zenon
by COPA-DATA

Manuale di zenon

Elemento WPF di zenon

v.8.20

© 2020 Ing. Punzenberger COPA-DATA Srl

Tutti i diritti riservati.

La distribuzione e la copia di questo documento - indifferentemente dal metodo usato - è consentita solo con il permesso scritto della società COPA-DATA. I dati tecnici servono solo per la descrizione del prodotto e non sono caratteristiche garantite in senso legale. Ci riserviamo il diritto di apportare modifiche - anche per quanto riguarda gli aspetti tecnici.

Indice

1 Benvenuti nell'help di COPA-DATA 5

2 Elemento WPF 5

3 Nozioni di base 6

3.1 WPF nella visualizzazione di processo 7

3.2 Assembly referenziati 8

3.3 Workflow 10

3.3.1 Workflow con Microsoft Expression Blend 11

3.3.2 Workflow con Adobe Illustrator 11

4 Guida per i progettisti 12

4.1 Workflow con Microsoft Expression Blend 12

4.1.1 Creare un pulsante come file XAML con Microsoft Expression Blend 12

4.2 Workflow con Adobe Illustrator 16

4.2.1 Illustrazione bargraf 16

4.2.2 Esportazione WPF 18

4.2.3 Animazione in Blend 20

5 Guida per gli sviluppatori 24

5.1 Creazione di un semplice controllo utente WPF con funzionalità code behind 24

5.2 Debug del controllo utente WPF a Runtime 29

5.3 Scambio di dati fra zenon e controlli utenti WPF 34

5.3.1 Scambio di dati via Dependency Properties 34

5.3.2 Scambio di dati via VSTA 39

5.4 Accesso al modello oggetto zenon (Runtime) da un controllo utente WPF 41

5.4.1 Accesso via VSTA "Variable Link" 41

5.4.2 Accesso via Marshalling 46

6 Progettazione nell´Editor di zenon 49

6.1 CDWPF-file (file collection) 49

6.2 Creare un elemento WPF 50

6.3 Configurazione del collegamento 51

6.3.1 Proprietà 53

6.3.2 Eventi 59

6.3.3 Trasformazione 61

6.4 Validità dei file XAML 63
6.5 Elementi predefiniti 65
6.5.1 Orologio analogico - AnalogClockControl 66
6.5.2 Bargraf verticale - VerticalBargraphControl 66
6.5.3 Barra stato di avanzamento - ProgressBarControl 67
6.5.4 COMTRADE-Viewer 68
6.5.5 Diagramma - Energy class 81
6.5.6 Diagramma di Pareto 82
6.5.7 Visualizzazione circolare - CircularGaugeControl 85
6.5.8 Diagramma di Sankey 89
6.5.9 Visualizzazione della temperatura - TemperatureIndicatorControl 91
6.5.10 Slider universale - UniversalReglerControl 92
6.5.11 Diagramma a cascata 94
6.5.12 BACnet WPF Control 96
6.6 Visualizzazione di elementi WPF in zenon Web Client 112
6.6.1 Configurazione nell´Editor di zenon: 112
6.6.2 Codice VSTA (complesso) 113
6.6.3 Codice VSTA (semplificato) 114
6.7 Esempi: integrazione di WPF in zenon 116
6.7.1 Implementare un bargraf come WPF-XAML in zenon. 116
6.7.2 Implementare un pulsante come WPF-XAML in zenon. 120
6.7.3 Implementare DataGrid Control in zenon 126
6.8 Gestione degli errori 133

1 Benvenuti nell'help di COPA-DATA

ZENON VIDEO-TUTORIAL

Esempi pratici di progettazione con zenon si trovano nel nostro canale YouTube (https://www.copadata.com/tutorial_menu). I tutorial sono raggruppati per tema e forniscono una panoramica di come si lavora con i diversi moduli di zenon. Tutti i tutorial sono disponibili in inglese.

GUIDA GENERALE

Se mancano informazioni in questo capitolo dell'help o se avete richieste di aggiunte, contattate documentation@copadata.com via E-Mail.

SUPPORTO TECNICO ALLA PROGETTAZIONE

Per domande relative a progetti concreti, si prega di contattare il Customer Service via E-Mail all'indirizzo support@copadata.com.

LICENZE E MODULI

Se avete bisogno di moduli aggiuntivi o licenze, il nostro staff di sales@copadata.com sarà lieto di assistervi.

2 Elemento WPF

Con l´elemento dinamico **WPF** si possono implementare e visualizzare file WPF/XAML validi in zenon.

Nota: nell´Editor di zenon non viene visualizzato il tooltip standard per l´elemento WPF se è collegato un file **.wpf**. Nel Runtime di zenon, inoltre, non viene supportato il tooltip zenon per gli elementi WPF.

Info

Tutti i marchi e i nomi dei prodotti in questa documentazione sono marchi o marchi registrati dei rispettivi proprietari.

3 Nozioni di base

XAML

XAML è l'acronimo di **Extensible Application Markup Language** e si pronuncia [ˈzæːməl]. Questo linguaggio di markup basato su XML sviluppato da Microsoft definisce elementi grafici, animazioni, trasformazioni, rappresentazione di sfumature di colore ecc., nelle interfacce Silverlight e WPF. L´uso di XAML permette di separare in modo rigoroso design e programmazione. Ad esempio, il designer prepara l'interfaccia grafica e crea animazioni di base, che vengono poi rilevate dallo sviluppatore/progettista che crea la logica dell'applicazione.

WPF

WPF è l'acronimo di **Windows Presentation Foundation**, e definisce un framework grafico che è una parte di Windows .NET Framework.

- Il WPF fornisce un modello completo per il programmatore.
- XAML descrive, basandosi su XML, la gerarchia delle interfacce come linguaggio markup. A seconda della struttura del file XAML, è possibile collegare proprietà, eventi e trasformazioni di elementi WPF con variabili e funzioni di zenon.
- Il Framework riunisce le diverse aree di una presentazione, come interfaccia utente, disegno, grafica, audio, video, documenti e tipografia.

Per l´esecuzione in zenon è necessaria la versione 4.6.2 o successiva di Microsoft .NET Framework.

Info

Trasparenza

Affinché i controlli WPF per cui è stato definito uno sfondo trasparente, siano visualizzati in modo trasparente, devono essere soddisfatti i seguenti requisiti sul computer (sia per l´Editor che per il Runtime):

- Il sistema operativo deve essere almeno Windows 8.1.
- Deve essere installata almeno la versione 4.6.2 di .NET Framework.

Con Windows 7 o 8, i controlli WPF non vengono visualizzati trasparenti. Le aree trasparenti vengono riempite con il colore impostato come colore di sfondo nell´immagine di zenon.

3.1 WPF nella visualizzazione di processo

XAML rende possibili diverse opzioni grafiche per zenon, Gli elementi di visualizzazione e gli elementi dinamici possono essere adattati graficamente indipendentemente dalla progettazione. Ad esempio, le illustrazioni complesse vengono prima create dai designer e poi importate come file XAML in zenon e collegate alla logica desiderata. Ci sono molte possibilità di applicazione concreta:

ELEMENTI DINAMICI IN ANALOG-LOOK.

La grafica non deve più essere disegnata in zenon, ma può essere importata direttamente come file XAML. Ciò consente l'uso di elementi complessi, disegnati in modo complesso nella visualizzazione del processo. Riflessi, ombreggiature, effetti 3D ecc, ad esempio, vengono supportati come grafica. Gli elementi, adattati al rispettivo ambiente industriale, permettono all´operatore una gestione intuitiva, analoga alla gestione tramite i comandi sul macchinario.

ILLUSTRAZIONI COMPLESSE PER UNA GESTIONE INTUITIVA

L´implementazione di elementi di visualizzazione basati su XAML migliora la veste grafica dei progetti e consente una visualizzazione di processo più semplice e chiara. Gli elementi, ottimizzati in fatto di

usabilità, facilitano la gestione. La visualizzazione chiara dei dati semplifica la comprensione di contenuti complessi. La possibilità flessibile di modifica ed adeguamento dei singoli elementi aumenta il comfort della gestione. Ciò consente al progettista di determinare in modo indipendente i valori, le scale e le unità di visualizzazione.

CHIARA PRESENTAZIONE DEI DATI E RIEPILOGHI

Elementi di visualizzazione raggruppata consentono di visualizzare in modo chiaro i dati di processo più importanti, in modo che l'operatore dell'impianto sia sempre informato sulle sequenze di processo attuali. Analisi grafiche, visualizzazione valori e slider possono essere raggruppati in un solo elemento, consentendo così una gestione veloce e semplice.

VISUALIZZAZIONE DI PROCESSO SPECIFICA PER I DIVERSI SETTORI INDUSTRIALI

Elementi come termometri, scale o bargraf, sono tra gli elementi di base della visualizzazione del processo. XAML consente di adattarli graficamente alle diverse esigenze dei differenti settori industriali. In questo modo, nella visualizzazione del processo sul terminale, gli operatori ritrovano quegli elementi cui sono abituati e che conoscono già per via del loro lavoro con i macchinari.

ADATTAMENTO A CORPORATE DESIGN

Le illustrazioni possono essere adattate alle specifiche di stile dell'azienda, al fine di ottenere un aspetto coerente della visualizzazione del processo. Si possono usare come base gli elementi di controllo standard di zenon, che potranno poi essere modificati adeguando colori, caratteri, stile delle illustrazioni a quelli tipici e caratteristici per una determinata impresa.

3.2 Assembly referenziati

Mediante gli **elementi WPF** possono essere visualizzati non solo gli oggetti standard (rettangoli, grafiche, ecc.), o effetti (gradienti di colore, animazioni, ecc.), ma anche controlli utente personalizzati (con logica nel code behind), che vengono referenziati come assembly.

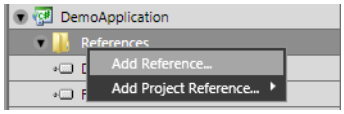
Ad esempio, è possibile creare un controllo utente che ha l´aspetto di un tachimetro che offre proprietà ed effetti visivi speciali, per esempio una proprietà **Value** che fa muovere la lancetta del tachimetro e/o che fa sì che il valore venga visualizzato in una didascalia.

Procedura:

- Disegnare l'aspetto del controllo utente con gli oggetti standard forniti dal WPF.
- Programmare le proprietà e le interazioni.
- Compilare l'intero pacchetto.

  Il risultato è disponibile come assembly .NET.

Questo assembly può essere utilizzato per progetti WPF. Per farlo, deve essere referenziato (collegato) nell'editor WPF (per es: Microsoft Expression Blend). Selezionare l´assembly nella finestra di dialogo di selezione file di zenon:



A partire da questo momento, i controlli utente WPF dell´assembly possono essere selezionati nella toolbox alla voce **Custom UserControls** e utilizzati nel progetto WPF.
Per informazioni più dettagliate su questo tema, consultare il capitolo **Guida dello sviluppatore** (A pagina: 24).

UTILIZZARE ASSEMBLY REFERENZIATI IN ZENON

Per utilizzare un assembly in zenon, questo deve essere reso disponibile come file.
I file collettivi in formato **.cdwpf** gestiscono gli assembly in modo indipendente. Non sono necessarie ulteriori configurazioni. Per i file **.xaml**, invece, gli assembly devono essere aggiunti alla cartella *File*:

- Nella struttura ad albero del progetto, cliccare sul nodo *File*.
- Selezionare *Altri*.
- Nel menù contestuale, selezionare **Aggiungi file...**

  Si apre la finestra di dialogo per la selezione dei file.
- Aggiungere l´assembly desiderato.

Quando si visualizza un file WPF nell´**elemento WPF** (Editor e Runtime), gli assembly vengono caricati da questa cartella. Questo garantisce inoltre che tutti gli assembly referenziati siano disponibili sul computer di destinazione durante il trasferimento dei file Runtime tramite **Trasporto Remoto**.

Un file collettivo (**.cdwpf**) e un file XAML con lo stesso nome possono esistere in parallelo. Tutti gli assembly (***.dll**) di tutti i file collettivi e della cartella *Altri* vengono copiati in una cartella di lavoro. Se esistono diversi assembly con lo stesso nome, viene utilizzato solo quello con la versione file più alta.

Suggerimento

Le DLL appartenenti ad un elemento WPF possono anche essere sostituite anche durante il funzionamento dell´Editor. Il referenziamento avviene tramite un link nel file XAML.

Per sostituire una DLL:

- Chiudere tutte le immagini di zenon che utilizzano l'elemento WPF.
- Chiudere tutti i simboli che utilizzano un elemento WPF.
- In Explorer, sostituire la DLL nella cartella *\wpfache* dei file Editor. Questa cartella si trova nella directory SQL, alla voce: *...\PROJECT-GUID\FILES\zenon\custom\wpfcache*

In alternativa alla sostituzione tramite Explorer, è anche possibile sostituire il file direttamente nell'Editor di zenon. Procedura:

- Nelle impostazioni di progetto di Visual Studio, aumentare la versione del file della DLL.
- Ricostruire la DLL.
- Chiudere tutte le immagini di zenon che utilizzano l'elemento WPF.
- Chiudere tutti i simboli che utilizzano un elemento WPF.
- Nell´Editor di zenon, cancellare la DLL dalla cartella *\File\Altri* e incollare il file con il numero di versione aumentato.

GESTIONE MULTI-PROGETTO

Nella gestione multi-progetto, deve essere usato lo stesso assembly in tutti i progetti. Se in un progetto un assembly viene sostituito da un'altra versione, deve essere sostituito anche in tutti gli altri progetti caricati nell'Editor o nel Runtime.

3.3 Workflow

La tecnologia WPF/XAML rende possibili nuovi workflow nella visualizzazione di processo. La separazione tra design e funzionalità assicura una chiara divisione dei compiti tra progettista e designer; le specifiche relative al design possono essere facilmente soddisfatte utilizzando progetti esistenti che non devono essere più modificati dal progettista.

Alla creazione di elementi WPF in zenon partecipano le seguenti persone:

- Designer
  - Concepisce l´aspetto grafico degli elementi.
  - Fornisce l'editing grafico per MS Expression Design

- MS Expression Blend Operator
  - Si occupa dell´animazione degli elementi.
  - Crea variabili in zenon per l'animazione degli elementi WPF, a cui il progettista può accedere.
- Progettista
  - Implementa gli elementi in zenon.
  - Crea logica e funzionalità.

Si distingue fra:

- Workflow con Microsoft Expression Blend (A pagina: 11)
- Workflow con Adobe Illustrator (A pagina: 11)

3.3.1 Workflow con Microsoft Expression Blend

Quando si utilizza Microsoft Expression Blend, la creazione di un elemento WPF avviene in quattro passaggi:

1. Illustrazione dell´elemento in **MS Expression Blend** (A pagina: 12)
2. Aprire l´elemento in **MS Expression Design** ed esportarlo come WPF.
3. Animazione in **MS Expression Blend** (A pagina: 12)
4. Integrazione in zenon (A pagina: 120)

Per un esempio di come creare un elemento WPF utilizzando Microsoft Expression Blend, vedere il capitolo Creare un pulsante come file XAML usando Microsoft Expression Blend (A pagina: 12).

3.3.2 Workflow con Adobe Illustrator

Con **Adobe Illustrator** è possibile utilizzare il seguente workflow:

1. Illustrazione degli elementi in **Adobe Illustrator** (A pagina: 16)
2. Importazione di file **.ai** ed elaborazione in **MS Expression Design** (A pagina: 18)
3. Esportazione WPF da **MS Expression Design** (A pagina: 18)
4. Animazione in **MS Expression Blend** (A pagina: 20)
5. Integrazione in zenon (A pagina: 116)

Un esempio di questa procedura si trova nel capitolo Workflow con Adobe Illustrator (A pagina: 16).

4 Guida per i progettisti

Questa sezione fornisce informazioni sulla corretta creazione di file WPF in Microsoft Expression Blend e Adobe Illustrator. Le esercitazioni per la creazione di un elemento pulsante (A pagina: 12) e di un elemento bargraf (A pagina: 16) mostrano come creare in pochi passaggi file WPF completamente funzionali per zenon partendo da grafiche già esistenti.

A questo scopo vengono utilizzati i seguenti strumenti:

- Adobe Illustrator CS3 (AI)
- Microsoft Expression Design 4 (ED)
- Microsoft Expression Blend 4 (EB)
- zenon

> **Info**
>
> Se in WPF devono essere utilizzati oggetti referenziati (assembly), fare riferimento alle note del capitolo Assembly referenziati (A pagina: 8).

4.1 Workflow con Microsoft Expression Blend

Con Microsoft Expression Blend, un elemento WPF viene:

- Illustrato
- Convertito in formato WPF tramite **MS Expression Design**
- Animato

L'esempio seguente mostra l'illustrazione e la conversione di un elemento pulsante in un file XAML.

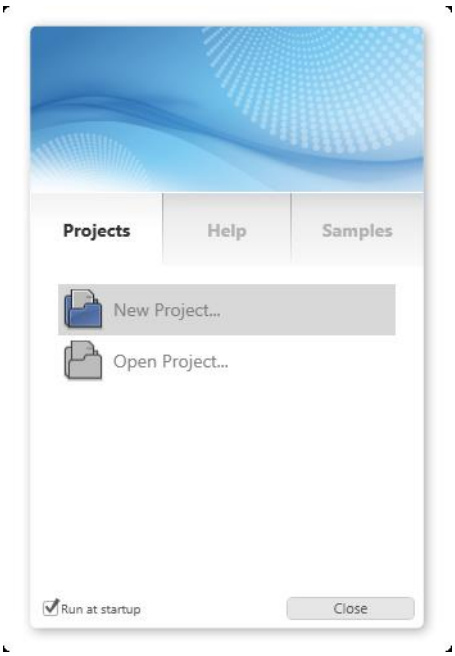
Nota: una versione di prova di "Microsoft Expression Blend" è disponibile per il download nel sito Web di Microsoft.

4.1.1 Creare un pulsante come file XAML con Microsoft Expression Blend

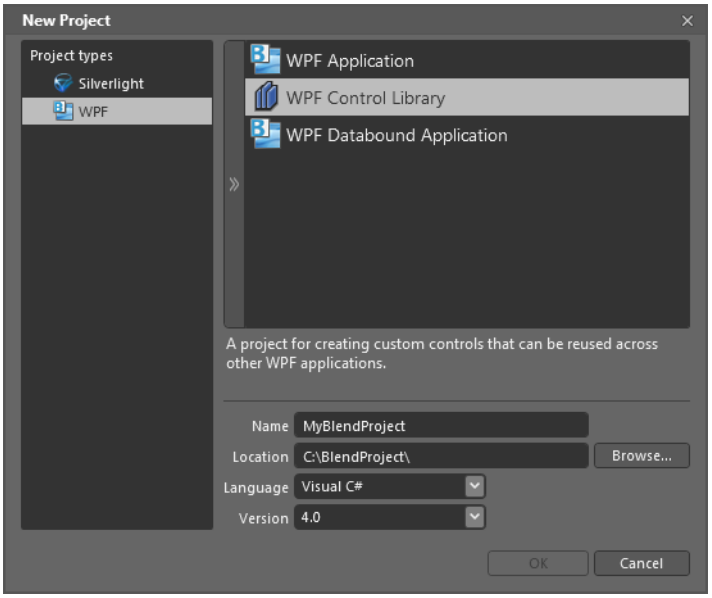
CREARE UN PULSANTE

1. Avviare Expression Blend.

2. Selezionare l´opzione **New Project**



3. Come tipo di progetto, selezionare *WPF*..
4. Assegnare qualsiasi percorso e nome (per es. MyBlendProject)



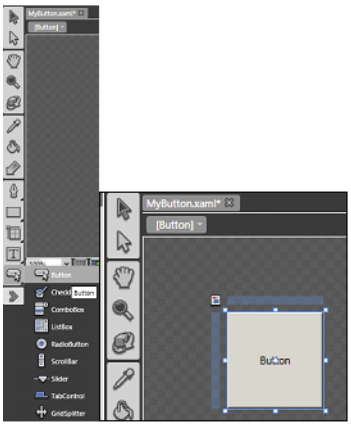
Le impostazioni **Language** e **Version** possono essere ignorate perché non deve essere programmata nessuna funzionalità.

5. Dopo che la finestra di dialogo è stata chiusa con un clic su **OK**, Microsoft Blend crea un nuovo progetto con le impostazioni selezionate. Expression Blend aggiunge un file XAML vuoto, che contiene già un riferimento di classe.
6. Eliminare il file CS associato al file XAML tramite il menù contestuale.

7. Rinominare il file XAML da **MainControl.xaml** a **MyButton.xaml**.
8. La dimensione development del file è impostata su 640 per 480 pixel per impostazione predefinita e deve ancora essere modificata:
   a) Passare alla visuale **XAML**
   b) Correggere la dimensione portandola a 100 volte 100 pixel.
   c) Cancellare il riferimento di classe x:Class="MyBlendProject.MyButton".

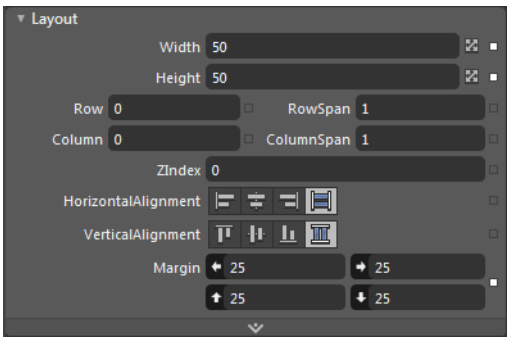
```
MyButton.xaml
1  <UserControl
2      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
5      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
6      mc:Ignorable="d"
7      x:Name="UserControl"
8      d:DesignWidth="100" d:DesignHeight="100">
9
10     <Grid x:Name="LayoutRoot" />
11
12 </UserControl>
```

9. Passare alla visuale **Design**.

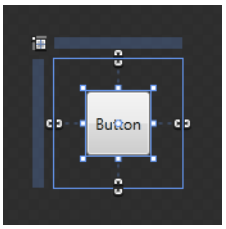


10. Inserire un pulsante tramite la barra degli strumenti.
11. Definire le proprietà.
    - Width: 50
    - Height: 50

- Margins: 25



Con queste impostazioni il pulsante si troverà al centro del controllo.



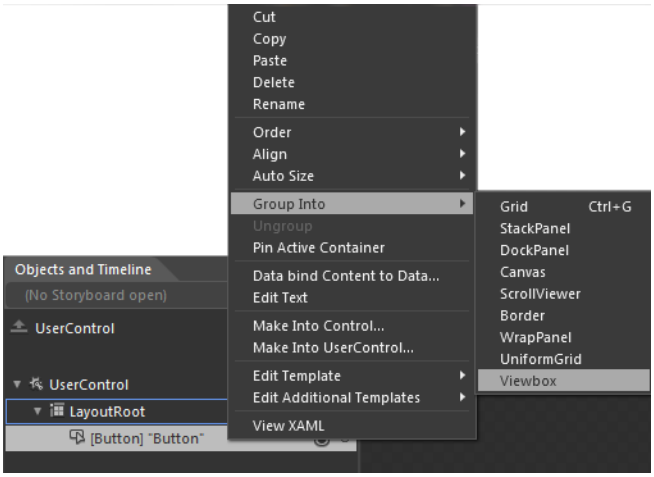
12. Salvare le modifiche e aprire il file per il controllo su Internet Explorer. Si può vedere che il pulsante è visualizzato nella dimensione di 50 x 50 pixel.

RENDERE IL PULSANTE SCALABILE

Se si include questo file in zenon in questo stato, il pulsante avrà sempre la dimensione esatta di 50 x 50 pixel. Poiché il pulsante deve essere utilizzato in modo scalabile, tornare a Expression Blend

1. Selezionare il pulsante nella struttura ad albero.
2. Nel menù contestuale del pulsante, selezionare la voce **Group Into->Viewbox**
3. il pulsante viene inserito in una **Viewbox**.
4. Definire le proprietà della viewbox.
   - Width: Auto
   - Height: Auto

5. Salvare il file.



6. Se ora si apre il file in Internet Explorer, il pulsante viene ridimensionato automaticamente quando si ridimensiona la finestra IE. Questo file ora si adatta automaticamente anche in zenon ai cambiamenti di dimensione dell'**elemento WPF**.

RINOMINARE

Prima di poter implementare il file in zenon, si deve nominare l´**elemento WPF**. Per impostazione predefinita, gli **elementi WPF** non sono denominati in Expression Blend e sono identificati da parentesi quadre e il loro tipo. Si assegna content zenon a content WPF utilizzando il nome degli **elementi WPF**:

- Nella struttura ad albero,modificare il nome:
  - Del pulsante, nominandolo **MyButton**.
  - Della ViewBox, nominandola **MyViewBox**.

Questo pulsante può ora essere incluso come file XAML in zenon (A pagina: 120).

4.2 Workflow con Adobe Illustrator

Quando si utilizza **Adobe Illustrator**, un elemento WPF viene:

- Illustrato in **Adobe Illustrator**.
- Convertito in un WPF in **MS Expression Design**.
- Animato in **MS Expression Blend**.

L´esempio seguente mostra l´illustrazione e la conversione di un elemento bargraf in un file XAML.

4.2.1 Illustrazione bargraf

Un bargraf viene creato in Adobe Illustrator.

1. **AI:elemento di partenza bargraf**

Illustrato in Adobe Illustrator CS3.

2. **AI:Bargraph Vista Percorso in Adobe Illustrator**

- Tutti gli effetti devono essere convertiti (**Object -> Convert appearance**)
- Tutte le linee sono convertite in percorsi (**Object -> Path -> Contour line**)
- Non utilizzare filtri come ombreggiature, sfocature, ecc.

NOTE SULLA COMPATIBILITÀ

Le illustrazioni create con Adobe Illustrator sono fondamentalmente adatte per l'esportazione WPF. Tuttavia, non tutti gli effetti possibili in Illustrator possono essere convertiti in modo corrispondente in Expression Design/Blend. Tenere presente quanto segue:

Effetto	Descrizione
Maschere di ritaglio	Le maschere di ritaglio create in Adobe Illustrator non vengono interpretate correttamente da Expression Design. Per lo più, vengono visualizzate in Blend come delle superfici di color nero. Si consiglia di creare le illustrazioni senza maschere di ritaglio.
Filtri e effetti	Non tutti i filtri di Adobe Illustrator sono adottati in Expression Design in modo corrispondente: Ad esempio, i filtri di sfocatura, ombreggiatura e gli effetti angolari di Illustrator non funzionano in Expression Design. Soluzione: • Con il comando **Oggetto -> Converti aspetto** in Adobe Illustrator, è possibile convertire la maggior parte degli effetti in modo che possano essere letti correttamente da Expression Design. • Gli effetti per gli angoli di Adobe Illustrator vengono interpretati correttamente da MS Design quando vengono convertiti in percorsi in AI.
Campi di testo	Per poter collegare campi di testo a codice, questi devono essere

Effetto	Descrizione
	creati separatamente in Expression Blend. "**Labels**" sono necessari per i testi dinamici; per le informazioni statiche, invece, sono sufficienti semplici "**Campi testo**". La possibilità di creare etichette di testo non è offerta in MS Design. Devono essere create direttamente in MS Blend.
Trasparenze e trasparenze di gruppo	Potrebbero verificarsi problemi relativi all´interpretazione corretta delle impostazioni di trasparenza, in particolare quelle di gruppo, in Adobe Illustrator. MS Expression Blend e MS Expression Design offrono la possibilità di creare nuove impostazioni di trasparenza.
Moltiplicare i livelli	Queste impostazioni di livello in Adobe Illustrator non vengono sempre visualizzate correttamente da MS Expression Blend. Tuttavia, è possibile impostare "**Moltiplica livelli**" direttamente in Expression Design.
Strumenti analogici e posizioni standard	Per preparare la grafica in modo ottimale per l'animazione, le lancette e le manopole devono sempre trovarsi nella posizione di partenza, di solito a *ore 0* o *12:00*. In questo modo i parametri di posizione per le rotazioni ecc. vengono visualizzati correttamente in Blend e l´animazione può avvenire senza la necessità di una conversione dei dati di posizione.

4.2.2 Esportazione WPF

I file WPF sono necessari per l'animazione in Microsoft Expression Blend. Si consiglia Microsoft Expression Design per questa esportazione, perché fornisce buoni risultati e la maggior parte degli effetti Illustrator sono interpretati correttamente.

Nota: Internet offre un plug-in gratuito per l'esportazione diretta di file WPF da Adobe Illustrator. Questo plug-in offre un'esportazione veloce e semplice da Illustrator, ma è meno adatto per l'applicazione corrente a causa delle perdite grafiche. Sono possibili anche differenze di colore rispetto al documento originale.

