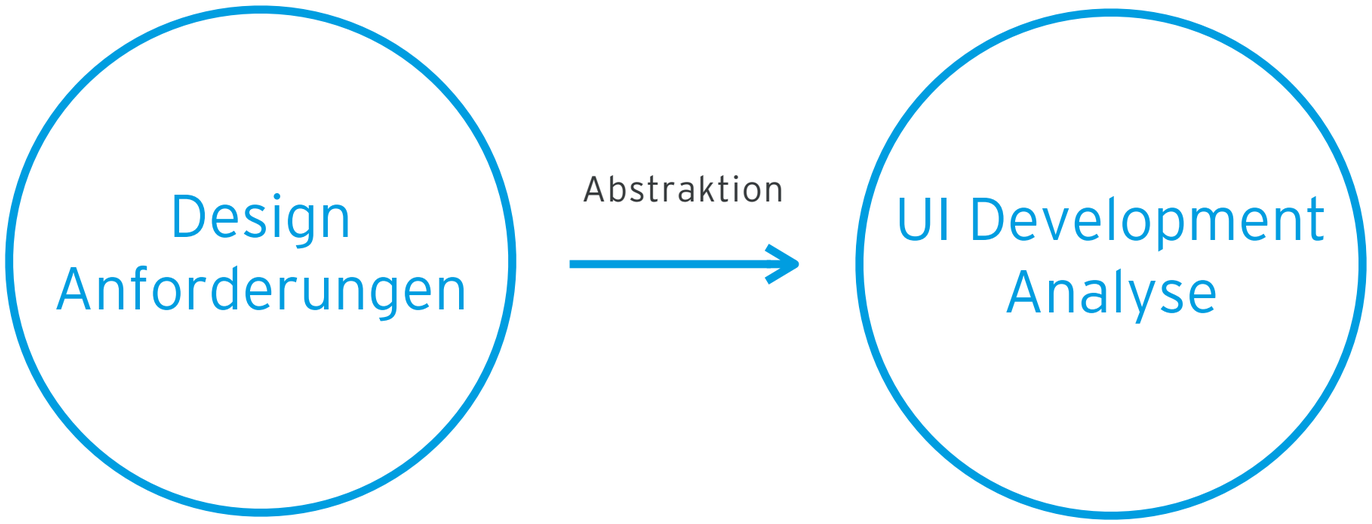
Abstraktes Vorgehen UI Developer - Styling und Custom Control Development

Abstraktes Vorgehen UI Developer - Styling und Custom Control Development

Abstraktes Vorgehen UI Developer - Styling und Custom Control Development



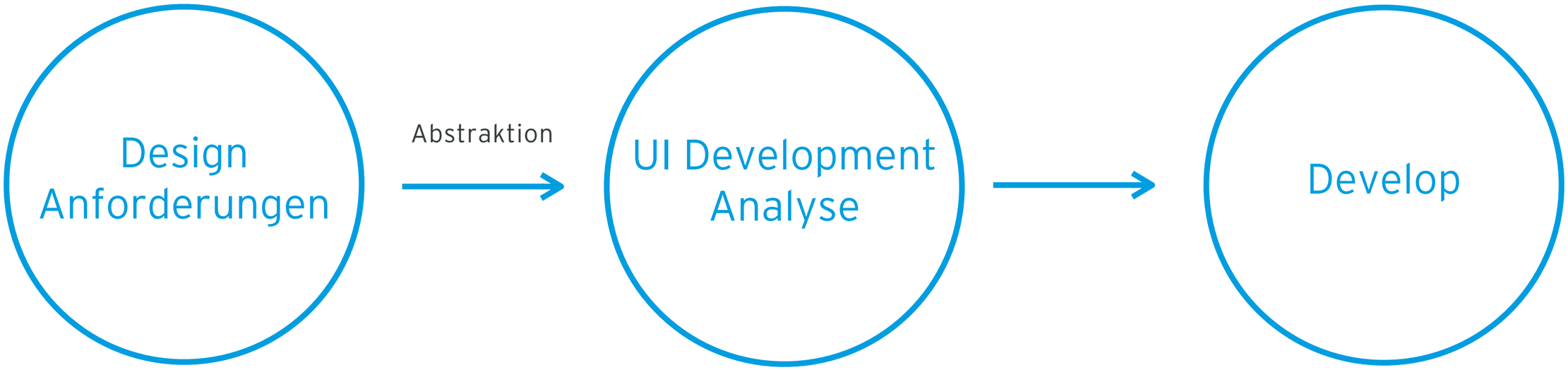
Abstraktes Vorgehen UI Developer - Styling und Custom Control Development



Identifikation

- Standard UI Elemente
- Custom Controls
  - Anforderungsanalyse pro Custom Control
  - Funktionale Anforderungen, Properties, Events, ...
- Views

Abstraktes Vorgehen UI Developer - Styling und Custom Control Development

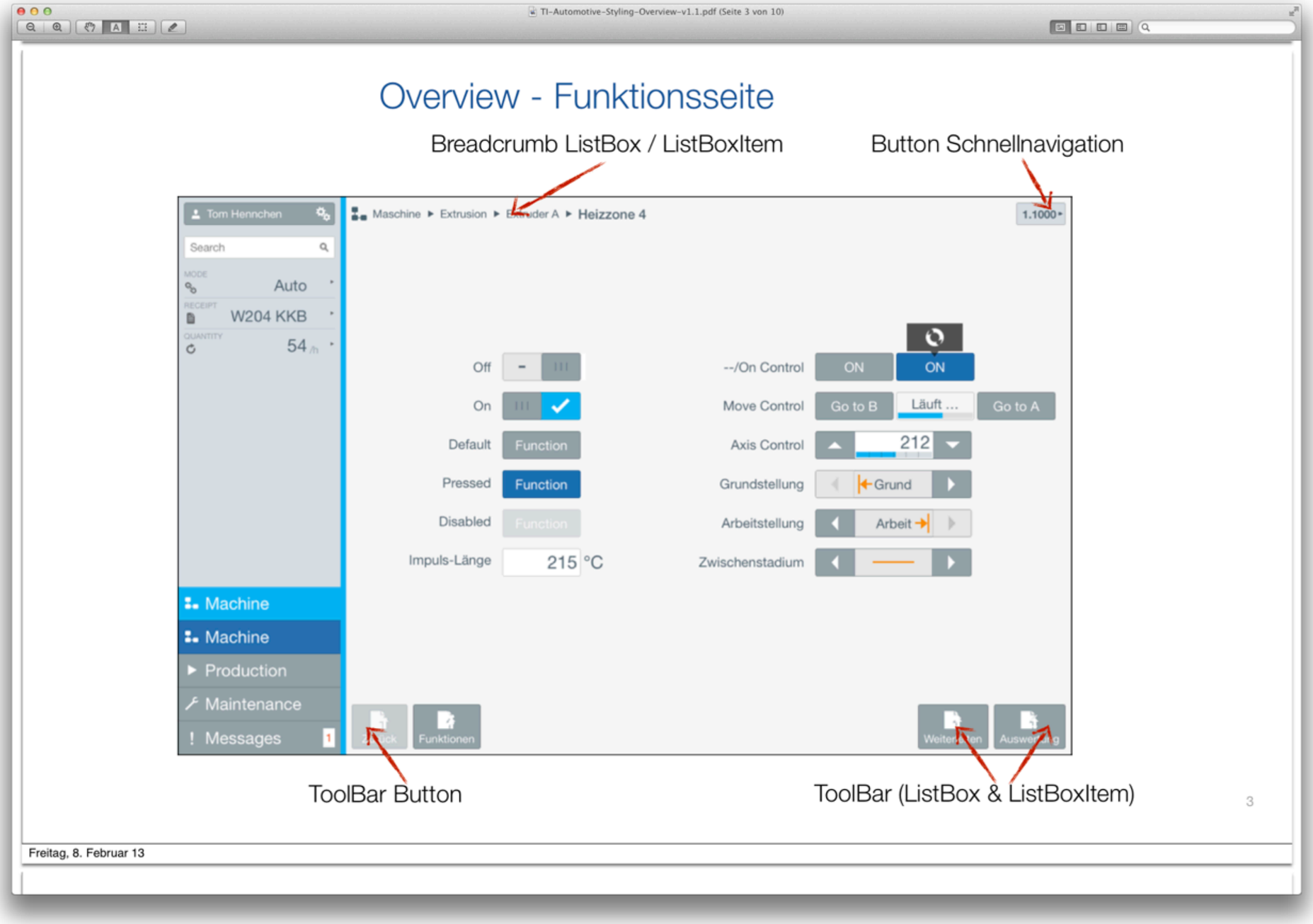


Identifikation

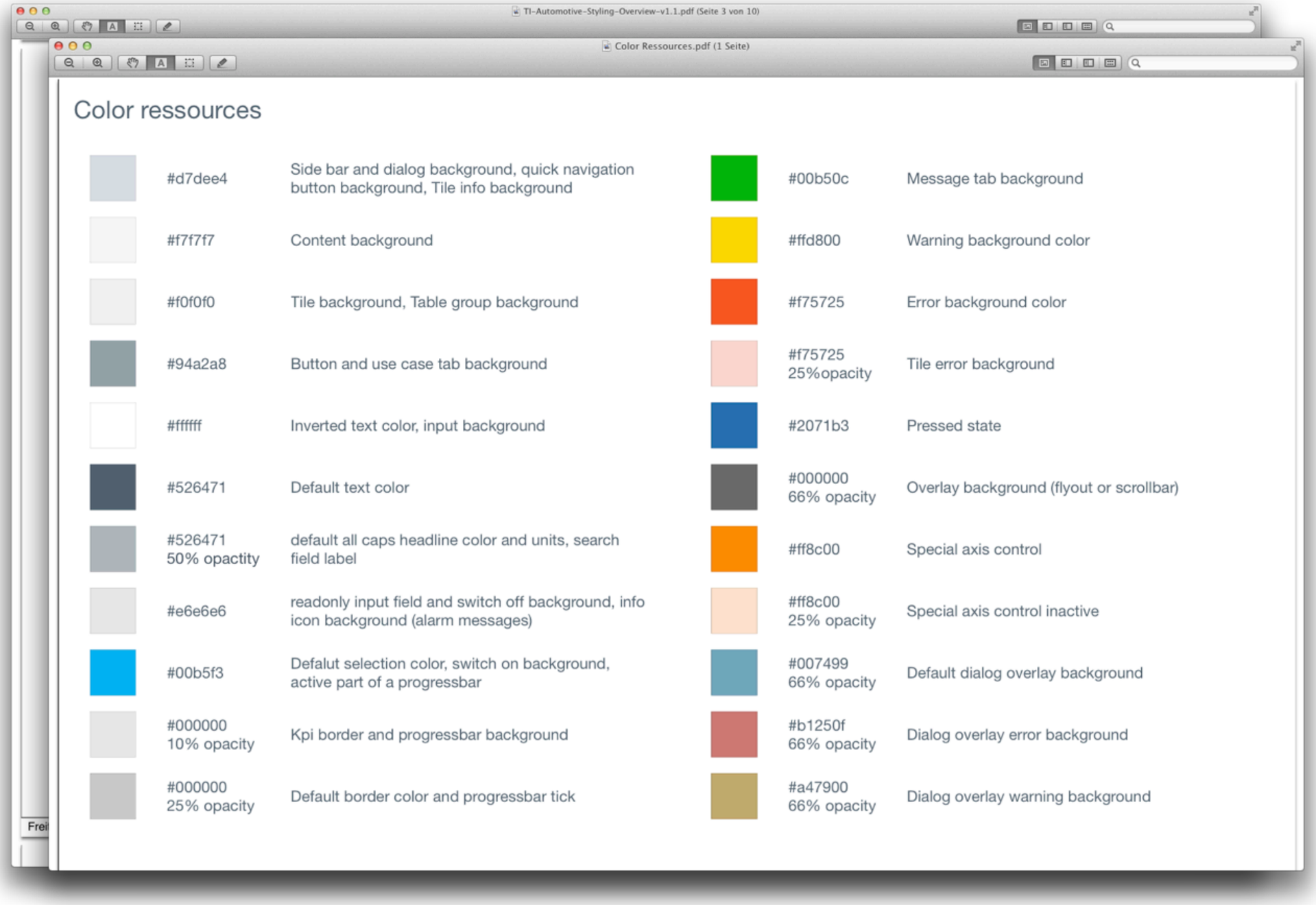
- Standard UI Elemente
- Custom Controls
  - Anforderungsanalyse pro Custom Control
  - Funktionale Anforderungen, Properties, Events, ...
- Views

Designer und UI Developer Schnittstellen...

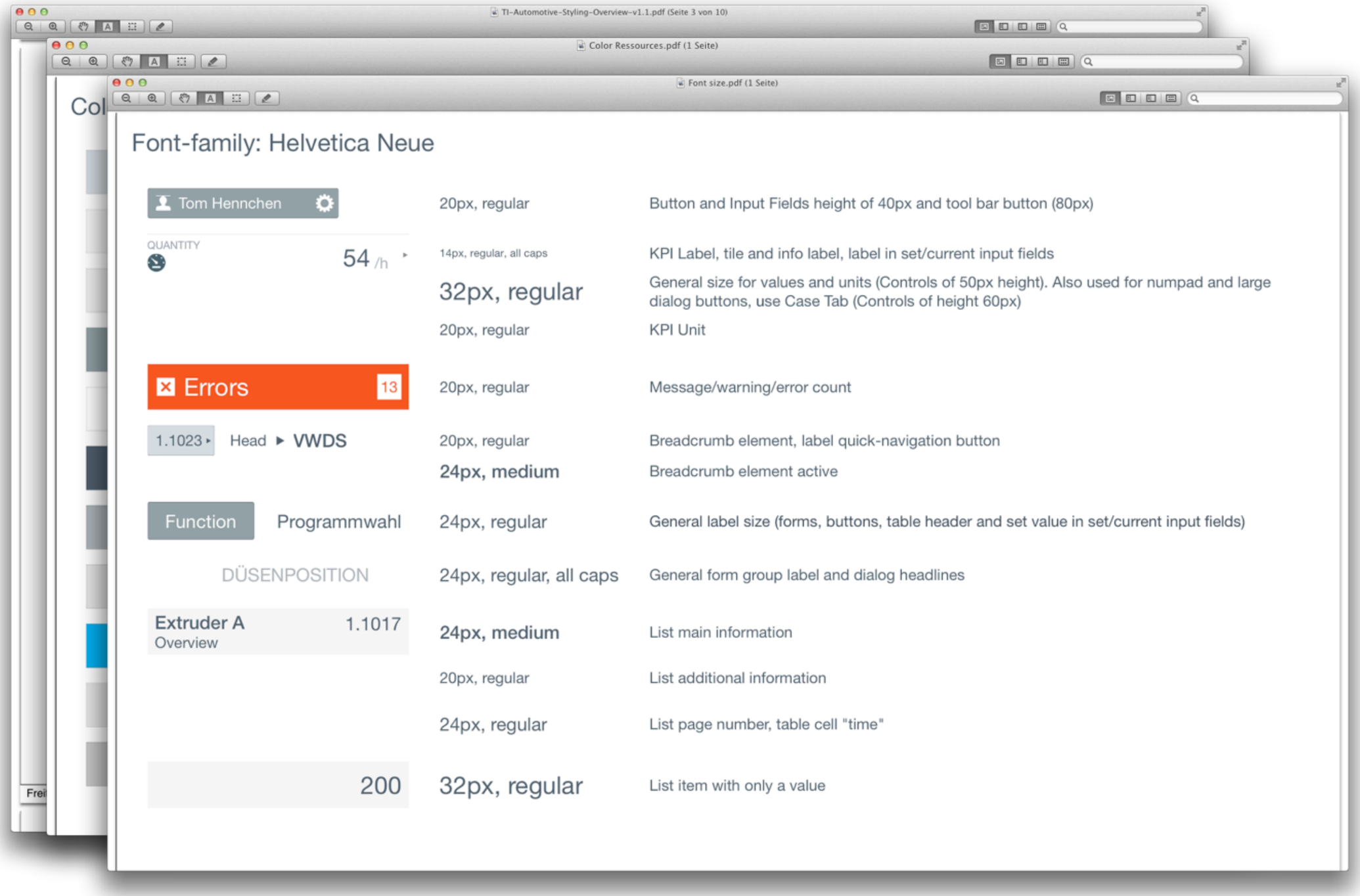
Designer und UI Developer Schnittstellen...



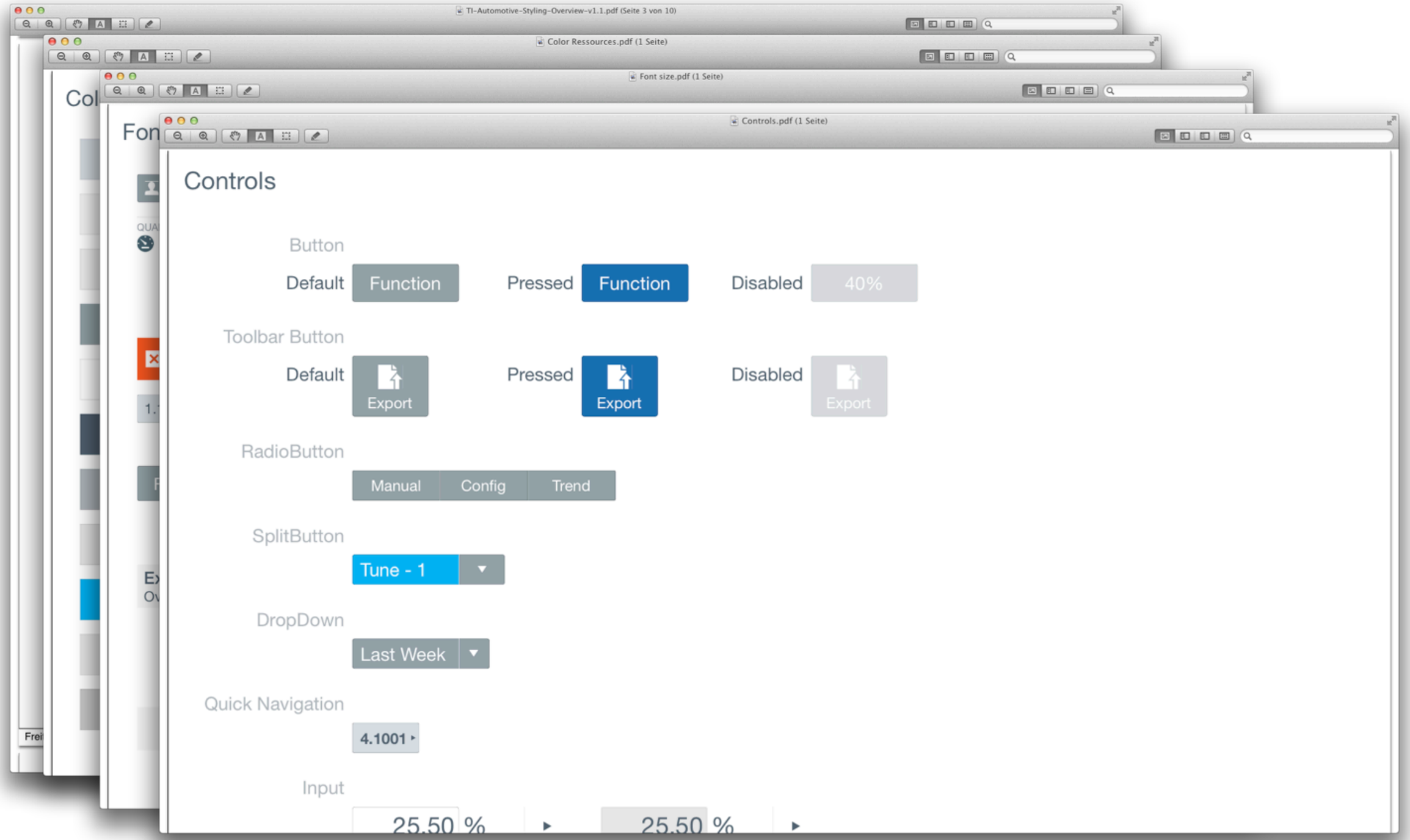
Designer und UI Developer Schnittstellen...