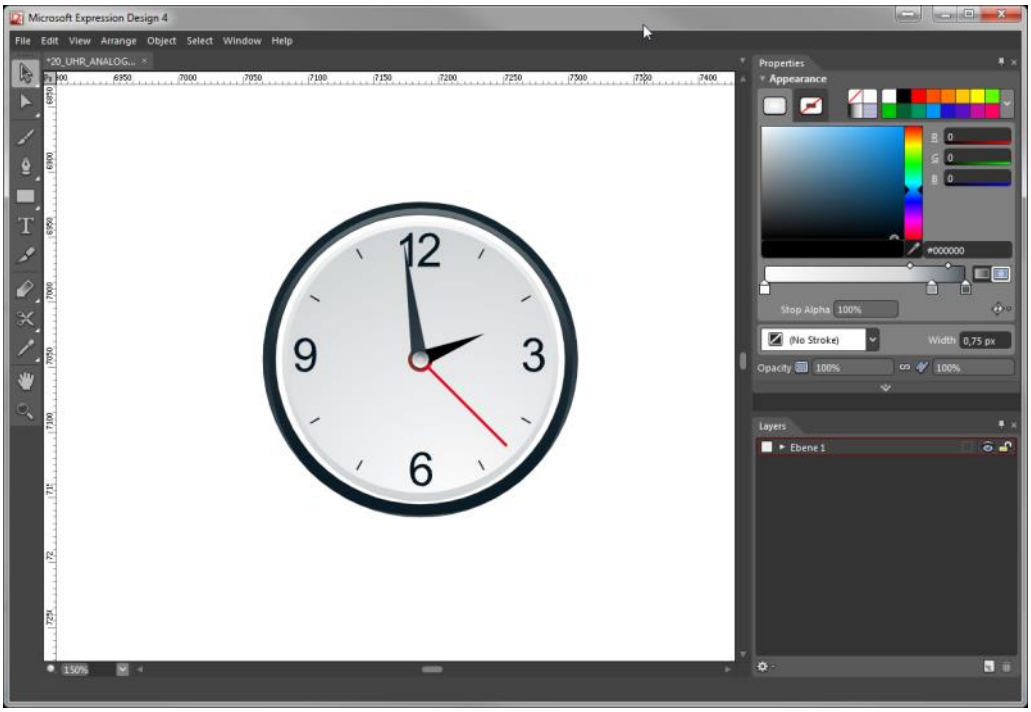
I file in formato **.ai** possono essere importati regolarmente in Expression Design. I percorsi rimangono gli stessi.
Attenzione tuttavia, alcuni effetti Illustrator comuni non possono essere visualizzati correttamente da Expression Design (vedere capitolo Illustrazione (A pagina: 16)).

In 5 passaggi esportiamo l'elemento bargraph già creato:

1. **ED:importazione**

   - Importare il file Illustrator (A pagina: 16) preparato in **Microsoft Expression Design** tramite **File -> Importa**

2. **ED:ottimizzazione**



   - Se il file di origine non è visualizzato correttamente in MS Expression Design, può essere modificato e ottimizzato qui.

3. **ED:selezione**

   - Utilizzare la freccia di **Selezione diretta** in MS Expression Design per evidenziare l'elemento per l'esportazione WPF, in questo caso l'intero orologio.

4. **ED:avviare l´esportazione**

   - Avviare l'esportazione tramite **File -> Esporta**

- Si apre la finestra di dialogo per la configurazione delle impostazioni di esportazione.

5. **ED:impostazioni di esportazione**

- Effettuare le seguenti impostazioni di esportazione:

a) **Formato**: *XAML Silverlight 4 / WPF Canvas*

Assegna sempre un nome agli oggetti: attivare con segno di spunta.

Posiziona gli oggetti raggruppati in un contenitore di layout XAML: attivare con segno di spunta.

b) **Testo:** *Controlli TextBlock modificabili*

c) **Effetti linea:** *rasterizza tutto.*

Il file esportato ha l´estensione **.xaml.** Nella fase successiva, viene preparato e animato (A pagina: 20) in MS Expression Blend.

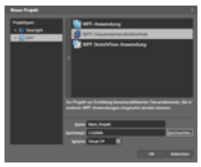
4.2.3 Animazione in Blend

MS Expression Blend:

- Anima i file XAML statici di MS Expression Design.
- Crea variabili per controllare gli effetti che possono essere indirizzati da zenon.

In 13 passaggi si passa dal file XAML statico all'elemento animato, che può essere incluso in zenon:

1. **EB: creare un progetto**



a) Aprire Microsoft Expression Blend.

a) Creare un nuovo progetto.

b) Selezionare **WPF-> WPF- Control Library** come **tipo di progetto**.

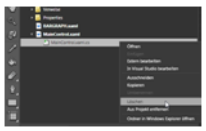
c) Assegnare un nome (nel nostro tutorial: **My_Project**)

d) Selezionare una posizione di memorizzazione.

e) Selezionare una lingua (nel nostro tutorial: C#)

f) Selezionare un framework supportato da zenon

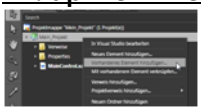
2. **EB:cancellare MainControl.xaml.cs**



a) Passare a **MainControl.xaml.cs**.

b) Cancellare questo file con il comando **Elimina** del menù contestuale.

3. **EB: aprire il file XAML esportato**

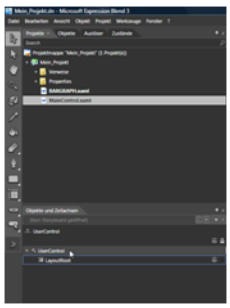


a) Aprire il menù contestuale di **My_Project** (con il tasto destro del mouse).

b) Selezionare **Aggiungi elemento esistente...**

c) Selezionare il file XAML esportato da Microsoft Expression Design per aprirlo in Microsoft Expression Blend

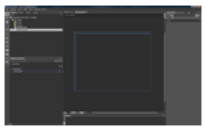
4. **EB:aprire MainControl.xaml**



a) Aprire **MainControl.xaml** generato automaticamente.

b) Nell´area **Oggetti e sequenza temporale**, passare alla voce **UserControl**.

5. **EB:adattare il codice XAML**



a) Cliccare con il tasto destro del mouse su **UserControl**.

b) Nel menù contestuale, selezionare **Visualizza XAML**.

c) Cancellare le righe 7 e 9 del codice XAML:

```
x:Class="Mein_Projekt.MainControl"

d:DesignWidth="640" d:DesignHeight="480"
```

6. **EB: controllare il codice XAML**

- Ora il codice XAML dovrebbe essere il seguente:

```
<UserControl

    xmlns=http://schemas.microsoft.com/winfx/2006/xaml/presentation
    xmlns:x=http://schemas.microsoft.com/winfx/2006/xaml
    xmlns:d=http://schemas.microsoft.com/expression/blend/2008
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"



    mc:Ignorable="d"
    x:Name="UserControl">



    <Grid x:Name="LayoutRoot"/>

</UserControl>
```

7. **EB:copiare elementi**

a) Aprire il file XAML importato da Expression Design.

b) Selezionare tutti gli elementi.

c) Nel menù contestuale, selezionare **Copia**.

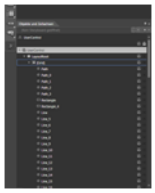
d) Tornare al file XAML creato automaticamente.

8. **EB:inserisci elementi**

a) Cliccare con il tasto destro del mouse su **Layout Root**.

b) Selezionare **Incolla**.

9. **EB:personalizzare il tipo di layout**

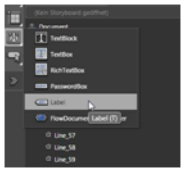


a) Cliccare con il tasto destro del mouse su **LayoutRoot -> Modifica tipo di layout -> Viewbox**

b) Adesso la struttura dovrebbe avere il seguente aspetto: **UserControl -> LayoutRoot -> Grid -> Elementi**

c) Assegnare un nome a **LayoutRoot** e **Grid** cliccando due volte sul nome.

10. **EB: testi e valori**



- I testi dinamici e statici sono disegnati con campi di testo.
- Valori (numeri) vengono emessi con **Labels (etichette)**.

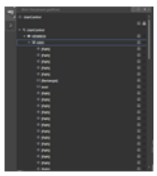
11. **EB:inserire labels (etichette)**

- Label (etichette) sostituiscono i numeri che devono essere collegati in seguito con variabili Int (deve essere fatto per tutti gli elementi numerici).

12. **EB:impostare proprietà**

- Per visualizzare il 100%, impostare la proprietà **MaxHeight** dell'elemento di visualizzazione bargraf sul valore *341* (l'altezza massima dell'elemento di visualizzazione è *340*).

13. **EB: preparare all´uso in zenon**



a) Eliminare tutti i nomi (i nomi possono essere assegnati solo agli elementi che devono essere indirizzati tramite zenon).

b) Salvare il file XAML con un nome qualsiasi.

c) Implementare il file XAML in zenon (A pagina: 116)

Suggerimento per eseguire un controllo: se il file XAML viene visualizzato correttamente in Microsoft Internet Explorer e si adatta anche alle dimensioni della finestra di Internet Explorer, verrà utilizzato correttamente anche in zenon.

5 Guida per gli sviluppatori

In questa sezione viene descritta la creazione di un semplice controllo utente WPF con funzionalità code behind utilizzando Microsoft Visual Studio e il debug di questo controllo utente a Runtime.

A questo scopo vengono utilizzati i seguenti strumenti:

- Microsoft Visual Studio 2015.
- zenon

> **Info**
>
> Suggerimento: utilizzare Microsoft Visual Studio versione 2012 o successiva. In questa versione XAML Designer è integrato meglio.

5.1 Creazione di un semplice controllo utente WPF con funzionalità code behind

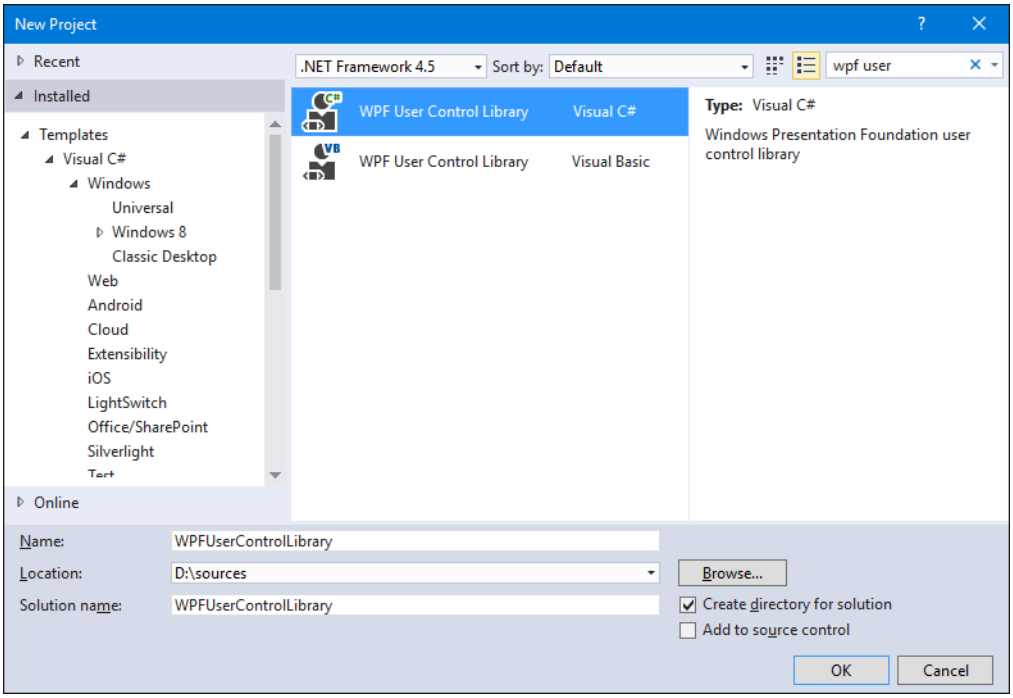
Questo capitolo descrive la creazione e l'integrazione di un semplice controllo utente WPF. Poiché qui viene descritto solo il meccanismo/la procedura di base per l'integrazione in zenon, la funzionalità del controllo utente è limitata all'aggiunta di due valori. Per mantenere la semplicità di questo esempio, sono stati volutamente omessi la gestione estesa degli errori e l´illustrazione delle opzioni nella loro completezza.

CREARE UN CONTROLLO UTENTE WPF

1. In Visual Studio, creare una nuova **Solution**, ed in questa un progetto **WPF User Control Library**.

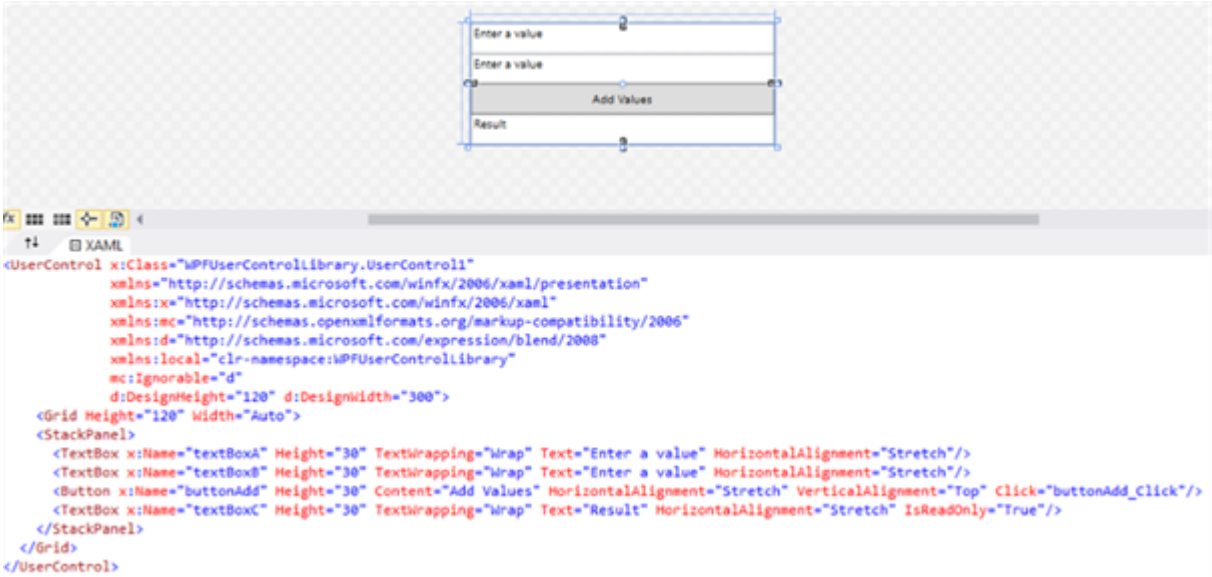
   In questo esempio si è optato per .NET Framework Version 4. È possibile scegliere anche una versione diversa che deve essere installata sul sistema di destinazione sul quale sarà poi avviato il Runtime.

Info: se il modello di progetto corrispondente non appare nell'elenco dei modelli disponibili, può essere aggiunto tramite la ricerca (campo nell'angolo in alto a destra della finestra di dialogo).



Nel nostro esempio, il progetto è denominato **WPFUserControlLibrary**.

2. Creare 3 caselle di testo e un pulsante nel file UserControl1.xaml:



3. Inserire il codice seguente nell´evento click del pulsante::

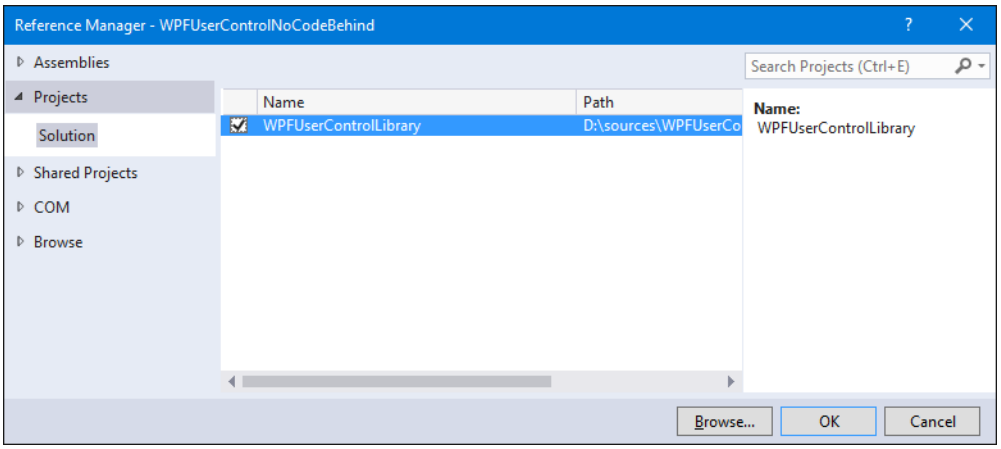
```
private void buttonAdd_Click(object sender, RoutedEventArgs e)
{
  try
  {
    textBoxC.Text = (Convert.ToInt32(textBoxA.Text) + Convert.ToInt32(textBoxB.Text)).ToString();
  }
  catch (Exception ex)
  {
    textBoxC.Text =  "Error adding values: " + ex.Message;
  }
}
```

Ora è disponibile l´User Control che fornisce la funzionalità richiesta. Tuttavia, poiché zenon può visualizzare solo file XAML che non sono collegati a un file code-behind, è necessario un file XAML aggiuntivo che faccia riferimento alla libreria (assembly) appena creata.

CREAZIONE DEL FILE XAML (SENZA CODE-BEHIND) PER ZENON

Per creare il file XAML necessario in zenon, procedere come segue:

1. Create anzitutto un altro progetto, ancora una volta del tipo **WPF User Control Library**.
2. In questo esempio, viene chiamato **WPFUserControlNoCodeBehind**.
3. Includere un riferimento al progetto appena creato in questo nuovo progetto.



4. Il file XAML (**UserControl1.xaml**) ha il seguente aspetto:

```
<UserControl x:Class="WPFUserControlNoCodeBehind.UserControl1"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
             xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
             xmlns:local="clr-namespace:WPFUserControlNoCodeBehind"
             mc:Ignorable="d"
             d:DesignHeight="300" d:DesignWidth="300">
  <Grid>

  </Grid>
</UserControl>
```

5. Poiché tutti i contenuti necessari sono contenuti nella DLL creata in precedenza e non è possibile utilizzare un file code behind, eliminare le righe seguenti:

   x:Class=`"WPFUserControlNoCodeBehind.UserControl1"`

xmlns:local="clr-namespace:WPFUserControlNoCodeBehind"

6. Inoltre, eliminare le seguenti righe (per le impostazioni della dimensione del Designer):

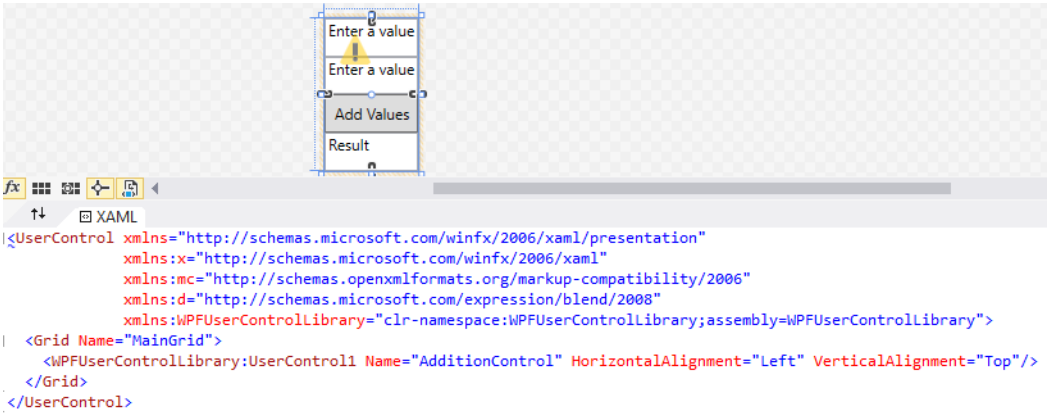
   mc:Ignorable="d"

   d:DesignHeight="300" d:DesignWidth="300"

7. Eliminare il file code-behind (**UserControl1.xaml.cs**) in questo progetto.
8. Trascinare l'UserControl precedentemente creato (dal progetto **WPFUserControlLibrary**) su XAML Designer tramite la toolbox .
9. Assegnare un nome a Grid, e UserControl.

   Attenzione: se qui non viene assegnato nessun nome, questi elementi non vengono visualizzati nella finestra di dialogo del collegamento nell'Editor di zenon e non possono quindi essere dinamizzati.

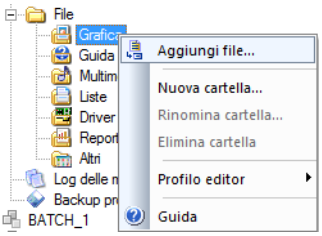
10. Il file XAML dovrebbe avere il seguente aspetto:



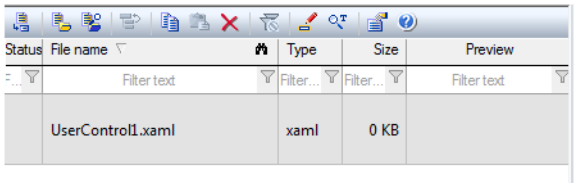
Il prossimo passaggio illustra come implementare la DLL, e il file XAML in zenon.

PASSAGGI IN ZENON

1. Aprire l´Editor di zenon.
2. Passare al nodo *File -> Grafica*.
3. Nel menù contestuale, selezionare **Aggiungi file...**



4. Scegliere il file XAML nella posizione di memorizzazione (**UserControl1.xaml** del progetto **WPFUserControlNoCodeBehind**) e aggiungerlo:

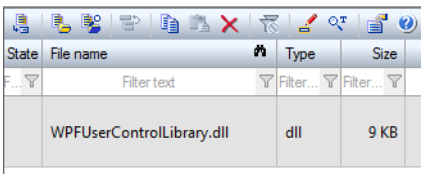


5. Aggiungere la DLL con la funzionalità per il file XAML.

   Procedura:

   a) Nel menù contestuale del nodo File -> Altri, selezionare **Aggiungi file...**.

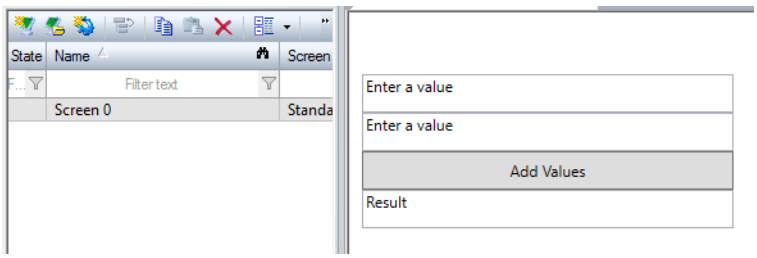
   b) Selezionare il file **WPFUserControlLibrary.dll** (dal percorso di output) del primo progetto (**WPFUserControlLibrary**).



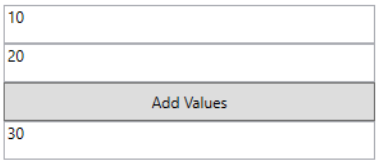
6. Creare un´immagine di zenon.

7. Aggiungere un elemento WPF e selezionare il file XAML precedentemente implementato.

   Adesso si dovrebbe vedere quanto segue nell´Editor di zenon:



8. Avviare il Runtime di zenon per testare il controllo anche in quella sede.



Info

Il file XAML e gli assembly referenziati possono essere salvati anche insieme come file *.cdwpf. Così dovrà essere importato solo un file nell´Editor (alla voce *File -> Grafica*). Ulteriori informazioni su questo argomento si trovano nel capitolo File CDWPF (file collection) (A pagina: 49).

Suggerimento: durante lo sviluppo di un controllo utente WPF, di solito è più pratico inserire separatamente il file XAML e la (le) DLL referenziata (e). Questo facilita lo scambio della DLL e il debug. Ulteriori informazioni sul debug si trovano nel capitolo Debug del controllo utente WPF a Runtime (A pagina: 29).

Suggerimento

Le DLL appartenenti ad un elemento WPF possono anche essere sostituite anche durante il funzionamento dell´Editor. Il referenziamento avviene tramite un link nel file XAML.

Per sostituire una DLL:

- Chiudere tutte le immagini di zenon che utilizzano l'elemento WPF.
- Chiudere tutti i simboli che utilizzano un elemento WPF.
- In Explorer, sostituire la DLL nella cartella *\wpfache* dei file Editor. Questa cartella si trova nella directory SQL, alla voce: *...\PROJECT-GUID\FILES\zenon\custom\wpfcache*

In alternativa alla sostituzione tramite Explorer, è anche possibile sostituire il file direttamente nell'Editor di zenon. Procedura:

- Nelle impostazioni di progetto di Visual Studio, aumentare la versione del file della DLL.
- Ricostruire la DLL.
- Chiudere tutte le immagini di zenon che utilizzano l'elemento WPF.
- Chiudere tutti i simboli che utilizzano un elemento WPF.
- Nell´Editor di zenon, cancellare la DLL dalla cartella *\File\Altri* e incollare il file con il numero di versione aumentato.

Ulteriori esempi sono riportati nel capitolo Esempi: integrazione di WPF in zenon (A pagina: 116).

5.2 Debug del controllo utente WPF a Runtime

Per eseguire il debug del del WPF User Control a Runtime, procedere come segue.

Questo esempio utilizza il controllo descritto nel capitolo Creazione di un semplice controllo utente WPF con funzionalità code behind (A pagina: 24).

DEBUG TRAMITE ATTACH TO PROCESS

1. Assicurarsi che il Runtime di zenon sia avviato e che sia stata aperta un'immagine con il controllo utente WPF.

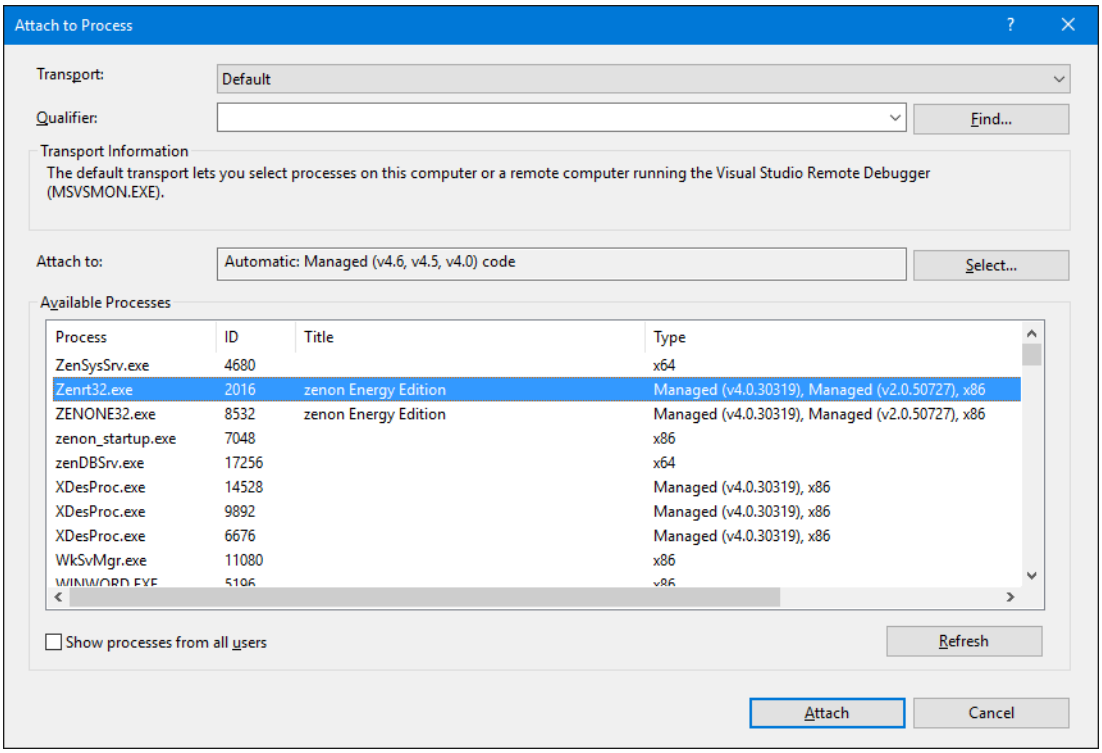
   Assicurarsi poi che la DLL attualmente utilizzata corrisponda alla build corrente (versione) del progetto User Control (**WPFUserControlLibrary**).