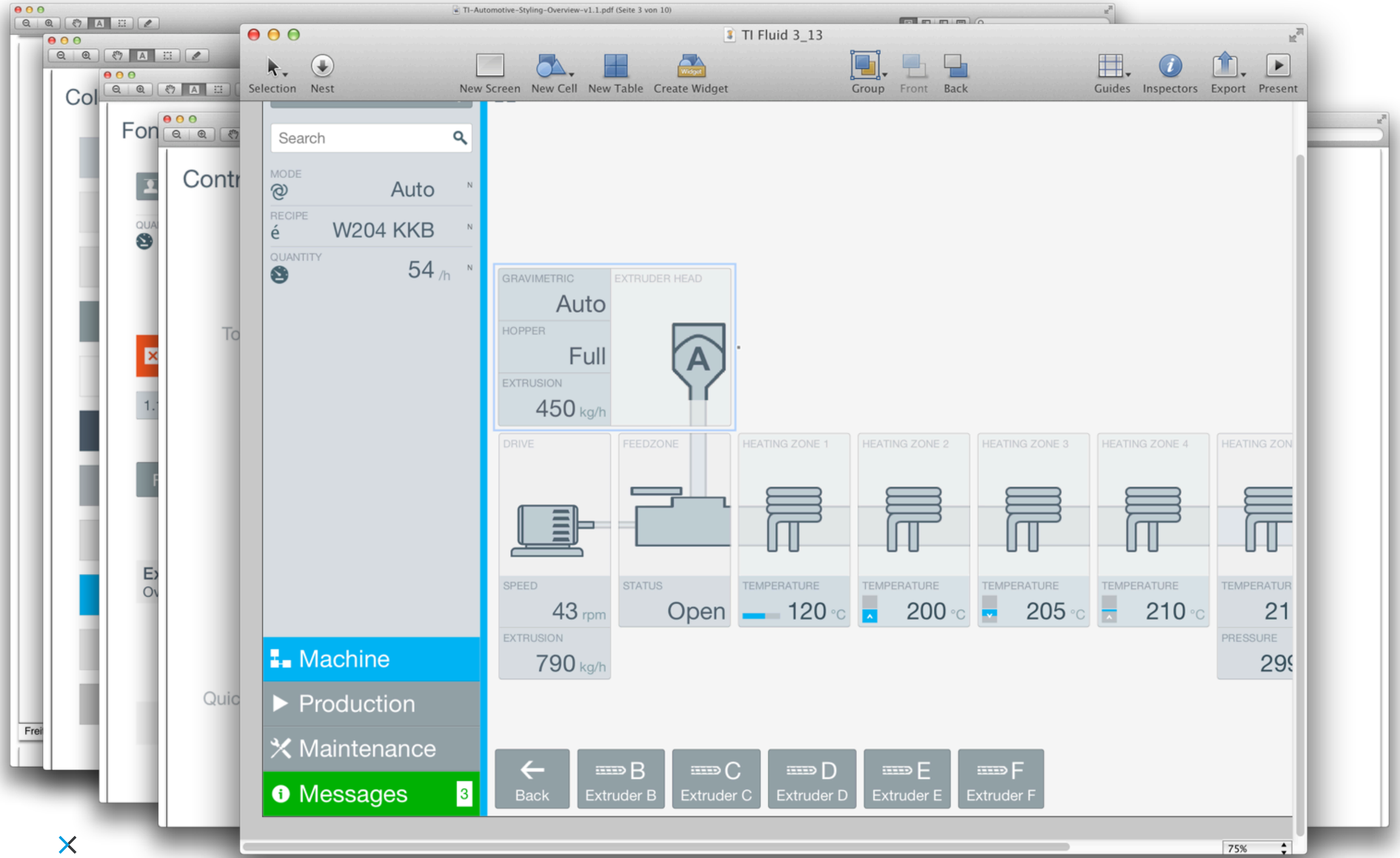
Designer und UI Developer Schnittstellen…



Designer und UI Developer Schnittstellen...



Designer und UI Developer Schnittstellen...



ERGOSIGN

EIN UI...

André Lanninger
UIDeveloper

Datagraph 11

ChangeStyle

Load

62

- Bubble 18
  0, 20
- Bubble 19
  10, 15
- Bubble 20
  20, **62**
- Bubble 21
  30, 10
- Bubble 22
  40, 27
- Bubble 23
  50, 77
- Bubble 24
  60, 73
- Bubble 25
  70, 90
- Bubble 26
  80, 33

Styling?

Custom Controls?

DER WEG ZUM CUSTOM CONTROL

Reminder - Eigentlich “nur”

- Ableitung von einer konkreten **Klasse != UserControl**
- Styling- und Template- Möglichkeiten
- Visueller Aufbau im Control Template
- Default Style möglich Generic.xaml
- Zusammenfassung in einer Control Library möglich

WPF Standard Control?

WPF Standard Control?

WPF Standard Controls gruppieren?

WPF Standard Control?

WPF Standard Controls gruppieren?

Styling & Templating?

WPF Standard Control?

WPF Standard Controls gruppieren?

Styling & Templating?

ValueConverter/Markup Extensions?

WPF Standard Control?

WPF Standard Controls gruppieren?

Styling & Templating?

ValueConverter/Markup Extensions?

Attached Properties/Behaviors?

WPF Standard Control?

WPF Standard Controls gruppieren?

Styling & Templating?

ValueConverter/Markup Extensions?

Attached Properties/Behaviors?

Drittanbieter?

WPF Standard Control?

WPF Standard Controls gruppieren?

Styling & Templating?

ValueConverter/Markup Extensions?

Attached Properties/Behaviors?

Drittanbieter?

vs.

My Custom Control

Weitere Faktoren zur Entscheidung Custom Control

- Prototyp vs. Produktiv-Code
- Wiederverwendbarkeit
- Konsistenz im User Interface

André Lanninger
UIDeveloper

Datagraph 11

ChangeStyle

Load

62

- Bubble 18
  0, 20
- Bubble 19
  10, 15
- Bubble 20
  20, **62**
- Bubble 21
  30, 10
- Bubble 22
  40, 27
- Bubble 23
  50, 77
- Bubble 24
  60, 73
- Bubble 25
  70, 90
- Bubble 26
  80, 33

SearchTextBox,

aber nicht als Custom Control!

SEARCHTEXTBOX

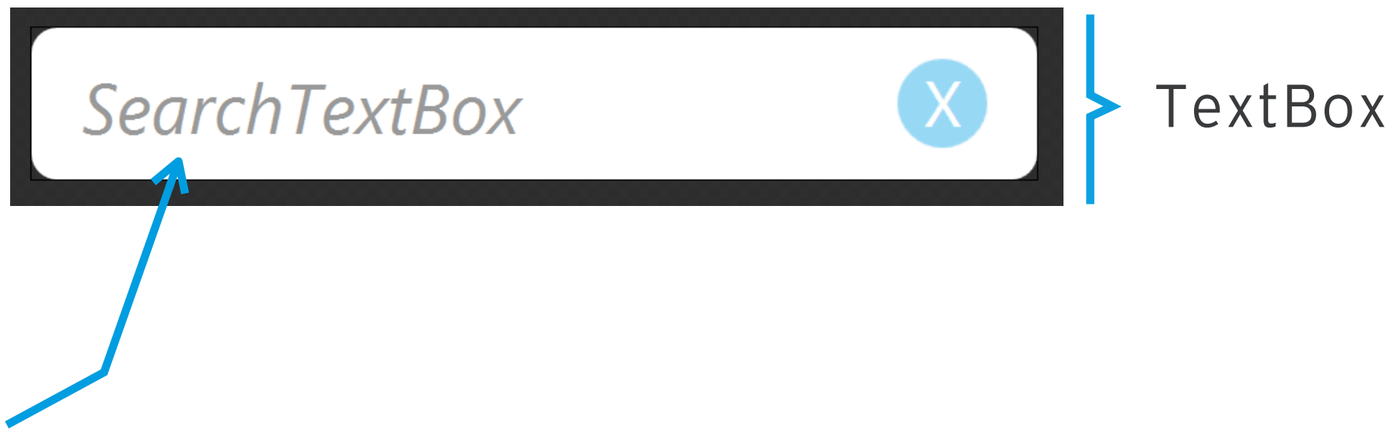
ANALYSE

View Model

View Model

View Model

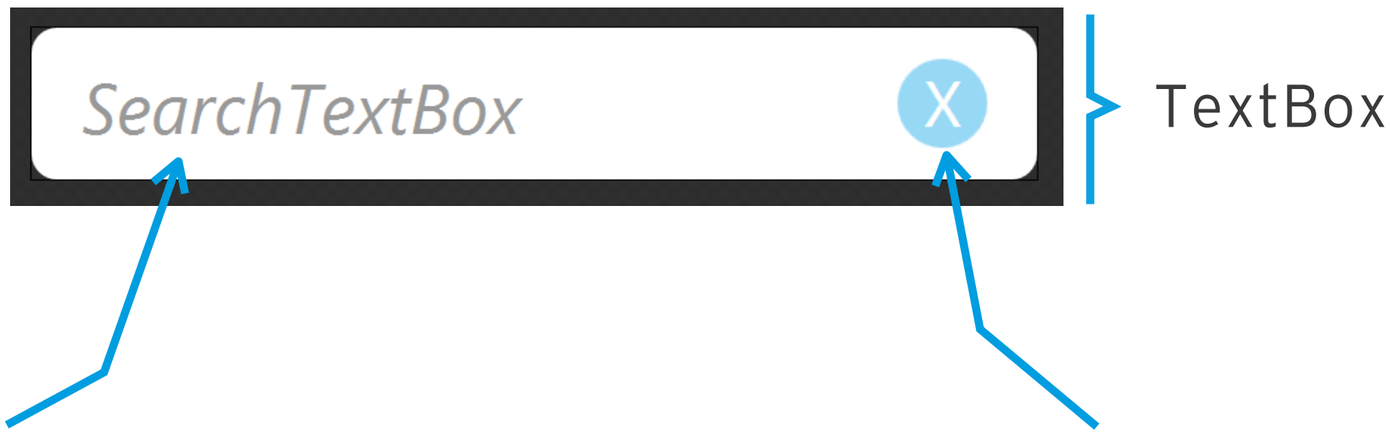
View Model



View Model

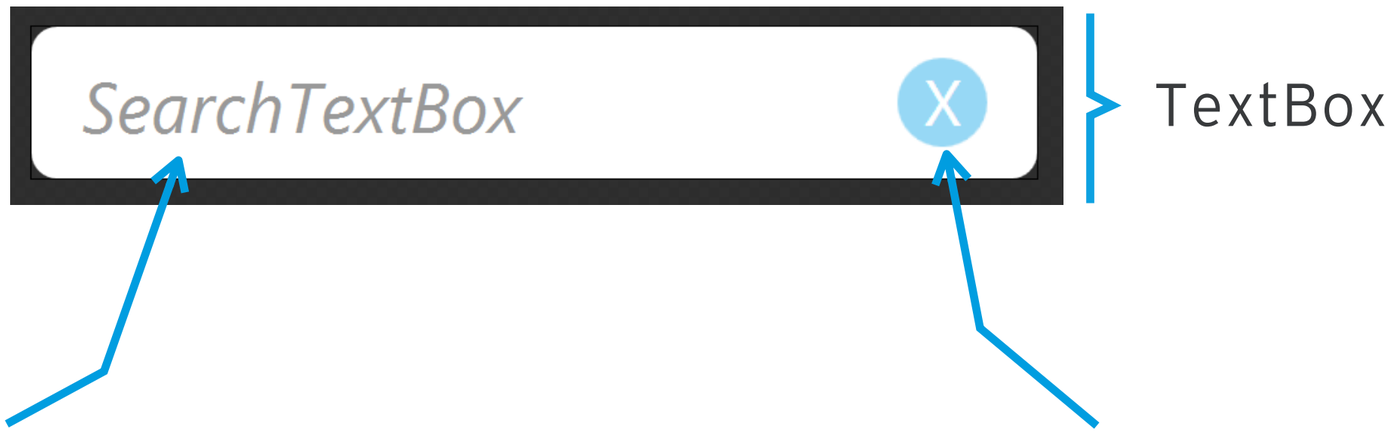
- Watermark
- WatermarkFontStyle
- WatermarkFontWeight
- WatermarkForeground

View Model



- Watermark
- WatermarkFontStyle
- WatermarkFontWeight
- WatermarkForeground

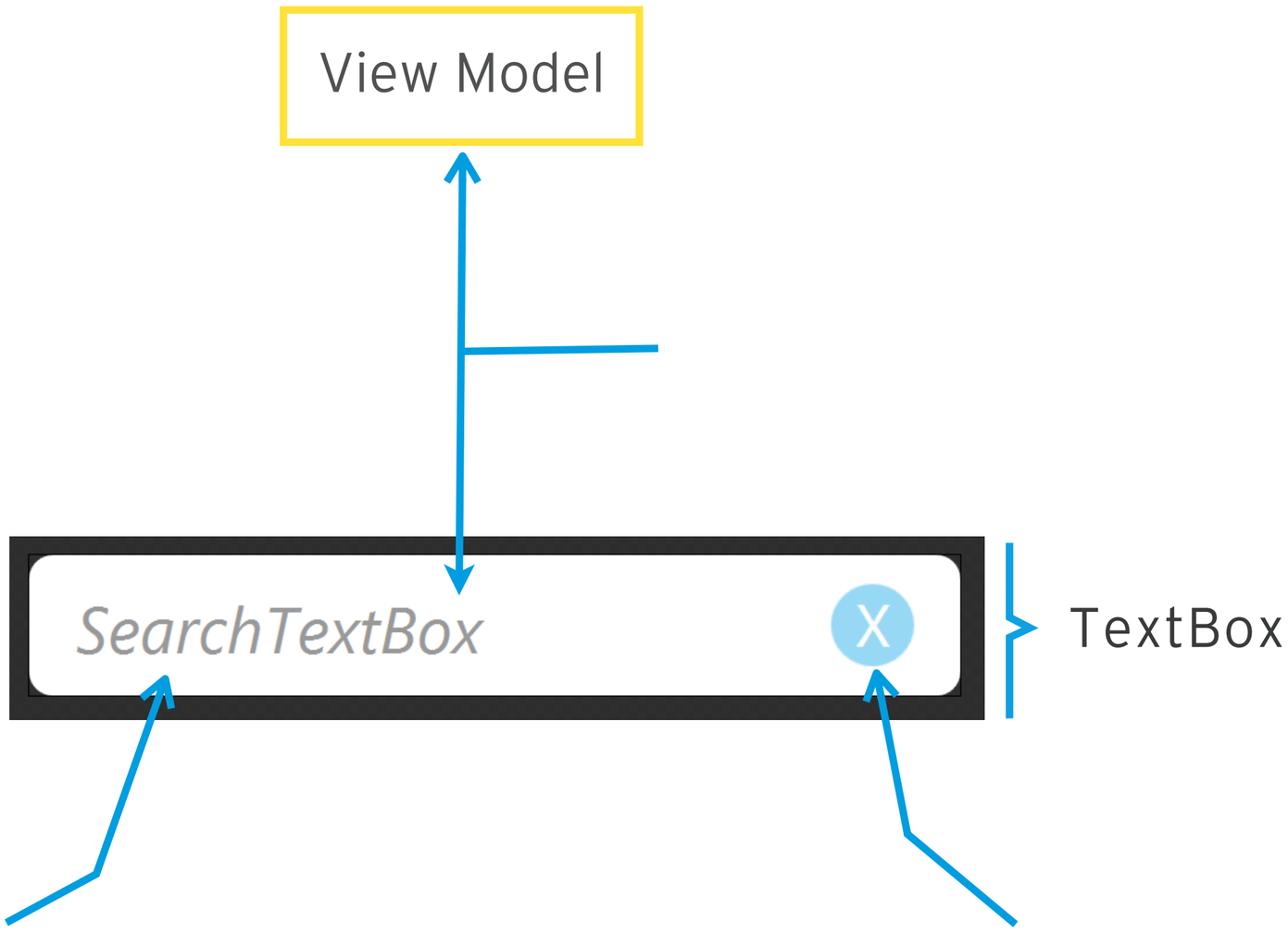
View Model



- Watermark
- WatermarkFontStyle
- WatermarkFontWeight
- WatermarkForeground

ClearCommand

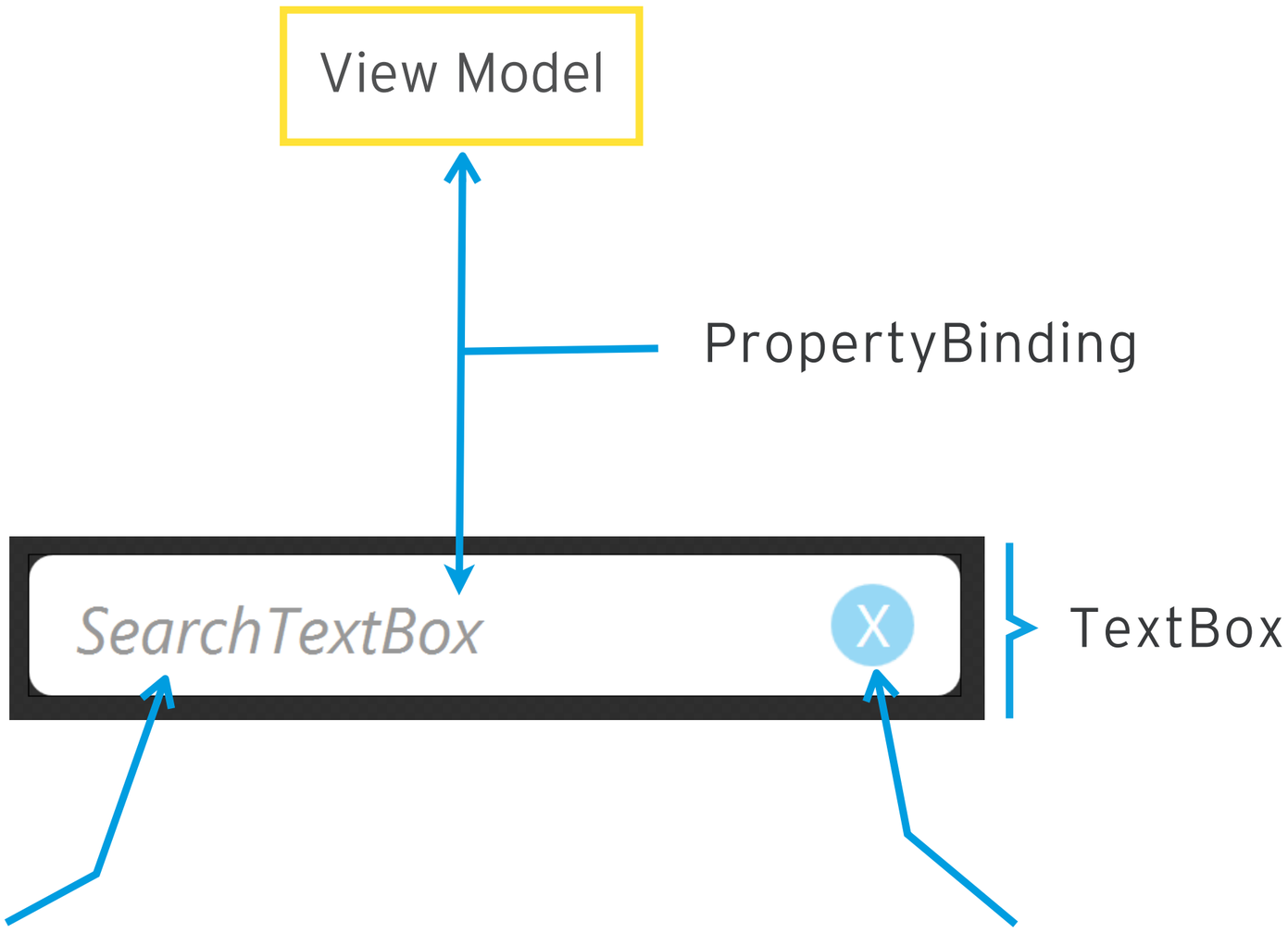
(Zurücksetzen des Texts)



- Watermark
- WatermarkFontStyle
- WatermarkFontWeight
- WatermarkForeground

ClearCommand

(Zurücksetzen des Texts)