2. Nel progetto di Visual Studio, impostare un punto di interruzione (breakpoint) nell'evento click del pulsante.

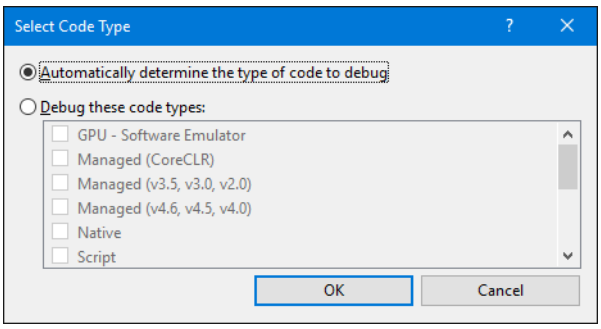
```
28    private void buttonAdd_Click(object sender, RoutedEventArgs e)
29    {
30      try
31      {
32        textBoxC.Text = (Convert.ToInt32(textBoxA.Text) + Convert.ToInt32(textBoxB.Text)).ToString();
33      }
34      catch (Exception ex)
35      {
36        textBoxC.Text =  "Error adding values: " + ex.Message;
37      }
38    }
```

3. In Visual Studio, in **Debug** , selezionare la voce di menù **Attach to Process**.

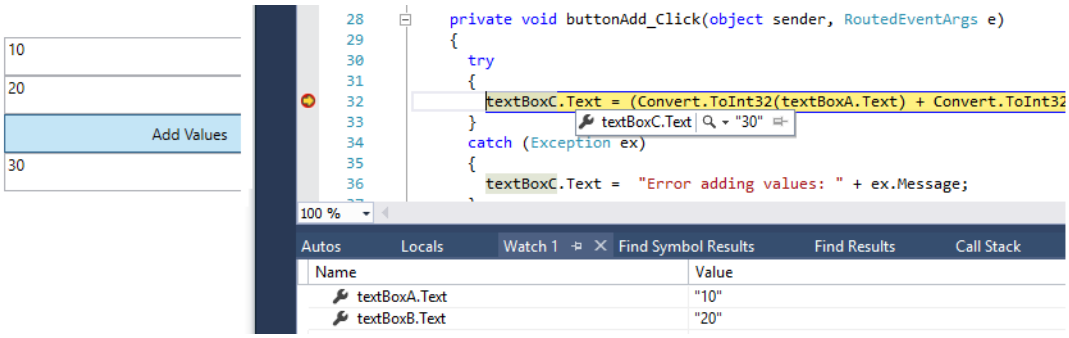
4. Selezionare il processo zenon Runtime.



5. Alla voce **Attach to**, selezionare **Automatico**, o la versione appropriata di .NET Framework.



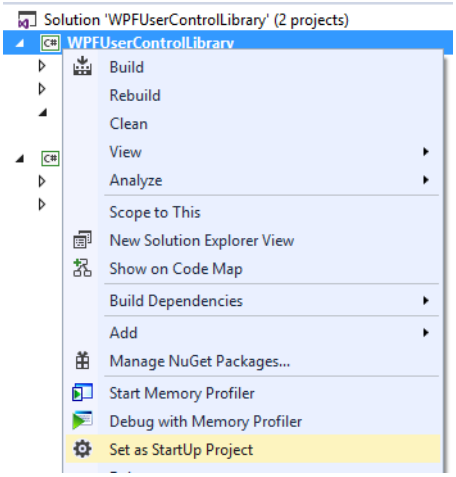
6. Cliccare su **Attach**.
7. Attivare ora il punto di interruzione (breakpoint) inserendo valori nel controllo WPF nel Runtime di zenon e cliccando sul pulsante.



DEBUG TRAMITE START EXTERNAL PROGRAM

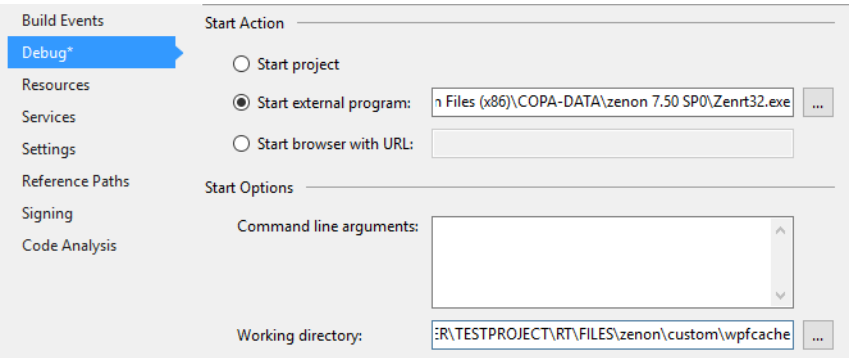
1. Assicurarsi che il Runtime di zenon sia terminato.
2. Assicurarsi che il progetto contenente il controllo utente WPF sia impostato come progetto di partenza nell`Editor di zenon.

3. Assicurarsi che il progetto User Control (**WPFUserControlLibrary**) sia impostato come progetto di partenza in Visual Studio.

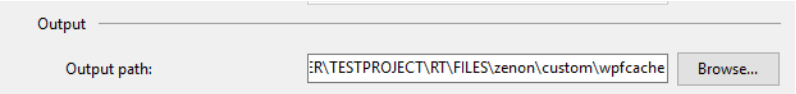


4. Nelle proprietà del progetto Visual Studio, in **Debug**, alla voce**Start action**: **Start external program**
5. Selezionare per **Start external program** il percorso dell´applicazione zenon Runtime.
6. Nella Working Directory, selezionare la cartella **\wpfcache** dei file Runtime (**...\PROJECTNAME\RT\FILES\zenon\custom\wpfcache**)

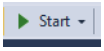
   Suggerimento: a progetto selezionato nell´Editor di zenon, premere la combinazione di tasti **CTRL+ALT+R** per passare direttamente alla directory principale dei file Runtime.



7. Nelle proprietà di progetto, alla voce **Build** specificare la cartella **\wpfache** dei file Runtime anche come **Output path**.

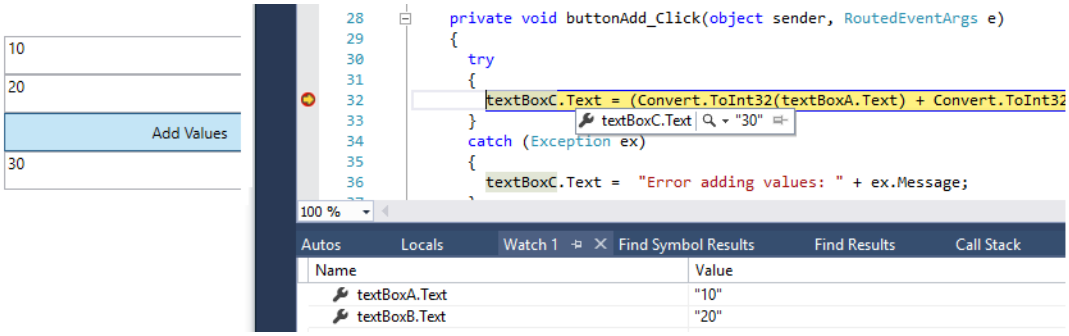


8. Compilare il progetto in Visual Studio.
9. Avviare il debug in Visual Studio tramite **Start**.



10. Il Runtime di zenon verrà avviato automaticamente.

11. Attivare il breakpoint inserendo i valori nel controllo WPF nel Runtime di zenon e cliccando sul pulsante.



Info

Quando si avvia il Runtime di zenon, gli assembly (DLL) referenziati nel controllo utente WPF della cartella **\FILES\zenon\custom\additional**, o gli assembly dei file **CDWPF** vengono copiati nella cartella **\FILES\zenon\custom\wpfcache**. Se la versione del file della DLL nella cartella **\wpfache** è uguale o superiore alla versione del "file originale", non verrà sostituita!

Per il debug è quindi sufficiente sostituire solo il file direttamente nella cartella **\wpfache**.

Per la consegna è necessario assicurarsi che la versione attuale della DLL sia disponibile nella cartella **\additional**, o nel file **CDWPF**!

Attenzione: se la DLL nella cartella **\additional** o in **CDWPF** viene solo aggiornata, ma non viene aumentato il numero di versione, la DLL nella cartella **\wpcache** deve essere prima eliminata manualmente, altrimenti non verrà aggiornata (a causa del meccanismo descritto sopra).

Suggerimento

Le DLL appartenenti ad un elemento WPF possono anche essere sostituite anche durante il funzionamento dell´Editor. Il referenziamento avviene tramite un link nel file XAML.

Per sostituire una DLL:

- Chiudere tutte le immagini di zenon che utilizzano l'elemento WPF.
- Chiudere tutti i simboli che utilizzano un elemento WPF.
- In Explorer, sostituire la DLL nella cartella *\wpfache* dei file Editor. Questa cartella si trova nella directory SQL, alla voce: *...\PROJECT-GUID\FILES\zenon\custom\wpfcache*

In alternativa alla sostituzione tramite Explorer, è anche possibile sostituire il file direttamente nell'Editor di zenon. Procedura:

- Nelle impostazioni di progetto di Visual Studio, aumentare la versione del file della DLL.
- Ricostruire la DLL.
- Chiudere tutte le immagini di zenon che utilizzano l'elemento WPF.
- Chiudere tutti i simboli che utilizzano un elemento WPF.
- Nell´Editor di zenon, cancellare la DLL dalla cartella *\File\Altri* e incollare il file con il numero di versione aumentato.

5.3 Scambio di dati fra zenon e controlli utenti WPF

Esistono diversi modi per scambiare dati tra zenon e i controlli utenti WPF.

- Scambio di dati via Dependency Properties (A pagina: 34)
- Scambio di dati via VSTA (A pagina: 39)

5.3.1 Scambio di dati via Dependency Properties

Il modo più elegante e più sicuro di scambiare dati fra zenon e controlli utenti WPF di creazione propria, è quello di usare le *Dependency Properties*.

Il WPF User Control Projekt (**WPFUserControlLibrary**) creato nel capitolo Creazione di un semplice controllo utente WPF con funzionalità code behind (A pagina: 24) serve da base.

Anche in questo capitolo ci concentreremo solamente sulla tematica di fondo (in questo caso, le *Dependency Properties* e lo scambio di dati fra il controllo utente e zenon). Non descriveremo

specifiche funzionalità di WPF, come Databinding (Associazione dati) ecc., né tratteremo in modo esplicito la problematica della risoluzione dei problemi.

ESTENSIONI NEL CODICE

1. Create l´evento *TextChanged* per l´elemento `textBoxA` nel file **UserControl1.xaml**

   ```
   TextChanged="textBoxA_TextChanged"
   ```

2. Inserire le seguenti righe di codice nella classe `UserControl1` del file code-behind (**UserControl1.xaml.cs**).

```
/// <summary>
/// Gets or sets the ValueA.
/// </summary>
public double ValueA
{
get
{
return (double)GetValue(ValueADependencyProperty);
}
set
{
SetValue(ValueADependencyProperty, value);
}
}

/// <summary>
/// Dependency property for ValueA
/// </summary>
public static readonly DependencyProperty ValueADependencyProperty =
DependencyProperty.Register("ValueA", typeof(double),
typeof(UserControl1), new FrameworkPropertyMetadata(0.0, new
PropertyChangedCallback(OnValueADependencyPropertyChanged)));

/// <summary>
/// Called when [value a dependency property changed].
/// </summary>
/// <param name="source">The source.</param>
/// <param name="e">The <see cref="DependencyPropertyChangedEventArgs"/> instance containing the event
data.</param>
private static void OnValueADependencyPropertyChanged(DependencyObject source,
DependencyPropertyChangedEventArgs e)
```

```
{
UserControl1 control = source as UserControl1;
if (control != null)
{
try
{
control.ValueA = (double)e.NewValue;
control.textBoxA.Text = control.ValueA.ToString();
}
catch (Exception)
{}
}
}

/// <summary>
/// Handles the TextChanged event of the textBoxA control.
/// </summary>
/// <param name="sender">The source of the event.</param>
/// <param name="e">The <see cref="TextChangedEventArgs"/> instance containing the event data.</param>
private void textBoxA_TextChanged(object sender, TextChangedEventArgs e)
{
try
{
ValueA = Convert.ToDouble(textBoxA.Text);
}
catch (Exception)
{}
}
```

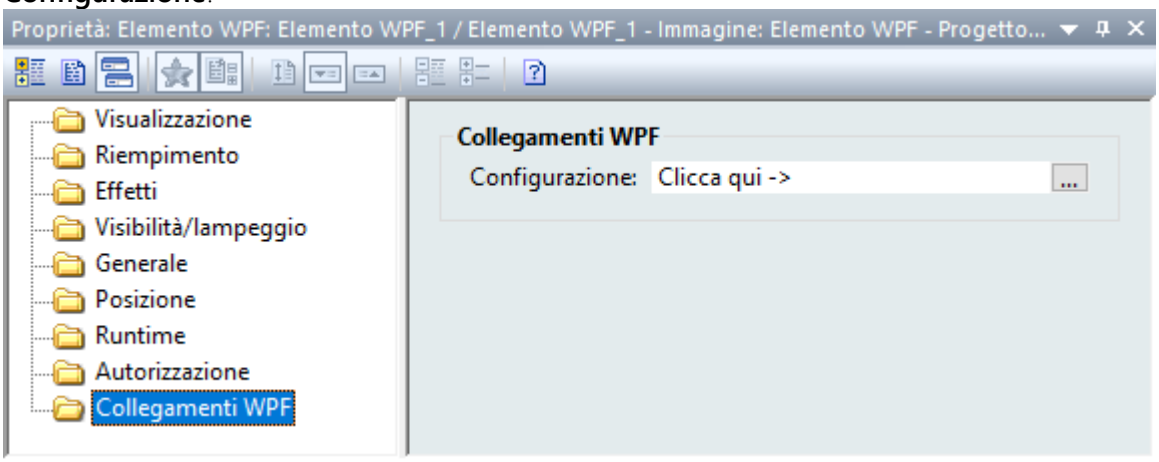
Poi compilare la solution.

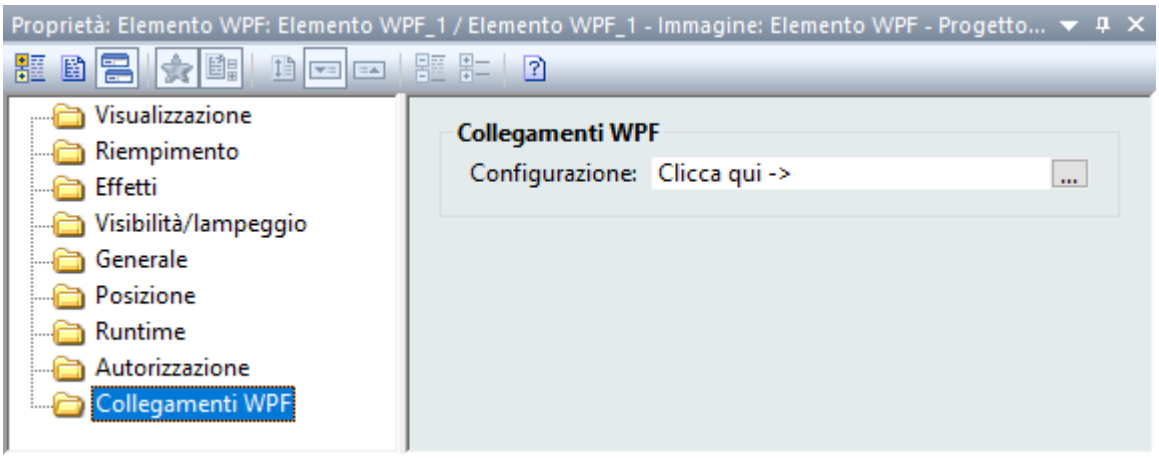
> **Info**
>
> Questo esempio utilizza una proprietà numerica (`double`). Possono anche essere usati altri tipi di dato semplici (come `bool`, `string`, `int`, ecc.).

COLLEGAMENTO IN ZENON

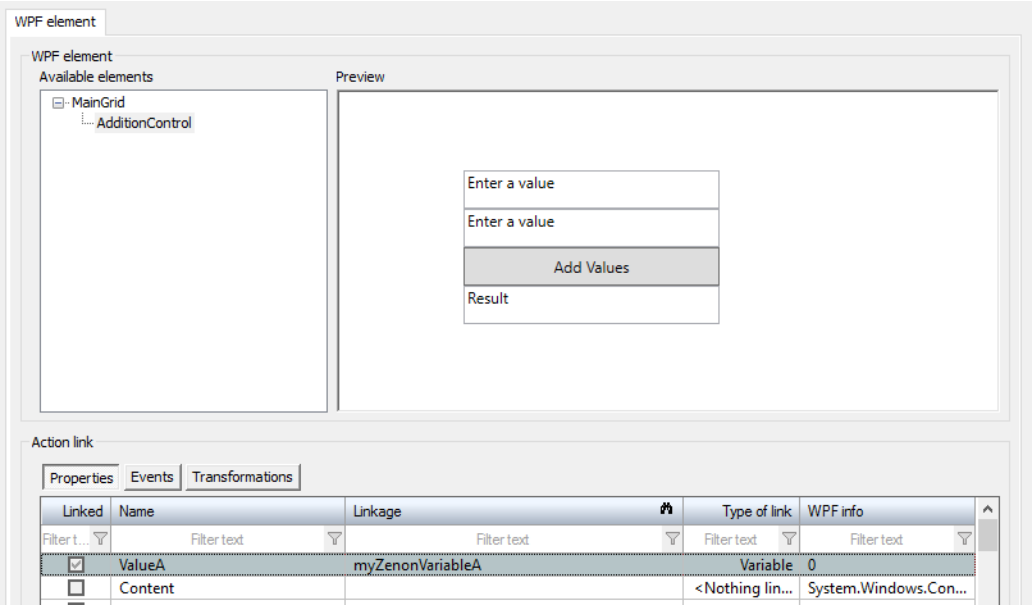
1. Aggiornare il controllo utente WPF (DLL) nell´Editor di zenon.

2. Procedere come descritto nel capitolo Creazione di un semplice controllo utente WPF con funzionalità code (A pagina: 24).
3. Creare una variabile numerica in zenon. Collegare questa variabile a un elemento di testo dinamico.
   Posizionare quest´ultimo nell'immagine accanto all'elemento WPF con il controllo utente.
4. Aprire l'immagine contenente l'elemento WPF.
5. Nel gruppo **WPF-Verknüpfungen** delle proprietà dell'elemento WPF, selezionare **Configurazione**.



6. 

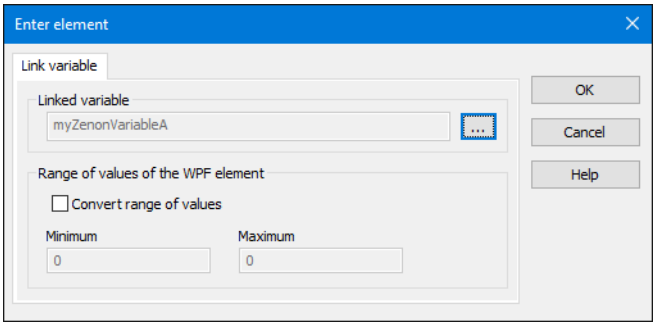
Espandere il nodo nell'angolo in alto a sinistra della struttura ad albero e selezionare **AdditionControl**.



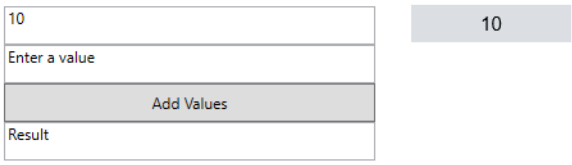
7. Selezionare la riga con *ValueA* (questo è il nome della proprietà precedentemente creata nel codice) e selezionare alla voce **Type of link: Variable.**

Suggerimento: assegnare alle proprietà un prefisso, in modo da poterle trovare più facilmente, ad es: *_ValueA*

8. Nella colonna **Linkage** ("Collegamento") selezionare la variabile creata precedentemente in zenon.



9. Confermare la finestra di dialogo cliccando su OK e compilare i file Runtime.
10. Avviare il Runtime per testare il controllo utente WPF.



11. Se cambia il valore nel controllo utente, cambia automaticamente anche quello in zenon e viceversa.
12. Naturalmente, è possibile eseguire il debug del controllo come descritto nel capitolo Debug del controllo utente WPF a Runtime (A pagina: 29), e creare ulteriori Dependency Properties.

> **Info**
>
> Per accedere (automaticamente) ai valori di DependencyProperties durante l'inizializzazione (ad esempio, quando si apre il controllo utente), è possibile utilizzare `UserControl_Loaded` Event.

5.3.2 Scambio di dati via VSTA

I dati possono anche essere scambiati tra zenon e i controllo utenti WPF anche tramite VSTA.

A questo scopo vengono utilizzati i metodi dell´elemento API

- `get_WPFProperty` (lettura di valori)
- `set_WPFProperty` (scrittura di valori)

.

L'esempio qui utilizzato si basa sull'esempio discusso nel capitolo Scambio di dati via Dependency Properties (A pagina: 34).

CREAZIONE DI UNA MACRO VSTA PER LO SCAMBIO DI DATI FRA ZENON E IL CONTROLLO UTENTE WPF.

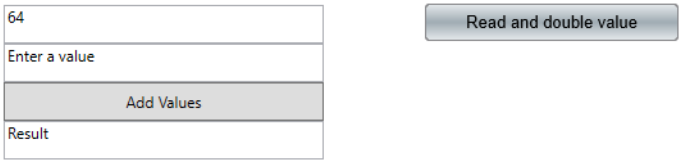
1. Creare la seguente macro VSTA nel ProjectAddin del progetto di zenon.

```
/// <summary>
/// Sample Macro for data exchange between VSTA and a WPF User Control
/// </summary>
public void MacroWPFAccess()
{
    //Get the Screen and Element hosting the WPF User Control
    zenOn.IDynPicture myWPFScreen = this.DynPictures().Item("Screen");
    zenOn.IElement myWPFElement = myWPFScreen.Elements().Item("WPF_Element");

    //Read the current value from the WPF Element property
    double currentValue = Convert.ToDouble(myWPFElement.get_WPFProperty("AdditionControl", "ValueA"));

    //Double the value and write it back to the WPF Element property
    myWPFElement.set_WPFProperty("AdditionControl", "ValueA", currentValue * 2);
}
```

   Tenere presente quanto segue:

   - `"Screen"` è il nome dell´immagine di zenon in cui si trova l´elemento WPF.
   - `"WPF_Element"` è il nome dell´elemento WPF che contiene il controllo utente WPF.
   - `"AdditionControl"` è il nome del controllo utente WPF stesso (definito nel file **UserControl1.xaml**).
   - `"ValueA"` è il nome dell´User Control Properties.

2. Creare una funzione "Lanciare Macro-VSTA" e collegarla a un pulsante nell'immagine in cui si trova l'elemento WPF.
3. Avviare il Runtime per testare le modifiche.



   Quando viene eseguita la macro, il valore viene letto dal controllo, raddoppiato e riscritto.

> **Info**
>
> Le User Control Properties utilizzate in questo metodo per lo scambio dei dati non devono necessariamente essere eseguite come Dependency Properties, come illustrato in questo esempio. È possibile utilizzare anche le Standard Properties; a questo proposito, si veda il capitolo Accesso via VSTA "Variable Link" (A pagina: 41).

5.4 Accesso al modello oggetto zenon (Runtime) da un controllo utente WPF.

Esistono diversi modi per accedere al modello oggetto zenon da un controllo utente WPF. Questi sono spiegati più dettagliatamente nei capitoli seguenti.

Attenzione

Quando si utilizzano gli oggetti COM di zenon nei controlli utente o nelle applicazioni esterne di creazione propria, questi devono essere rilasciati utilizzando il metodo **Marshal.ReleaseComObject**. Il rilascio tramite il metodo **Marshal.FinalReleaseComObject** non deve essere utilizzato, perché Add-In di zenon smettono di funzionare quando si usa questo metodo.

5.4.1 Accesso via VSTA "Variable Link"

Per accedere all´interfaccia Runtime COM di zenon Runtime tramite "Variablen-Link", procedere come segue. L'esempio riportato nel capitolo Creazione di un semplice controllo utente WPF con funzionalità code-behind (A pagina: 24) serve come esempio di partenza.

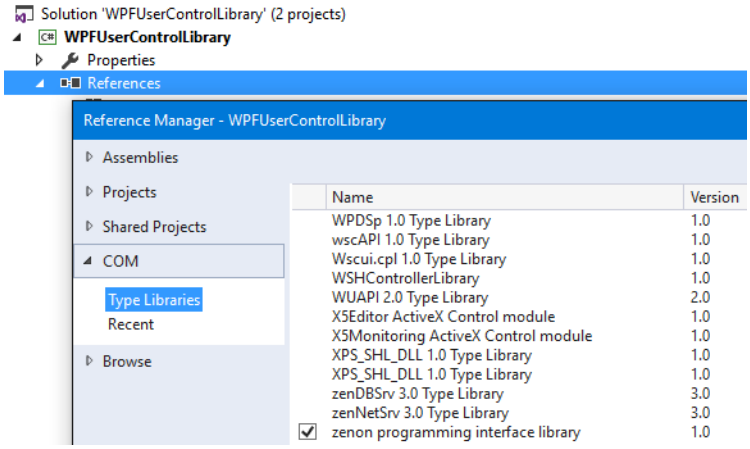
Info

Il seguente codice mostra come può essere realizzato l´accesso COM a zenon Runtime ed è limitato alle funzionalità di base. Non si è affrontato il tema della risoluzione dei problemi.

ADEGUAMENTI E MODIFICHE NECESSARIE NEL CONTROLLO UTENTE WPF.

I seguenti passaggi sono necessari nel progetto WPF User Control (**WPFUserControlLibrary**).

In primo luogo, deve essere incluso un riferimento all'interfaccia COM zenon.



Poi il seguente codice deve essere inserito nella classe `UserControl1` :

```
//The zenon Project
zenOn.Project zenonProject = null;

/// <summary>
/// Property for the Variable link via VSTA
/// </summary>
public object zenonVariableLink
{
get { return null; }
set
{
if (value != null && zenonProject == null)
{
zenOn.Variable zenonVariable;
try
{
zenonVariable = (zenOn.Variable)value;
}
catch (Exception)
{
return;
}
if ((zenonVariable!= null) && (!string.IsNullOrEmpty(zenonVariable.Name)))
```

```
{
zenonProject = zenonVariable.Parent.Parent;
}
}
}
}

/// <summary>
/// Trigger used to notify the control from VSTA to release the COM resources
/// </summary>
public object zenonReleaseTrigger
{
get { return null; }
set
{
if ((bool)value && zenonProject != null)
{
try
{
Marshal.ReleaseComObject(zenonProject);
}
catch (Exception)
{
return;
}
zenonProject = null;
GC.Collect();
GC.WaitForPendingFinalizers();
GC.Collect();
}
}
}
```

In questo modo le proprietà `zenonVariableLink` (per inizializzare l'oggetto COM) e `zenonReleaseTrigger` (per rilasciare l'oggetto COM) saranno accessibili successivamente da VSTA (scrittura).

Per testare l´accesso COM velocemente ed in modo molto semplice, si può aggiungere nell´evento click del pulsante esistente del controllo utente la seguente riga di codice.

```
private void buttonAdd_Click(object sender, RoutedEventArgs e)
```

```
{
if (zenonProject != null)
{
MessageBox.Show(zenonProject.Name);
}
return;
...
```

> **Info**
>
> Questo esempio utilizza una variabile di tipo `zenOn.Project`. Naturalmente, è possibile utilizzare altri oggetti, eventi, ecc. del modello di oggetti zenon.

AGGIUSTAMENTI NECESSARI IN ZENON PROJECT/VSTA CODE

Nel codice VSTA sono necessari i seguenti passaggi:

Creazione di una macro VSTA per l'inizializzazione

```
/// <summary>
/// Macro for API initialization in the WPF User Control
/// </summary>
public void MacroWPFInit()
{
zenOn.IDynPicture myWPFScreen = this.DynPictures().Item("Screen");
zenOn.IElement myWPFElement = myWPFScreen.Elements().Item("WPF_Element");
myWPFElement.set_WPFProperty("AdditionControl", "zenonVariableLink", this.Variables().Item(0));
}
```

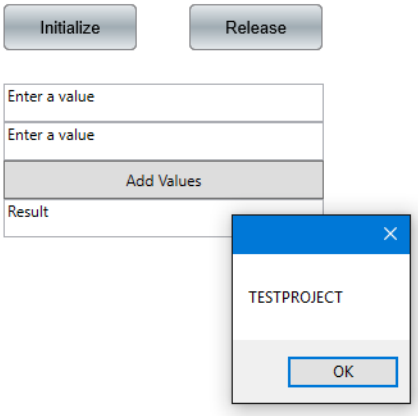
Creazione di una macro VSTA per il rilascio

```
/// <summary>
/// Macro for API release in the WPF User Control
/// </summary>
public void MacroWPFRelease()
{
zenOn.IDynPicture myWPFScreen = this.DynPictures().Item("Screen");
zenOn.IElement myWPFElement = myWPFScreen.Elements().Item("WPF_Element");
myWPFElement.set_WPFProperty("AdditionControl", "zenonReleaseTrigger", true);
}
```

Creare due funzioni "Lanciare Macro-VSTA" e collegarle a pulsanti che si trovano nella stessa immagine dell'elemento WPF.

Ora avviare il Runtime per testare la funzionalità.

- Eseguire la macro di inizializzazione.
- Fare clic sul pulsante nel controllo utente WPF; viene visualizzata una finestra di messaggio con il nome del progetto



- Eseguire la macro di rilascio.

Per eseguire il debug del controllo utente, è possibile procedere come descritto nel capitolo Debug del controllo utente WPF a Runtime (A pagina: 29).