- Watermark
- WatermarkFontStyle
- WatermarkFontWeight
- WatermarkForeground

ClearCommand

(Zurücksetzen des Texts)

STYLE & TEMPLATE

```
<Style x:Key="WatermarkAttachedTextBoxStyle" TargetType="{x:Type TextBox}" BasedOn="{StaticResource FrameworkElementStyle}">
    <Setter Property="Height" Value="30" />
    <Setter Property="Foreground" Value="{StaticResource HeaderBrush}" />
    <Setter Property="FontSize" Value="{StaticResource Layer4FontSize}" />
    <Setter Property="FontFamily" Value="{StaticResource PrimaryFontFamily}" />
    <Setter Property="FontWeight" Value="{StaticResource RegularFontWeight}" />
    <Setter Property="Background" Value="{StaticResource WhiteBrush}" />
    <Setter Property="HorizontalContentAlignment" Value="Stretch" />
    <Setter Property="VerticalContentAlignment" Value="center" />
    <Setter Property="Padding" Value="10 0" />
    <Setter Property="AllowDrop" Value="true" />
    <Setter Property="ScrollViewer.PanningMode" Value="VerticalFirst" />
    <Setter Property="Stylus.IsFlicksEnabled" Value="False" />
    <Setter Property="AttachedProperties.WatermarkFontStyle" Value="Italic" />
    <Setter Property="AttachedProperties.WatermarkFontWeight" Value="{Binding FontWeight, RelativeSource={RelativeSource Self}}" />
    <Setter Property="AttachedProperties.WatermarkForeground" Value="{Binding Foreground, RelativeSource={RelativeSource Self}}" />
    <Setter Property="Template">
        <Setter.Value>
            <ControlTemplate TargetType="{x:Type TextBox}">
                <Border x:Name="Bd"
                        Background="{TemplateBinding Background}"
                        BorderBrush="{TemplateBinding BorderBrush}"
                        BorderThickness="{TemplateBinding BorderThickness}"
                        CornerRadius="5"
                        SnapsToDevicePixels="true">
                    <Grid>
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition />
                            <ColumnDefinition Width="auto" />
                        </Grid.ColumnDefinitions>
                        <Grid x:Name="TextContainer">
                            <TextBlock x:Name="Watermark"
                                       Margin="{TemplateBinding Padding}"
                                       HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}"
                                       VerticalAlignment="{TemplateBinding VerticalContentAlignment}"
                                       FontStyle="{TemplateBinding AttachedProperties.WatermarkFontStyle}"
                                       FontWeight="{TemplateBinding AttachedProperties.WatermarkFontWeight}"
                                       Foreground="{TemplateBinding AttachedProperties.WatermarkForeground}"
                                       Opacity=".5"
                                       Text="{TemplateBinding AttachedProperties.Watermark}"
                                       Visibility="Hidden" />
                            <ScrollViewer x:Name="PART_ContentHost" SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" />
                        </Grid>
                        <Button x:Name="ClearButton"
                                Grid.Column="1"
                                Width="17"
                                Height="17"
                                Margin="0 0 5 0"
                                Command="{TemplateBinding AttachedProperties.ClearCommand}"
                                CommandParameter="{Binding RelativeSource={RelativeSource TemplatedParent}}"
                                Content="X"
                                Focusable="False"
                                Padding="0 -2 0 0"
                                Style="{StaticResource AttachedSearchTextBoxButtonStyle}" />
                    </Grid>
                </Border>
                <ControlTemplate.Triggers>
                    <MultiTrigger>
```

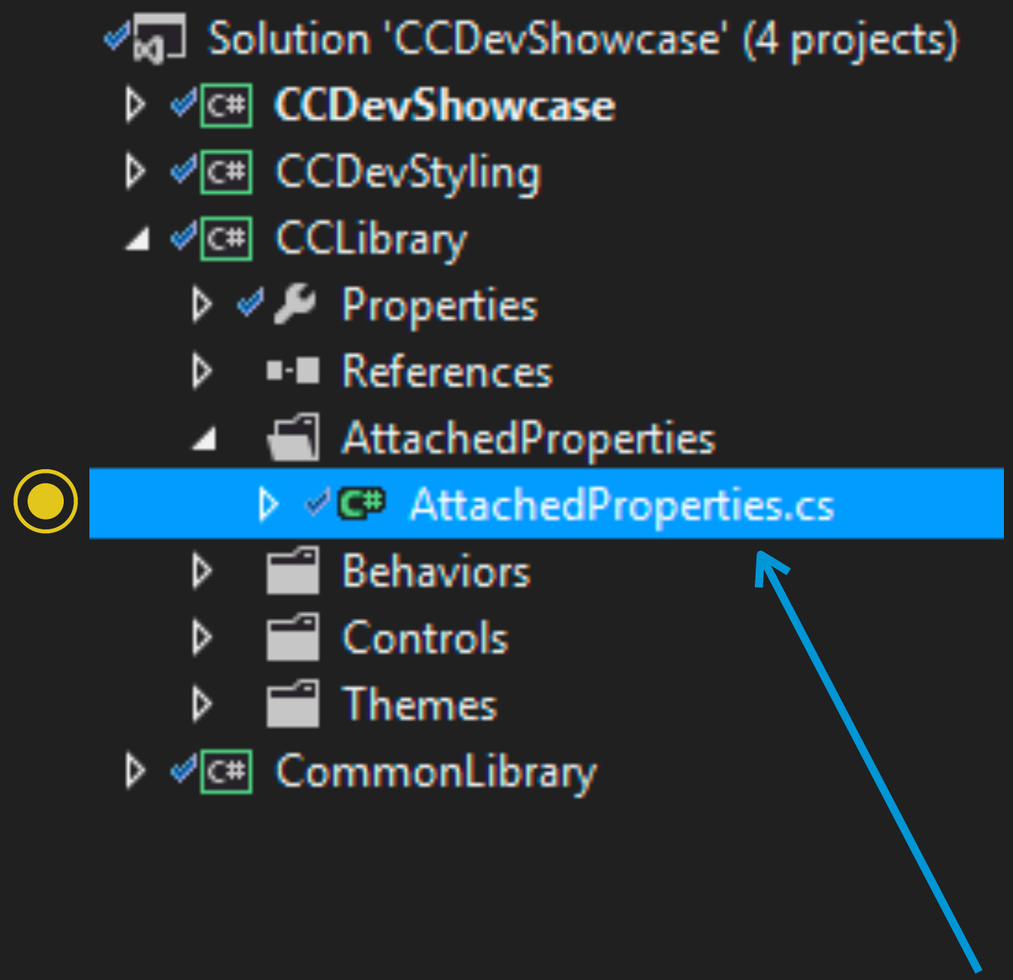
Style & Template

(TextBox.xaml)

ATTACHED PROPERTIES

Attached Properties

- Gleich wie Dependency Properties
  - Metadaten und Speicherperformance, Änderungsbenachrichtigungen, Grundlage für Trigger, Data Binding, Animationen etc.
- Definition nicht für eine bestimmtes Control sondern global in einer zentralen Klasse für alle Controls
- Möglichkeit auf einem beliebigen Control zusetzen
- Häufig verwendet bei Layout Panels
  - *Grid.Row*, *Grid.Column*, …



- Zentrale Klasse anlegen

Attached Properties anlegen 1/4

```
/// <summary> ...
public static readonly DependencyProperty WatermarkProperty = DependencyProperty.RegisterAttached(
    "Watermark",
    typeof(string),
    typeof(AttachedProperties),
    new FrameworkPropertyMetadata(string.Empty));

/// <summary> ...
public static readonly DependencyProperty WatermarkFontStyleProperty = DependencyProperty.RegisterAttached(
    "WatermarkFontStyle",
    typeof(FontStyle),
    typeof(AttachedProperties),
    new FrameworkPropertyMetadata(FontStyles.Italic));

/// <summary> ...
public static readonly DependencyProperty WatermarkFontWeightProperty = DependencyProperty.RegisterAttached(
    "WatermarkFontWeight",
    typeof(FontWeight),
    typeof(AttachedProperties),
    new FrameworkPropertyMetadata(FontWeights.Bold));

/// <summary> ...
public static readonly DependencyProperty WatermarkForegroundProperty = DependencyProperty.RegisterAttached(
    "WatermarkForeground",
    typeof(Brush),
    typeof(AttachedProperties),
    new FrameworkPropertyMetadata(Brushes.Black));
```

- Registrieren über statische Funktion
  - Name
  - Type
  - Zugehörige Klasse
  - Metadaten

Attached Properties anlegen 2/4

(AttachedProperties.cs)

```
/// <summary> ...
[Description("Gets the Watermark."), Category(PROPERTY_CATEGORY)]
public static string GetWatermark(UIElement element)
{
    return (string)element.GetValue(WatermarkProperty);
}

/// <summary> ...
[Description("Sets the Watermark."), Category(PROPERTY_CATEGORY)]
public static void SetWatermark(UIElement element, string value)
{
    element.SetValue(WatermarkProperty, value);
}

/// <summary> ...
[Description("Gets the WatermarkFontStyle."), Category(PROPERTY_CATEGORY)]
public static FontStyle GetWatermarkFontStyle(UIElement element)
{
    return (FontStyle)element.GetValue(WatermarkFontStyleProperty);
}

/// <summary> ...
[Description("Sets the WatermarkFontStyle."), Category(PROPERTY_CATEGORY)]
public static void SetWatermarkFontStyle(UIElement element, FontStyle value)
{
    element.SetValue(WatermarkFontStyleProperty, value);
}

/// <summary> ...
[Description("Gets the WatermarkFontWeight."), Category(PROPERTY_CATEGORY)]
public static FontWeight GetWatermarkFontWeight(UIElement element)
{
    return (FontWeight)element.GetValue(WatermarkFontWeightProperty);
}

/// <summary> ...
[Description("Sets the WatermarkFontWeight."), Category(PROPERTY_CATEGORY)]
public static void SetWatermarkFontWeight(UIElement element, FontWeight value)
{
    element.SetValue(WatermarkFontWeightProperty, value);
```

- Statische Funktionen für den Zugriff
- Verwendung der Methoden GetValue(), SetValue()

Attached Properties anlegen 3/4

(AttachedProperties.cs)

```
<Style x:Key="WatermarkAttachedTextBoxStyle" TargetType="{x:Type TextBox}" BasedOn="{StaticResource FrameworkElementStyle}">
    <Setter Property="Height" Value="30" />
    <Setter Property="Foreground" Value="{StaticResource HeaderBrush}" />
    <Setter Property="FontSize" Value="{StaticResource Layer4FontSize}" />
    <Setter Property="FontFamily" Value="{StaticResource PrimaryFontFamily}" />
    <Setter Property="FontWeight" Value="{StaticResource RegularFontWeight}" />
    <Setter Property="Background" Value="{StaticResource WhiteBrush}" />
    <Setter Property="HorizontalContentAlignment" Value="Stretch" />
    <Setter Property="VerticalContentAlignment" Value="center" />
    <Setter Property="Padding" Value="10 0" />
    <Setter Property="AllowDrop" Value="true" />
    <Setter Property="ScrollViewer.PanningMode" Value="VerticalFirst" />
    <Setter Property="Stylus.IsFlicksEnabled" Value="False" />
    <Setter Property="AttachedProperties.WatermarkFontStyle" Value="Italic" />
    <Setter Property="AttachedProperties.WatermarkFontWeight" Value="{Binding FontWeight, RelativeSource={RelativeSource Self}}" />
    <Setter Property="AttachedProperties.WatermarkForeground" Value="{Binding Foreground, RelativeSource={RelativeSource Self}}" />
    <Setter Property="Template">
        <Setter.Value>
            <ControlTemplate TargetType="{x:Type TextBox}">
                <Border x:Name="Bd"
                        Background="{TemplateBinding Background}"
                        BorderBrush="{TemplateBinding BorderBrush}"
                        BorderThickness="{TemplateBinding BorderThickness}"
                        CornerRadius="5"
                        SnapsToDevicePixels="true">
                    <Grid>
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition />
                            <ColumnDefinition Width="auto" />
                        </Grid.ColumnDefinitions>
                        <Grid x:Name="TextContainer">
                            <TextBlock x:Name="Watermark"
                                       Margin="{TemplateBinding Padding}"
                                       HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}"
                                       VerticalAlignment="{TemplateBinding VerticalContentAlignment}"
                                       FontStyle="{TemplateBinding AttachedProperties.WatermarkFontStyle}"
                                       FontWeight="{TemplateBinding AttachedProperties.WatermarkFontWeight}"
                                       Foreground="{TemplateBinding AttachedProperties.WatermarkForeground}"
                                       Opacity=".5"
                                       Text="{TemplateBinding AttachedProperties.Watermark}"
                                       Visibility="Hidden" />
                            <ScrollViewer x:Name="PART_ContentHost" SnapsToDevicePixels="{TemplateBinding SnapsToDevicePixels}" />
                        </Grid>
                        <Button x:Name="ClearButton"
                                Grid.Column="1"
                                Width="17"
                                Height="17"
                                Margin="0 0 5 0"
                                Command="{TemplateBinding AttachedProperties.ClearCommand}"
                                CommandParameter="{Binding RelativeSource={RelativeSource TemplatedParent}}"
                                Content="X"
                                Focusable="False"
                                Padding="0 -2 0 0"
                                Style="{StaticResource AttachedSearchTextBoxButtonStyle}" />
                    </Grid>
                </Border>
                <ControlTemplate.Triggers>
                    <MultiTrigger>
                        <MultiTrigger.Conditions>
```

- Wie DependencyProperties verwendbar
- TemplateBinding möglich
- Klasse.PropertyName

Attached Properties anlegen 4/4

(TextBox.xaml)

READ-ONLY ATTACHED PROPERTIES

Read-Only Attached Properties

- Von Außen nur lesbar und nicht schreibbar
- Verwaltung über internen Schlüssel
  - DependencyPropertyKey
  - Besteht aus Schlüssel & Dependency Property

› Schlüssel registrieren über statische Funktion

```
/// <summary> ...
private static readonly DependencyPropertyKey ClearCommandPropertyKey = DependencyProperty.RegisterAttachedReadOnly(
    "ClearCommand",
    typeof(DelegateCommand),
    typeof(AttachedProperties),
    new FrameworkPropertyMetadata(new DelegateCommand(ExecuteClearCommand)));
```

```
/// <summary> ...
public static readonly DependencyProperty ClearCommandProperty = ClearCommandPropertyKey.DependencyProperty;
```

› Anlegen der Property

Read-Only Attached Properties anlegen 1/2

(AttachedProperties.cs)

```
/// <summary> ...
public static DelegateCommand GetClearCommand(DependencyObject obj)
{
    return (DelegateCommand)obj.GetValue(ClearCommandProperty);
}

/// <summary> ...
private static void SetClearCommand(DependencyObject obj, DelegateCommand value)
{
    obj.SetValue(ClearCommandPropertyKey, value);
}
```

- Verwendung der Methoden GetValue(), SetValue()

Attached Properties anlegen 2/2

(AttachedProperties.cs)

```
/// <summary> ...
private static readonly DependencyPropertyKey ClearCommandPropertyKey = DependencyProperty.RegisterAttachedReadOnly(
    "ClearCommand",
    typeof(DelegateCommand),
    typeof(AttachedProperties),
    new FrameworkPropertyMetadata(new DelegateCommand(ExecuteClearCommand)));
```

› Referenz über CommandParameter

```
/// <summary> ...
private static void ExecuteClearCommand(object obj)
{
    TextBox self = obj as TextBox;

    if (self == null)
        return;

    self.Clear();
}
```

CommandAction definieren

(AttachedProperties.cs)

- Setzen der Bindings
  - Command
  - CommandParameter (Referenz)

```
<Button x:Name="ClearButton"
        Grid.Column="1"
        Width="17"
        Height="17"
        Margin="0 0 5 0"
        Command="{TemplateBinding AttachedProperties.ClearCommand}"
        CommandParameter="{Binding RelativeSource={RelativeSource TemplatedParent}}"
        Content="X"
        Focusable="False"
        Padding="0 -2 0 0"
        Style="{StaticResource AttachedSearchTextBoxButtonStyle}" />
```

TemplateBinding

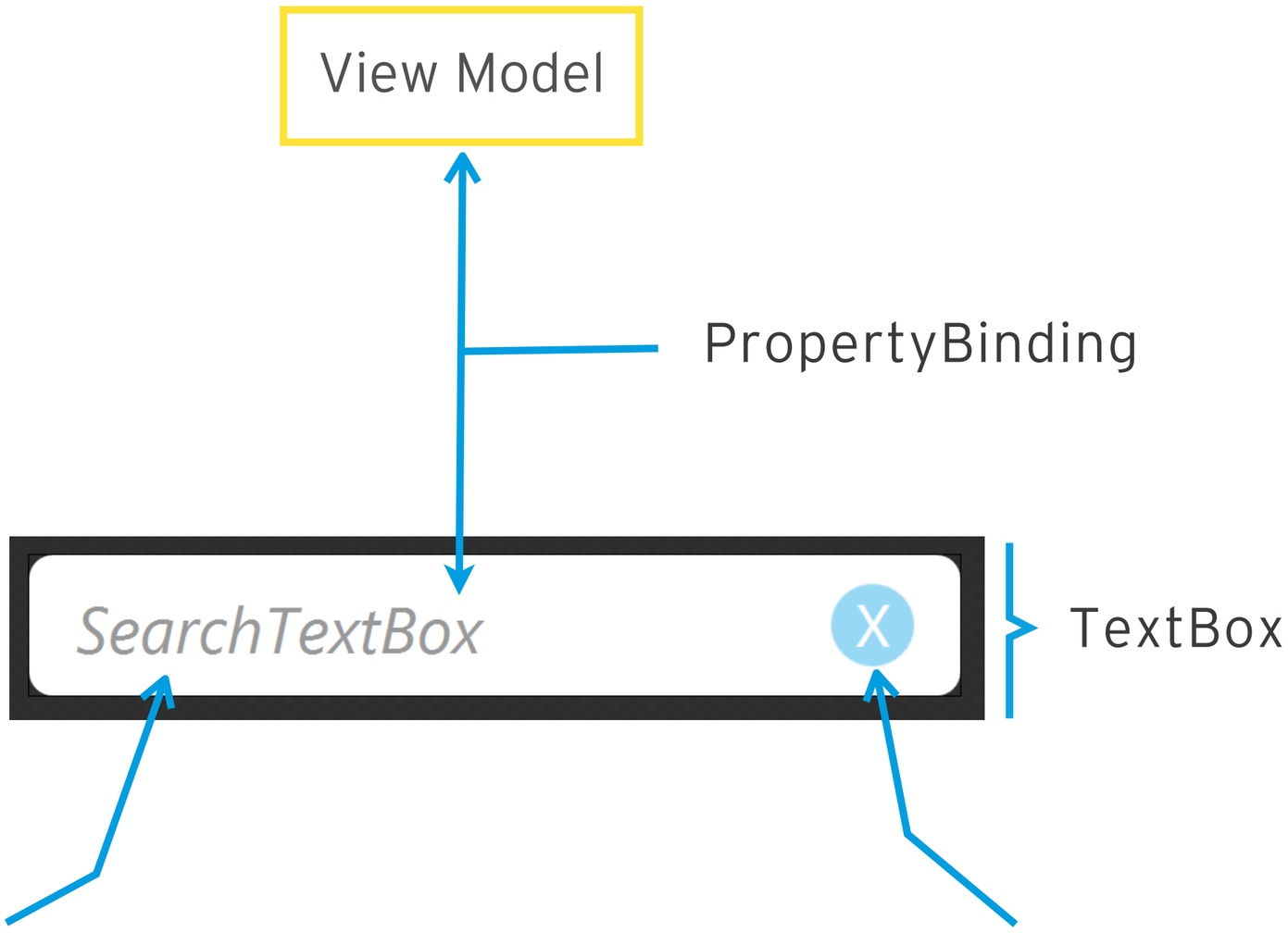
(TextBox.xaml)

```
<TextBox x:Name="SearchTextBox"
         Margin="0 0 0 10"
         AttachedProperties.Watermark="Search Points"
         Text="{Binding SearchString,
                        UpdateSourceTrigger=PropertyChanged}" />
```

- Binding an ViewModel Property

ViewModel Kommunikation

(MainView.xaml)



- Watermark
- WatermarkFontStyle
- WatermarkFontWeight
- WatermarkForeground

ClearCommand

(Zurücksetzen des Texts)

WAS FINDEN WIR NOCH… ?