Suggerimento

Ai fini di una chiara dimostrazione, l´inizializzazione ed il rilascio dell´oggetto COM avvengono nel presente esempio semplicemente tramite funzioni macro VSTA. A seconda dell´impiego concreto, gli eventi in VSTA possono rivelarsi più idonei.

Ad esempio, il code per l´inizializzazione può essere eseguito nell´evento `_Open` dell´immagine con l´elemento WPF, e il code per il rilascio nell´evento `_Close`.

Il meccanismo qui descritto viene usato anche nel capitolo Visualizzazione di elementi WPF nel Web Client di zenon (A pagina: 112).

Attenzione

Se gli oggetti COM vengono utilizzati nei controlli utente WPF, devono sempre essere esplicitamente rilasciati prima che il controllo utente WPF venga distrutto (prima della chiusura dell´immagine, del termine del Runtime e della ricarica).

5.4.2 Accesso via Marshalling

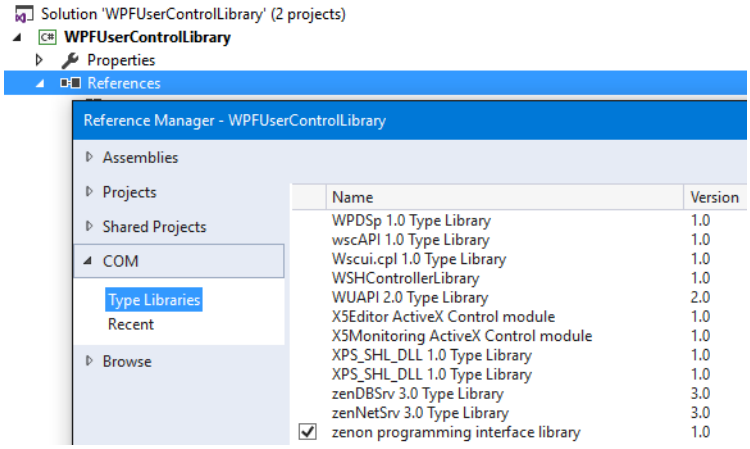
Per accedere all´interfaccia Runtime COM di zenon Runtime tramite "Marshalling", procedere come segue. L'esempio riportato nel capitolo Creazione di un semplice controllo utente WPF con funzionalità code-behind (A pagina: 24) serve come esempio di partenza.

> **Info**
>
> Il seguente codice mostra come può essere realizzato l´accesso COM a zenon Runtime ed è limitato alle funzionalità di base. Non si è affrontato il tema della risoluzione dei problemi.

ADEGUAMENTI E MODIFICHE NECESSARIE NEL CONTROLLO UTENTE WPF.

I seguenti passaggi sono necessari nel progetto WPF User Control (**WPFUserControlLibrary**).

In primo luogo, deve essere incluso un riferimento all'interfaccia COM zenon.



Poi il seguente codice deve essere inserito nella classe `UserControl1` :

```
//The zenon Project
zenOn.Project zenonProject = null;
```

Inoltre, il costruttore del controllo utente deve essere esteso con l´aggiunta delle righe sottostanti (per inizializzare l'oggetto COM):

```
/// <summary>
/// Constructor for UserControl1, initialize COM Object
/// </summary>
```

```
public UserControl1()
{
InitializeComponent();

try
{
zenonProject =
((zenOn.Application)Marshal.GetActiveObject("zenOn.Application")).Projects().Item("TESTPROJECT");
}
catch (Exception)
{
}
}
```

L´oggetto COM deve essere rilasciato nell´evento `UserControl_Unloaded`:

```
/// <summary>
/// Release COM Object
/// </summary>
private void UserControl_Unloaded(object sender, RoutedEventArgs e)
{
try
{
if (zenonProject != null)
{
Marshal.ReleaseComObject(zenonProject);
zenonProject = null;
}
}
catch (Exception)
{
}
}
```

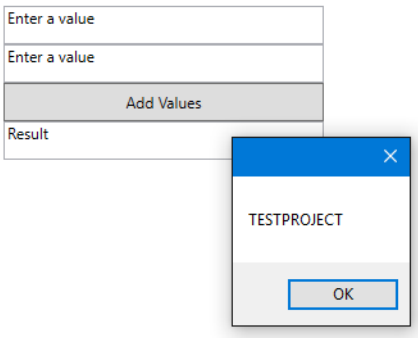
Per testare l´accesso COM velocemente ed in modo molto semplice, si può aggiungere nell´evento click del pulsante esistente del controllo utente la seguente riga di codice.

```
private void buttonAdd_Click(object sender, RoutedEventArgs e)
{
if (zenonProject != null)
```

```
{
MessageBox.Show(zenonProject.Name);
}
 return;

...
```

Compilare adesso la soluzione e aggiornare il controllo utente WPF nel progetto di zenon.

Avviare il Runtime per testare il controllo utente.



Per eseguire il debug del controllo utente, è possibile procedere come descritto nel capitolo Debug del controllo utente WPF a Runtime (A pagina: 29).

Info

Questo esempio utilizza una variabile di tipo `zenOn.Project`. Naturalmente, è possibile utilizzare altri oggetti, eventi, ecc. del modello di oggetti zenon.

Attenzione

Se gli oggetti COM vengono utilizzati nei controlli utente WPF, devono sempre essere esplicitamente rilasciati prima che il controllo utente WPF venga distrutto (prima della chiusura dell´immagine, del termine del Runtime e della ricarica).

Info

In zenon WebClient non è possibile nessun accesso tramite Marshalling. Se è necessario l'accesso all'interfaccia COM, è necessario utilizzare il metodo descritto nel capitolo Accesso via VSTA "Variable Link" (A pagina: 41).

6 Progettazione nell´Editor di zenon

Per utilizzare i controlli utente WPF in zenon, la versione 4.6.2 di Microsoft Framework (o superiore, a seconda della versione del .NET Framework utilizzato nel controllo utente) deve essere installata sul computer Editor e sul computer Runtime.

CONDIZIONI PER VISUALIZZAZIONE WPF IN ZENON

La dinamizzazione è attualmente disponibile per tipi di variabili semplici (tipi di dati numerici e stringhe) Array e strutture non possono essere dinamizzati.

Pertanto, le seguenti funzioni WPF possono essere utilizzate in zenon:

- Proprietà dell'elemento che corrispondono a tipi di dati semplici, come *String*, *Int*, *Double*, *Bool* ecc.
- Proprietà dell´elemento di tipo "Object", che possono essere impostate con tipi di dati semplici.
- Eventi dell´elemento possono essere utilizzati con funzioni; i parametri degli eventi, però, non sono disponibili/analizzabili in zenon.
- Trasformazione di elementi per i quali esiste un **RenderTransform** per l´elemento nel file XAML.

  Attenzione: se il Content si trova al di fuori dell'area dell'elemento WPF durante le trasformazioni, non viene disegnato.

Note sull´ombra: Non è possibile visualizzare ombre per gli elementi WPF in zenon.

> **⚠Attenzione**
>
> Se i file Runtime di un progetto vengono compilati per una versione precedente alla 6.50, gli **elementi WPF** esistenti non sono inclusi nelle immagini a Runtime.

6.1 CDWPF-file (file collection)

Un file CDWPF (con estensione ***.cdwpf**) è un file ZIP rinominato che contiene i seguenti componenti:

- File XAML (per referenziare l´User Control Assembly)
- File DLL (il vero e proprio WPF User Control, facoltativo)
- Anteprima grafica (per l'anteprima, opzionale)

Regole per l'uso di file collettivi:

- I file (XAML, DLL, grafica di anteprima) possono essere posizionati direttamente nel file CDWPF o in una cartella condivisa.
- Il nome del file collettivo deve corrispondere al nome del file XAML.
- Può essere contenuto un solo file XAML.
- L'anteprima grafica deve essere piccola e non deve superare i 64 pixel di altezza.
  Nome del file di anteprima: **preview.png** o il nome del file XAML con l'estensione **png**.
- È possibile utilizzare un numero qualsiasi di assembly. La distinzione è basata sulla versione del file.
- I file collettivi non devono necessariamente contenere un assembly.
- Vengono analizzate tutte le sottocartelle e vengono presi in considerazione solo i file di tipo ***.dll**, ***.xaml**, o ***.png**.
- Se un file collettivo (***.cdwpf**) viene sostituito da un file con una versione diversa, devono essere adattati tutti i file CDWPF corrispondenti in tutti i simboli e le immagini di tutti i progetti.

6.2 Creare un elemento WPF

Per creare un elemento WPF:

1. Nella barra degli strumenti, selezionare il simbolo per l'elemento **Elemento WPF**, o scegliere l´inserimento corrispondente nel menù **Elementi**
2. Selezionare il punto di partenza nella finestra principale.
3. Disegnare l´elemento con il mouse.
4. Nel gruppo **Darstellung** delle proprietà dell´elemento, selezionare la proprietà **XAML-Datei**.
5. Si apre la finestra dialogo di selezione file.
6. Selezionare il file desiderato.
   I file validi sono nel formato:
   - *.xaml: Extensible Application Markup Language
   - *.cdwpf: File collettivo WPF, mostra anche l'immagine in anteprima

   (Il file deve essere già esistere nel nodo **File/Grafia** del Manager di progetto, o deve essere creato nella finestra di dialogo).
7. Configurare i collegamenti (A pagina: 51).

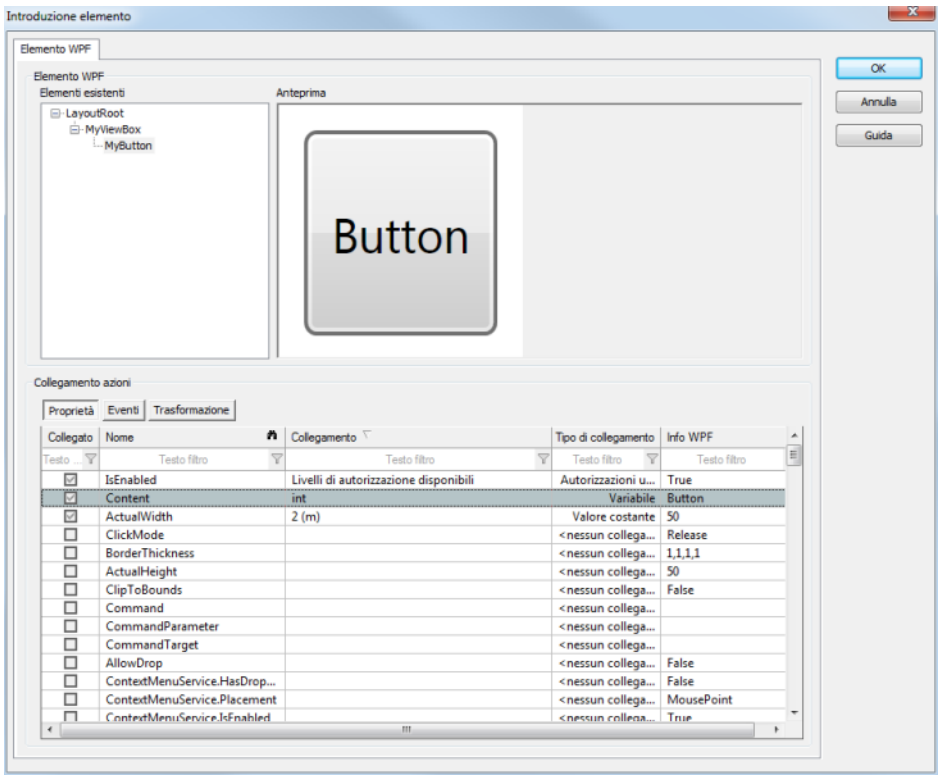
> **Info**
>
> Se si desidera utilizzare oggetti referenziati (assembly), fare riferimento alle note del capitolo Assembly referenziati (A pagina: 8).

6.3 Configurazione del collegamento

Per configurare un elemento WPF

1. Nel gruppo **WPF-Verknüpfungen** delle proprietà dell´elemento, selezionare la proprietà **Konfiguration**.
2. Si apre una finestra di dialogo con le schede, con un'anteprima del file XAML e degli elementi presenti nel file stesso.

FINESTRA DI DIALOGO DI CONFIGURAZIONE



Parametro	Descrizione
Elementi esistenti	Visualizza gli elementi con nome del file in una struttura ad albero. L'elemento selezionato può essere collegato ai dati di processo. L´attribuzione di **WPF** ai dati di processo si basa sul nome dell'elemento.. Pertanto, gli elementi vengono visualizzati solo se essi e gli elementi che li compongono hanno un nome. Le attribuzioni vengono configurate e visualizzate nelle schede

Parametro	Descrizione
	Proprietà, **Eventi** e **Trasformazione**. **Suggerimento:** se gli elementi corrispondenti non sono visualizzati, verificare che gli elementi non visualizzati nel file XAML abbiano un nome (per es.: *<Grid Name="GridName">*).
Anteprima	L'elemento selezionato lampeggia nell'anteprima.
Proprietà (A pagina: 53)	Configurazione e visualizzazione delle proprietà (variabili, autorizzazioni operative, interblocchi, valori collegati).
Eventi (A pagina: 59)	Configurazione e visualizzazione degli eventi (funzioni).
Trasformazioni (A pagina: 61)	Configurazione e visualizzazione delle trasformazioni.
Nome	Nome della proprietà.
Collegamento	Selezione del collegamento.
Tipo di collegamento	Tipo di collegamento (variabile, autorizzazione, funzione)
Info WPF	Per le proprietà, visualizza il valore corrente nel WPF-Content. L'utente può vedere direttamente che tipo di proprietà è (booleana, stringa, ecc).
Collegato	Indica se una proprietà è attualmente in uso. Se non inclusa nella vista, la voce di colonna può essere selezionata tramite il **Menù contestuale-> Selezione colonne**.

Info

Solo gli oggetti logici possono essere visualizzati nella finestra di configurazione. Gli oggetti visivi non vengono visualizzati. Per informazioni di background e su come gli oggetti visivi possono essere dinamizzati, vedere Attribuzione di oggetto zenon a WPF Content.

MODIFICARE COLLEGAMENTI

Tutti i collegamenti configurati possono essere modificati tramite le proprietà dell'elemento. Per fare questo, fare clic sull'elemento e aprire il gruppo di proprietà **WPF-Verknüpfungen**. Qui è possibile configurare ulteriormente i collegamenti senza dover aprire la finestra di dialogo.

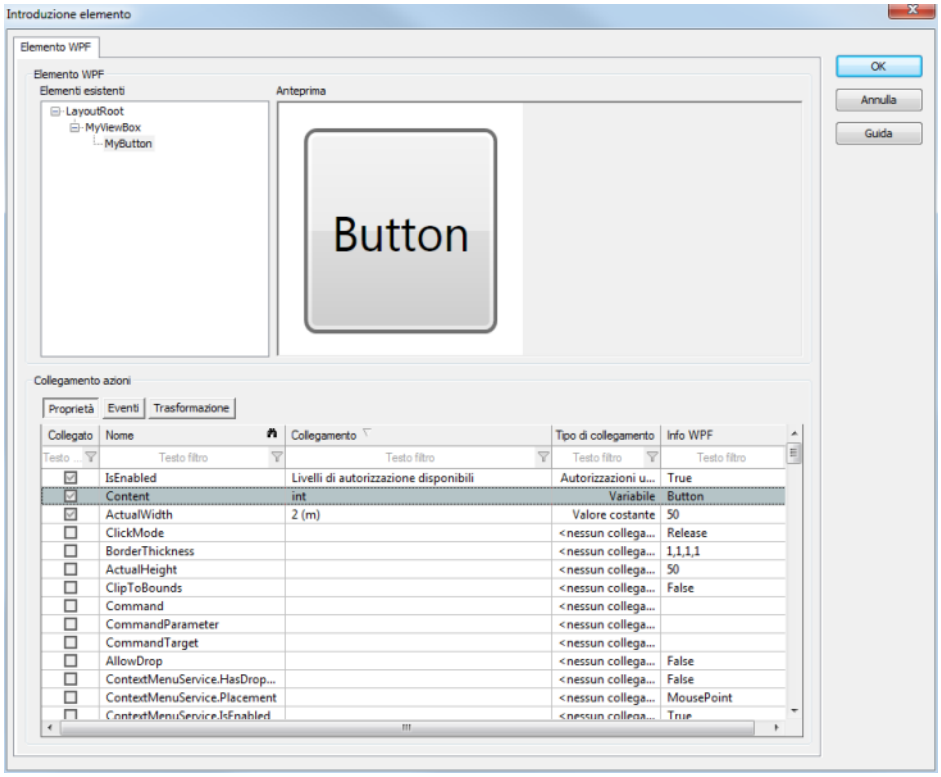
Restrizioni:

- Il tipo di collegamento non può essere modificato in questa sede.
- Nuovi collegamenti possono essere creati solo tramite la finestra di dialogo di configurazione.
- Inserimento di un elemento WPF in un simbolo: i collegamenti WPF non possono essere esportati.

6.3.1 Proprietà

Le proprietà consentono il collegamento di:

- Variabili (A pagina: 55)
- Valori (A pagina: 56)
- Autorizzazioni e interblocchi (A pagina: 58)



Parametro	Descrizione
Nome	Nome della proprietà.
Verknüpfung	Variabile collegata, autorizzazione operativa collegata, o valore collegato. Cliccando nella colonna si apre la rispettiva finestra di selezione, in base alla voce nella colonna **Tipo di collegamento**.
Verknüpfungsart	Selezione di un collegamento.
WPF-Info	Per le proprietà, visualizza il valore corrente nel WPF-Content. L'utente può vedere direttamente che tipo di proprietà è (booleana, stringa, ecc).
Verknüpft	Indica se una proprietà è attualmente in uso. Se non inclusa nella vista, la voce di colonna può essere selezionata tramite il **Menù contestuale-> Selezione colonne**.

CREARE UN COLLEGAMENTO

Per creare un collegamento:

1. Selezionare la riga con la proprietà da collegare.
2. Cliccare nella cella **Tipo di collegamento**.
3. Selezionare il collegamento desiderato nel menù a tendina.

   Le opzioni disponibili sono:

   - *<Nessun collegamento>* (elimina un collegamento esistente)
   - *Autorizzazione/Interblocco*
   - *Valore costante*
   - *Variabile*
4. Cliccare nella cella **Collegamento**.
5. Si apre a finestra di dialogo per la configurazione del collegamento desiderato.

Info

Le proprietà WPF e di zenon possono essere differenti. Se si collega, ad esempio, la proprietà **Visibilità**, sono a disposizione in .NET tre valori:

- 0 - visibile
- 1 - invisibile
- 2 - compressa

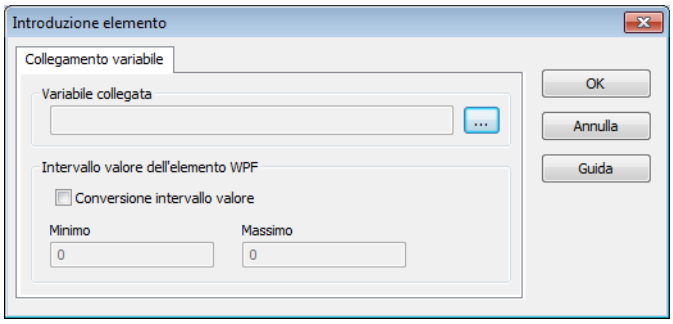
Questi valori devono essere mappati utilizzando la variabile di zenon collegata.

6.3.1.1 Collegamento di una variabile

Per collegare una variabile ad una proprietà WPF:

1. Selezionare la riga con la proprietà da collegare.
2. Cliccare nella cella **Tipo di collegamento**.
3. Nel menù a tendina, selezionare **Variabile**.
4. Cliccare nella cella **Collegamento**.
5. Si apre la finestra di dialogo per la configurazione delle variabili.

Questa finestra di dialogo si applica anche alla selezione delle variabili nelle trasformazioni (A pagina: 61). La configurazione permette anche la conversione di intervalli valori da zenon a WPF.



Parametro	Descrizione
Variabile collegata	Selezione della variabile da collegare. Fare clic sul pulsante **...** per aprire la finestra di dialogo per la selezione.
Campo valori dell'elemento WPF	Dati per la conversione dei valori variabile in valori WPF.
Conversione intervallo di valori	*Attiva*: la conversione unità WPF è attiva.

Parametro	Descrizione
	Effetto a Runtime: il valore corrente di zenon (inclusa unità zenon) viene convertito nel campo WPF mediante valori minimi e massimi standardizzati. **Esempio:** il valore di una variabile varia da 100 a 200. Per la variabile, l'intervallo normalizzato è impostato su 100 - 200. Lo scopo è quello di visualizzare questa variazione di valore usando una manopola WPF. Procedura: ▸ Alla voce **Trasformazione**, collegare la proprietà **RotateTransform.Angle** alla variabile. ▸ Attivare l´opzione **Adjust value** ▸ Configurare un campo di valori WPF da *0* a *360*. Ora, se, ad esempio, il valore è 150, la manopola viene ruotata di 180 gradi.
Minimo	Definisce il valore WPF più basso.
Massimo	Definisce il valore WPF più alto.
OK	Accetta le impostazioni e chiude la finestra di dialogo.
Annulla	Annulla le impostazioni e chiude la finestra di dialogo.
Guida	Apre la guida online.

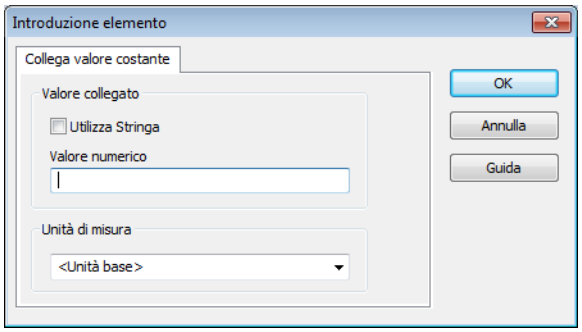
6.3.1.2 Collegare valori

I valori collegati possono essere una **String**, o un valore numerico di tipo **Double**. Quando viene visualizzata l'immagine, dopo il caricamento del WPF-Content, il valore selezionato nello stesso viene impostato.

Per collegare un valore ad una proprietà WPF:

1. Selezionare la riga con la proprietà da collegare.
2. Cliccare nella cella **Tipo di collegamento**.

3. Nel menù a tendina, selezionare l´opzione **Valore costante**.
4. Cliccare nella cella **Collegamento**.
5. Si apre la finestra di dialogo per la configurazione del collegamento valori.



Parametro	Descrizione
Valore collegato:	Inserimento di un valore numerico o di un valore stringa.
Utilizza Stringa	*Attivo*: al posto del valore numerico viene utilizzato un valore di stringa. I valori stringa sono traducibili. A Runtime, il testo viene tradotto quando l'immagine viene visualizzata e memorizzato nel WPF content. Se si esegue un cambio di lingua mentre l'immagine è aperta, il valore della stringa viene tradotto di nuovo e salvato.
Valore stringa/numerico	A seconda della selezione eseguita per la proprietà **Utilizza Stringa**, in questo campo viene immesso un valore numerico o un valore stringa. Per i valori numerici, è inoltre possibile selezionare un'unità di misura.
Unità di misura:	Selezione di un´unità di misura dal menù a tendina. Questa deve essere stata definita in precedenza nel Cambio unità di misura. L´unità di misura viene assegnata al valore numerico. Se a Runtime viene eseguito un cambio unità, il valore viene convertito nella nuova unità di misura e salvato nel WPF content.

CHIUDERE LA FINESTRA DI DIALOGO

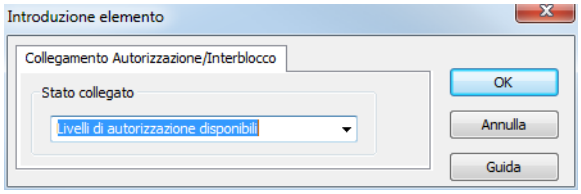
Opzione	Descrizione
OK	Applica le impostazioni e chiude la finestra di dialogo.
Annulla	Annulla tutte le modifiche e chiude la finestra di dialogo.
Guida	Apre la guida online.

6.3.1.3 Collegare autorizzazione o interblocco

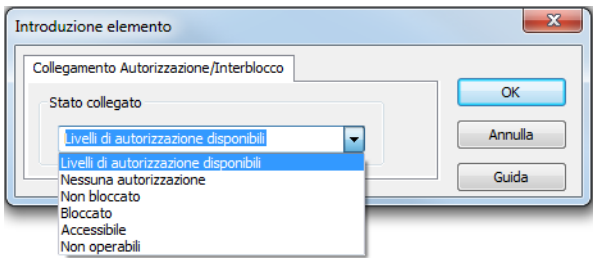
Le autorizzazioni operative non possono essere assegnate per l'intero elemento WPF. All´elemento viene assegnato un livello utente. Le autorizzazioni vengono concesse nell´ambito del livello utente per i singoli controlli. Se un´autorizzazione è attiva, il valore *1* viene scritto nella proprietà nell'elemento.

Per collegare un´autorizzazione o un interblocco ad una proprietà WPF:

1. Selezionare la riga con la proprietà da collegare.
2. Cliccare nella cella **Tipo di collegamento**.
3. Nel menù a tendina. selezionare **Autorizzazione utente/Interblocco**
4. Cliccare nella cella **Collegamento**.
5. Si apre la finestra di dialogo per la configurazione dell´autorizzazione.



Parametro	Descrizione
Collega autorizzazione/interblocco	Definizione dell´autorizzazione.
Stato collegato	Selezione dal menù a tendina di un´autorizzazione che viene collegata ad un controllo WPF. Ad esempio, la visibilità e l'operabilità di un pulsante WPF possono dipendere dallo stato dell'utente.

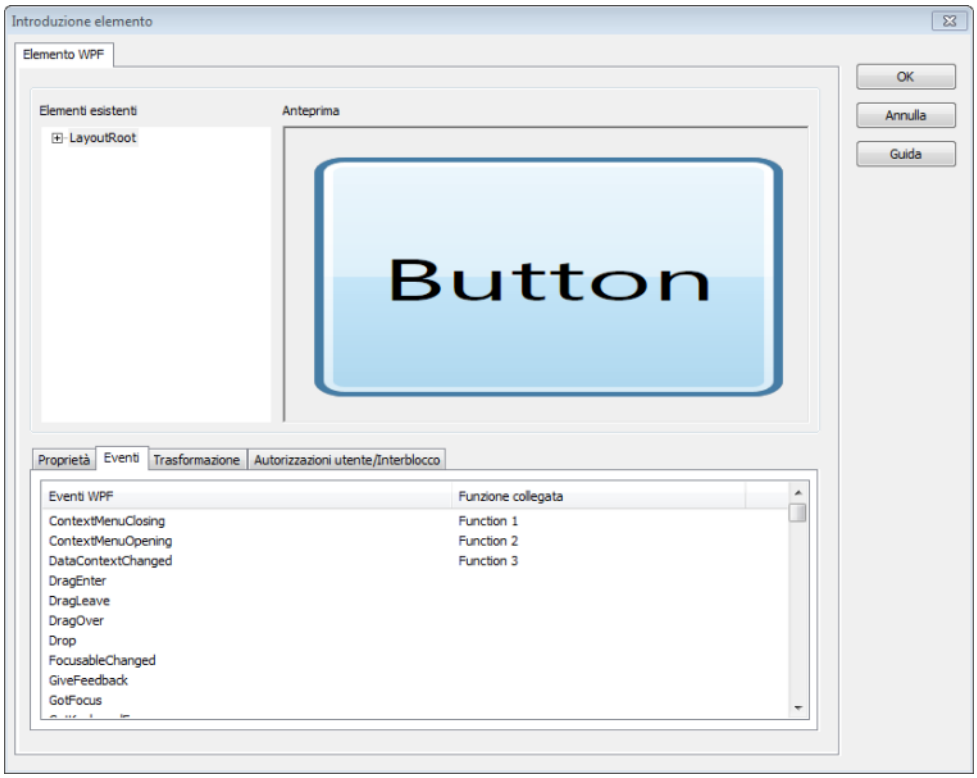


Autorizzazione	Descrizione
Autorizzazione operativa disponibile	Se l'utente ha diritti sufficienti per utilizzare l´**elemento WPF**, il valore *1* viene scritto nella proprietà.
Autorizzazione operativa non disponibile	Se l'utente non ha diritti sufficienti per utilizzare l´**elemento WPF**, il valore *1* viene scritto nella proprietà.

Autorizzazione	Descrizione
Non bloccato	Se l'elemento non è bloccato da un interblocco, il valore *1* viene scritto nella proprietà.
Bloccato	Se l'elemento è bloccato da un interblocco, il valore *1* viene scritto nella proprietà.
Accessibile	Se esiste l´autorizzazione e l´elemento non è bloccato, il valore *1* viene scritto nella proprietà.
Non operabili	Se manca l´autorizzazione o se l´elemento è bloccato, il valore *1* viene scritto nella proprietà.

6.3.2 Eventi

Gli eventi consentono il collegamento delle funzioni di zenon ad un elemento WPF.