André Lanninger
UIDeveloper

Datagraph 11

ChangeStyle

Load

62

- Bubble 18
  0, 20
- Bubble 19
  10, 15
- Bubble 20
  20, **62**
- Bubble 21
  30, 10
- Bubble 22
  40, 27
- Bubble 23
  50, 77
- Bubble 24
  60, 73
- Bubble 25
  70, 90
- Bubble 26
  80, 33

Styling?

Custom Controls?

André Lanninger
UIDeveloper

Datagraph 11

ChangeStyle

Load

Search Points

- Bubble 2
  0, 14
- Bubble 3
  10, 61
- Bubble 4
  20, 20
- Bubble 5
  30, 73
- Bubble 6
  40, 48
- Bubble 7
  50, 37
- Bubble 8
  60, 86
- Bubble 9
  70, 19
- Bubble 10
  80, 93

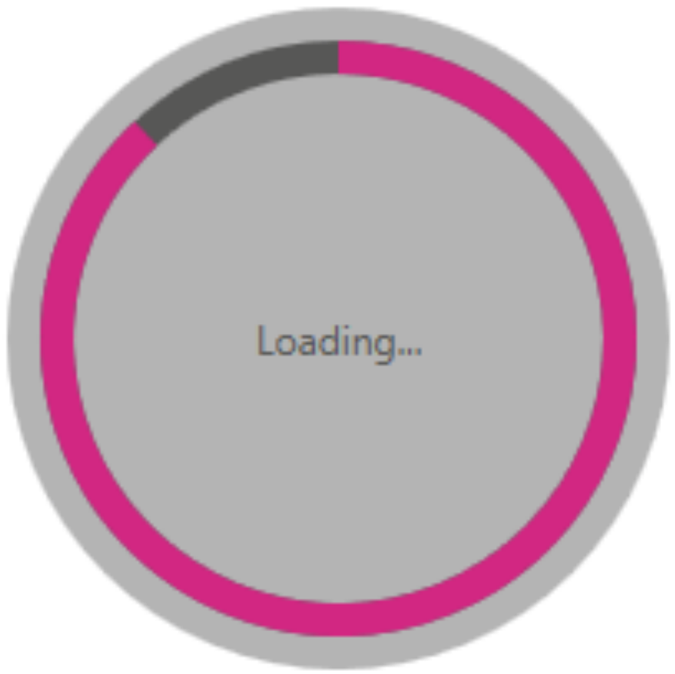
Loading...

CircularProgressBar

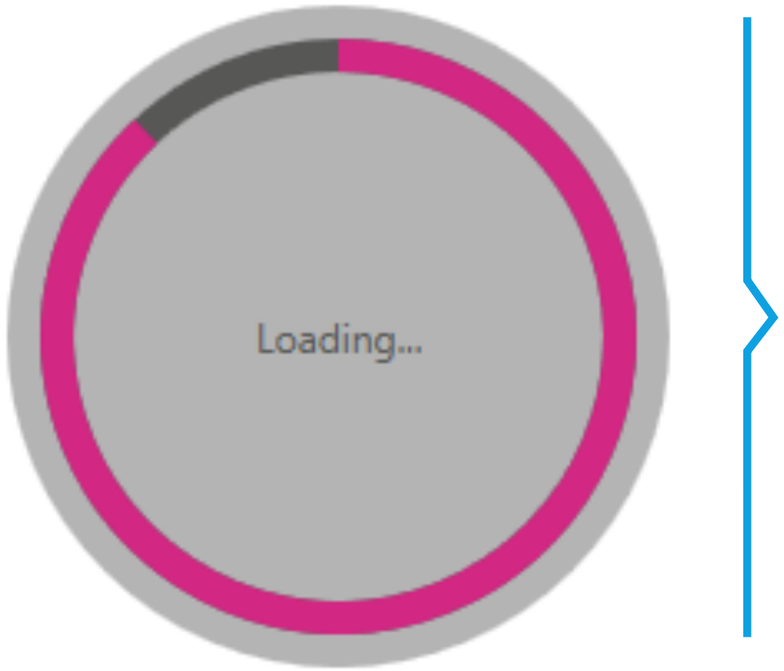
CIRCULAR PROGRESSBAR

ANALYSE

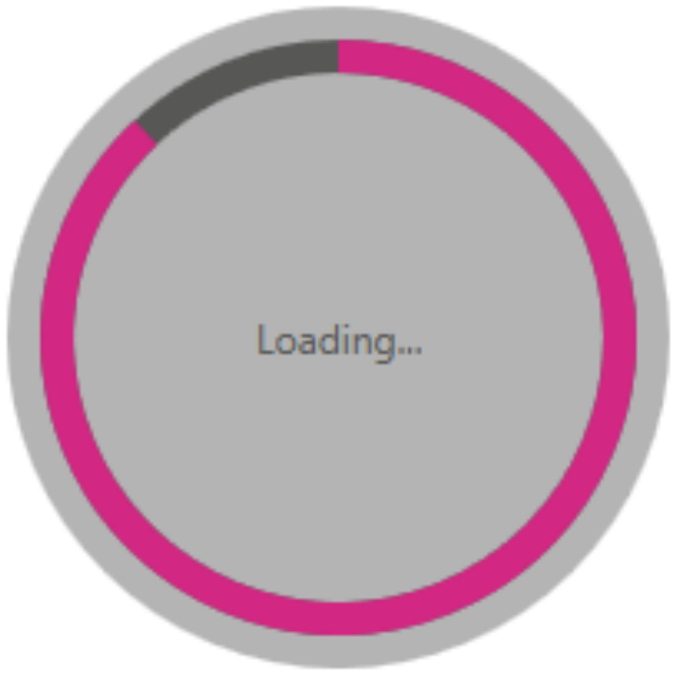
View Model



View Model

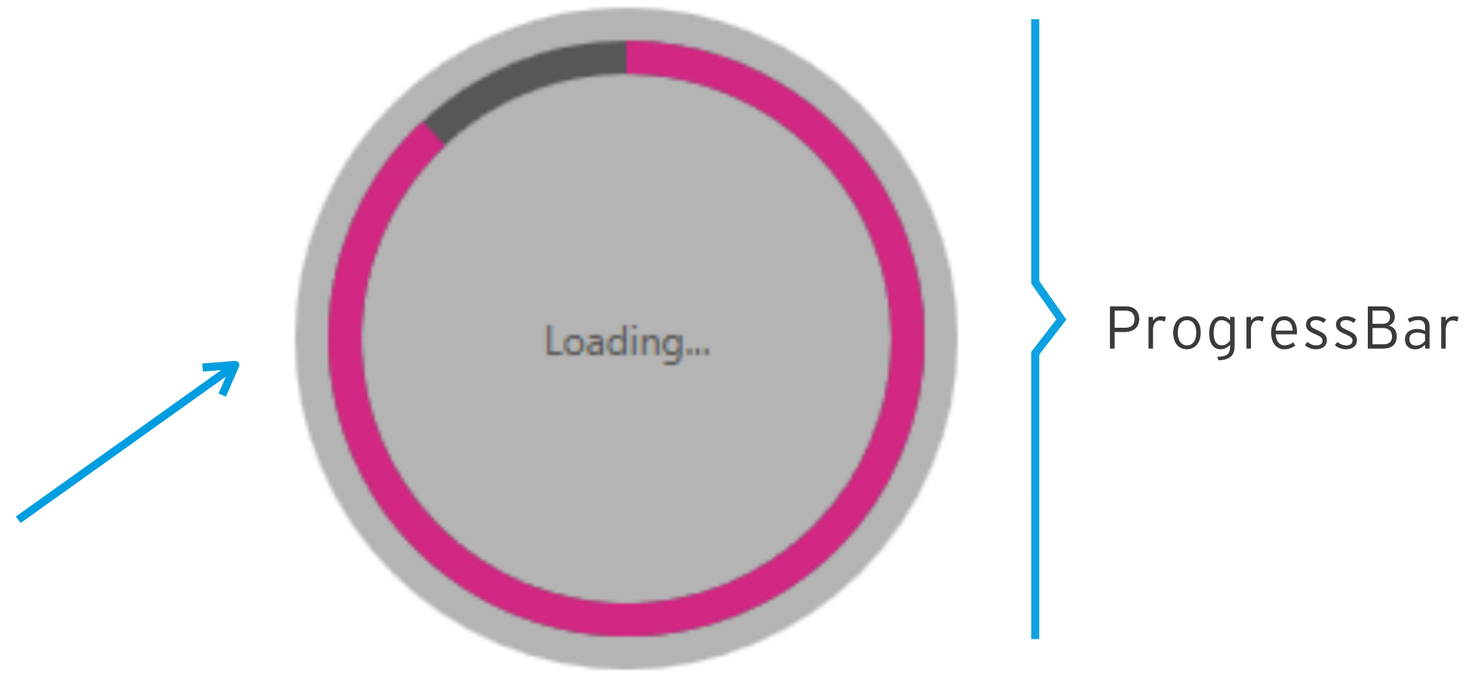


View Model

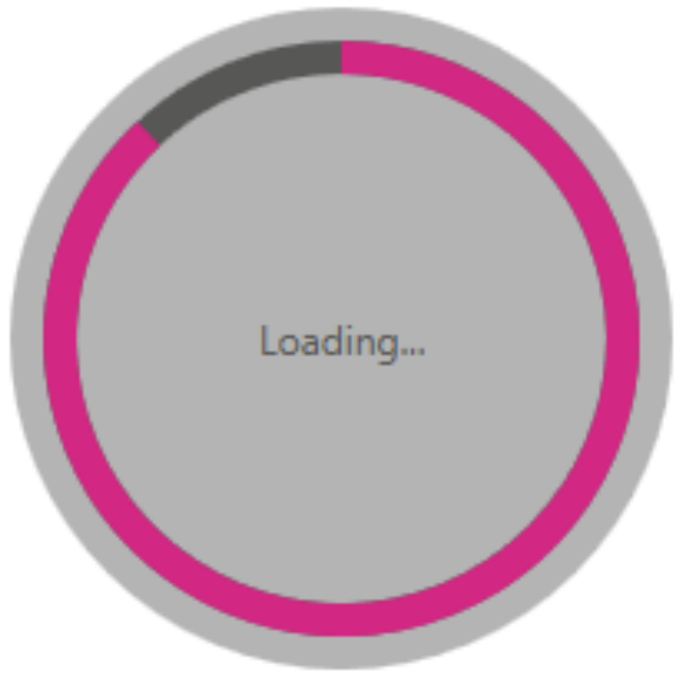


ProgressBar

View Model



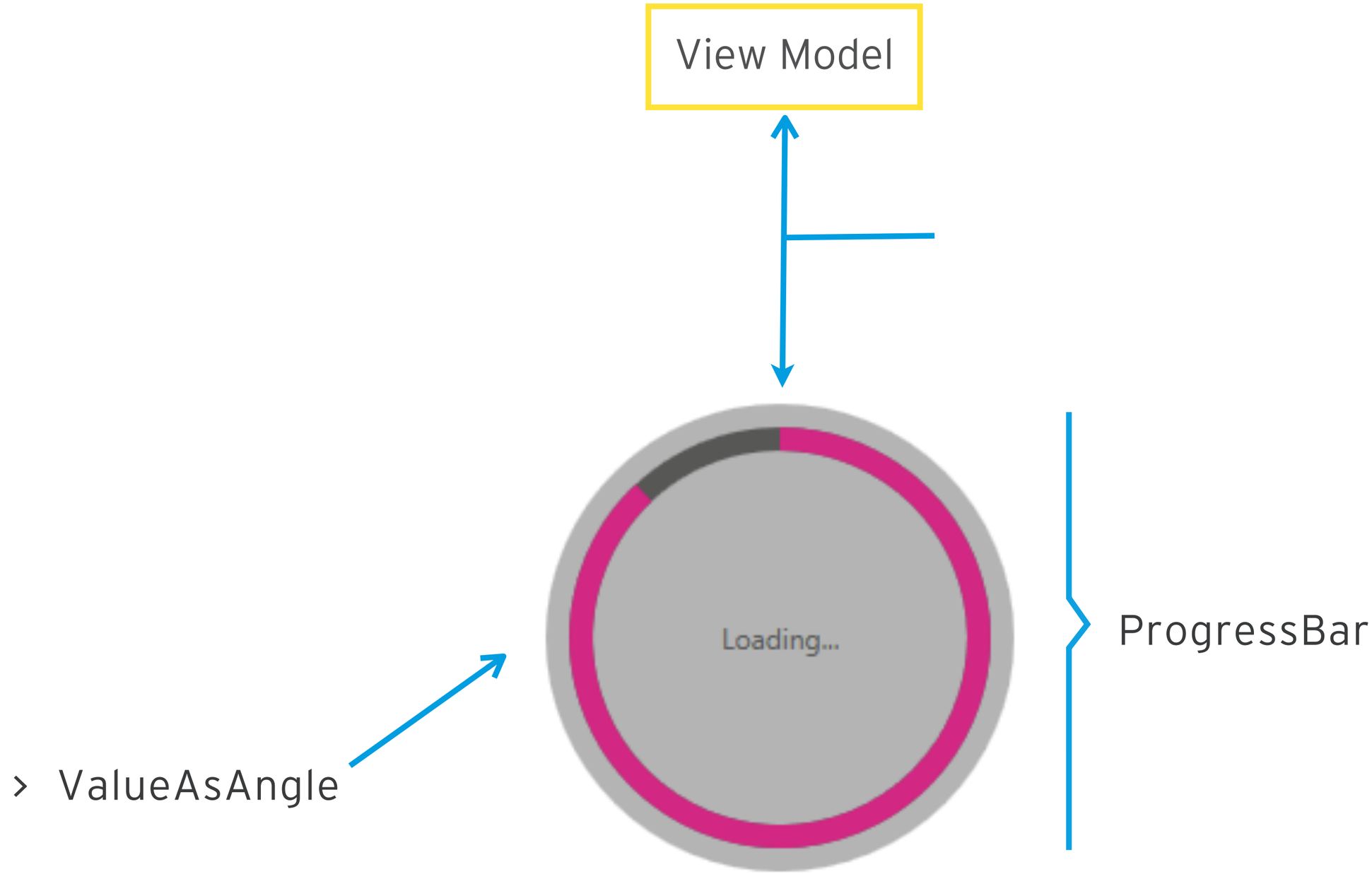
View Model

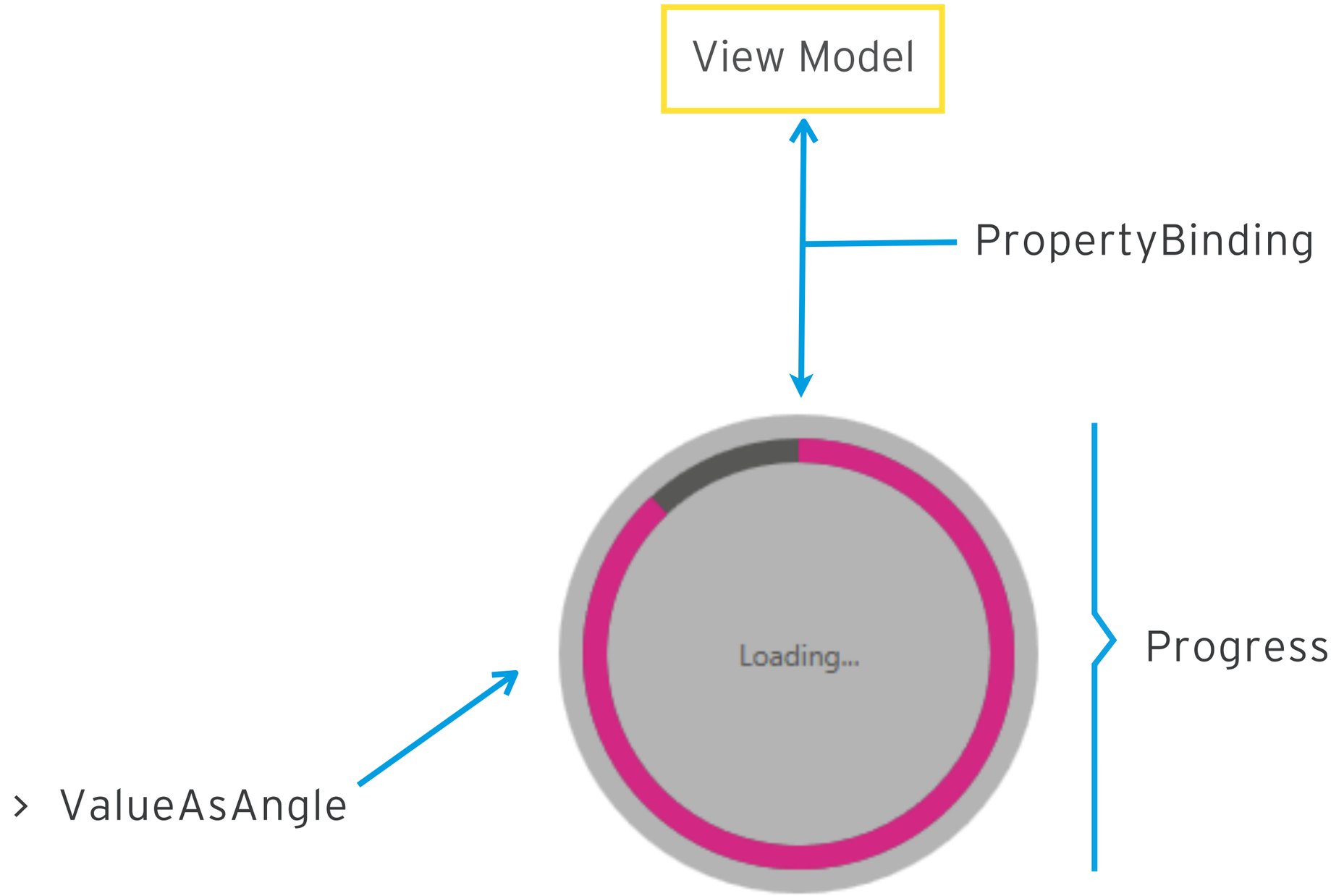


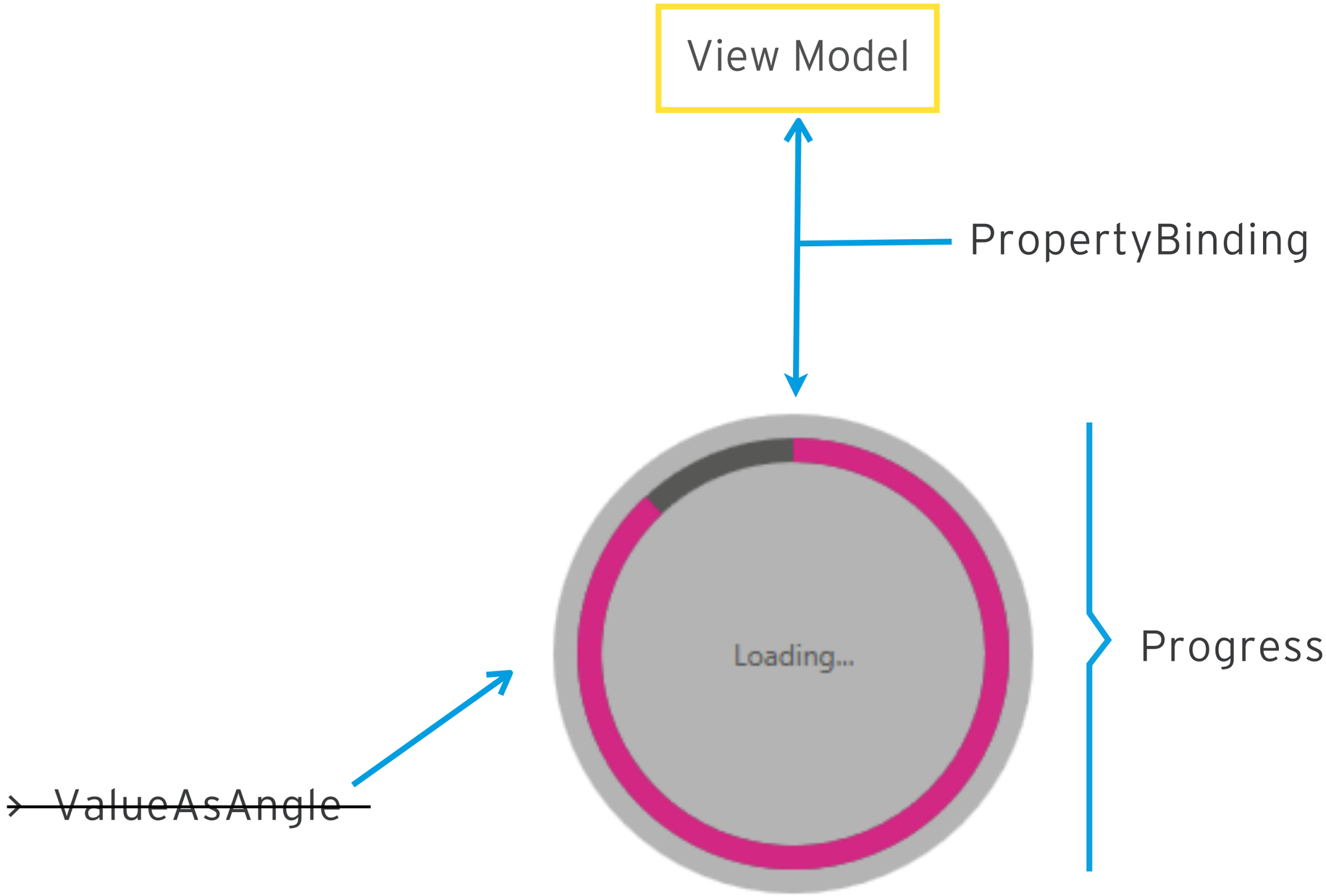
ProgressBar

› ValueAsAngle

CIRCULAR PROGRESSBAR ANALYSE




View Model
PropertyBinding
Loading...
ProgressBar
› ValueAsAngle


View Model
PropertyBinding
Loading...
ProgressBar
ValueAsAngle

STYLE & TEMPLATE

```xaml
<Style x:Key="CircularConverterProgressBarStyle" TargetType="{x:Type ProgressBar}" BasedOn="{StaticResource FrameworkElementStyle}">
    <!--  <Setters ...  -->
    <Setter Property="Template">
        <Setter.Value>
            <ControlTemplate TargetType="{x:Type ProgressBar}">
                <Grid>
                    <Grid Width="{TemplateBinding MinWidth}"
                          Height="{TemplateBinding MinHeight}"
                          Margin="{TemplateBinding Padding}"
                          HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}"
                          VerticalAlignment="{TemplateBinding VerticalContentAlignment}">
                        <ed:Arc ArcThickness="1"
                                ArcThicknessUnit="Percent"
                                EndAngle="360"
                                Fill="{TemplateBinding Background}"
                                StartAngle="0"
                                Stretch="None" />
                        <ed:Arc Margin="10"
                                ArcThickness="10"
                                ArcThicknessUnit="Pixel"
                                EndAngle="360"
                                Fill="{TemplateBinding BorderBrush}"
                                StartAngle="0"
                                Stretch="None" />
                        <ed:Arc Margin="10"
                                ArcThickness="10"
                                ArcThicknessUnit="Pixel"
                                Fill="{StaticResource Red01Brush}"
                                StartAngle="0"
                                Stretch="None">
                            <ed:Arc.EndAngle>
                                <MultiBinding Converter="{StaticResource ValueAsAngleConverter}">
                                    <MultiBinding.Bindings>
                                        <Binding Path="Minimum" RelativeSource="{RelativeSource TemplatedParent}" />
                                        <Binding Path="Maximum" RelativeSource="{RelativeSource TemplatedParent}" />
                                        <Binding Path="Value" RelativeSource="{RelativeSource TemplatedParent}" />
                                    </MultiBinding.Bindings>
                                </MultiBinding>
                            </ed:Arc.EndAngle>
                        </ed:Arc>
                        <TextBlock HorizontalAlignment="Center"
                                   VerticalAlignment="Center"
                                   Text="Loading..." />
                    </Grid>
                </Grid>
            </ControlTemplate>
        </Setter.Value>
    </Setter>
</Style>
```

- MultiBinding
  - Minimum
  - Maximum
  - Value

=> ValueAsAngle

Style & Template

(ProgressBar.xaml)

CONVERTER

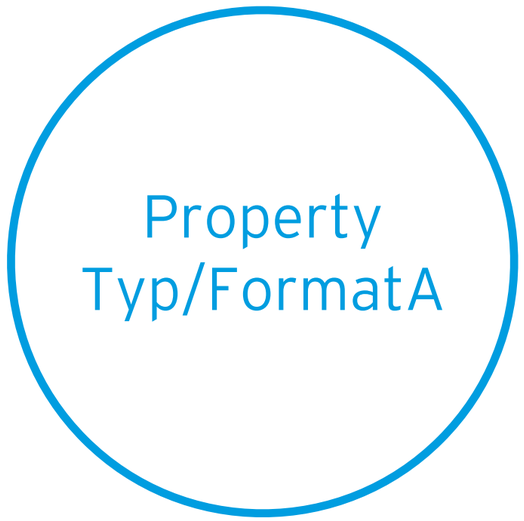
Converter

- Implementieren des entsprechenden Interfaces
  - *IValueConverter*
  - *IMultiValueConverter*

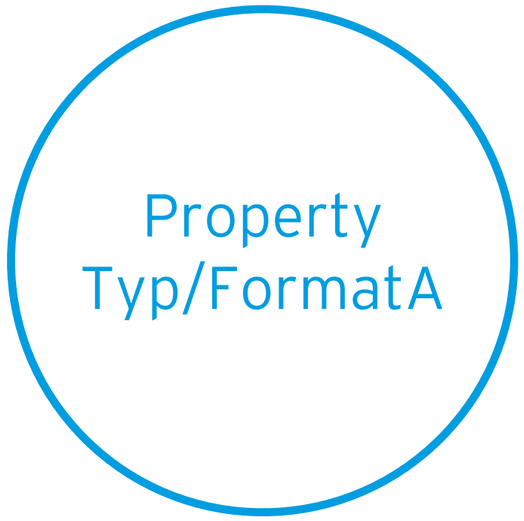
- Konvertieren einen/mehrere Wert/e in das gewünschte Format bzw. den gewünschten Typ