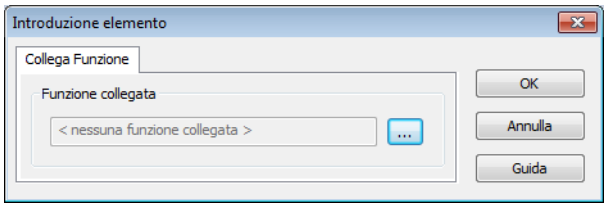
Parametro	Descrizione
Nome	Nome della proprietà.
Collegamento	Funzione collegata. Cliccando nella cella, si apre la finestra di dialogo di configurazione .

Parametro	Descrizione
Tipo di collegamento	Selezione di un collegamento. Fare clic nella cella per aprire la finestra di dialogo di selezione.
Info WPF	Per le proprietà, visualizza il valore corrente nel WPF-Content. Per l´utente è direttamente visibile di quale tipo di proprietà si tratta (booleana, stringa, ecc).
Collegato	Indica se una proprietà è attualmente in uso. Se non inclusa nella vista, la voce di colonna può essere selezionata tramite il **Menù contestuale-> Selezione colonne**.

COLLEGARE FUNZIONI

Per creare un collegamento:

1. Selezionare la riga con la proprietà da collegare.
2. Cliccare nella cella **Tipo di collegamento**.
3. Dal menù a tendina, selezionare la voce "Funzione".
4. Cliccare nella cella **Collegamento**.
5. Si apre la finestra di dialogo per la configurazione della funzione.

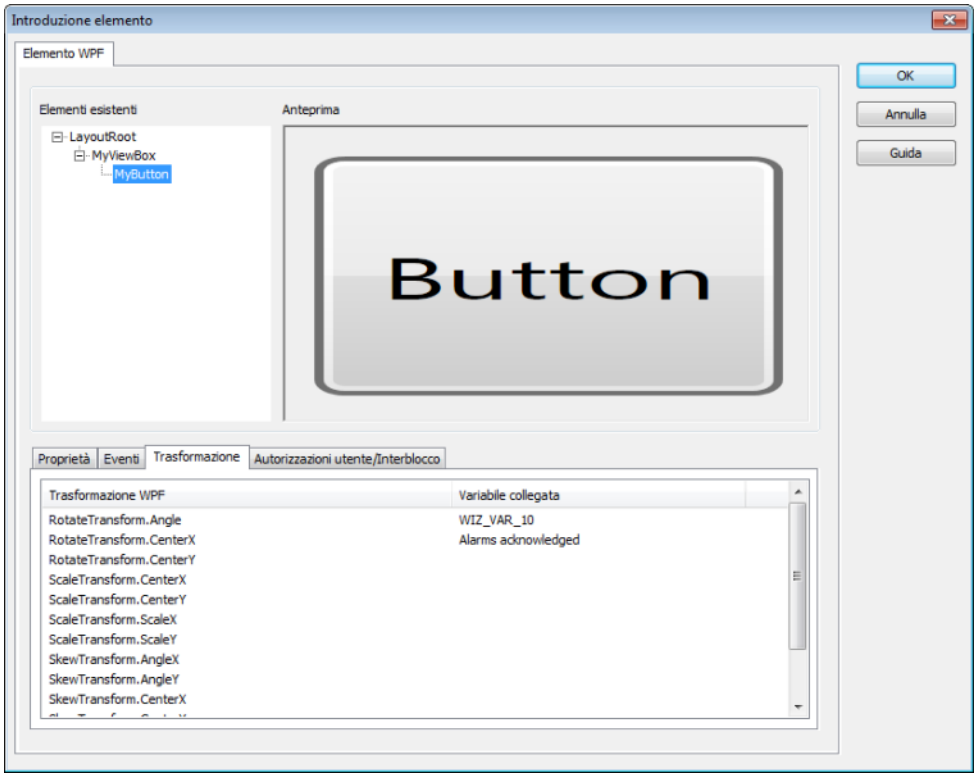


Parametro	Descrizione
Funzione collegata	Selezione della funzione da collegare. Cliccando sul pulsante ... si apre la finestra di dialogo per selezionare una funzione.
OK	Applica la selezione e chiude la finestra di dialogo.
Annulla	Elimina le modifiche e chiude la finestra di dialogo.
Guida	Apre la guida in linea.

6.3.3 Trasformazione

L´**elemento WPF** non supporta la funzionalità di rotazione. Ad esempio, se l´**elemento WPF** si trova in un simbolo e il simbolo viene ruotato, l'**elemento WPF** non ruota. Proprio per questo motivo, per WPF è stato introdotto un meccanismo proprio, detto **trasformazione**, che consente la rotazione oppure una trasformazione degli elementi. Queste trasformazioni vengono configurate nella scheda **Trasformazione**.

Attenzione: se il Content non rientra nell'ambito dell´**elemento WPF**, quella parte del contenuto viene persa o non disegnata.



Parametro	Descrizione
Nome	Nome della proprietà.
Collegamento	Selezione della variabile collegata. Le trasformazioni vengono visualizzate in XAML come oggetti di trasformazione con proprietà proprie. Se un elemento supporta una trasformazione, le possibili proprietà degli oggetti di trasformazione vengono rappresentate nella visualizzazione elenco. (Ulteriori informazioni in merito: Implementare un pulsante come WPF-XAML in zenon. (A pagina: 120) Se la variabile collegata viene impostata, ad esempio, sul valore *10*, questo valore viene scritto sulla destinazione WPF e l´elemento WPF

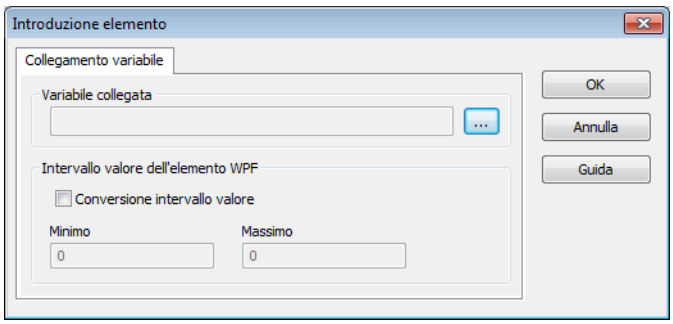
Parametro	Descrizione
	viene ruotato di 10°.
Tipo di collegamento	Selezione del tipo di collegamento “trasformazione”.
Info WPF	Per le proprietà, visualizza il valore corrente nel WPF-Content. Per l´utente è direttamente visibile di quale tipo di proprietà si tratta (booleana, stringa, ecc).
Collegato	Indica se una proprietà è attualmente in uso. Se non inclusa nella vista, la voce di colonna può essere selezionata tramite il **Menù contestuale-> Selezione colonne**.

COLLEGARE TRASFORMAZIONE

Per collegare una trasformazione ad una proprietà WPF:

1. Selezionare la riga con la proprietà da collegare.
2. Cliccare nella cella **Tipo di collegamento**.
3. Selezionate dal menù a tendina **trasformazione**
4. Cliccare nella cella **Collegamento**.
5. Si apre la finestra di dialogo per la configurazione delle variabili.

La configurazione permette anche la conversione di intervalli valori da zenon a WPF.



Parametro	Descrizione
Variabile collegata	Selezione della variabile da collegare. Fare clic sul pulsante **...** per aprire la finestra di dialogo per la selezione.
Campo valori dell'elemento WPF	Dati per la conversione dei valori variabile in valori WPF.
Conversione intervallo di valori	*Attiva*: la conversione unità WPF è attiva. **Effetto a Runtime:** il valore corrente di zenon

Parametro	Descrizione
	(inclusa unità zenon) viene convertito nel campo WPF mediante valori minimi e massimi standardizzati. **Esempio:** il valore di una variabile varia da 100 a 200. Per la variabile, l'intervallo normalizzato è impostato su 100 - 200. Lo scopo è quello di visualizzare questa variazione di valore usando una manopola WPF. Procedura: ▸ Alla voce **Trasformazione**, collegare la proprietà **RotateTransform.Angle** alla variabile. ▸ Attivare l´opzione **Adjust value** ▸ Configurare un campo di valori WPF da *0* a *360*. Ora, se, ad esempio, il valore è 150, la manopola viene ruotata di 180 gradi.
Minimo	Definisce il valore WPF più basso.
Massimo	Definisce il valore WPF più alto.
OK	Accetta le impostazioni e chiude la finestra di dialogo.
Annulla	Annulla le impostazioni e chiude la finestra di dialogo.
Guida	Apre la guida online.

6.4 Validità dei file XAML

La validità dei file XAML è soggetta a determinate condizioni:

- Namespaces corretti
- Nessun riferimento class
- Scalabilità

NAMESPACES CORRETTI

L´elemento WPF può visualizzare esclusivamente WPF-Content, cioè:

Solamente i file XAML con uno spazio dei nomi WPF corretto, possono essere visualizzati nell´**elemento WPF**. I file che usano un Silverlight-Namespace, non possono essere caricati o visualizzati in modo corretto. Nella maggior parte dei casi è però sufficiente sostituire Silverlight-Namespace con lo spazio dei nomi WPF.

WPF-Namespaces:

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

NESSUN USO DI RIFERIMENTI DI CLASSE

Poiché i file XAML sono caricati dinamicamente, non è possibile utilizzarli che contengono riferimenti alle classi (chiave "Class" nell'intestazione). La funzionalità programmata in file C# proprio non può essere utilizzata.

Per utilizzare i controlli utente WPF con Code Behind, procedere come descritto in Creazione di un semplice controllo utente WPF con funzionalità code behind (A pagina: 24).

SCALABILITÀ

Se il contenuto di un **elemento WPF** deve essere regolato in base alle dimensioni dell´**elemento WPF**, i comandi dell´**elemento WPF** devono essere annidati in un controllo che offra questa funzionalità, ad esempio un **ViewBox**. Inoltre, è necessario assicurarsi che **altezza** e **larghezza** di questi elementi siano configurate su **automatico**.

VERIFICA DI CORRETTEZZA UN FILE XAML

Per controllare se un file XAML ha il formato corretto:

- Aprir il file XAML in Internet-Explorer.
  - Se può essere aperto senza plugin aggiuntivi (Java o simili), allora questo file può anche essere caricato o visualizzato con alta sicurezza da zenon.
  - Se si verificano problemi durante il caricamento del file, vengono visualizzati in Internet Explorer ed è chiaramente visibile in quale riga ci sono problemi.

Anche il fattore di scala può essere testato in questo modo: Se il file è strutturato correttamente, il contenuto si adatta alle dimensioni della finestra di Internet Explorer.

MESSAGGIO DI ERRORE

Se in zenon viene utilizzato un file non valido, nella finestra di emissione viene visualizzato un messaggio di errore quando il file viene caricato nell'elemento WPF.

Esempio:

„Errore nel caricamento di *xaml-Datei:C:\ProgramData\COPA-DATA\SQL\781b1352-59d0-437e-a173-08563c3142e9\FILES\zenon\custom\media\UserControl1.xaml*

Non è possibile trovare l´attributo "Class" nel XML-Namespace "http://schemas.microsoft.com/winfx/2006/xaml".Riga 7 Posizione 2."

6.5 Elementi predefiniti

I partner registrati di zenon possono ottenere esempi di WPF tramite **MyArea** (**https://www.copadata.com/en/my-area/overview/**) sul sito web COPA-DATA.

C´è una serie di elementi WPF forniti con zenon. Altri possono essere scaricati dal web shop.

Tutti gli elementi WPF hanno delle proprietà che determinano l´aspetto grafico dell´elemento in questione. (Dependency Properties). Settando i valori con un file XAML, oppure collegando la proprietà via zenon, si può modificare l´aspetto direttamente nel Runtime. Le tabelle nella descrizione dei singoli elementi contengono le relative Dependency Properties, , in dipendenza del control.

Elementi disponibili:

- Orologio analogico (A pagina: 66)
- Bargraf verticale (A pagina: 66)
- BACnet WPF Control
- Comtrade-Viewer (A pagina: 68)
- Diagramma - Energy class (A pagina: 81)
- Barra di avanzamento (A pagina: 67)
- Diagramma di Pareto (A pagina: 82)
- Diagramma di Sankey (A pagina: 89)
- Visuale rotonda (A pagina: 85)
- Indicatore della temperatura (A pagina: 91)
- Slider universale (A pagina: 92)
- Diagramma a cascata (A pagina: 94)

SOSTITUISCI ASSEMBLY CON NUOVA VERSIONE

Per ogni progetto, si può usare sia nell´Editor che nel Runtime di zenon solamente un *Assembly* per un elemento WPF. Se in un progetto esistono due versioni di un *Assembly*, verrà usato il file caricato per primo. Il sistema provvederà a chiedere all´utente quale versione deve essere utilizzata. Non sono necessarie ulteriori azioni se la versione precedentemente utilizzata viene mantenuta. Se viene selezionata una versione più recente, tutti i file CDWPF corrispondenti in tutti i simboli e le immagini di tutti i progetti devono essere adattati.

Nota sulla gestione multi-progetto: Se in un progetto un *Assembly* viene sostituito da una versione più recente, deve essere sostituito anche in tutti gli altri progetti caricati nell'Editor o nel Runtime.

6.5.1 Orologio analogico - AnalogClockControl

Property	Funzione	Valore
ElementStyle	Forma/tipo dell´elemento.	*Enum:* ▸ *SmallNumbers* ▸ *BigNumbers* ▸ *No*
ElementBackgroundBrush	Colore dello sfondo dell´elemento.	*Brush*
ElementGlasReflection	Attivazione dell´effetto glas reflection sull´elemento.	*Visibility*
Offset	Valore in ore (h) che rappresenta la differenza di tempo rispetto all´orologio di sistema.	*Int16*
OriginText	Testo visualizzato nell'orologio (ad esempio, posizione).	*String*

6.5.2 Bargraf verticale - VerticalBargraphControl

Property	Funzione	Valore
CurrentValue	Valore attuale che deve essere visualizzato.	*Double*
MinValue	Valore minimo della scala.	*Double*
MaxValue	Valore massimo della scala.	*Double*

Property	Funzione	Valore
MajorTicksCount	Numero delle suddivisioni principali della scala.	*Integer*
MinorTicksCount	Numero delle suddivisioni secondarie della scala.	*Integer*
MajorTickColor	Colore delle suddivisioni principali della scala.	*Color*
MinorTickColor	Colore delle suddivisioni secondarie della scala.	*Color*
ElementBorderBrush	Colore della cornice dell´elemento.	*Brush*
ElementBackgroundBrush	Colore dello sfondo dell´elemento.	*Brush*
ElementGlasReflection	Attivazione dell´effetto glas reflection sull´elemento.	*Visibility*
ElementFontFamily	Elemento font.	*Font*
ScaleFontSize	Dimensione carattere della scala.	*Double*
ScaleFontColor	Colore carattere della scala.	*Color*
IndicatorBrush	Colore di riempimento del bargraf.	*Brush*
BargraphSeparation	Numero di suddivisioni del bargraf.	*Integer*
BargraphSeparationColor	Colore delle suddivisione della scala.	*Color*

6.5.3 Barra stato di avanzamento - ProgressBarControl

Property	Funzione	Valore
CurrentValue	Valore attuale che deve essere visualizzato.	*Double*
MinValue	Valore minimo dell'intervallo di valori.	*Double*
MaxValue	Valore massimo dell'intervallo di valori.	*Double*
ProgressbarDivisionCount	Numero di suddivisioni della barra di avanzamento (Boxed).	*Integer*
VisibilityText	Visibilità della visualizzazione valore.	*Boolean*
TextSize	Dimensione carattere della visualizzazione valore.	*Double*
TextColor	Colore della visualizzazione valore.	*Color*

Property	Funzione	Valore
ProgressBarBoxedColor	Colore della cornice della barra stato di avanzamento (Boxed).	*Color*
ProgressBarMarginDistance	Distanza della barra di avanzamento dal bordo dell'elemento (sinistra, superiore, destra, inferiore).	*Double*
ProgressBarInactiveBrush	Colore indicatore non attivo.	*Brush*
ProgressBarActiveBrush	Colore indicatore attivo.	*Brush*
ProgressBarPadding	Distanza della barra di stato avanzamento dal Progressbar-Box (sinistra, sopra destra, sotto).	*Double*
ElementBorderBrush	Colore della cornice dell´elemento.	*Brush*
ElementBackgroundBrush	Colore dello sfondo dell´elemento.	*Brush*

6.5.4 COMTRADE-Viewer

Il **COMTRADE-Viewer** L'elemento WPF è disponibile per i partner di COPA-DATA ed è a loro disposizione tramite la COPA-DATA Partner Community (https://www.copadata.com/en-us/partner-community/).

Informazioni sulla licenza

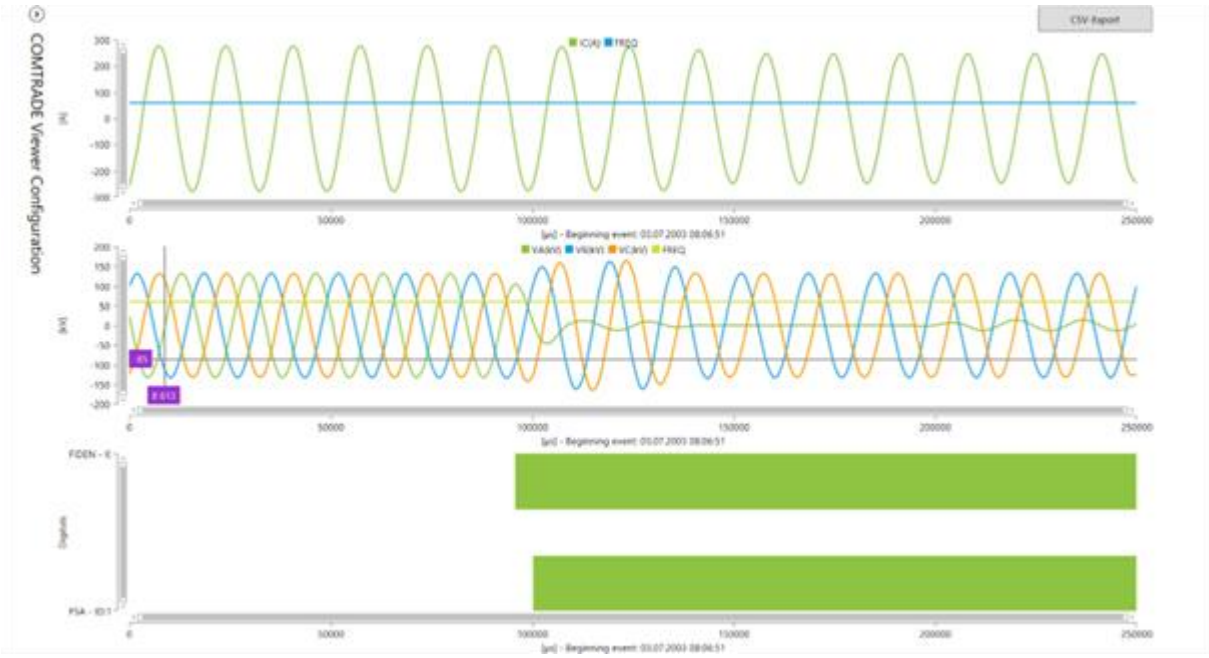
COMTRADE-Viewer può essere configurato solamente in un Editor di zenon, con una licenza Energy Edition valida. Se non esiste una licenza valida, il WPF sarà disattivato nell'Editor. Anche per la visualizzazione a Runtime è necessaria una licenza valida per la Energy Edition.

Il controllo supporta l'analisi grafica di registrazioni di disturbi digitali e record di dati di eventi di un file COMTRADE.

Info

Il controllo supporta i file conformi allo standard IEEE C37.111 (IEEE Standard Common Format for Transient Data Exchange (COMTRADE) for Power Systems). I file ASCII o binari possono essere visualizzati in edizione 1999 o 2013.

I file meno recenti o quelli senza un identificatore di anno non sono supportati. Quando si seleziona un file non valido o non supportato, viene visualizzata una finestra di dialogo di avviso.



Possibilità del **COMTRADE-Viewer** WPF nel Runtime di zenon:

- Selezione di un file in formato COMTRADE.
- Visualizzazione del file COMTRADE selezionato.
  Nota: i colori del display sono configurabili nell´Editor di zenon.
  - Corrente (rappresentazione sinusoidale)
  - Tensione (rappresentazione sinusoidale)
  - Segnali digitali (visualizzazione di diagramma a barre binario)
  - Visualizzazione di valori in corrispondenza di una determinata posizione del cursore.
  - Se viene selezionato un elemento che non rappresenta né corrente né tensione (ad es. frequenza), esso viene visualizzato in entrambi i campi analogici (corrente e tensione).
- Navigazione
  - Zoom avanti e zoom indietro con la rotella del mouse e con gesti multi-touch.
  - Ingrandimento di un´area.
    Selezione di quest´area con un clic del mouse.
  - Spostare l'area di visualizzazione con il tasto destro del mouse, la barra di scorrimento o i movimenti multi-touch.
- Esportazione degli oggetti selezionati in un file CSV.

Suggerimento

Per trasferire file COMTRADE su un computer su cui è in esecuzione il Runtime di zenon, è possibile utilizzare anche il trasferimento file del **driver IEC 61850**, oppure il **FTP function block** di zenon Logic .

Per ulteriori informazioni, consultare la documentazione del driver IEC 61850, o la documentazione di zenon Logic.

6.5.4.1 Visualizzazione a Runtime:

L´elemento WPF **COMTRADE** offre due viste a Runtime:

- Vista di configurazione
  - Selezione di un file di configurazione COMTRADE.
  - Selezione degli elementi da visualizzare.
- Vista grafico
  - Zoom avanti e zoom indietro.
  - Visualizzazione di valori in corrispondenza di una determinata posizione del cursore.
  - Esportazione degli elementi selezionati in un file CSV.

Info

Il passaggio da una visuale all´altra è integrato nell´elemento WPF. Non è dunque necessario progettare una funzione di cambio immagine.

6.5.4.2 Installazione

Requisiti:

- Si deve essere un partner registrato di COPA-DATA
- È necessario un accesso già installato alla COPA-DATA Partner Community.

PER SCARICARE UN ELEMENTO WPF:

1. Aprire COPA-DATA Partner Community (https://www.copadata.com/en-us/partner-community/).
2. Accedere all'area **MY AREA**.

3. Immettere "Partner" come termine di ricerca nella lente d'ingrandimento.

   COPA-DATA Partner Community viene visualizzata insieme a un link nell´area dei risultati.
4. Fare clic sul link.
5. Alla voce **Partner Login Area**, cliccare su **MORE**.

   Viene aperta la scheda **Overview**.
6. Passare alla scheda **Documents & Downloads**.
7. Alla voce **SELECT CATEGORY**, attivare la checkbox **Elemento WPF**.
8. Cliccare su **Search**.

   La didascalia scompare.

   Una cornice arancione delimita il campo di ricerca.
9. Fare clic nella cornice e immettere **COMTRADE**.
10. **COMTRADE Viewer** viene visualizzato.
11. Fare clic sull'icona arancione in **Actions** per scaricare **COMTRADE Viewer**.
12. Salvare la cartella localmente.
13. Decomprimere il file.
14. Aprire la cartella **COMTRADE** che contiene il file **Comtrade.cdwpf**.

6.5.4.3 Installazione

Per creare un elemento WPF:

1. Aprire l´Editor di zenon.
2. Passare al nodo **File** e **Grafica** del Manager di progetto.
3. Scegliere la voce **Aggiungi file...**.
4. Selezionare il file **Comtrade.cdwpf** da importare.
5. Confermare la selezione cliccando su **Apri**.
6. Aprire l'immagine in cui deve essere inserito l'elemento WPF.
7. Selezionare l'elemento WPF (**WPF**) nella barra dei menù verticale.
8. Disegnare l'elemento WPF nell'immagine con il tasto sinistro del mouse.

   Si apre la finestra **Scegli file**.
9. Selezionare il file **Comtrade.cdwpf** e confermare l'immissione cliccando su **OK**.

   L'elemento WPF è inserito.

6.5.4.4 Proprietà configurabili del controllo - Visualizzazione colori

CONFIGURAZIONE NELL'EDITOR

L´elemento chiamato **COMTRADE.CDWPF** può essere configurato e collocato in ogni tipo di immagine di zenon.
La progettazione della **Breite [Pixel]** e della **Höhe [Pixel]** dell´elemento sono proporzionalmente dipendenti. In questo modo si evita che **COMTRADE-Viewer** venga visualizzato in modo distorto a Runtime.

Nota: quando si configura il progetto, assicurarsi di specificare dimensioni sufficienti in modo da mantenere una chiara visione d'insieme.

ADATTAMENTI GRAFICI

Si configura il design grafico nelle proprietà dell´elemento WPF.
Per ulteriori informazioni, fare riferimento al capitolo Configurazione del collegamento (A pagina: 51) di questa guida.

Possibili valori del colore:

- È possibile utilizzare valori di colore esadecimali nei seguenti formati:
  #RRGGBB e *#AARRGGBB*
  I valori di trasparenza sono ammessi.
- **Esempi di valori per i colori:**
  #000000 = nero
  #FFFFFF = bianco
  #FF0000 = rosso.
- Valori di colore per nome
  Riferimento: https://msdn.microsoft.com/en-us/library/system.drawing.color.aspx (https://msdn.microsoft.com/en-us/library/system.drawing.color.aspx)

> **Suggerimento**
>
> Le proprietà per l´elemento WPF **COMTRADE-Viewer** hanno la lettera iniziale "*z*". Per una visione più chiara, utilizzare il filtro dei nomi durante la configurazione del collegamento.

PAGINA DI CONFIGURAZIONE

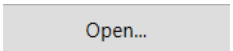
Testo e colore di sfondo della pagina di configurazione.

Analog Channels

Parametro	Descrizione	Valore
zConfiguratinPageTextColor	Colore del testo della pagina di configurazione.	*String*
zConfigurationPageBackgroundColor	Colore di sfondo della pagina di configurazione.	*String*

PULSANTI

Colore del testo e colore di sfondo dei pulsanti.



Parametro	Descrizione	Valore
zButtonTextColor	Colore del testo dei pulsanti.	*String*
zButtonBackgroundColor	Colore di sfondo dei pulsanti.	*String*

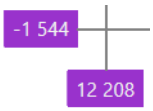
CHART

Colore del testo delle scritte dell´asse o legenda e colore di sfondo.

Parametro	Descrizione	Valore
zChartTextColor	Colore del testo dell'etichetta dell'asse.	*String*
zChartBackgroundColor	Colore di sfondo dell'etichetta dell'asse	*String*

LABEL

Colore del testo e dello sfondo della visualizzazione dei valori in corrispondenza della posizione selezionata del cursore.



Parametro	Descrizione	Valore
zChartLabelTextColor	Colore del testo della visualizzazione del valore.	*String*
zChartLabelBackgroundColor	Colore di sfondo della	*String*

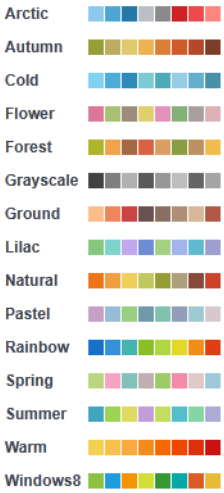
Parametro	Descrizione	Valore
	visualizzazione del valore.	

CHART

Tavolozza di colori della visuale grafici e della corrispondente legenda.

Parametro	Descrizione	Valore
zChartPalette	Tavolozza dei colori per grafici e legenda. Riferimento a nomi tavolozze dei colori (vedi visione generale). Default: se non è configurata nessuna tavolozza di colori, questa viene rilevata dal sistema operativo del computer.	*String*

POSSIBILI TAVOLOZZE DEI COLORI - PANORAMICA



6.5.4.5 Proprietà configurabili del controllo - Impostazioni lingua e visualizzazione

Tramite le seguenti impostazioni si può definire definire quanto segue:

- Se utilizzare la finestra di dialogo standard o una finestra di dialogo definita dall'utente per aprire i file di Windows.

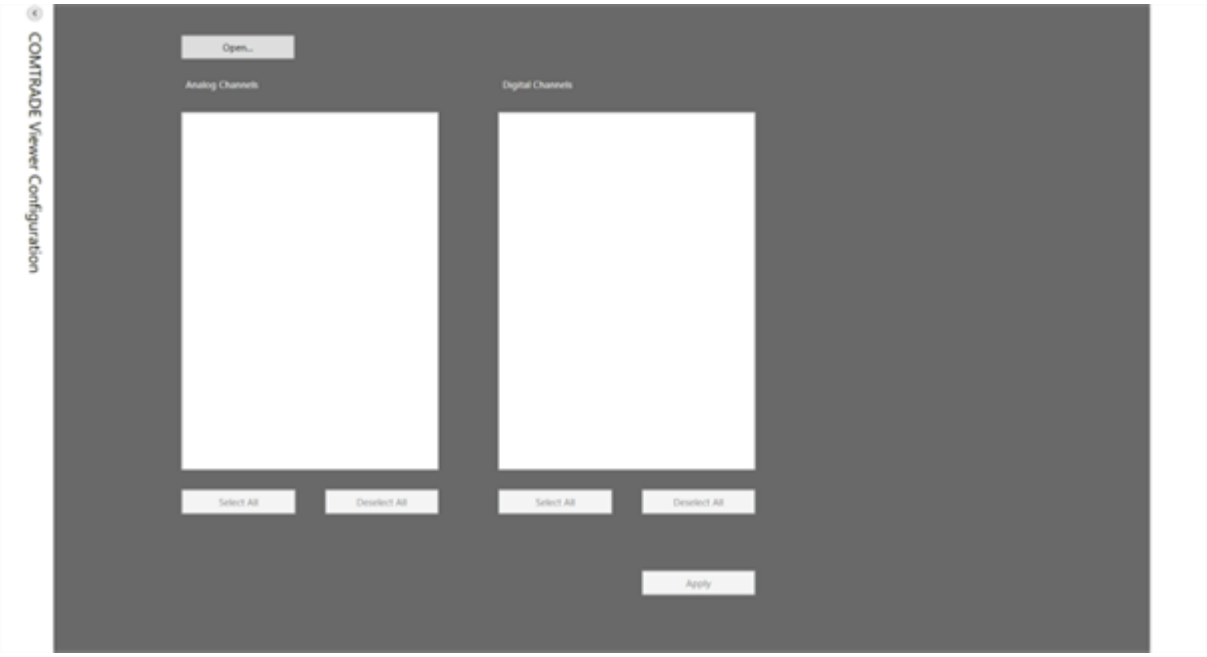
- Se l'interfaccia utente deve essere visualizzata in polacco.
- Se il pulsante **Esportazione CSV** deve essere visualizzato a Runtime.

Parametro	Descrizione
zLimitFileOpenAccessTo	Se il parametro è impostato su un valore diverso da "vuoto", si apre una finestra di dialogo personalizzata per aprire i file di Windows. Questa finestra di dialogo visualizza tutti i file ***.CFG** per i quali esiste un corrispondente file ***.DAT** nella cartella indirizzata dal parametro. Default: *vuoto*
zUiLanguage	Quando il parametro è impostato su uno dei seguenti valori, l'interfaccia utente di **COMTRADE-Viewer** e la finestra di dialogo personalizzata di apertura dei file vengono visualizzate in polacco: ▸ *pl* ▸ *pL* ▸ *Pl* ▸ *PL* **Nota:** nella finestra di dialogo standard di apertura file, solo il titolo viene visualizzato in polacco. Le altre voci vengono visualizzate secondo le impostazioni della lingua di Windows. Default: *en*
zButtonCSVExportVisibility	Definisce la visualizzazione del pulsante **Esportazione CSV** a Runtime: ▸ *True*: il pulsante viene visualizzato. ▸ *False*: il pulsante non viene visualizzato. Default: *True*

6.5.4.6 Visuale Runtime - Pagina di configurazione

Quando si apre a Runtime un´immagine in cui è stato configurato un elemento WPF **COMTRADE-Viewer**, la relativa pagina di configurazione è sempre vuota.

Nota: questo vale anche se nel Runtime di zenon si passa da un'altra immagine a questa eseguendo la funzione di cambio immagine..



CONFIGURAZIONE DI COMTRADE VIEWER

Il comando **COMTRADE Viewer Configuration** posto lateralmente in alto, cambia la visualizzazione dalla configurazione alla visualizzazione grafica e viceversa.

SELEZIONARE UN FILE

Il pulsante **Open...** apre la finestra di selezione per selezionare un file.

Per la visualizzazione nella finestra di selezione file viene eseguita una preselezione:

- Le coppie di file **.cfg-* e **.dat* vengono riconosciute.
  Nota: i file opzionali del tipo **.hdr* o **.inf* non vengono presi in considerazione.
- Vengono visualizzati solo i file **.dat* corrispondenti.
- Cliccando sul file desiderato e sul pulsante **OK**, vengono caricati tutti i file associati (**.dat*, **.cfg*).
- È possibile caricare un solo file alla volta.
- Caricato il file, il suo contenuto viene visualizzato nelle due colonne **Analog Channels** e **Digital Channels**.
  I nomi e le unità degli elementi hanno la loro origine nella configurazione **COMTRADE** e non possono essere modificati.

ULTERIORI INFORMAZIONI SULL´ELABORAZIONE DI FILE _*.CFG- E *.DAT.

Le informazioni del file *.cfg consentono l´analisi del file *.dat. Quest´ultimo contiene i dati delle varie serie di misure analogiche e digitali di correnti e tensioni. I dati sono suddivisi in singoli dataset e rappresentati in formato Hex.

File *.cfg

- L´ultimo inserimento di un file di questo tipo di dato è un moltiplicatore di tempo. Questa voce viene moltiplicata per il time stamp di ciascuna voce del file *.dat durante la lettura di un record di errore. Se non c'è un moltiplicatore di tempo, si assume internamente un fattore con il valore 1. Il file *.cfg non viene modificato.
- Per l'immissione dei valori di misura digitali valgono determinate norme. Esempio di immissione di un valore di misura digitale conforme allo standard: 1,LOPHC,,,0. Tuttavia, se lo zero non è presente alla fine della voce, **COMTRADE-Viewer** lo aggiunge internamente. Il file *.cfg non viene modificato.

File *.dat

- **COMTRADE-Viewer** è in grado di leggere file di questo tipo di dati che iniziano con l´indice 0 o >1. Nel corso di questo processo, il sistema analizza sempre se questi dataset sono numerati consecutivamente in passi discreti di 1. Se i record non sono numerati correttamente, il file non può essere letto.

ANALOG CHANNELS

Parametro	Descrizione
[Liste der verfügbaren Kanäle]	Selezione degli elementi da visualizzare. Selezione multipla facendo clic sulla voce desiderata dell'elenco. Gli elementi selezionati sono visualizzati su uno sfondo colorato. Un altro clic del mouse rimuove la selezione della voce.
Select All	Seleziona tutti gli elementi della lista.
Deselect All	Disattiva la selezione esistente di elementi.

DIGITAL CHANNELS

Parametro	Descrizione
[Liste der verfügbaren Kanäle]	Selezione degli elementi da visualizzare. Selezione multipla facendo clic sulla voce desiderata dell'elenco. Gli elementi selezionati sono visualizzati su uno sfondo colorato. Un altro clic del

Parametro	Descrizione
	mouse rimuove la selezione della voce.
Select All	Seleziona tutti gli elementi della lista.
Deselect All	Disattiva la selezione esistente di elementi.

VISUALIZZARE LA SELEZIONE

Per visualizzare la selezione nella visualizzazione grafica, fare clic sul pulsante **Apply**.

Nota: un clic sull'interruttore disposto verticalmente **COMTRADE Viewer Configuration** cambia solo la vista. La modifica della selezione dei canali non viene presa in considerazione.

6.5.4.7 Visuale Runtime - Visualizzazione di dati COMTRADE

I canali selezionati vengono visualizzati nella visualizzazione grafica dell'elemento WPF **COMTRADE-Viewer**. I colori da utilizzare possono essere configurata nell'Editor zenon.

I file conformi allo standard **IEEE C37.111-2013** (o **IEC 60255-24:2013**) possono essere importati.

Per la visualizzazione sono necessari i valori X che rappresentano i time stamp del punto dato registrato.

I valori X possono essere memorizzati direttamente nel file ***.dat**.

In caso contrario, i time stamp sono calcolati sulla base delle seguenti voci del file ***.cfg**:

- Ora di inizio della registrazione del record di guasto (data e ora del primo punto dati).
- Numero di punti di dati registrati (**Samplerate**)
- Numero di punti dati per ogni interrogazione (**Rate**)

ESPORTAZIONE DEI DATI SELEZIONATI

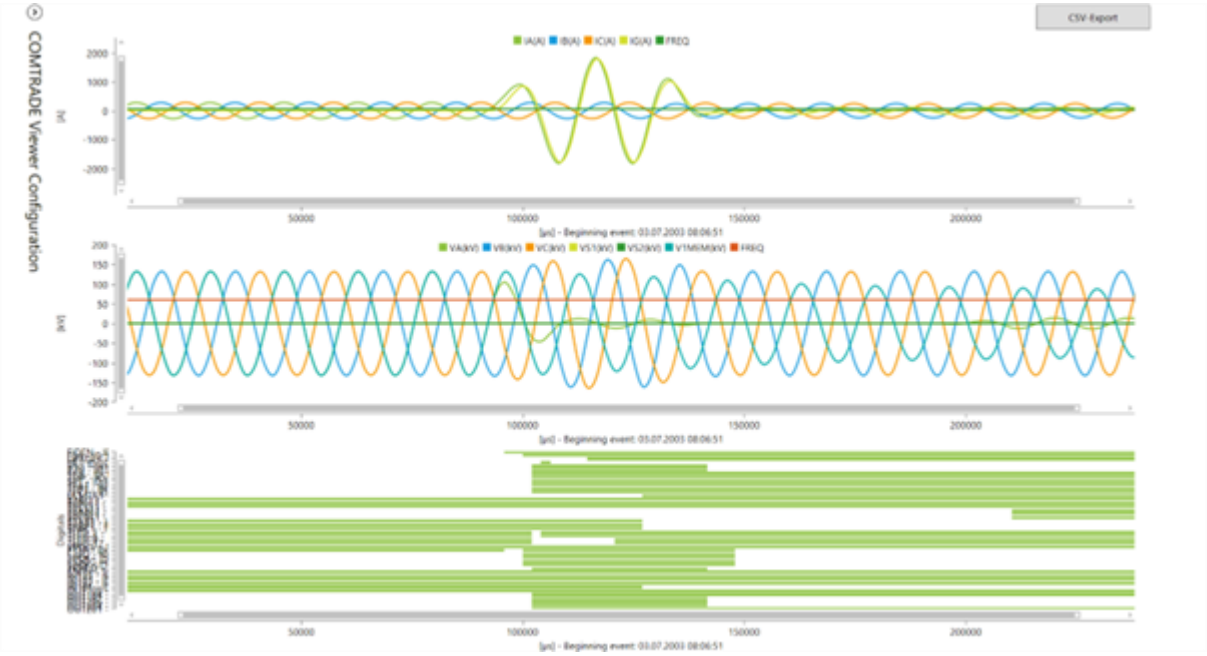
I canali analogici e digitali selezionati possono essere esportati in un file CSV con il pulsante **CSV-Export**.

VISTA GRAFICO

La vista grafica di **COMTRADE-Viewer** è divisa in tre sezioni:

- Intensità di corrente
  Sezione in alto.
- Tensione
  Sezione centrale.

- Canali digitali
  Sezione in basso.



ETICHETTA ASSE

- Asse orizzontale
  L´asse orizzontale rappresenta l´intero periodo di tempo, così come registrato nel file COMTRADE (*.*dat*).
  La scala di questo asse tempo dipende dal fattore di ingrandimento. Più grande è l'ingrandimento, più dettagliata diventa la visualizzazione dell'ora.
- Asse verticale
  L´asse verticale rappresenta i valori.
  - La scala dell'asse dei valori dipende dal fattore di ingrandimento. Più grande è il fattore di ingrandimento selezionato, più dettagliata diventa la visualizzazione del valore.
  - Il nome dei canali analogici viene visualizzato verticalmente accanto ai valori e corrisponde all'unità di misura definita nel file COMTRADE (*.*cfg*).
  - I canali digitali vengono visualizzati nella successione definita nel file COMTRADE (*.*cfg*).
    Il *Channel identifier* del file COMTRADE serve da identificazione.

LEGENDA

IA(A) IB(A) IC(A) IG(A)

La legenda colori dei grafici appare sopra ai grafici stessi.

- La denominazione dei canali digitali corrisponde a quella definita nel file COMTRADE (*.*cfg*).

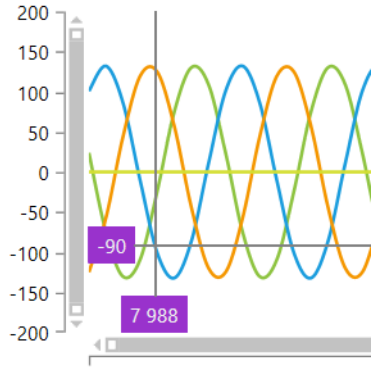
- L'assegnazione dei colori per canale avviene automaticamente con la tavolozza dei colori configurata.
- L'ora viene visualizzata in un piè di pagina sotto il grafico. Il tempo iniziale viene visualizzato in forma di testo.

NAVIGAZIONE E ZOOM

La navigazione (scrolling e zoom) è applicata sempre a tutte e tre le sezioni della rappresentazione grafica.

- Utilizzare la barra di scorrimento per spostare il display all'interno della linea del tempo orizzontale.
- Zoom avanti e zoom indietro.
  - Utilizzare la rotellina del mouse per ingrandire la visualizzazione del grafico nella posizione corrente del puntatore del mouse o per ridurre l'ingrandimento.
  - Selezionando un'area di visualizzazione tenendo premuto il tasto del mouse si visualizza l'area selezionata.
    Nota: la visualizzazione dei valori è sempre adattata all'intervallo selezionato.. Ciò può portare ad un appiattimento della curva nella vista del grafico ingrandita.
  - Facendo doppio clic sulla barra di scorrimento si resetta l´ingrandimento al fattore iniziale.

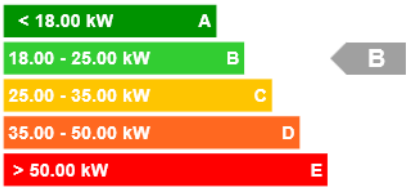
ANALISI



I valori esatti nella posizione del puntatore del mouse sono visualizzati in forma di blocchi di valore. Un mirino fornisce supporto ottico aggiuntivo per la determinazione esatta della posizione di lettura.

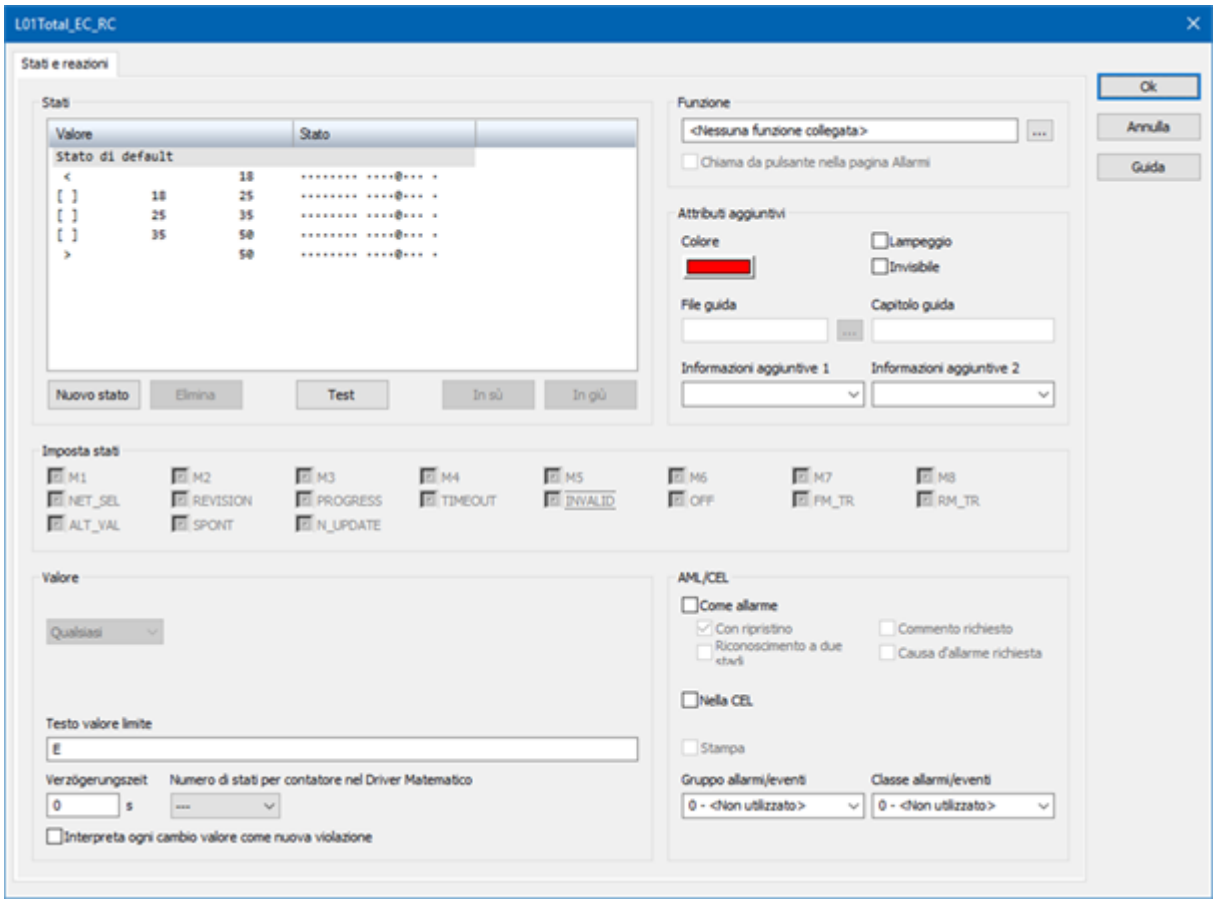
6.5.5 Diagramma - Energy class

Il diagramma Energy class, L'elemento WPF è disponibile per i partner di COPA-DATA ed è a loro disposizione tramite la COPA-DATA Partner Community (https://www.copadata.com/en-us/partner-community/).



Per modellare un diagramma energy class è necessario utilizzare una matrice di reazione. Questa matrice di reazione deve essere collegata alla variabile il cui valore è previsto per la visualizzazione e la classificazione in classi energetiche. Il nome della variabile deve essere trasferito alla proprietà "**zVariableName**".

MATRICE DI REAZIONE PER DIAGRAMMA ENERGY CLASS.



La matrice di reazione collegata deve essere conforme allo schema seguente:

- Il primo stato deve essere un´area, oppure una definizione "meno di".

- Poi si possono definire quante aree si vuole.
- L´ultimo stato deve essere un´area, oppure una definizione "maggiore di".

Per la progettazione vale quanto segue:

1. Se il primo stato è un'area e il valore della variabile scende al di sotto di questo intervallo, viene ancora visualizzato il primo stato nel diagramma. Lo stesso vale, viceversa, per l´ultimo stato.
2. I colori usati dal diagramma WPF per le classi sono i colori di valore limite definiti nella matrice di reazione.
3. Le lettere usate per le classi sono in ordine alfabetico partendo dalla "A".

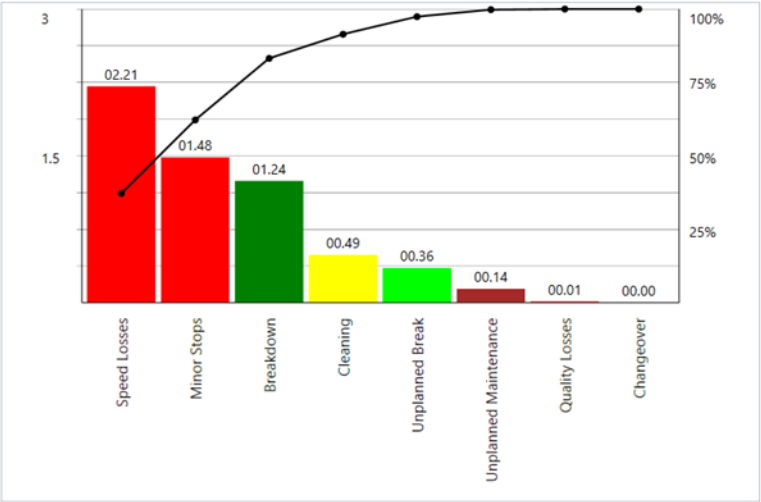
Property	Descrizione	Valore
zenonFontID	ID di un font dalla prima lista caratteri (la dimensione carattere non viene presa in considerazione).	Integer
zenonNumberOfDecimalPlaces	Numero delle cifre decimali visualizzate.	Integer
zenonVariableName	Nome della variabile da visualizzare.	Stringa

Nota: per visualizzare il diagramma Energy class in zenon Web Client è necessaria un'ulteriore programmazione VSTA. Per i dettagli, vedere Visualizzazione di elementi WPF in zenon Web Client (A pagina: 112).

6.5.6 Diagramma di Pareto

Il diagramma di Pareto, L'elemento WPF è disponibile per i partner di COPA-DATA ed è a loro disposizione tramite la COPA-DATA Partner Community (https://www.copadata.com/en-us/partner-community/).

Di seguito è riportato un esempio di un diagramma di Pareto a Runtime:



Le seguenti impostazioni possono essere eseguite nella finestra di configurazione WPF sotto **COPADATA-ELEMENT**:

Property	Funzione	Valore
zenonBarColor1	Colore della 1° Barra	*COLOR (STRING)*
zenonBarColor2	Colore della 2° Barra	*COLOR (STRING)*
zenonBarColor3	Colore della 3° Barra	*COLOR (STRING)*
zenonBarColor4	Colore della 4° Barra	*COLOR (STRING)*
zenonBarColor5	Colore della 5° Barra	*COLOR (STRING)*
zenonBarColor6	Colore della 6° Barra	*COLOR (STRING)*
zenonBarColor7	Colore della 7° Barra	*COLOR (STRING)*
zenonBarColor8	Colore della 8° Barra	*COLOR (STRING)*
zenonBarColor9	Colore della 9° Barra	*COLOR (STRING)*
zenonBarColor10	Colore della 10° Barra	*COLOR (STRING)*
zenonColorPercentageLine	Colore della percentage line (relative sum frequency).	*COLOR (STRING)*
zenonInitialisationTime	Specifica per quanto tempo attendere per inizializzare il controllo.	*INTEGER* Default: 150 ms
zenonLineVisibility	Visibilità della percentage line (relative sum frequency).	*BOOLEAN*

Property	Funzione	Valore
zenonVariable1_Label	Etichetta per la 1° Barra	*STRING*
zenonVariable1_Value	Valore della 1° Barra	*DOUBLE*
zenonVariable2_Label	Etichetta per la 2° Barra	*STRING*
zenonVariable2_Value	Valore della 2° Barra	*DOUBLE*
zenonVariable3_Label	Etichetta per la 3° Barra	*STRING*
zenonVariable3_Value	Valore della 3° Barra	*DOUBLE*
zenonVariable4_Label	Etichetta per la 4° Barra	*STRING*
zenonVariable4_Value	Valore della 4° Barra	*DOUBLE*
zenonVariable5_Label	Etichetta per la 5° Barra	*STRING*
zenonVariable5_Value	Valore della 5° Barra	*DOUBLE*
zenonVariable6_Label	Etichetta per la 6° Barra	*STRING*
zenonVariable6_Value	Valore della 6° Barra	*DOUBLE*
zenonVariable7_Label	Etichetta per la 7° Barra	*STRING*
zenonVariable7_Value	Valore della 7° Barra	*DOUBLE*
zenonVariable8_Label	Etichetta per la 8° Barra	*STRING*
zenonVariable8_Value	Valore della 8° Barra	*DOUBLE*
zenonVariable9_Label	Etichetta per la 9° Barra	*STRING*
zenonVariable9_Value	Valore della 9° Barra	*DOUBLE*
zenonVariable10_Label	Etichetta per la 10° Barra	*STRING*
zenonVariable10_Value	Valore della 10° Barra	*DOUBLE*

I seguenti eventi possono essere usati e collegati a funzioni di zenon:

Event	Funzione	Valore
zenonBar1Click	Funzione che viene eseguita quando si clicca sulla 1° barra.	Funzione
zenonBar2Click	Funzione che viene eseguita quando si clicca sulla 2° barra.	Funzione
zenonBar3Click	Funzione che viene eseguita quando si	Funzione

Event	Funzione	Valore
	clicca sulla 3° barra.	
zenonBar4Click	Funzione che viene eseguita quando si clicca sulla 4° barra.	Funzione
zenonBar5Click	Funzione che viene eseguita quando si clicca sulla 5° barra.	Funzione
zenonBar6Click	Funzione che viene eseguita quando si clicca sulla 6° barra.	Funzione
zenonBar7Click	Funzione che viene eseguita quando si clicca sulla 7° barra.	Funzione
zenonBar8Click	Funzione che viene eseguita quando si clicca sulla 8° barra.	Funzione
zenonBar9Click	Funzione che viene eseguita quando si clicca sulla 9° barra.	Funzione
zenonBar10Click	Funzione che viene eseguita quando si clicca sulla 10° barra.	Funzione

6.5.7 Visualizzazione circolare - CircularGaugeControl

Property	Funzione	Valore
CurrentValue	Valore attuale che deve essere visualizzato.	*Double*
IsReversed	Orientamento scale, in senso orario o antiorario.	*Boolean*
ElementFontFamily	Elemento font.	*Font*
MinValue	Valore minimo della scala.	*Double*
MaxValue	Valore massimo della scala.	*Double*
ScaleRadius	Raggio della scala.	*Double*
ScaleStartAngle	Angolo di inizio della scala.	*Double*
ScaleLabelRotationMode	Orientamento del titolo della scala.	*Enum:* ▸ *None* ▸ *Automatic*

Property	Funzione	Valore
		▸ *Surround In* ▸ *Surround Out*
ScaleSweepAngle	Intervallo angolare che definisce la dimensione della scala.	*Double*
ScaleLabelFontSize	Dimensione carattere dell´etichetta della scala.	*Double*
ScaleLabelColor	Colore del carattere dell´etichetta della scala.	*Color*
ScaleLabelRadius	Raggio di allineamento dell'etichetta della scala.	*Double*
ScaleValuePrecision	Precisione dell'etichetta della scala.	*Integer*
PointerStyle	Forma del pointer che indica il valore.	*Enum:* ▸ *Arrow* ▸ *Rectangle* ▸ *TriangleCap* ▸ *Pentagon* ▸ *Triangle*
MajorTickColor	Colore delle suddivisioni principali della scala.	*Color*
MinorTickColor	Colore delle suddivisioni secondarie della scala.	*Color*
MajorTickSize	Dimensione delle suddivisioni principali della scala.	*Size*
MinorTickSize	Dimensione delle suddivisioni secondarie della scala.	*Size*
MajorTicksCount	Numero delle suddivisioni principali della scala.	*Integer*
MajorTicksShape	Forma/tipo delle suddivisioni principali della scala.	*Enum:* ▸ *Rectangle* ▸ *Trapezoid* ▸ *Triangle*
MinorTicksShape	Forma/tipo delle suddivisioni secondarie della scala.	*Enum:*

Property	Funzione	Valore
		▸ *Rectangle* ▸ *Trapezoid* ▸ *Triangle*
MinorTicksCount	Numero delle suddivisioni secondarie della scala.	*Integer*
PointerSize	Dimensione del pointer.	*Size*
PointerCapRadius	Dimensione del punto di fissaggio del pointer.	*Double*
PointerBorderBrush	Colore della cornice del pointer	*Brush*
PointerCapStyle	Forma/tipo del punto di fissaggio del pointer.	*Enum:* ▸ *BackCap* ▸ *FrontCap* ▸ *Screw*
PointerCapBorderBrush	Colore del punto di fissaggio del pointer.	*Brush*
PointerBrush	Colore del´indicatore.	*Brush*
GaugeBorderBrush	Colore della cornice dell´elemento.	*Brush*
GaugeBackgroundBrush	Colore dello sfondo dell´elemento.	*Brush*
PointerCapColorBrush	Colore del punto di fissaggio del pointer.	*Brush*
GaugeMiddlePlate	Raggio della placca centrale dello sfondo dell'elemento.	*Double*
PointerOffset	Offset del pointer (spostamento).	*Double*
RangeRadius	Raggio dell'area totale di visualizzazione.	*Double*
RangeThickness	Spessore dell'area totale di visualizzazione.	*Double*
RangeStartValue	Valore iniziale dell'area totale di visualizzazione.	*Double*
RangexEndValue	Valore finale di un intervallo. x indica l'area. Il valore finale è anche il valore iniziale dell'intervallo successivo:	*Double*

Property	Funzione	Valore
	▸ **Range1EndValue**: Valore finale della 1° area e valore di start della 2° area. ▸ **Range2EndValue**: Valore finale della 2° area e valore di start della 3° area. ▸ **Range3EndValue**: Valore finale della 3° area e valore di start della 4° area. ▸ **Range4EndValue**: Valore finale della 4° area e valore di start della 5° area. ▸ **Range5EndValue**: Valore finale della 5° area e valore di start della 6° area. ▸ **Range6EndValue**: Valore finale della 6° area.	
RangexColorBrush	Colore della rispettiva area. x indica l'area. Sono disponibili: ▸ **Range1ColorBrush**: Colore della 1° area. ▸ **Range2ColorBrush**: Colore della 2° area. ▸ **Range3ColorBrush**: Colore della 3° area. ▸ **Range4ColorBrush**: Colore della 4° area. ▸ **Range5ColorBrush**: Colore della 5° area. ▸ **Range6ColorBrush**: Colore della 6° area. **Nota:** non può essere impostato in zenon. Per informazioni sull'uso di CDWPF per modificare gli intervalli di colori, fare riferimento a **Assembly referenziati** (A pagina: 8), **File CDWP** (A pagina: 49) e **Proprietà configurabili del controllo - Visualizzazione colori** (A pagina: 72).	*Brush*
ScaleOuterBorderBrush	Colore del bordo della scala.	*Brush*
ScaleBackgroundBrush	Colore dello sfondo della scala.	*Brush*
ValueTextFrameStyle	Forma/tipo della visualizzazione valore.	*Enum:* ▸ *LargeFrame* ▸ *SmallFrame*

Property	Funzione	Valore
		▸ *None*
ValueTextContent	Contenuto della visualizzazione valore.	*Enum:* ▸ *Text* ▸ *TextValue* ▸ *Value*
ValueTextSize	Dimensione carattere della visualizzazione valore.	*Double*
ValueTextColor	Colore carattere della visualizzazione valore.	*Color*
IsGlasReflection	Attivazione dell´effetto glas reflection sull´elemento.	*Boolean*
GaugeOffsett	Abbassamento del punto di rotazione di tutto l´elemento.	*Double*

6.5.8 Diagramma di Sankey

Il diagramma di Sankey, L'elemento WPF è disponibile per i partner di COPA-DATA ed è a loro disposizione tramite la COPA-DATA Partner Community (https://www.copadata.com/en-us/partner-community/).