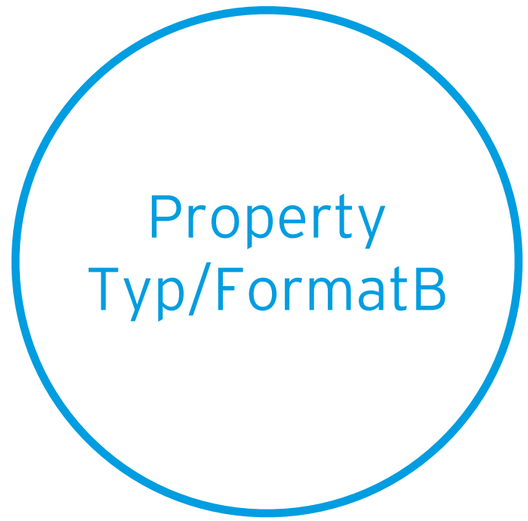
Converter - IValueConverter

Converter - IValueConverter

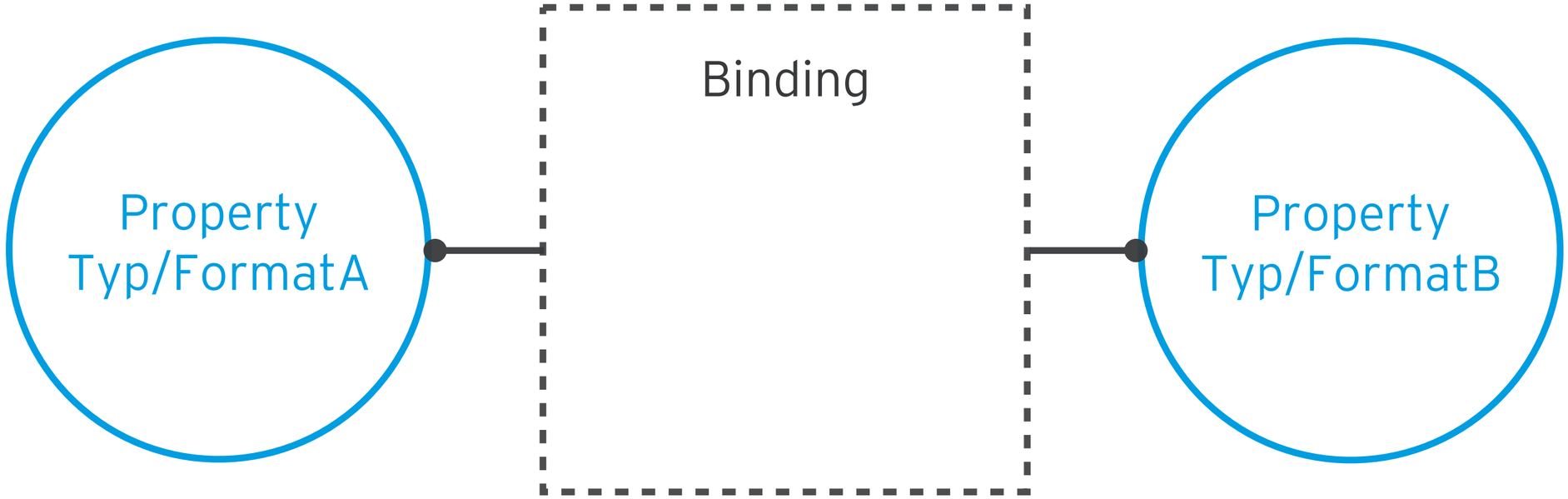


Converter - IValueConverter

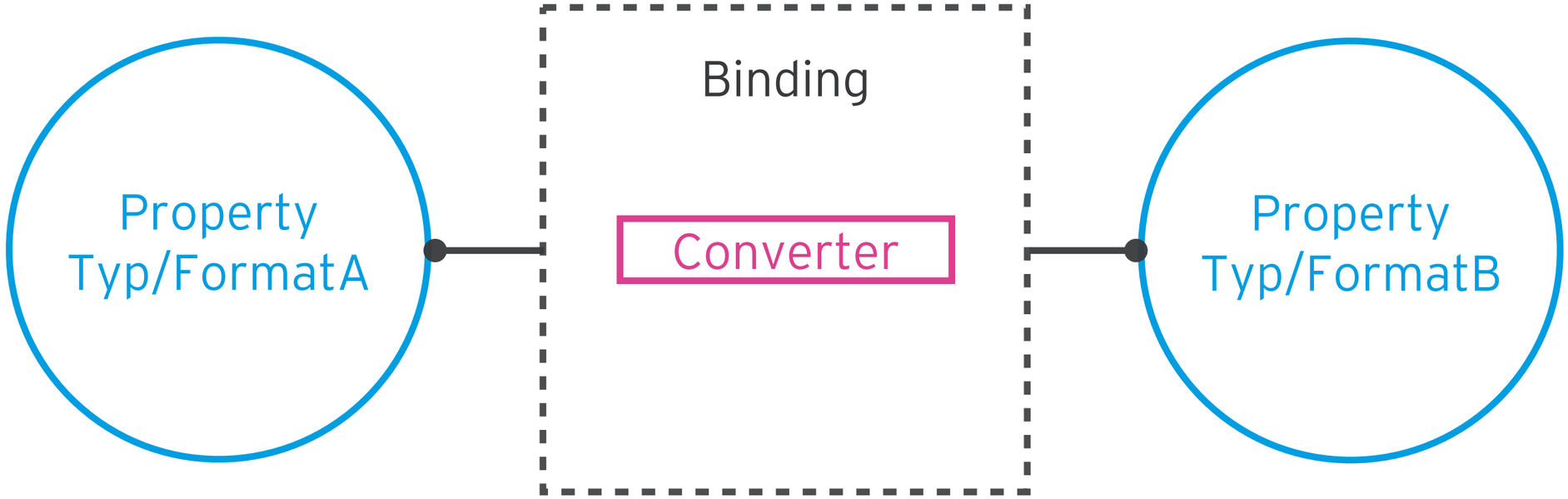




Converter - IValueConverter

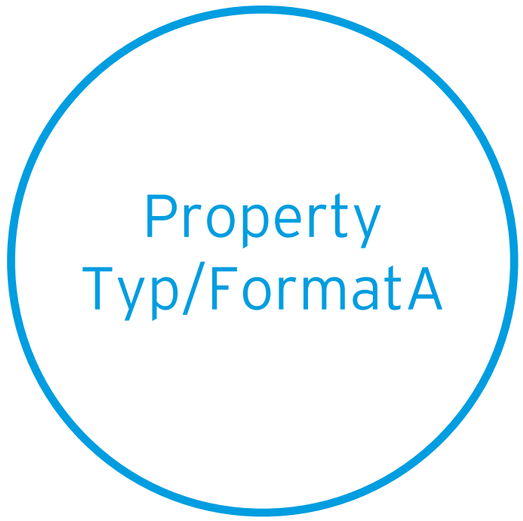


Converter - IValueConverter

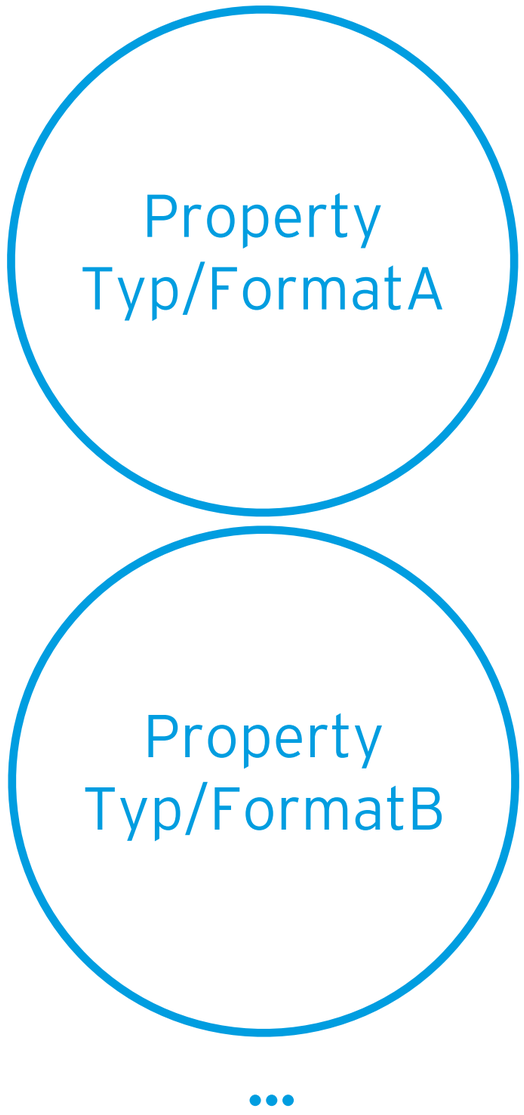


Converter - IMultiValueConverter

Converter - IMultiValueConverter

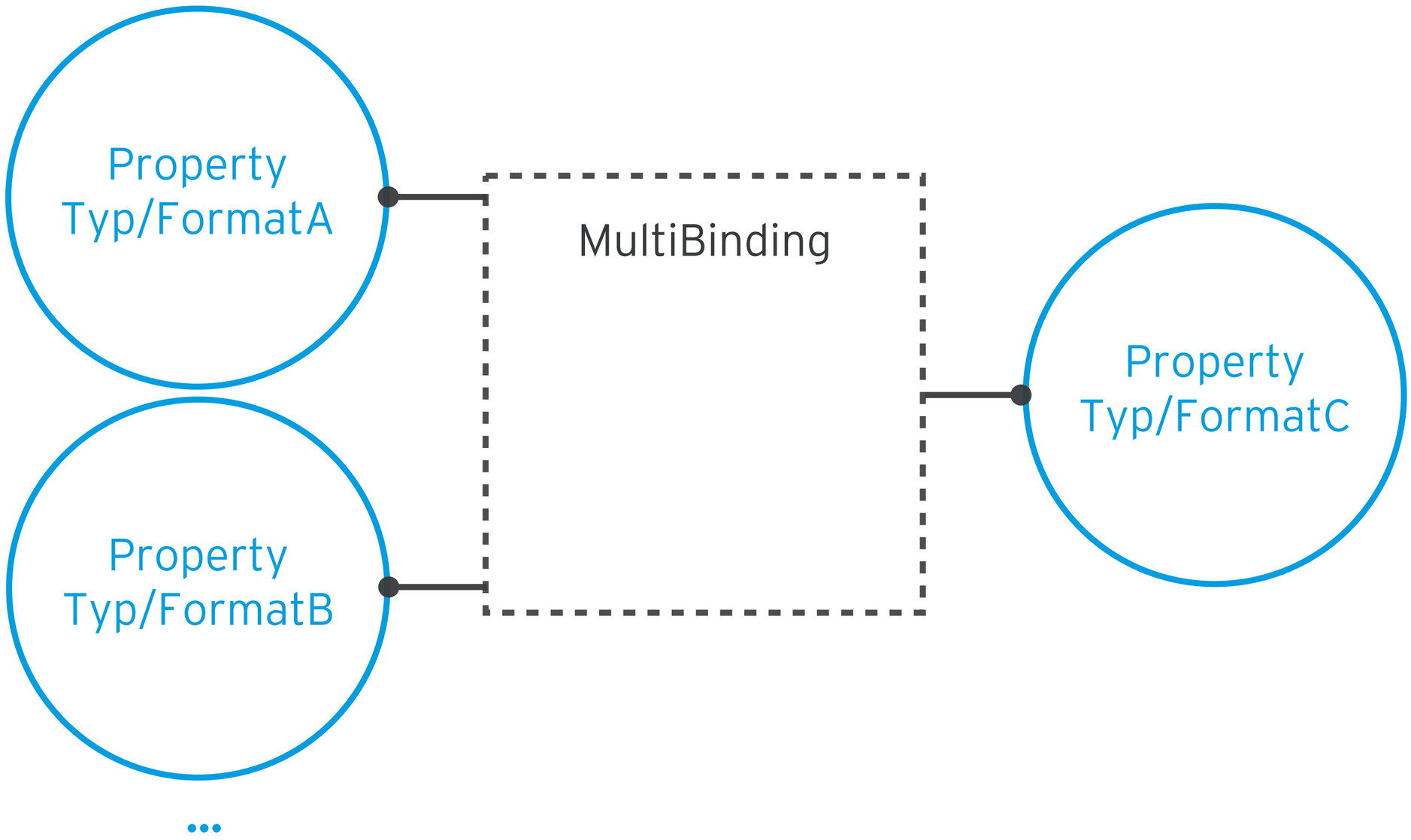


Converter - IMultiValueConverter

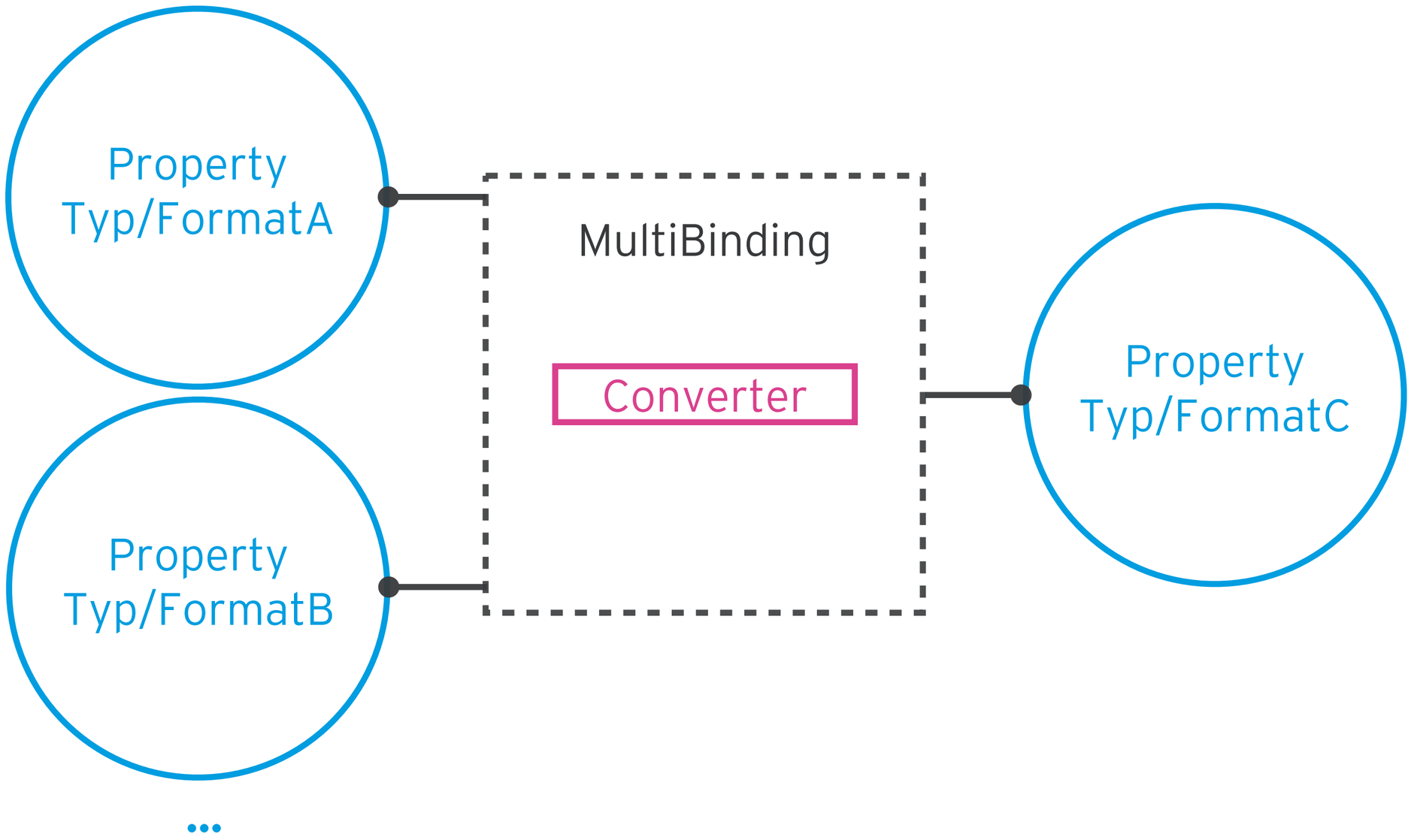


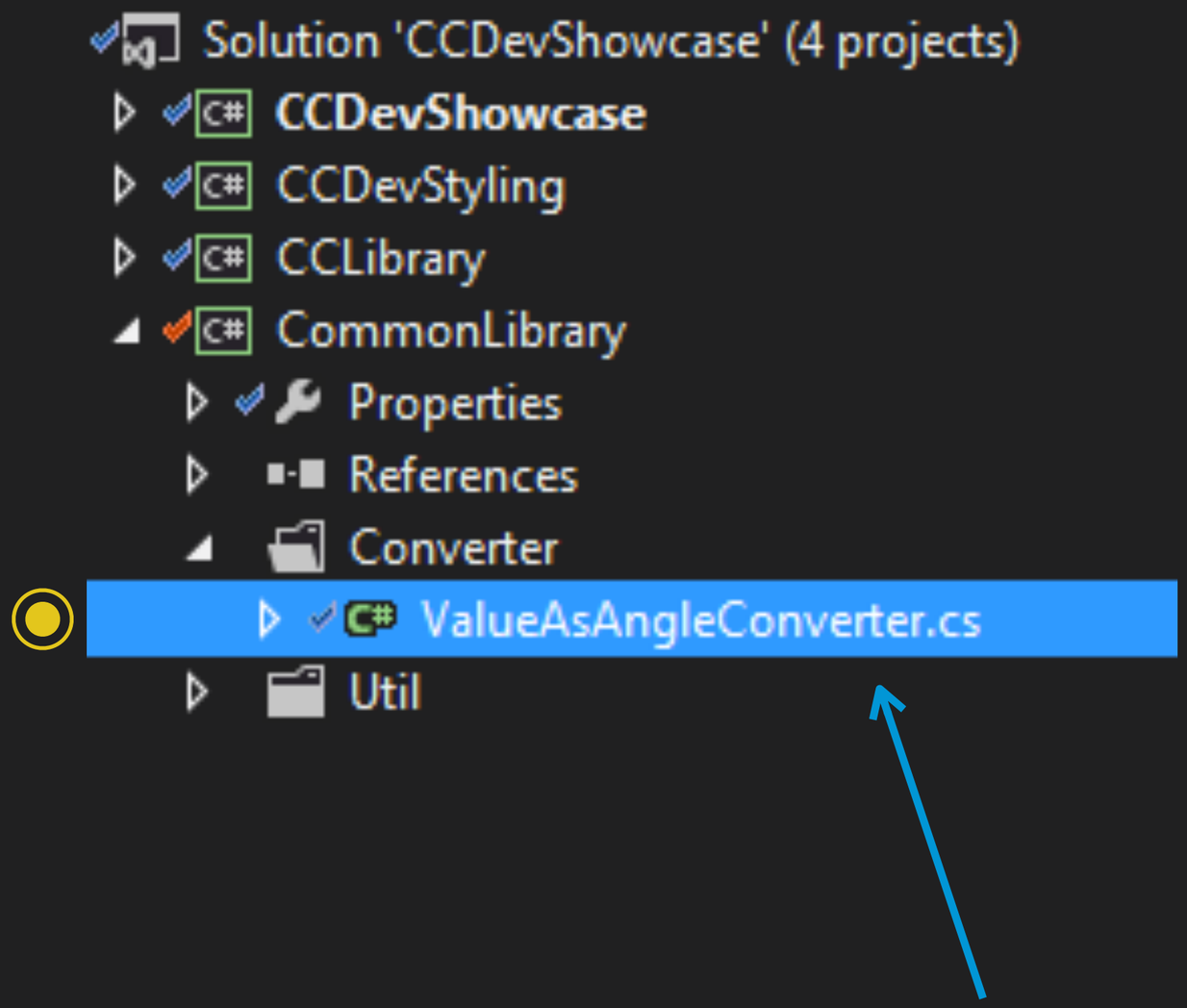
Converter - IMultiValueConverter

Converter - IMultiValueConverter



Converter - IMultiValueConverter

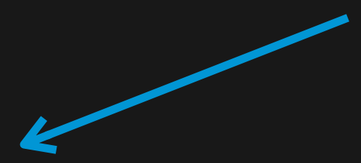




› Klasse anlegen

Converter 1/3

- Interface implementieren
  - Convert(...)
  - ConvertBack(...)



```
/// <summary> ...
public class ValueAsAngleConverter : IMultiValueConverter
{
    public object Convert(object[] values, Type targetType, object parameter, System.Globalization.CultureInfo culture)
    {
        double min, max, value = 0.0;

        if (values[0] == null || values[1] == null || values[2] == null)
            return 0.0;

        if (!double.TryParse(values[0].ToString(), out min) ||
            !double.TryParse(values[1].ToString(), out max) ||
            !double.TryParse(values[2].ToString(), out value))
            return 0.0;

        return MathHelpers.ValueAsAngle(min, max, value);
    }

    public object[] ConvertBack(object value, Type[] targetTypes, object parameter, System.Globalization.CultureInfo culture)
    {
        throw new NotImplementedException();
    }
}
```

Converter 2/3

(ValueAsAngleConverter.cs)

- Anlegen als Resource
- Converter Property setzen

```
<ValueAsAngleConverter x:Key="ValueAsAngleConverter" />


<ed:Arc.EndAngle>
    <MultiBinding Converter="{StaticResource ValueAsAngleConverter}">
        <MultiBinding.Bindings>
            <Binding Path="Minimum" RelativeSource="{RelativeSource TemplatedParent}" />
            <Binding Path="Maximum" RelativeSource="{RelativeSource TemplatedParent}" />
            <Binding Path="Value" RelativeSource="{RelativeSource TemplatedParent}" />
        </MultiBinding.Bindings>
    </MultiBinding>
</ed:Arc.EndAngle>
```

Converter 3/3

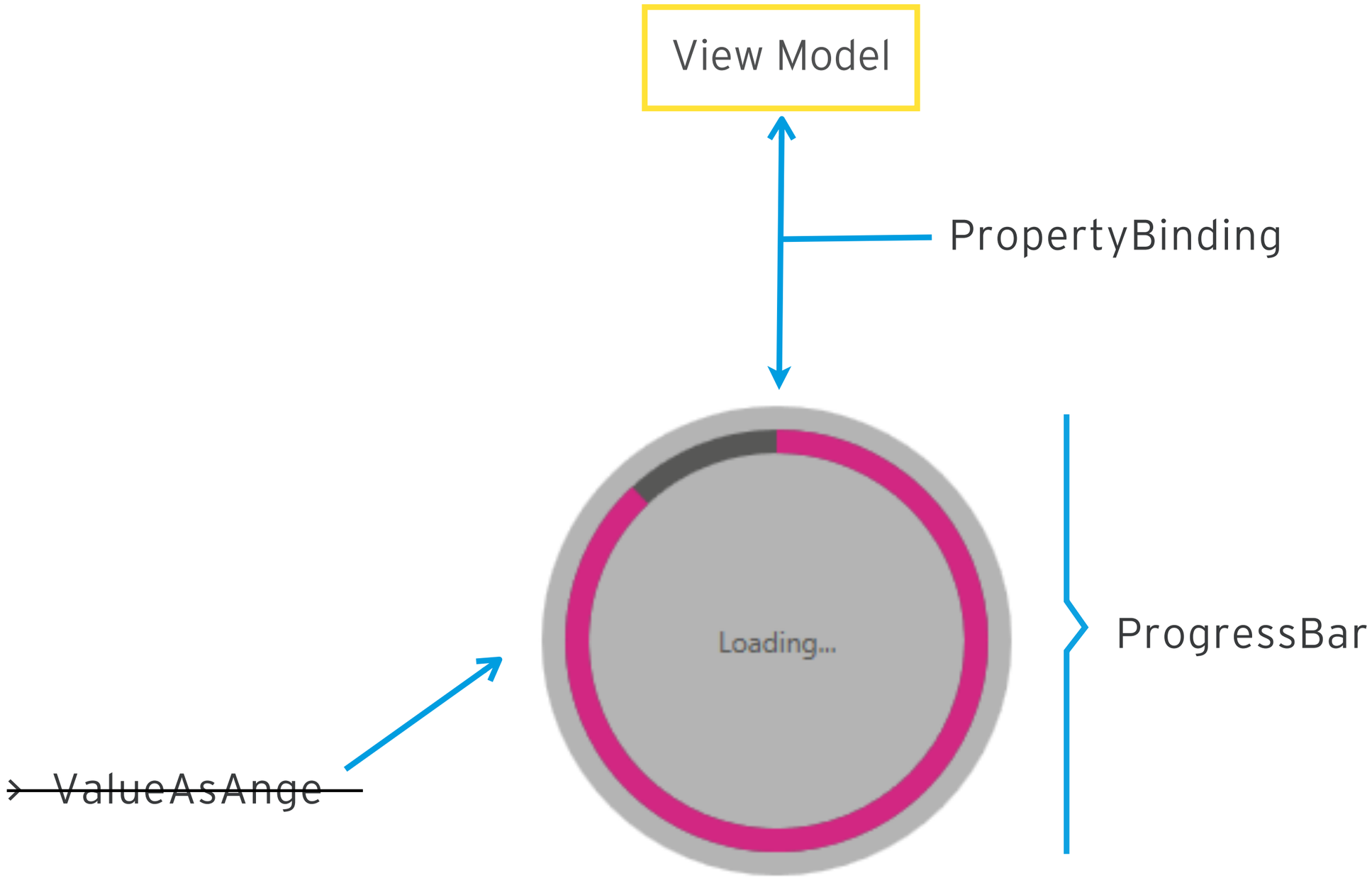
(Converters.xaml, ProgressBar.xaml)

```xaml
<DataTemplate x:Key="LoadingTemplate">
    <ProgressBar Maximum="100"
                 Minimum="0"
                 Value="{Binding ProgressValue}" />
</DataTemplate>
```

- Binding an ViewModel Property

ViewModel Kommunikation

(DataTemplates.xaml)


View Model
PropertyBinding
Loading...
ProgressBar
> ~~ValueAsAnge~~

LOADING INDICATOR

ANALYSE

André Lanninger
UIDeveloper

Datagraph 11

ChangeStyle

Load

62

Bubble 18
0, 20

Bubble 19
10, 15

Bubble 20
20, **62**

Bubble 21
30, 10

Bubble 22
40, 27

Bubble 23
50, 77

Bubble 24
60, 73

Bubble 25
70, 90

Bubble 26
80, 33

André Lanninger
UIDeveloper

Datagraph 11

ChangeStyle

Load

62

Bubble 18
0, 20

Bubble 19
10, 15

Bubble 20
20, **62**

Bubble 21
30, 10

Bubble 22
40, 27

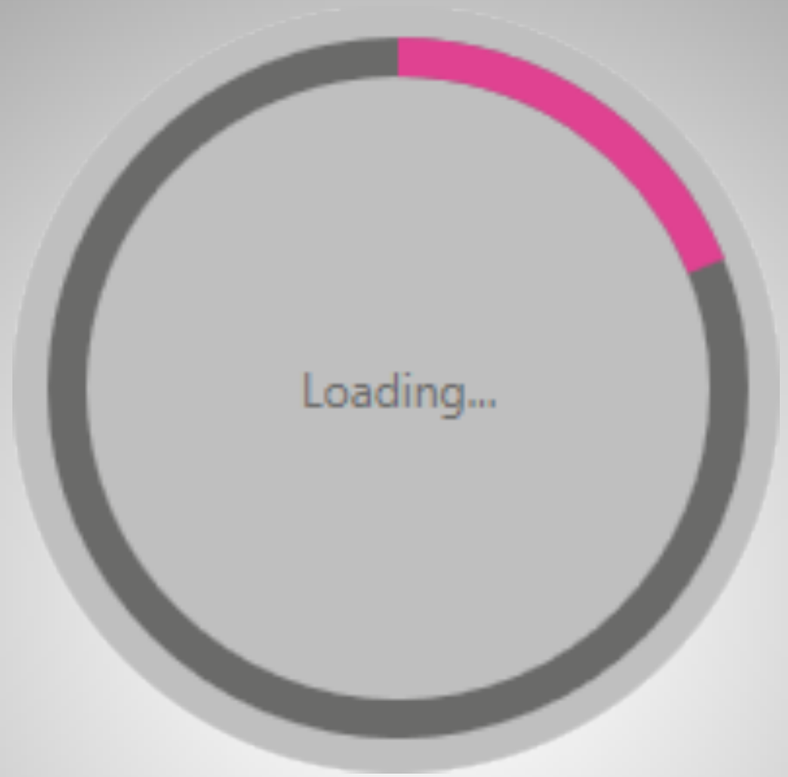
Bubble 23
50, 77

Bubble 24
60, 73

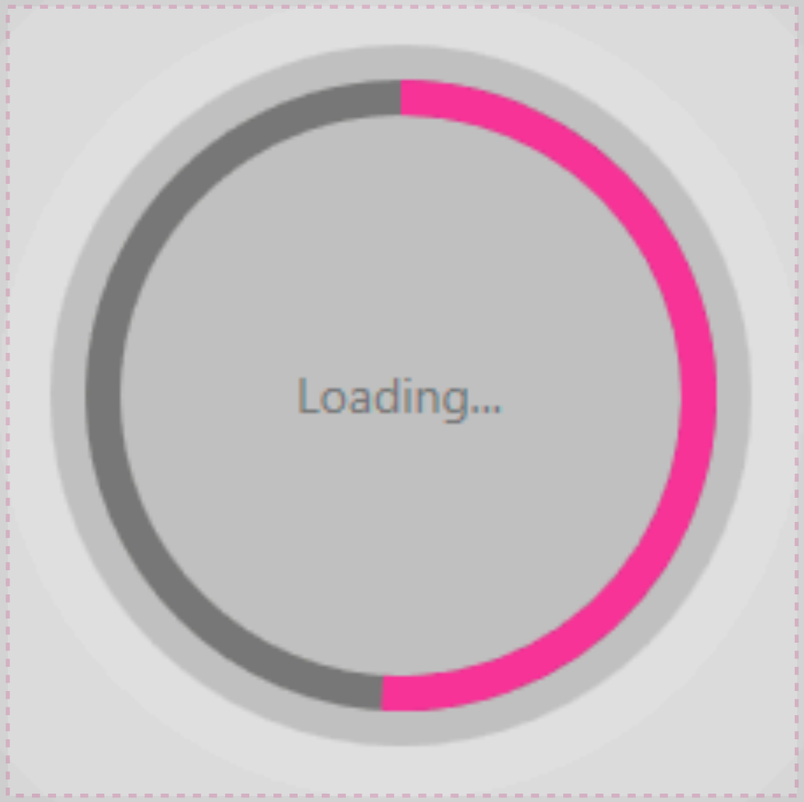
Bubble 25
70, 90

Bubble 26
80, 33

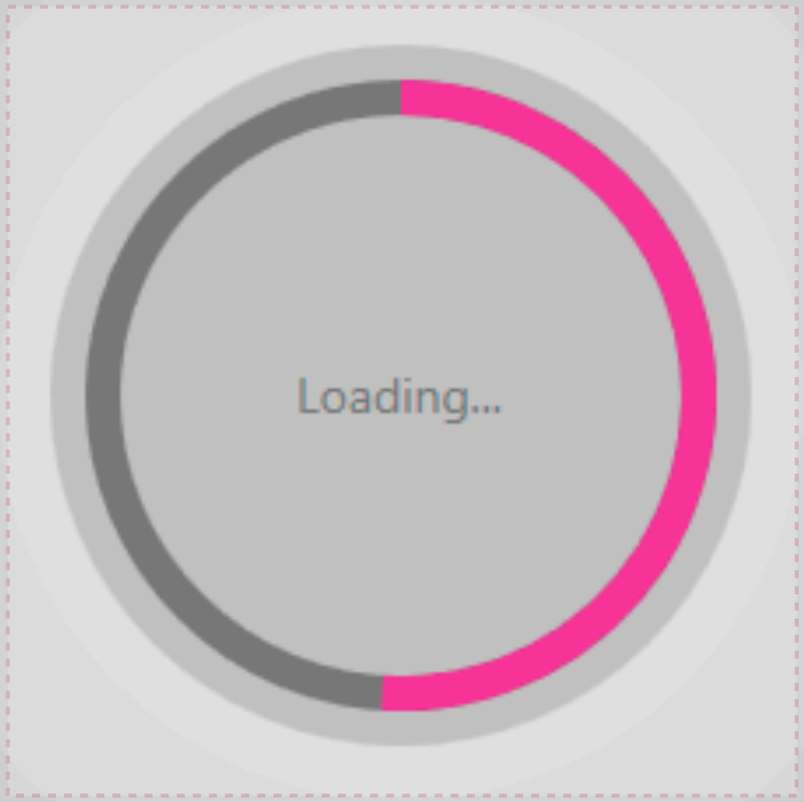
Loading...


Loading...