Per modellare un diagramma di Sankey, si deve usare il Sankey Wizard. Questo Wizard crea un file XML, che viene quindi valutato dall'elemento WPF. A questo scopo, la proprietà **zSankeyName** deve avere il nome del file XML. Il file XML deve trovarsi nella cartella *Altri* (nodo "File" del Manager di progetto) di un progetto del workspace. Il file viene salvato dal Wizard in questa cartella.

Di seguito è riportato un esempio di un diagramma di Sankey a Runtime:

Le seguenti impostazioni possono essere eseguite nella finestra di configurazione WPF sotto **COPADATA-ELEMENT**:

Property	Funzione	Valore
FontSize	Dimensione dei caratteri dei testi.	*Integer*
zBackgroundColor	Colore di sfondo del diagramma.	*Color (String)*
zFontColor	Colore dei testi.	*Color (String)*

Property	Funzione	Valore
zFontFamily	Tipo di carattere di tutti i testi.	*Font (String)*
zLossDetectionActive	Rilevamento automatico delle perdite attivato/disattivato. Se *true*, le perdite in un nodo vengono automaticamente visualizzate come flussi.	*Bool*
zNoDataText	Testo visualizzato se non ci sono valori da visualizzare e **zPrevireActive** è *false*.	*String*
zNoValidXMLText	Testo visualizzato se non è stato trovato nessun file XML valido con il nome immesso e **zPreviewActive** è false.	*String*
zNumberOfDecimalPlaces	Specifica il numero di posizioni decimali visualizzate.	*Integer*
zPreviewActive	Visualizzazione di un´anteprima attivata/disattivata. L´anteprima viene visualizzata se Non ci sono dati da visualizzare (il diagramma modellato viene popolato con valori predefiniti), oppure Il file XML non è stato trovato, oppure Il file non contiene una definizione valida (viene visualizzato un esempio di diagramma di Sankey).	*Bool*
zRefreshRate	Refreshrate del diagramma in ms.	*Integer*
zSankeyName	Nome del file XML contenente la modellazione del diagramma.	*String*
zShowRelativeValues	Visualizzazione dei valori in assoluti *false* oppure relativi *true*.	*Bool*

Nota: per visualizzare il diagramma di Sankey in zenon Web Client è necessaria un'ulteriore programmazione VSTA. Per i dettagli, vedere Visualizzazione di elementi WPF in zenon Web Client (A pagina: 112).

6.5.9 Visualizzazione della temperatura - TemperatureIndicatorControl

Property	Funzione	Valore
CurrentValue	Valore attuale che deve essere visualizzato.	*Double*
MinValue	Valore minimo della scala.	*Double*
MaxValue	Valore massimo della scala.	*Double*
MajorTicksCount	Numero delle suddivisioni principali della scala.	*Integer*
MinorTicksCount	Numero delle suddivisioni secondarie della scala.	*Integer*
TickNegativColor	Colore della suddivisione principale negativa (che passa gradualmente a TickPositivColor).	*Color*
TickPositivColor	Colore della suddivisione principale positiva (che passa gradualmente a TickNegativColor).	*Color*
MinorTickColor	Colore delle suddivisioni secondarie	*Color*
ElementBorderBrush	Colore della cornice dell´elemento.	*Brush*
ElementBackgroundBrush	Colore dello sfondo dell´elemento.	*Brush*
ElementGlasReflection	Attivazione dell´effetto glas reflection sull´elemento.	*Visibility*
ElementFontFamily	Elemento font.	*Font*
IndicatorColor	Colore di riempimento dell´indicatore.	*Color*
IndicatorBorderColor	Colore della cornice dell´indicatore.	*Color*
MajorTickSize	Dimensione delle suddivisioni principali della scala.	*Size*
MinorTickSize	Dimensione delle suddivisioni secondarie della scala.	*Size*
ScaleLetteringDistance	Distanza della didascalia della scala (verticale); ogni x suddivisione principale deve essere etichettata.	*Integer*
IndicatorScaleDistance	Distanza fra indicatore e scala (orizzontale).	*Double*
ScaleFontSize	Dimensione carattere della scala.	*Double*
ScaleFontColor	Colore carattere della scala.	*Color*
Unit	Unità.	*String*
ElementStyle	Forma/tipo dell´elemento.	*Enum:* ▸ *SmallFra*

Property	Funzione	Valore
		me ▸ *Unit* ▸ *None*

6.5.10 Slider universale - UniversalReglerControl

Property	Funzione	Valore
CurrentValue	Valore attuale che deve essere visualizzato.	*Double*
ElementFontFamily	Elemento font.	*Font*
MinValue	Valore minimo della scala.	*Double*
MaxValue	Valore massimo della scala.	*Double*
Radius		*Double*
ScaleRadius	Raggio della scala.	*Double*
ScaleStartAngle	Angolo di inizio della scala.	*Double*
ScaleLabelRotationMode	Orientamento del titolo della scala.	*Enum:* ▸ *None* ▸ *Automatic* ▸ *SurroundIn* ▸ *SurroundOut*
ScaleSweepAngle	Intervallo angolare che definisce la dimensione della scala.	*Double*
ScaleLabelFontSize	Dimensione carattere dell´etichetta della scala.	*Double*
ScaleLabelColor	Colore del carattere dell´etichetta della scala.	*Color*
ScaleLabelRadius	Raggio di allineamento dell'etichetta della scala.	*Double*
ScaleValuePrecision	Precisione dell'etichetta della scala.	*Integer*
ElementStyle	Tipo di visualizzazione dell´elemento:	*Enum:* ▸ *Knob*

Property	Funzione	Valore
		▸ *Plate* ▸ *None*
MajorTickColor	Colore delle suddivisioni principali della scala.	*Color*
MinorTickColor	Colore delle suddivisioni secondarie della scala.	*Color*
MajorTickSize	Dimensione delle suddivisioni principali della scala.	*Size*
MinorTickSize	Dimensione delle suddivisioni secondarie della scala.	*Size*
MajorTicksCount	Numero delle suddivisioni principali della scala.	*Integer*
MajorTicksShape	Forma/tipo delle suddivisioni principali della scala.	*Enum:* ▸ *Rectangle* ▸ *Trapezoid* ▸ *Triangle*
MinorTicksShape	Forma/tipo delle suddivisioni secondarie della scala.	*Enum:* ▸ *Rectangle* ▸ *Trapezoid* ▸ *Triangle*
MinorTicksCount	Numero delle suddivisioni secondarie della scala.	*Integer*
BackgroundBorderBrush	Colore della cornice dell´elemento.	*Brush*
BackgroundBrush	Colore dello sfondo dell´elemento.	*Brush*
PointerCapColorBrush	Colore del punto di fissaggio del pointer.	*Brush*
GaugeMiddlePlate	Raggio della placca centrale dello sfondo dell'elemento.	*Double*
ValueFontSize	Dimensione carattere della visualizzazione valore.	*Double*
ValueFontColor	Colore carattere della visualizzazione valore.	*Color*
IsGlasReflection	Attivazione dell´effetto glas reflection sull´elemento.	*Boolean*
KnobBrush	Colore della manopola.	*Brush*

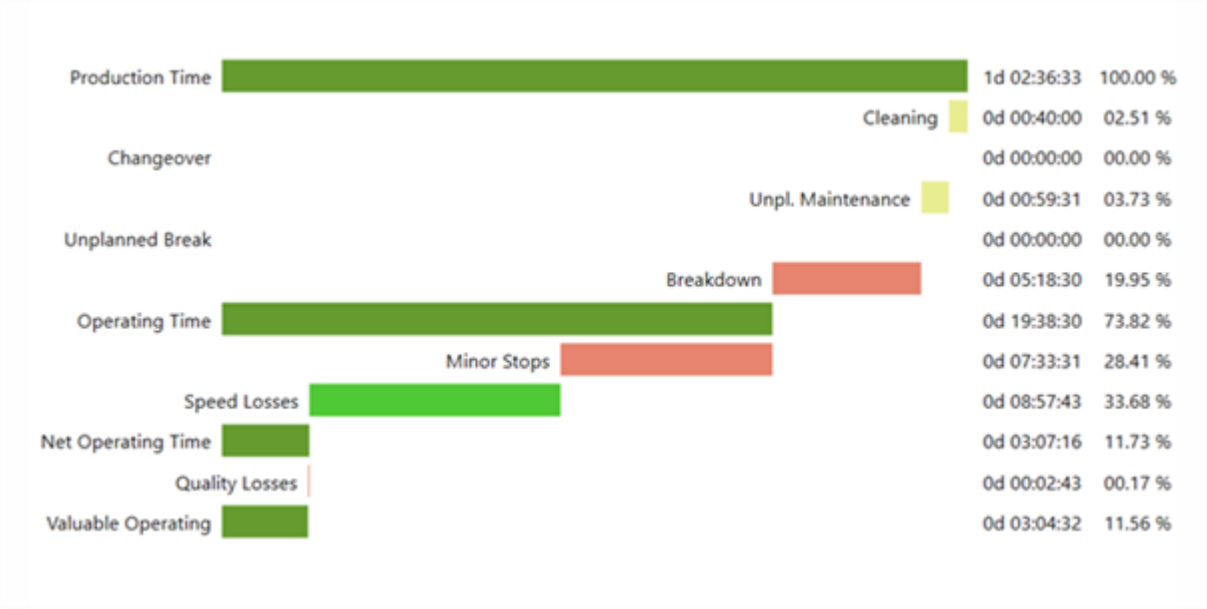
Property	Funzione	Valore
IndicatorBrush	Colore dell´indicatore.	*Brush*
IndicatorBackgroundBrush	Colore di sfondo dell´indicatore non attivo.	*Brush*
KnobSize	Diametro della manopola	*Double*
KnobIndicatorSize	Dimensioni dell'indicatore del pulsante.	*Size*
ElementSize	Grandezza dell´elemento.	*Size*
VisibilityKnob	Attivazione del pulsante.	*Boolean*
ValuePosition	Posizione della visualizzazione valore.	*Double*
ValueVisibility	Attivazione della visualizzazione valore.	*Boolean*

6.5.11 Diagramma a cascata

Diagramma a cascata L'elemento WPF è disponibile per i partner di COPA-DATA ed è a loro disposizione tramite la COPA-DATA Partner Community (https://www.copadata.com/en-us/partner-community/).

Per modellare un diagramma a cascata è necessario utilizzare il Meaning and Waterfall Chart Wizard. Con questa procedura guidata è possibile modellare un diagramma a cascata. Le informazioni vengono salvate direttamente nelle variabili interessate nella proprietà **Parametro per diagramma a cascata** (gruppo **Analyzer** delle proprietà delle variabili).

Di seguito è riportato un esempio di un diagramma a cascata a Runtime:



Nota: questo tool è a disposizione solamente in lingua inglese.

Le seguenti impostazioni possono essere eseguite nella finestra di configurazione WPF sotto **COPADATA-ELEMENT**:

Property	Funzione	Valore
zenonRefreshRate	Tempo tra gli aggiornamenti del grafico in millisecondi.	*Integer*
zenonWaterfallIdentifier	Nome del grafico a cascata.	*String*
zenonZSystemModel	Gruppo di impianti delle variabili usate.	*String*

Nota: per visualizzare il diagramma a cascata in zenon Web Client è necessaria un'ulteriore programmazione VSTA. Per i dettagli, vedere Visualizzazione di elementi WPF in zenon Web Client (A pagina: 112).

COLLEGARE LE BARRE AD UNA FUNZIONE DI ZENON

Le barre di un diagramma a cascata possono essere collegate a una funzione a Runtime. Per la visualizzazione a Runtime, per il diagramma a cascata possono essere configurati con un clic del mouse le barre, l'etichettatura e la visualizzazione dei valori per l'esecuzione di una funzione.

Eseguire la seguente configurazione per collegare le barre del diagramma a cascata a una funzione:

1. Configurare l´elemento WPF per il diagramma a cascata.
   Nota: si consiglia di utilizzare il Meaning and Waterfall Chart Wizard.
2. Creare una funzione di zenon:
   a) Creare una nuova funzione:

      Nella barra degli strumenti, o nel menù contestuale del nodo "Funzioni", selezionare **Nuova funzione**.
      Si apre la finestra di dialogo per la selezione di una funzione.

   b) Selezionare la funzione desiderata.

   c) Configurare i parametri della funzione.
3. Assegnare un nome alla funzione nella proprietà **Nome**.
   Tenere presente quanto segue: il nome della funzione deve contenere i parametri della variabile per il diagramma a cascata senza codice colore!
   Questi parametri si trovano anche nella proprietà della variabile **Parametro per diagramma a cascata** del gruppo **Analyzer**.
4. Collegare la funzione con lo stesso gruppo di impianti cui sono collegate le variabili.
   Nota: questo collegamento si trova nella proprietà **Gruppi d'impianto** della funzione.

Per questa progettazione valgono le seguenti regole:

- La funzione e le variabili collegate devono essere nello stesso progetto di zenon.
- Le variabili devono essere collegate ad un gruppo di impianti.
- La funzione e le variabili devono essere collegate allo stesso gruppo di impianti.

> **Esempio**
>
> Per una barra con la definizione di cascata *WF= WF1,02,05,#E9ED92;* per esempio, deve essere utilizzato il nome funzione *Function_WF1,02,05*.

6.5.12 BACnet WPF Control

Il **BACnet Schedule Control** L'elemento WPF è disponibile per i partner di COPA-DATA ed è a loro disposizione tramite la COPA-DATA Partner Community (https://www.copadata.com/en-us/partner-community/).

Il Control offre la possibilità di configurare gli oggetti BACnet Schedule in un'interfaccia grafica utente e di visualizzarli.

6.5.12.1 Installazione

Requisiti:

- Si deve essere un partner registrato di COPA-DATA
- Si ha bisogno di un accesso già esistente alla COPA-DATA Partner Community.

SCARICARE UN ELEMENTO WPF DAL PARTNER PORTAL DI COPA-DATA

Seguire questi passaggi per scaricare l'elemento WPF:

1. Aprire COPA-DATA Partner Community (https://www.copadata.com/en-us/partner-community/).
2. Accedere all'area **MY AREA**.
3. Immettere "Partner" come termine di ricerca nella lente d'ingrandimento.

   COPA-DATA Partner Community viene visualizzata insieme a un link nell´area dei risultati.
4. Fare clic sul link.
5. Alla voce **Partner Login Area**, cliccare su **MORE**.

   Viene aperta la scheda **Overview**.
6. Passare alla scheda **Documents & Downloads**.
7. Alla voce **SELECT CATEGORY**, attivare la checkbox **Elemento WPF**.
8. Cliccare su **Search**.

   La didascalia scompare.

   Una cornice arancione delimita il campo di ricerca.
9. Fare clic nella cornice e immettere **Bacnet**.
10. **BACNetWPFScheduler** viene visualizzato.
11. Fare clic sull'icona arancione in **Actions** per scaricare **BACnet Schedule Control**.
12. Salvare la cartella localmente.
13. Decomprimere il file.
14. Aprire la cartella contenente i file *BacnetSchedulerControl.xaml* e *BacnetSchedulerWPF.dll*.

IMPORTAZIONE DI BACNET SCHEDULE CONTROL IN ZENON EDITOR

Per creare un elemento WPF:

1. Aprire l´Editor di zenon.
2. Nel Manager di progetto, passare al nodo **File**.

3. Espandere la vista di questo nodo facendo clic sul pulsante *[+]*.
   La visualizzazione ad albero viene espansa e vengono visualizzati i sottonodi.
4. Passare al sottonodo **Grafica**.
   a) Scegliere **Importa file** dalla barra degli strumenti della visualizzazione dettagli o dal menù contestuale.
      Si apre la finestra di dialogo per la selezione file.
   b) Selezionare il file *BacnetSchedulerControl.xaml*.
   c) Confermare la selezione cliccando sul pulsante **Apri**.
5. Passare al sottonodo **Altri**.
   a) Scegliere **Importa file** dalla barra degli strumenti della visualizzazione dettagli o dal menù contestuale.
      Si apre la finestra di dialogo per la selezione file.
   b) Selezionare il file *BacnetSchedulerWPF.dll*.
   c) Confermare la selezione cliccando sul pulsante **Apri**.

6.5.12.2 Configurazione nell´Editor di zenon

Per utilizzare **BacnetScheduler** in zenon, eseguire i seguenti passaggi:

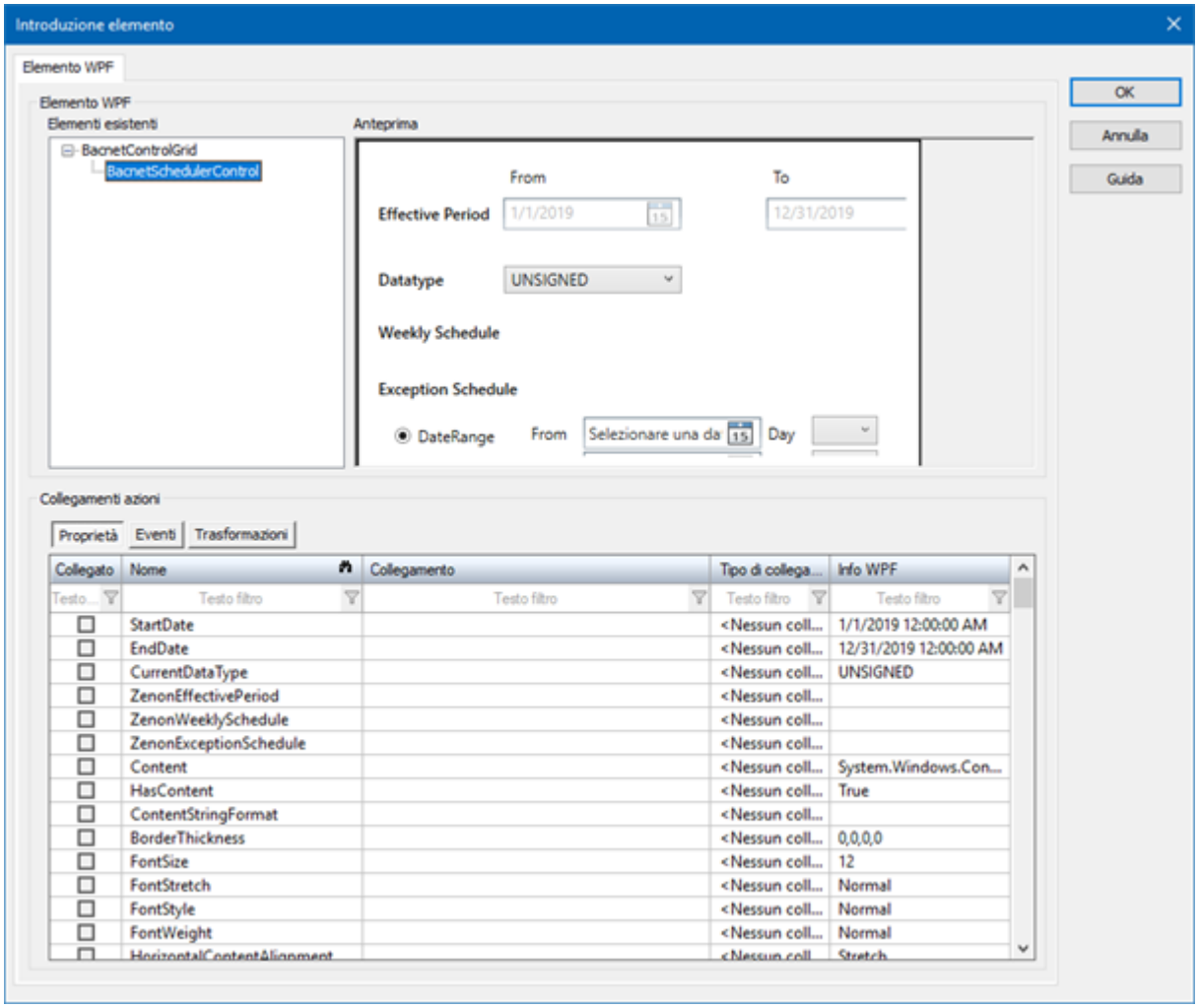
1. Creare una nuova immagine.

   Nella barra degli strumenti o nel menù contestuale del nodo **Immagini**, selezionare il comando **Nuova immagine**.
2. Modificare le proprietà dell´immagine:
   a) Assegnare un nome all´immagine nella proprietà **Name**.
   b) Nella proprietà **Bildtyp**, selezionare un qualsiasi tipo di immagine (tipo di immagine consigliato: *Standard*).
   c) Nella proprietà **Schablone** selezionare il modello desiderato.
3. Configurare i contenuti dell´immagine:
   a) Aprire l'immagine in cui deve essere inserito l'elemento WPF.
   b) Nella barra degli strumenti **Elementi**, selezionare “Elemento WPF” (**WPF**).
   c) Disegnare l'elemento WPF nell'immagine con il tasto sinistro del mouse.
   d) Si apre la finestra **Scegli file**.
   e) Selezionare il file **BacnetSchedulercontrol.xaml** e confermare l'immissione cliccando su **OK**.

f) L'elemento WPF è inserito.

4. Regolare la dimensione dell'elemento WPF nell'immagine zenon.
   a) Selezionare l´elemento WPF.
   b) Passare al gruppo delle proprietà **Position**.
   c) Parametrizzare la larghezza dell'elemento con la proprietà **Breite [Pixel]** e l'altezza con la proprietà **Höhe [Pixel]**.
   Dimensione minima consigliata: 920 x 920 Pixel

6.5.12.3 Proprietà configurabili del controllo



Eseguire le seguenti operazioni per parametrizzare il BACnet WPF Control:

1. Selezionare l´elemento WPF.
2. Passare al gruppo delle proprietà **WPF-Verknüpfungen**.
3. Fare clic sul pulsante **...** accanto alla proprietà **Konfiguration**.

4. Si apre la finestra di dialogo “Introduzione elemento”.
5. Espandere il nodo **BacnetControlGrid** nell'area **Elementi esistenti**.
   La voce **BacnetSchedulerControl** diventa visibile nel nodo.
6. Fare clic sulla voce **BacnetSchedulerControl**.
   La vista nell'area **Collegamenti azioni** viene aggiornata.
7. Parametrizzare il contenuto del WPF:
   a) Fare clic una volta nella rispettiva riga della colonna **Tipo di collegamento**.
      Viene visualizzato un menù a tendina.
   b) Selezionare il tipo di collegamento appropriato dal menù a tendina.
   c) Cliccando nella colonna **Collegamento** si apre la finestra di dialogo **Introduzione elemento**.
      Quest´ultima è adattata al tipo di collegamento configurato (ad es. selezione di una variabile, ingresso di un valore, ...).
8. Confermare la parametrizzazione cliccando su **OK**

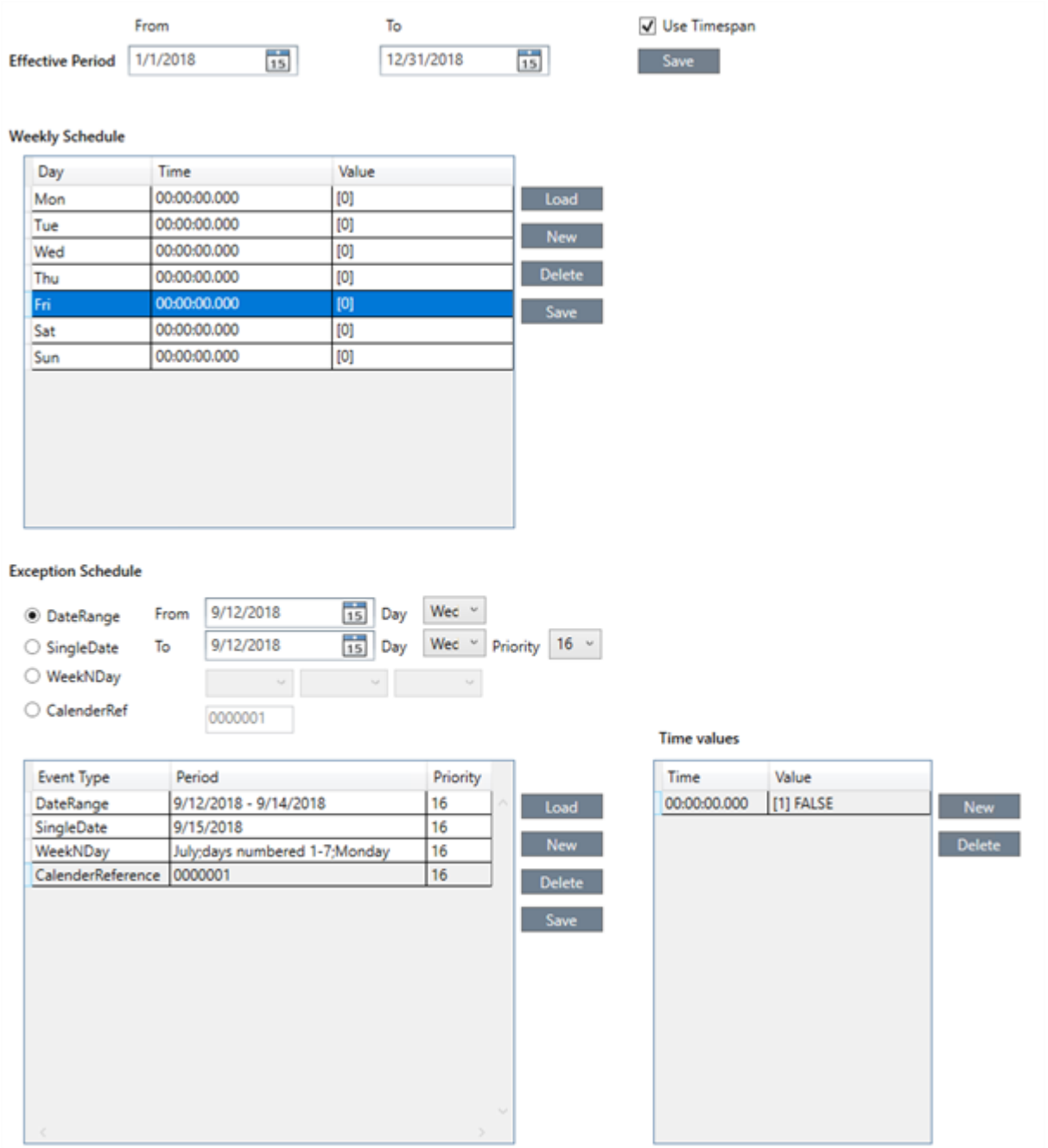
TIPI DI COLLEGAMENTO POSSIBILI

Per questo WPF è possibile collegare uno o più parametri:

- *StartDate*
- *EndDate*
- *ZenonEffectivePeriod*
- *ZenonWeeklySchedule*
- *ZenonExceptionSchedule*

> **Info**
>
> La parametrizzazione dell'elemento WPF viene visualizzata in zenon nel gruppo di proprietà **WPF-Verknüpfungen**dell'elemento WPF. Ogni collegamento WPF viene visualizzato con il proprio sottonodo o panoramica proprietà.

6.5.12.4 Visuale Runtime



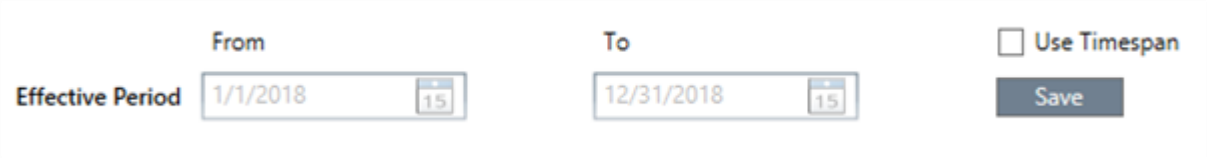
Nel Runtime di zenon, gli oggetti BACnet Schedule esistenti possono essere letti dal PLC, modificati a Runtime e salvati di nuovo nel PLC.