View Model

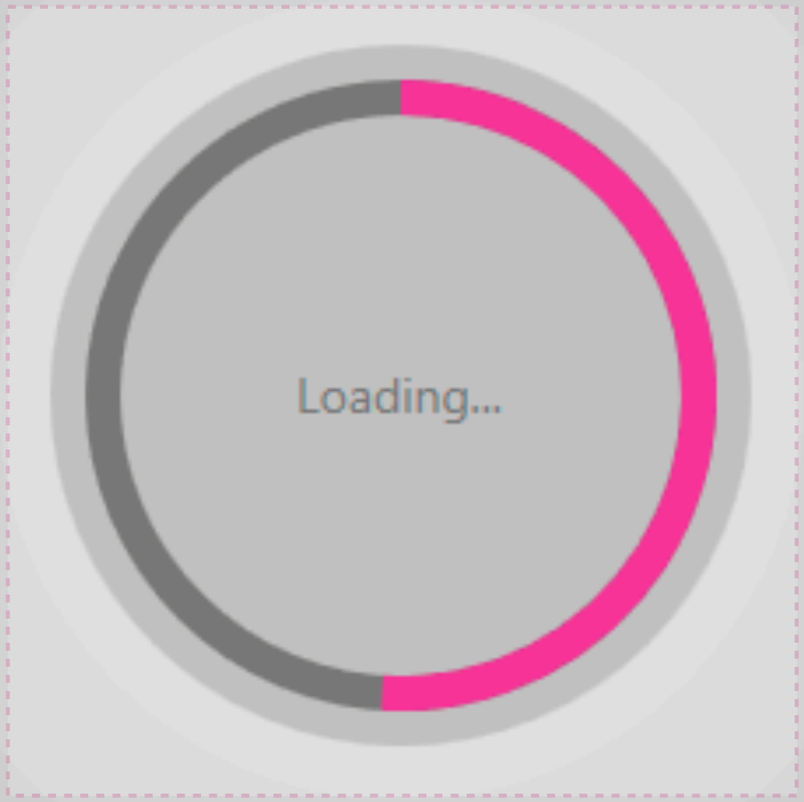


View Model



LOADING INDICATOR ANALYSE

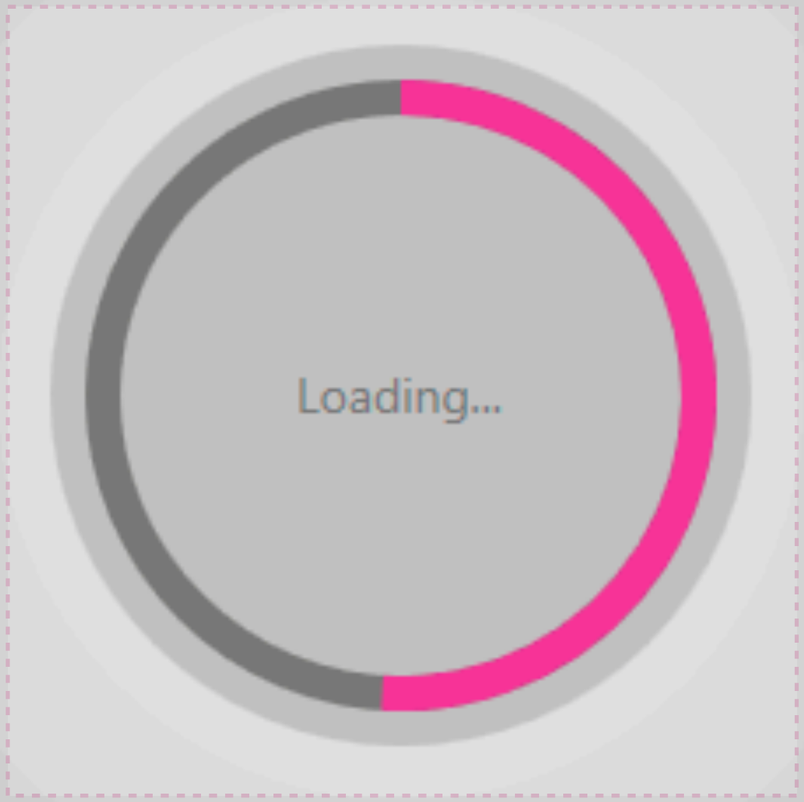
View Model



Beliebiger Content

LOADING INDICATOR ANALYSE

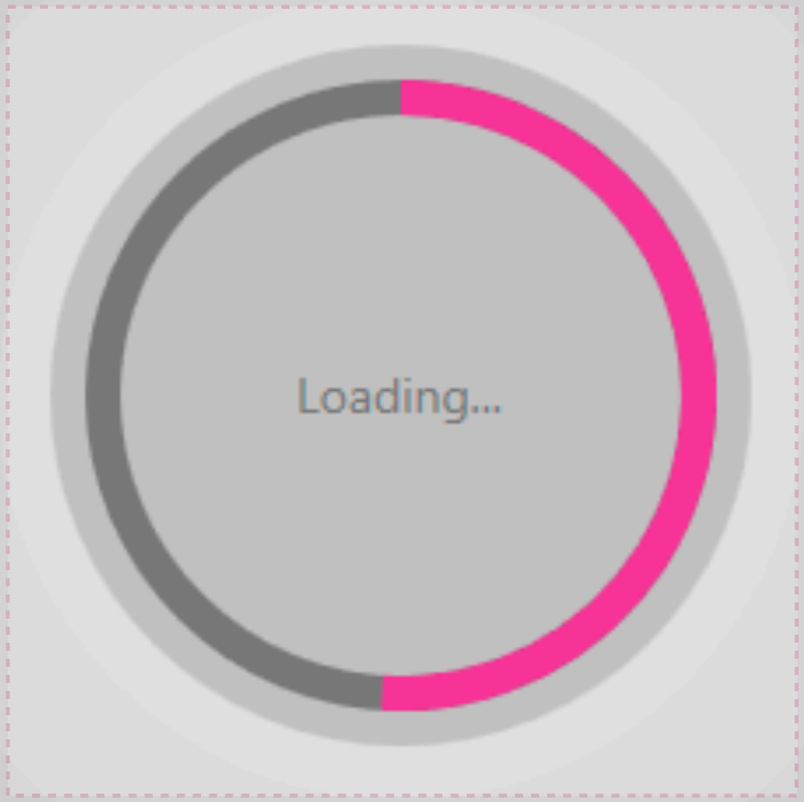
View Model



Beliebiger Content

LOADING INDICATOR ANALYSE

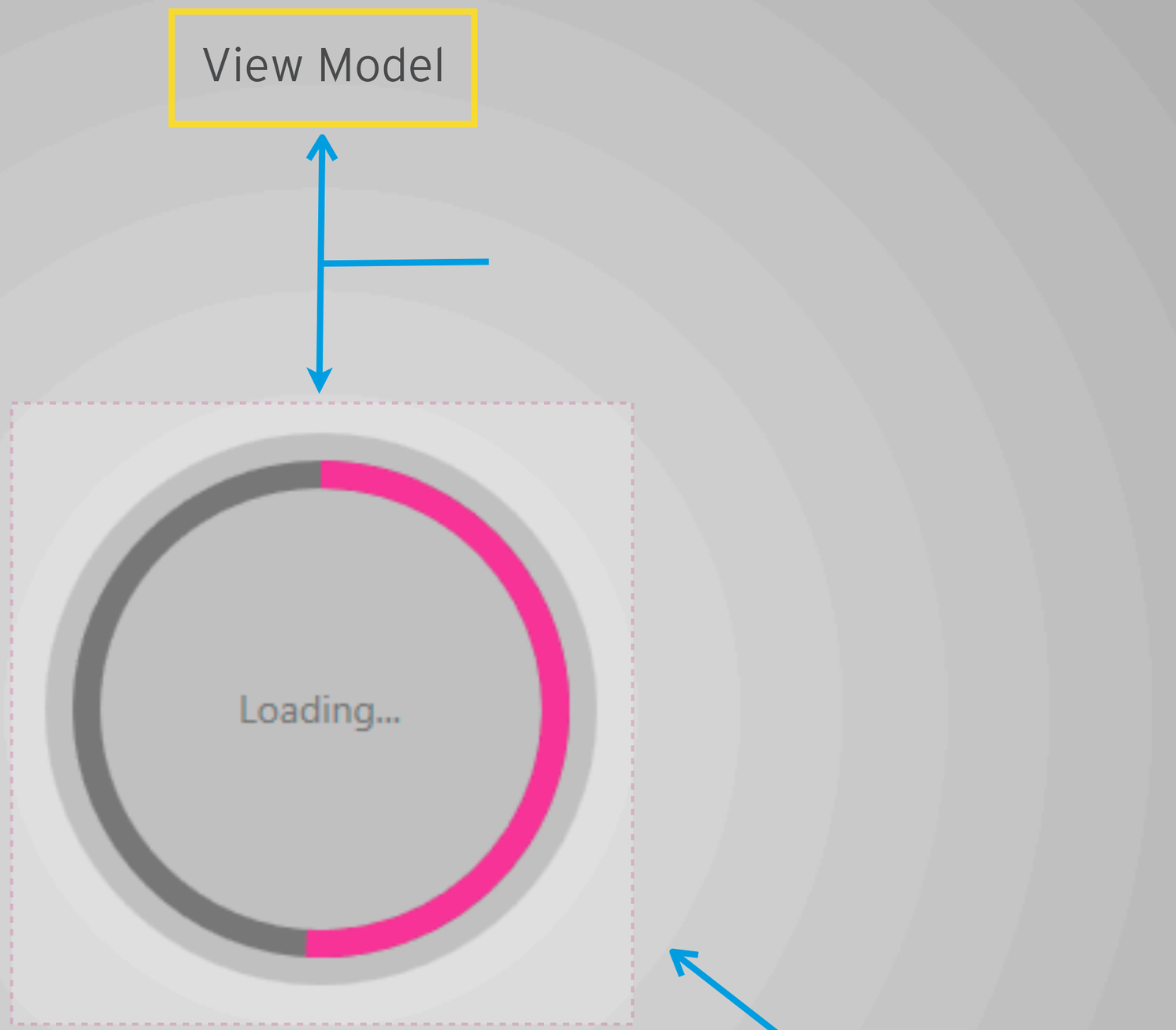
View Model



- IsLoading
- LoadingContent
- LoadingTemplate
- LoadingDataContext

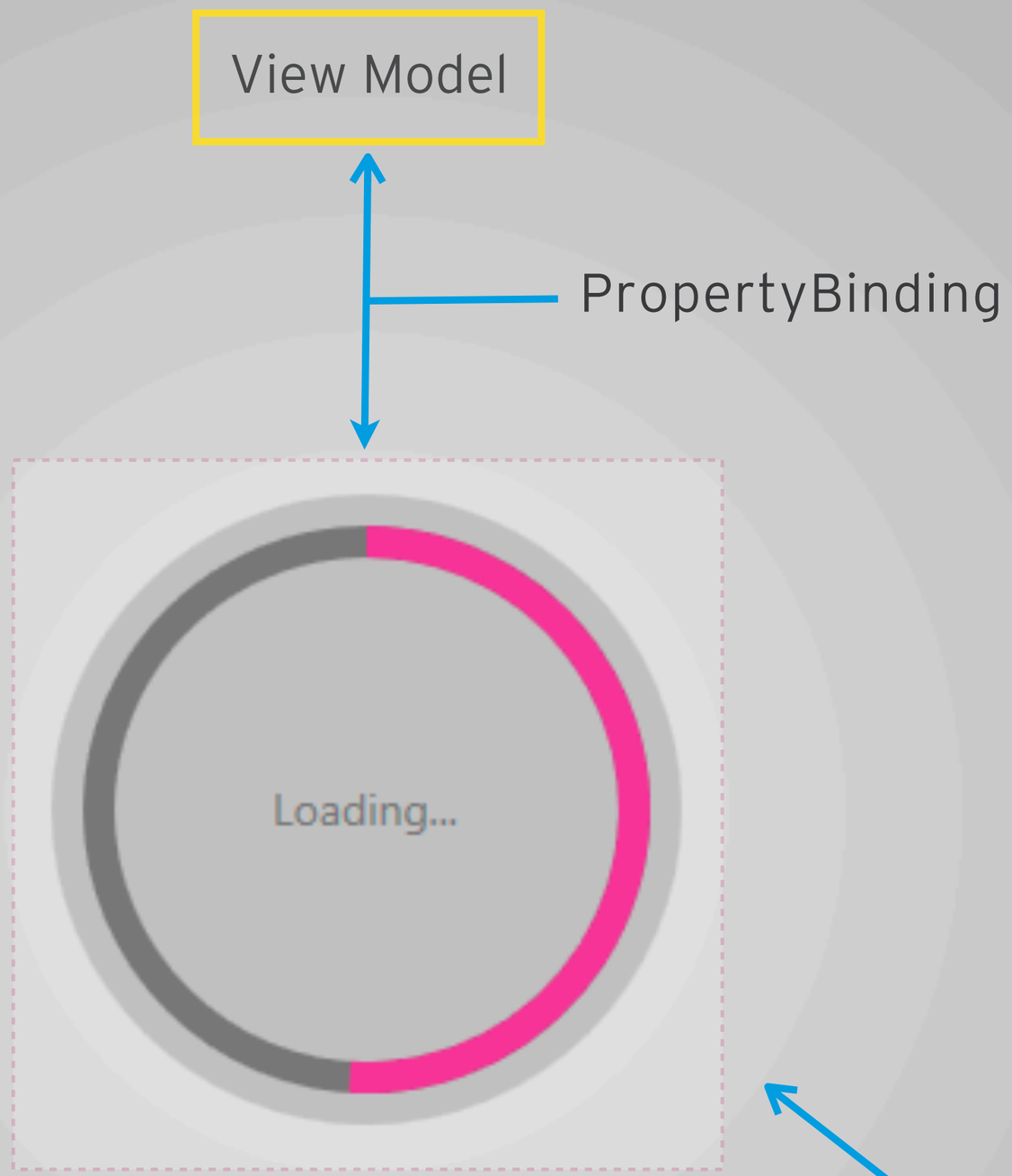
Beliebiger Content

LOADING INDICATOR ANALYSE



- IsLoading
- LoadingContent
- LoadingTemplate
- LoadingDataContext

LOADING INDICATOR ANALYSE



- IsLoading
- LoadingContent
- LoadingTemplate
- LoadingDataContext

Beliebiger Content

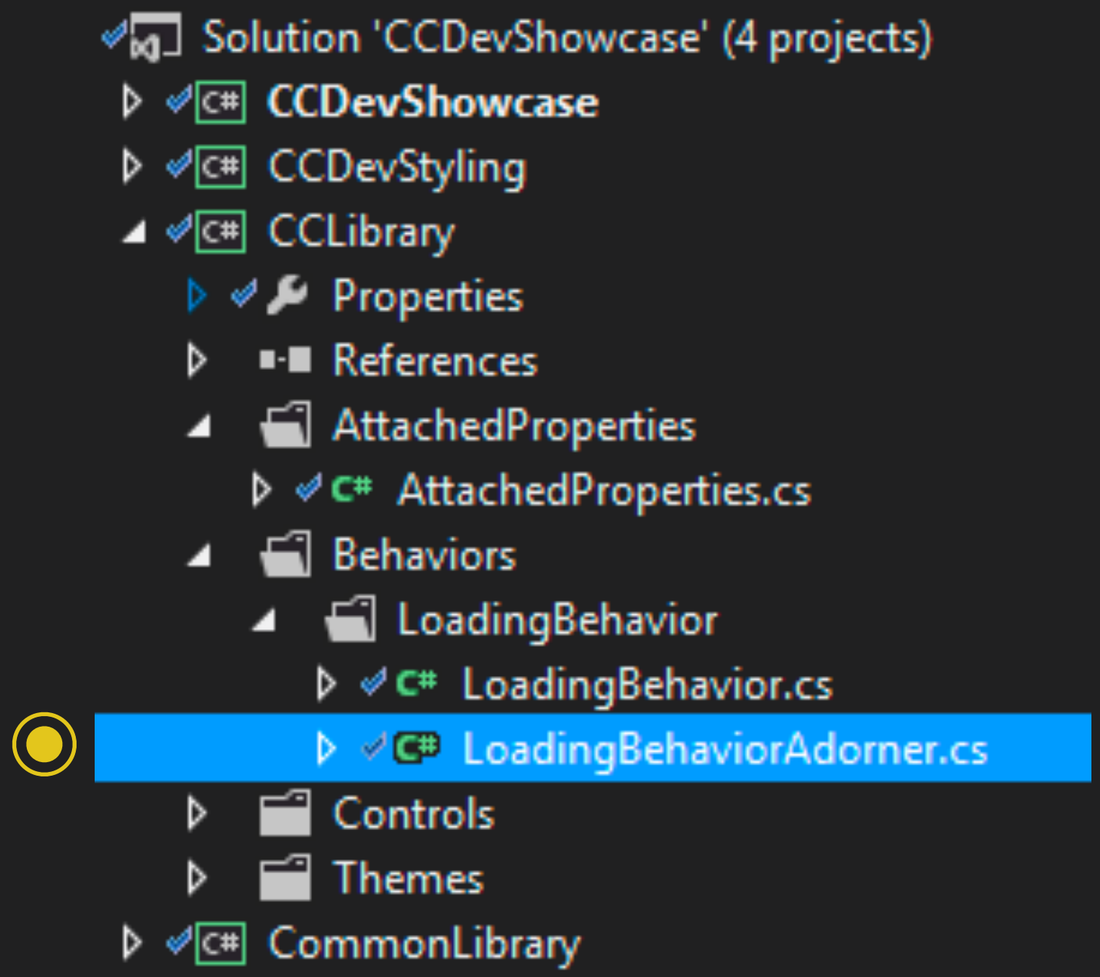
ADORNER LAYER UND ADORNER

Überblick

- AdornerLayer = zusätzliche **Zeichen/Renderebene**
- Möglichkeit Inhalte über einem Control darzustellen

- Adorner repräsentieren die Inhalte in einem AdornerLayer
- Adorner werden in den nächst liegenden AdornerDecorator platziert (*Default von Window*)

- Adorner selbst müssen implementiert werden!
  - Beispielsweise Darstellung eines Button im Adorner



› Klasse anlegen

Implementieren eines Adorners mit Content Data Template 1/5

```
/// <summary> ...
public class LoadingBehaviorAdorner : Adorner...
```

- Ableiten von Adorner
- Abstrakte Klasse
- Basisimplementierung

Implementieren eines Adorners mit Content Data Template 2/5

(LoadingBehaviorAdorner.cs)

```
/// <summary> ...
public class LoadingBehaviorAdorner : Adorner
{
    #region Members

    private List<UIElement> logicalChilds;
    private ContentControl contentControl;

    #endregion

    #region

    #region Properties

    /// <summary> ...
    public object ContentDataContext
    {
        get { return contentControl.DataContext; }
        set { contentControl.DataContext = value; }
    }

    /// <summary> ...
    public object Content
    {
        get { return contentControl.Content; }
        set { contentControl.Content = value; }
    }

    /// <summary> ...
    public DataTemplate ContentTemplate
    {
        get { return contentControl.ContentTemplate; }
        set { contentControl.ContentTemplate = value; }
    }

    #endregion
}
```

› Member definieren

› Properties definieren

Implementieren eines Adorners mit Content Data Template 3/5

(LoadingBehaviorAdorner.cs)

```
/// <summary> ...
protected override Size ArrangeOverride(Size finalSize)
{
    contentControl.Arrange(new Rect(finalSize));
    return finalSize;
}

/// <summary>
/// Get the visual children count.
/// </summary>
protected override int VisualChildrenCount
{
    get { return logicalChilds.Count; }
}

/// <summary>
/// Get children by index.
/// </summary>
/// <param name="index"></param>
/// <returns></returns>
protected override Visual GetVisualChild(int index)
{
    return logicalChilds[index];
}

/// <summary>
/// Enumerator for logical children.
/// </summary>
protected override IEnumerator LogicalChildren
{
    get { return logicalChilds.GetEnumerator(); }
}
```

- Überschreiben von Methoden zur korrekten Darstellung des Contents

Implementieren eines Adorners mit Content Data Template 4/5

(LoadingBehaviorAdorner.cs)

```
/// <summary> ...
public LoadingBehaviorAdorner(UIElement adornedElement) : base(adornedElement)
{
    contentControl = new ContentControl() { Background = Brushes.Transparent };

    logicalChilds = new List<UIElement>(1);
    logicalChilds.Add(contentControl);

    this.AddVisualChild(contentControl);
    this.AddLogicalChild(contentControl);
}
```

› Elemente initialisieren

Implementieren eines Adorners mit Content Data Template 5/5

(LoadingBehaviorAdorner.cs)

```
namespace CCDevShowcase.View
{
    /// <summary>
    /// Interaction logic for MainView.xaml
    /// </summary>
    public partial class MainView : UserControl
    {
        public MainView()
        {
            InitializeComponent();

            IsLoadingToggleButton.Click += IsLoadingToggleButton_Click;
        }

        void IsLoadingToggleButton_Click(object sender, RoutedEventArgs e)
        {
            var adornerLayer = AdornerLayer.GetAdornerLayer(Chart);
            var adorner = new LoadingBehaviorAdorner(Chart);

            adorner.Content = new ProgressBar() { Minimum = 0, Maximum = 100, Value = 50 };

            adornerLayer.Add(adorner);
        }
    }
}
```

Adorner manuell verwendet

(MainView.xaml.cs)

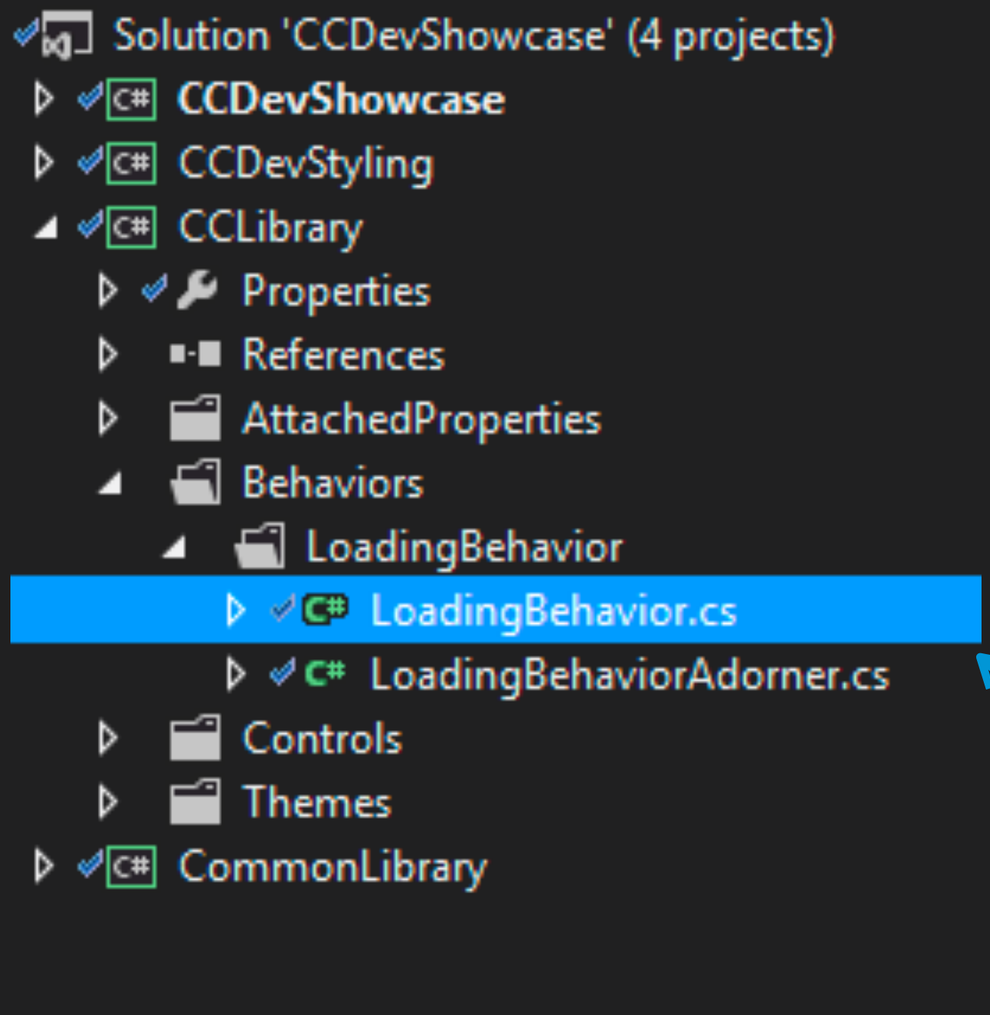
BEHAVIOR

Idee

- Behavior verwenden und den Source-Code zur Adorner-Erzeugung und Verwaltung zentral zu definieren
- Per Behavior die Circular ProgressBar innerhalb des Adorner darstellen
- Behavior steuert das Verhalten

Behavior

- Kapselt eine gewisse Funktionalität und Logik in eine wiederverwendbare Klasse
- Kann an eine beliebiges Control „angeheftet“ werden
- Repräsentiert durch die Klasse *Behavior<T>*
  - wobei *T* = Typ des AssociatedObjects
- Override Methoden *OnAttached()*, *OnDetaching()* und Property *AssociatedObject* bieten dem Entwickler volle Flexibilität sowie Kontrolle



› Klasse anlegen

Implementieren einer Behavior 1/6

```
/// <summary> ...
public class LoadingBehavior : Behavior<FrameworkElement>
{
    Members

    DependencyProperties

    PropertyChangedCallbacks

    Events

    Handlers

    Ctor

    Init/Cleanup

    #region OverrideMethods

    /// <summary> ...
    protected override void OnAttached()
    {
        base.OnAttached();

        this.AssociatedObject.Dispatcher.BeginInvoke(new Action(() =>
        {
            Init();
        }), DispatcherPriority.Loaded);
    }

    /// <summary> ...
    protected override void OnDetaching()
    {
        base.OnDetaching();

        CleanUp();
    }
```

- Ableiten von Behavior<T> wobei T den Typ des AssociatedObjects widerspiegelt
- Überschreiben der Methoden OnAttached(), OnDetaching()

Implementieren einer Behavior 2/6

(LoadingBehavior.cs)

```
/// <summary> ...
public static readonly DependencyProperty LoadingContentProperty = DependencyProperty.Register(
    "LoadingContent",
    typeof(object),
    typeof(LoadingBehavior),
    new FrameworkPropertyMetadata(null, LoadingContentPropertyChanged));

/// <summary> ...
public static readonly DependencyProperty LoadingTemplateProperty = DependencyProperty.Register(
    "LoadingTemplate",
    typeof(DataTemplate),
    typeof(LoadingBehavior),
    new FrameworkPropertyMetadata(null, LoadingContentTemplatePropertyChanged));

/// <summary> ...
public static readonly DependencyProperty LoadingDataContextProperty = DependencyProperty.Register(
    "LoadingDataContext",
    typeof(object),
    typeof(LoadingBehavior),
    new FrameworkPropertyMetadata(null, LoadingDataContextPropertyChanged));

/// <summary> ...
public static readonly DependencyProperty IsLoadingProperty = DependencyProperty.Register(
    "IsLoading",
    typeof(bool),
    typeof(LoadingBehavior),
    new FrameworkPropertyMetadata(false, IsLoadingPropertyChanged));
```

- Anlegen der Properties & Property Changed Callbacks

Implementieren einer Behavior 3/6

(LoadingBehavior.cs)

- Unser Adorner verwenden
- Get AdornerLayer

```
/// <summary> ...
private void Init()
{
    isAttached = true;

    loadingBehaviorAdornerLayer = AdornerLayer.GetAdornerLayer(this.AssociatedObject);
    loadingBehaviorAdorner = new LoadingBehaviorAdorner(this.AssociatedObject);

    UpdateAdorner();
    UpdateAdornerContent();
    UpdateAdornerContentTemplate();
    UpdateAdornerContentDataContext();
}
```

- Adorner anlegen
  - Parameter = AdorneredElement

Implementieren einer Behavior 4/6

(LoadingBehavior.cs)

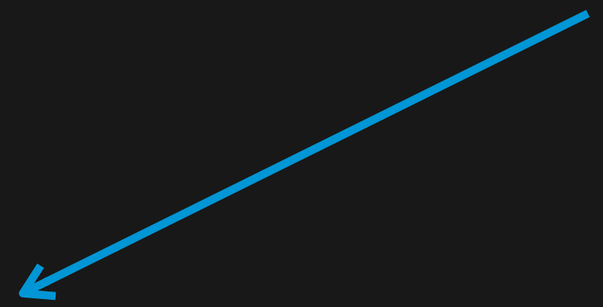
```
/// <summary> ...
public static readonly DependencyProperty IsLoadingProperty = DependencyProperty.Register(
    "IsLoading",
    typeof(bool),
    typeof(LoadingBehavior),
    new FrameworkPropertyMetadata(false, IsLoadingPropertyChanged));

private static void IsLoadingPropertyChanged(DependencyObject d, DependencyPropertyChangedEventArgs e)
{
    LoadingBehavior self = d as LoadingBehavior;

    if (!self.isAttached)
        return;

    self.UpdateAdorner();
}

private void UpdateAdorner()
{
    if (!IsLoading)
        loadingBehaviorAdornerLayer.Remove(loadingBehaviorAdorner);
    else
        loadingBehaviorAdornerLayer.Add(loadingBehaviorAdorner);
}
```

- Hinzufügen/Löschen des Adorners aus dem dazugehörigen AdornerLayer

Implementieren einer Behavior 5/6

(LoadingBehavior.cs)

- LoadingBehavior zum gewünschten Element hinzufügen



```
<Grid x:Name="Chart"
      Grid.Row="1"
      Grid.Column="1">
    <i:Interaction.Behaviors>
        <LoadingBehavior IsLoading="{Binding IsLoading}" LoadingTemplate="{StaticResource LoadingTemplate}" />
    </i:Interaction.Behaviors>
    <PointChart x:Name="MyPointChart"
                Panel.ZIndex="100"
                ItemContainerStyle="{StaticResource MyPointChartItemStyle}"
                ItemsSource="{Binding Points}"
                SelectedItem="{Binding SelectedItem}"
                SelectionChangedCommand="{Binding SelectionChangedCommand}"
                SelectionMode="Extended" />
</Grid>
```

Implementieren einer Behavior 6/6

(MainView.xaml)

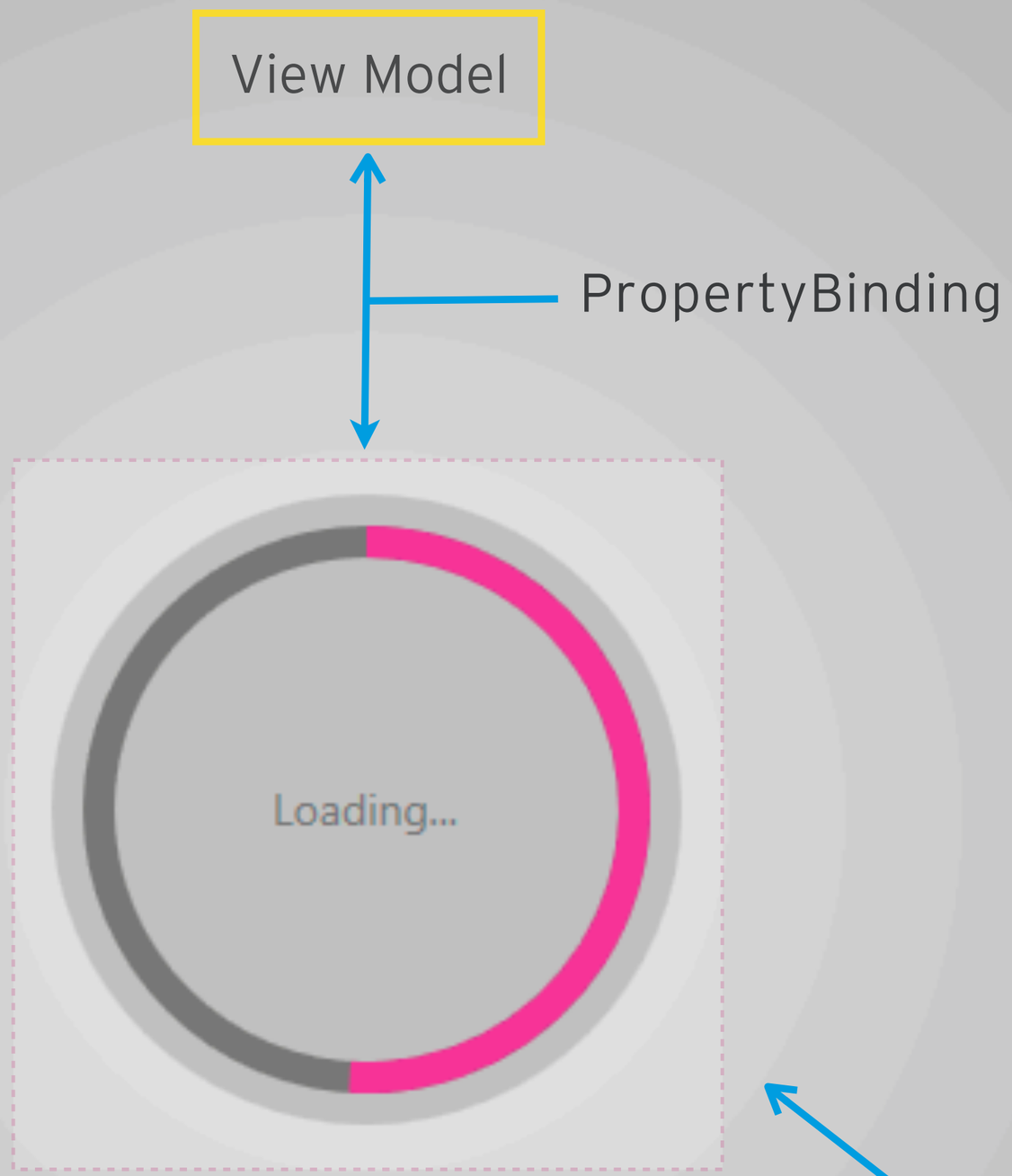
```xaml
<DataTemplate x:Key="LoadingTemplate">
    <ProgressBar Maximum="100"
                 Minimum="0"
                 Value="{Binding ProgressValue}" />
</DataTemplate>
```

- Binding an ViewModel Property

ViewModel Kommunikation

(DataTemplates.xaml)

LOADING INDICATOR ANALYSE



- IsLoading
- LoadingContent
- LoadingTemplate
- LoadingDataContext

Beliebiger Content

BEHAVIOR VS. ATTACHED PROPERTIES

AttachedProperties

- Da AttachedProperties statisch angelegt werden, sollten sie nur zur:
  - weiteren Eigenschaftsbeschreibung (Watermark, CornerRadius usw.)
  - oder zur Realisierung überschaubarer Zusatzfunktionalität (ClearCommand) verwendet werden.
- **Nachteil** Referenz Problem

Behavior

- Behaviors hingegen bieten die Möglichkeit komplexeren Code in eine wiederverwendbare Klassen zu kapseln (LoadingBehavior)
- Mittels *AssociatedObject*, *AssociatedType*, *OnAttached()*, *OnDetaching()* möglichst große Flexibilität und Komfort.

DA WAR DOCH NOCH WAS…

André Lanninger
UIDeveloper

Datagraph 11

ChangeStyle

Load

62

- Bubble 18
  0, 20
- Bubble 19
  10, 15
- Bubble 20
  20, **62**
- Bubble 21
  30, 10
- Bubble 22
  40, 27
- Bubble 23
  50, 77
- Bubble 24
  60, 73
- Bubble 25
  70, 90
- Bubble 26
  80, 33

PointChart

FAZIT

Fazit

- Denken Sie an einen benutzer-zentrierten Designprozess
- Schätzen Sie gutes UI Development
- Reizen Sie die Möglichkeiten von WPF aus! (Keine halben Sachen)
- Legen Sie Wert auf Struktur und Konsistenz während der Entwicklung

FRAGEN?

VIELEN DANK.

ERGOSIGN

VIELEN DANK.

www.ergosign.de
contact@ergosign.de

ERGOSIGN

Ergosign GmbH
Europaallee 12
66113 Saarbrücken
Germany

T +49 681 988412-0
F +49 681 988412-10

Ergosign GmbH
Bernhard-Nocht-Straße 109
20359 Hamburg
Germany

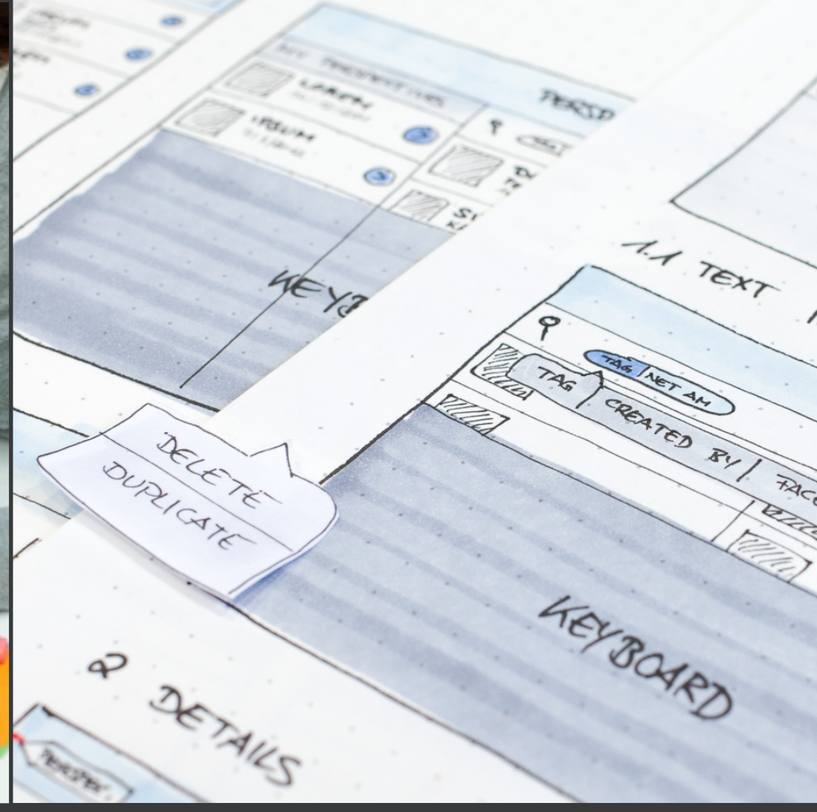
T +49 40 3179868-0
F +49 40 3179868-10

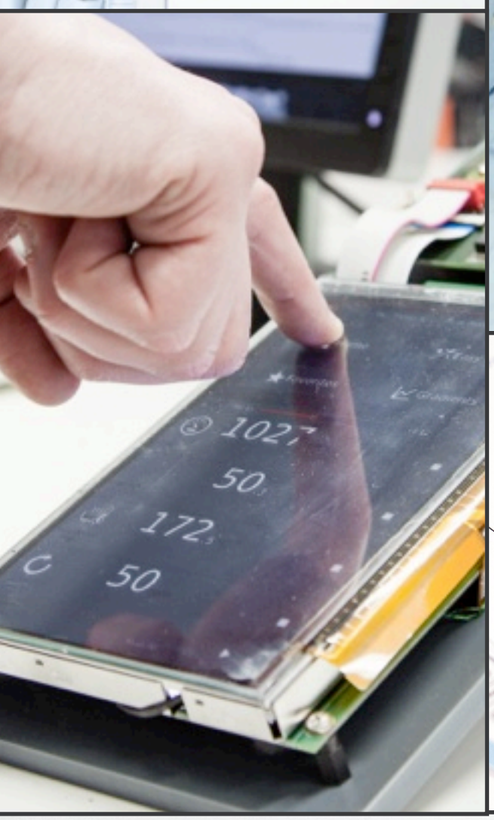
Ergosign GmbH
Adams-Lehmann-Straße 44
80797 München
Germany

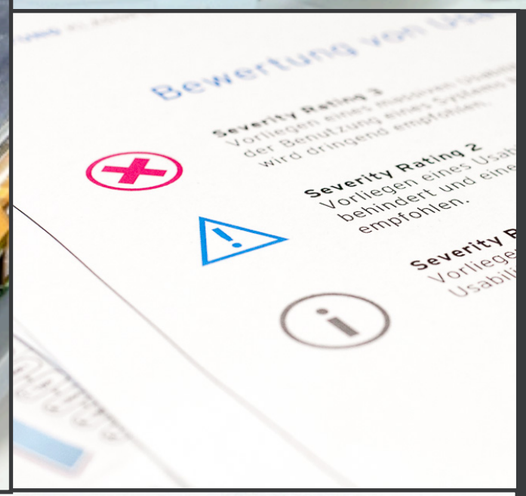
T +49 89 6890607-0
F +49 89 6890607-10

Ergosign Switzerland AG
Badenerstrasse 808
8048 Zürich
Switzerland

T +41 44 54293-04
F +41 44 54293-07







Ein benutzerzentrierter Prozess ist Voraussetzung für das Design einer positiven **User Experience**.