L'interfaccia utente a Runtime è divisa in tre aree:

- **Effective Period**
- **Weekly Schedule**
- **Exception Schedule**

Nota: l'interfaccia grafica utente è disponibile solo in inglese.

6.5.12.4.1 Visuale Runtime

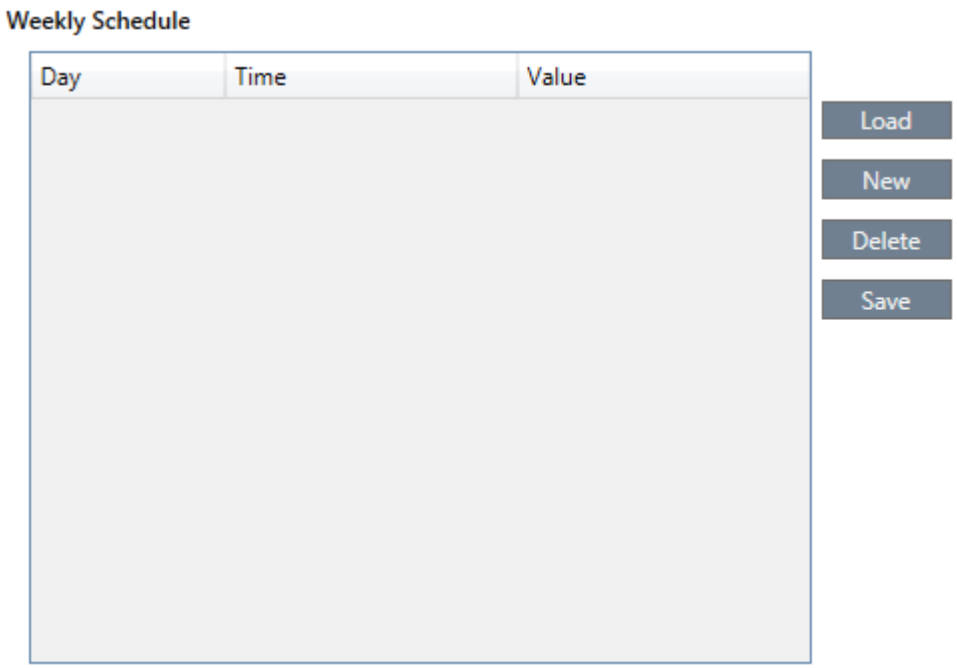


Quest'area visualizza il valore della variabile collegata all'opzione **ZenonEffectivePeriod** quando il Control è stato configurato (A pagina: 99) nell´Editor.

Parametro	Descrizione
Use Timespan	Effective Period è parametrizzato a Runtime. L'ora viene immessa nei campi di immissione **From** e **To**. ▸ *Non attivo*: Effective Period non viene utilizzato. ▸ *Attivo*: Effective Period viene inserito a Runtime. Default: *non attiva*.
From	Inizio di Effective Period. Inserimento della data. Inserimento tramite input manuale o selezione da calendario. Il calendario viene visualizzato come una finestra di dialogo a discesa visualizzata cliccando sull'icona del calendario. Default: *1. 1.2018* **Nota:** attivo solo se l'opzione **Use Timespan** è attiva.
To	Fine di Effective Period. Inserimento della data. Inserimento tramite input manuale o selezione da calendario. Il calendario viene visualizzato come una finestra di dialogo a discesa visualizzata cliccando sull'icona del calendario. Default: 31*/12./2018* **Nota:** attivo solo se l'opzione **Use Timespan** è attiva.
Save	Scrive nel PLC la configurazione corrente di

Parametro	Descrizione
	Effective Period.

6.5.12.4.2 Visuale Runtime



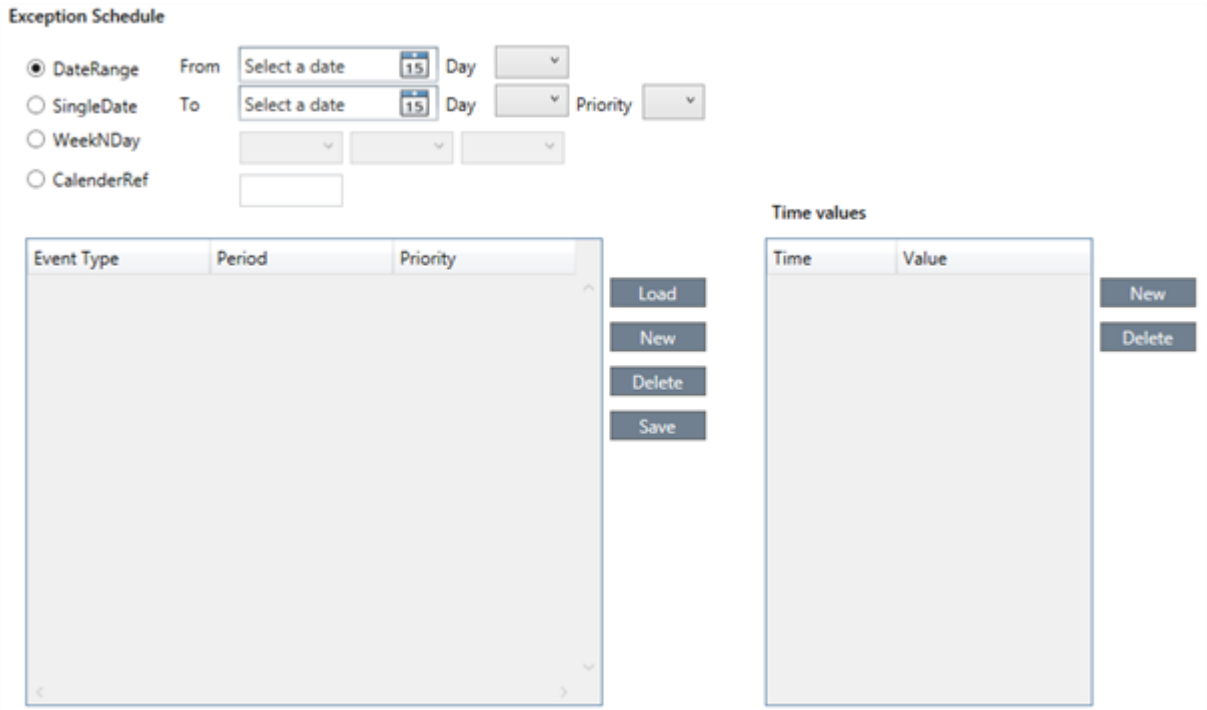
Quest'area visualizza il valore della variabile collegata all'opzione **ZenonWeeklySchedule** quando il Control è stato configurato (A pagina: 99) nell´Editor.

Parametro	Descrizione
[Elenco dei contenuti dell´Weekly Schedules caricata]	L´orario viene visualizzato come voce dell'elenco nella lista **Weekly Schedule**. La parametrizzazione avviene tramite la rispettiva voce della lista: ▸ *Day* Selezionare il giorno dall'elenco a discesa. ▸ *Time* Inserimento dell´ora Formato di input: *hh:mm:ss:ms* ▸ *Value* Valore della rispettiva pianificazione. Il valore rappresenta quello del tipo di dati:: *[0] = NULL* *[1] = BOOLEAN*

Parametro	Descrizione
	[2] = Unsigned *[3] = REAL* *[5] = Double* *[6] = OctetString* *[7] = CharacterString* *[8] = BitString* *[9] = Enumerated* *[10] = Date* *[11] = Time* *[12] = BACnetObjectIdentifier* Default: *[0]* **Nota:** il tipo di dati deve essere inserito nel parametro *Values* durante la configurazione. Il tipo di dati è specificato come numero secondo l'elenco di cui sopra. Si prega di notare che questo input deve essere eseguito inserendo il numero tra parentesi quadre []. Esempio: *[1]* per *BOOLEAN*. Oltre al tipo di dati, è possibile configurare anche il valore corrispondente del tipo di dati. Questo valore viene inserito in base al tipo di dati. Esempi: *[1] TRUE* Per booleano con valore TRUE. *[10] 2/3/2021* Per il tipo di dati "Data" con valore 3 febbraio 2021 Inserimento data in formato BACnet standard.
Load	Legge la configurazione corrente per **Weekly Schedule** dal PLC. La *Weekly Schedule* caricata viene visualizzata nell'elenco **Weekly Schedule**. **Nota:** quando si esegue un **Load**, tutte le configurazioni non salvate a Runtime sono sostituite dai valori del PLC senza richiesta di conferma.
New	Crea una nuova *Weekly Schedule*.

Parametro	Descrizione
Delete	Cancella la voce selezionata dall'elenco **Weekly Schedule**. La voce viene cancellata immediatamente senza una precedente richiesta di conferma. La selezione multipla è possibile, ma viene ignorata quando si clicca su **Delete**.
Save	Scrive nel PLC la configurazione corrente di **Weekly Schedules**.

6.5.12.4.3 Visuale Runtime



Quest'area visualizza il valore della variabile collegata all'opzione **ZenonExceptionSchedule** quando il Control è stato configurato (A pagina: 99) nell´Editor.

Parametro	Descrizione
Tipo di Exception Schedule	Tipo di pianificazione per eccezioni (Exception Schedule). Selezione dal box delle opzioni: ▸ *DateRange*

Parametro	Descrizione
	L´Exception Schedule è definita con un intervallo di tempo. L'intervallo di tempo è configurato con le opzioni **From**, **Day**, **To** e **Day** . ▸ *SingleDate* L´Exception Schedule è definita con un giorno esatto. Il giorno può essere configurato con le opzioni **Single** e **Day**. ▸ *WeekNDay* L´Exception Schedule è definita con più giorni. Questi giorni sono configurati nei menù a tendina per **WeekNDay** . ▸ *CalenderRef* L´Exception Schedule è definita con riferimento ad un oggetto di calendario BACNet esistente.
From (Per *Eventtype DateRange*) **Single** (Per *Eventtype SingleDate*)	Inizio dell'Eventtype *Exception Schedule*. Inserimento della data. Inserimento manuale o selezione da calendario. Il calendario viene visualizzato come una finestra di dialogo a discesa facendo clic sull'icona del calendario. Default: *vuoto (Select a date)* **Nota:** attivo solo se sono attive le opzioni **DateRange** o **SingleDate**.
Day (Nella riga **From**)	Giorno della settimana dell'inizio dell´Eventtype *Exception Schedule*. Se nell'opzione **From** è configurata una data, il giorno della settimana corrispondente viene impostato automaticamente per questa opzione. Selezione del giorno della settimana dal menù a tendina: ▸ *Mon* Lunedì ▸ *Tue*

Parametro	Descrizione
	Martedì - *Wed* Mercoledì - *Thu* Giovedì - *Fri* Venerdì - *Sat* Sabato - *Sun* Domenica Default: *vuoto* **Nota:** attivo solo se sono attive le opzioni **DateRange** o **SingleDate**.
To	Fine dell´Eventtype *Exception Schedule*. Inserimento della data. Inserimento tramite input manuale o selezione da calendario. Il calendario viene visualizzato come una finestra di dialogo a discesa visualizzata cliccando sull'icona del calendario. Default: *vuoto(Select a date)* **Nota:** attivo solo se l'opzione **DataRange** è attiva.
Day (Nella riga **To**)	Giorno della settimana della fine dell´Eventtype *Exception Schedule*. Se nell'opzione **To** è configurata una data, il giorno della settimana corrispondente viene impostato automaticamente per questa opzione. Selezione del giorno della settimana dal menù a tendina: - *Mon* Lunedì - *Tue* Martedì - *Wed*

Parametro	Descrizione
	Mercoledì - *Thu* Giovedì - *Fri* Venerdì - *Sat* Sabato - *Sun* Domenica **Nota:** attivo solo se l'opzione **DataRange** è attiva.
Priority	Priorità per l´Eventtype secondo lo standard BACnet. Selezione dal menù a tendina: Valori *1 - 16*
WeekNDay	Selezione della data per l´Eventtype *Exception Schedule* da menù a tendina. - Menù a tendina 1: Selezione del mese **Nota:** la voce *any month of year* corrisponde ad ogni mese. - Menù a tendina 2: Selezione del giorno in base a determinati criteri. Menù a tendina 3: Selezione del giorno della settimana **Nota:** la voce *any day of week* corrisponde ad ogni giorno di una settimana. **Nota:** attivo solo se l'opzione **WeekNDay** è attiva.
CalenderRef	Riferimento ad un oggetto BACnet esistente di tipo **CalendarObject**. Inserire l'*Object Instance* numerica dell'oggetto BACnet. Secondo lo standard BACnet, questo è sempre un numero bit di sette cifre. Campo di ingresso valido (secondo lo standard BACnet): *0 - 4194302*

Parametro	Descrizione
	Nota: A questo scopo, il numero *Object Instance* qui inserito nel controllo deve essere integrato inserendo 0 davanti. **Esempio:** **CalendarObject** *Instance Nummer = 1* input nel controllo = *0000001*
[Elenco dei contenuti dell´Exception Schedule caricata]	▸ L´orario viene visualizzato come voce dell'elenco nella lista **Expeption Schedule**. I parametri vengono impostati utilizzando le opzioni sopra elencate: ▸ *Event Type:* Corrisponde alla selezione dell´Event Type dal riquadro delle opzioni. ▸ *Period:* Visualizzazione della data. A seconda dell´Event Type selezionato, la visualizzazione può variare: [Date] - [Date] ▸ *Priority:* Priorità configurata
Load	Legge nel PLC la configurazione corrente di **Exception Schedules**. L´*Exception Schedule* caricata viene visualizzata nell'elenco. **Nota:** quando si esegue un **Load**, tutte le configurazioni non salvate a Runtime sono sostituite dai valori del PLC senza richiesta di conferma.
New	Acquisisce la configurazione di progetto dall'area **Exception Schedule** e crea una nuova voce nell'elenco.
Delete	Cancella la voce selezionata dall'elenco **Exception Schedule**. La voce viene cancellata immediatamente senza una precedente richiesta di conferma.

Parametro	Descrizione
	La selezione multipla è possibile, ma viene ignorata quando si clicca su **Delete**.
Save	Scrive nel PLC la configurazione corrente di **Exception Schedules**.

TIME VALUES

Valori temporali per un Exception Schedule Event configurato.

Parametro	Descrizione
[Elenco dei Time values configurati per l'Exception Schedule Event selezionato]	▸ *Time* Inserimento dell´ora Formato di input: *hh:mm:ss:ms* ▸ *Value* Valore della rispettiva pianificazione. Il valore rappresenta il valore del tipo di dati: *[0] = NULL* *[1] = BOOLEAN* *[2] = Unsigned* *[3] = REAL* *[5] = Double* *[6] = OctetString* *[7] = CharacterString* *[8] = BitString* *[9] = Enumerated* *[10] = Date* *[11] = Time* *[12] = BACnetObjectIdentifier* Default: *[0]* **Nota:** il tipo di dati deve essere inserito nel parametro *Values* durante la configurazione. Il tipo di dati è specificato come numero secondo l'elenco di cui sopra. Si prega di notare che questo input deve essere eseguito inserendo il numero tra parentesi quadre []. Esempio: *[1]* per *BOOLEAN*. Oltre al tipo di dati, è possibile configurare anche il valore corrispondente del tipo di dati. Questo

Parametro	Descrizione
	valore viene inserito in base al tipo di dati. Esempi: *[1] TRUE* Per booleano con valore TRUE. *[10] 2/3/2021* Per il tipo di dati "Data" con valore 3 febbraio 2021 Inserimento data in formato BACnet standard.
New	Crea un nuovo Time value per l´Exception Schedule Event selezionato.
Delete	Cancella la voce Time value selezionata. La voce viene cancellata immediatamente senza una precedente richiesta di conferma. La selezione multipla è possibile, ma viene ignorata quando si clicca su **Delete**.

6.5.12.5 Gestione nel Runtime di zenon

Possibilità del **BACnet Schedule Control** WPF nel Runtime di zenon:

- Periodo di validità della Schedule (*Effective Period*)
  - Periodo di validità configurabile in giorni
  - Lettura dell´*Effective Period* esistente dal PLC.
  - Salvataggio dell´*Effective Period* modificato sul PLC.
- Configurazione di Schedules settimanali:
  - Lettura delle *Weekly Schedules* esistenti dal PLC.
  - Proiezione di nuove voci per *Weekly Schedule*
  - Cancellazione di voci esistenti
  - Salvataggio delle *Weekly Schedules* modificate sul PLC
  - Possibilità di configurazione per ogni voce:

    - Giorno (Day of the Week)

    - Tempo per giorno (locale e UTC, a seconda del PLC)

    - Tipo di dati e valore
- Configurazione di eccezioni (*Exception Schedule*)

- Lettura dell'Exception Schedule esistente dal PLC.
- Configurazione di nuove voci per l´Exception Schedule.
- Cancellazione di voci esistenti.
- Salvataggio sul PLC dell´*Exception Schedule* modificata.
- Possibilità di configurazione per ogni voce:

  - Event Type (DateRange, SingleDate, WeekNDay, CalenderRef)

  - Period

  - Priority

  - Tempo

  - Tipo di dati e valore

6.6 Visualizzazione di elementi WPF in zenon Web Client

Per poter usare gli elementi WPF predefiniti **"Diagramma Energy class"**, **"Diagramma di Sankey**" e **"Diagramma a cascata"** anche per la visualizzazione in uno zenon Web Client, è necessario effettuare adeguamenti e modifiche nel progetto:

- Configurazione nell´Editor di zenon (A pagina: 112)
- Adeguamento del codice VSTA (A pagina: 113)

6.6.1 Configurazione nell´Editor di zenon:

Eseguire i seguenti passi di configurazione nell'Editor di zenon per poter visualizzare determinati elementi WPF in zenon Web Client:

POSIZIONARE L´ELEMENTO WPF NELL´IMMAGINE DI ZENON:

- Posizionare l´elemento WPF in un´immagine di zenon.
- Assegnare un nome univoco nella proprietà **Elementname**.
  Questa proprietà si trova nel gruppo **Allgemein**.
  Nota: se il nome di un elemento è già stato assegnato nell'immagine, viene visualizzata una finestra di avviso.
- Usare il nome dell'elemento assegnato qui nel codice VSTA.

6.6.2 Codice VSTA (complesso)

Per inserire il codice di programmazione per la visualizzazione di elementi WPF in zenon Web Client, effettuare le seguenti operazioni:

1. Nell´Editor di zenon, passare al nodo **Scripting**.
2. Selezionare il nodo **VSTA** e fare clic con il tasto destro del mouse per aprire la voce del menù contestuale **Apri Editor VSTA con ProjectAddin ...**
3. Si apre la finestra di dialogo per la creazione di un progetto VSTA.
4. Nella finestra di dialogo **Nuovo Progetto VSTA**, selezionare la voce C#.
5. Creare (copiare) il codice qui sotto.
6. Inserire il nome dell'elemento WPF nel codice.

Nota: all'apertura dell'Editor VSTA verificare se il contenuto del seguente codice è già presente nella configurazione. Per visualizzare l'elemento WPF nel Web Client, confrontare il codice esistente ed effettuare le aggiunte necessarie. Si prega di notare i commenti nel codice di esempio.

CODICE VSTA

```
//Als Member:
zenOn.IDynPictures zScreens = null;
string[] WPFElements ={"WPF_Control", "WPFWebclient_1", "WPFWebclient_2" }; //Nomi degli elementi ad immagine WPF che compaiono nel progetto zenon e necessitano accesso all´API (tanti/ tanto pochi si vuole)

//Aggiungere le seguenti tre righe di codice nella funzione project archive:
void ThisProject_Active()
{
zScreens = this.DynPictures();
zScreens.Open += new zenOn.DDynPicturesEvents_OpenEventHandler(zScreens_Open);
zScreens.Close += new zenOn.DDynPicturesEvents_CloseEventHandler(zScreens_Close);
}
//Aggiungere le seguenti due righe di codice nella funzione project inactive:
void ThisProject_Inactive()
{
zScreens.Open -= new zenOn.DDynPicturesEvents_OpenEventHandler(zScreens_Open);
zScreens.Close -= new zenOn.DDynPicturesEvents_CloseEventHandler(zScreens_Close);

API
FreeObjects();
}
```

```
//Aggiungere due nuovi Event-Handler:
void zScreens_Open(zenOn.IDynPicture obDynPicture)
{
foreach (string element in WPFElements)
{
if (obDynPicture.Elements().Item(element) != null)
{
obDynPicture.Elements().Item(element).set_WPFProperty("ELEMENT", "zenonVariableLink",
this.Variables().Item(0));
}
}
}
void zScreens_Close(zenOn.IDynPicture obDynPicture)
{
foreach (string element in WPFElements)
{
if (obDynPicture.Elements().Item(element) != null)
{
zenOn.IElement zWPFElement= obDynPicture.Elements().Item(element);
zWPFElement.set_WPFProperty("ELEMENT", "zenonTrigger", true);
zWPFElement = null;
}
}
}
```

6.6.3 Codice VSTA (semplificato)

Se si usa solamente un elemento WPF in un´immagine di zenon, in alternativa è possibile utilizzare il seguente codice più semplice. Per fare ciò, è necessario immettere i nomi dell'elemento WPF e dell´immagine in cui esso viene usato. Questo codice è consigliato se per ogni progetto viene utilizzato uno solo degli elementi WPF predefiniti.

CODICE VSTA

```
zenOn.IDynPicture zScreen = null;
string wpfElement = "WPF_Control"; //Nome dell´elemento WPF nell´immagine.
string wpfPicture = "@Details_Overview_Online"; //Nome dell´immagine di zenon

//Aggiungere nella funzione project active:
```

```
void ThisProject_Active()
{
zScreen = this.DynPictures().Item(wpfPicture);
zScreen.Open += new zenOn.OpenEventHandler(zScreen_Open);
zScreen.Close += new zenOn.CloseEventHandler(zScreen_Close);
}

//Aggiungere nella funzione project inactive:
void ThisProject_Inactive()
{
zScreen.Open -= new zenOn.OpenEventHandler(zScreen_Open);
zScreen.Close -= new zenOn.CloseEventHandler(zScreen_Close);

API
FreeObjects();
}

void zScreen_Open()
{
if (zScreen.Elements().Item(wpfElement) != null)
{
zScreen.Elements().Item(wpfElement).set_WPFProperty("ELEMENT", "zenonVariableLink",
this.Variables().Item(0));
}
}

void zScreen_Close()
{
if (zScreen.Elements().Item(wpfElement) != null)
{
zenOn.IElement zWPFElement = zScreen.Elements().Item(wpfElement);
zWPFElement.set_WPFProperty("ELEMENT", "zenonTrigger", true);
zWPFElement = null;
}
}
```

6.7 Esempi: integrazione di WPF in zenon

Gli esempi che seguono mostrano come i file XAML vengono creati e inclusi in zenon come elementi WPF:

- Implementare un pulsante come WPF-XAML in zenon. (A pagina: 120)
- Implementare un bargraf come WPF-XAML in zenon. (A pagina: 116)
- Implementare DataGrid Control in zenon (A pagina: 126)

6.7.1 Implementare un bargraf come WPF-XAML in zenon.

Struttura dell´esempio:

- Creazione di un bargraf (A pagina: 16) in Adobe Illustrator e conversione in WPF
- Integrazione in zenon
- Collegamento con variabile
- Adattamento dell´elemento bargraf-WPF

CREAZIONE DI UN BARGRAF

Creare in primo luogo un bargraf come descritto nel capitolo Workflow con Adobe Illustrator (A pagina: 16). Per utilizzare il file XAML in zenon, aggiungerlo nella cartella **File/Grafica** dell'albero del progetto .

INTEGRARE IL BARGRAF

Nota: la seguente descrizione utilizza un progetto di zenon con i seguenti contenuti:

- Un´immagine vuota come immagine di partenza
- 4 variabili del driver interno per
  - Scala *0*
  - Scala *media*
  - Scala *alta*
  - Valore attuale
- Una variabile del driver matematico per la visualizzazione del valore corrente *255*)

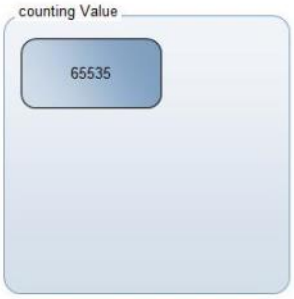
Per integrare il bargraf:

1. Anzitutto aprire l´immagine vuota

2. Posizionare un **elemento WPF** (A pagina: 50) nell´immagine
3. Selezionare **XAML-Datei** nella finestra delle proprietà
4. Selezionare il file XAML desiderato (per esempio **bargraph_vertical.xaml**) e chiudere la finestra di dialogo.

ADATTARE IL BARGRAF

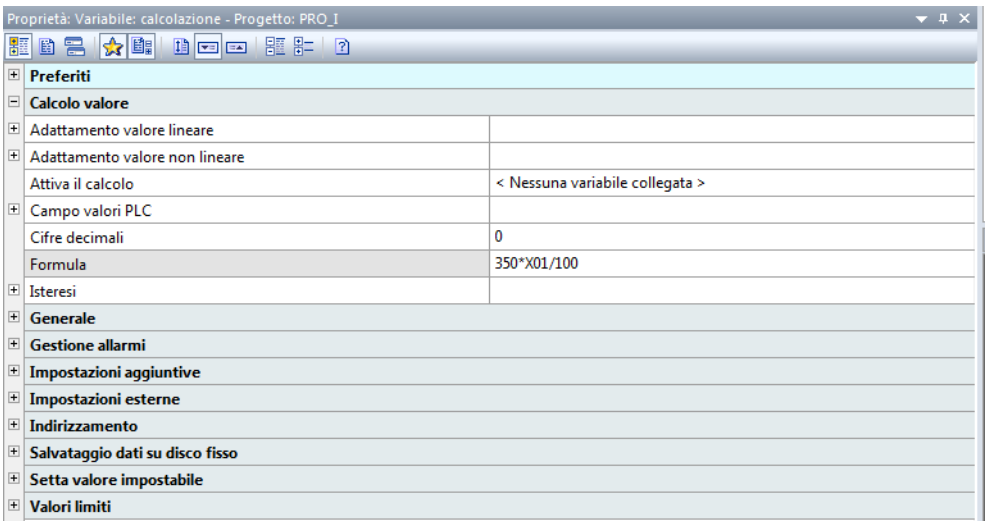
Prima della configurazione, se necessario, viene regolata la scala del file XAML:



Procedura:

- Creare una nuova variabile matematica che calcola il nuovo valore in relazione al fattore di scala, ad esempio:
- Variabile: 0-1000

- Variabile matematica {valore creato nel file Xaml}*Variabile/1000

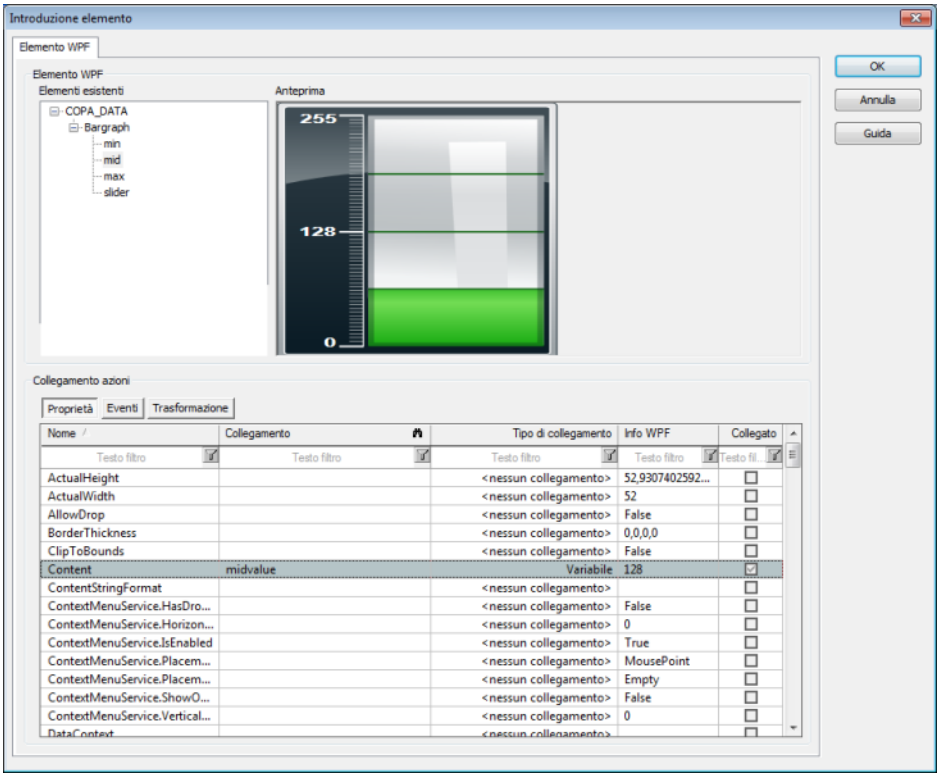


Quindi viene configurato il file XAML.

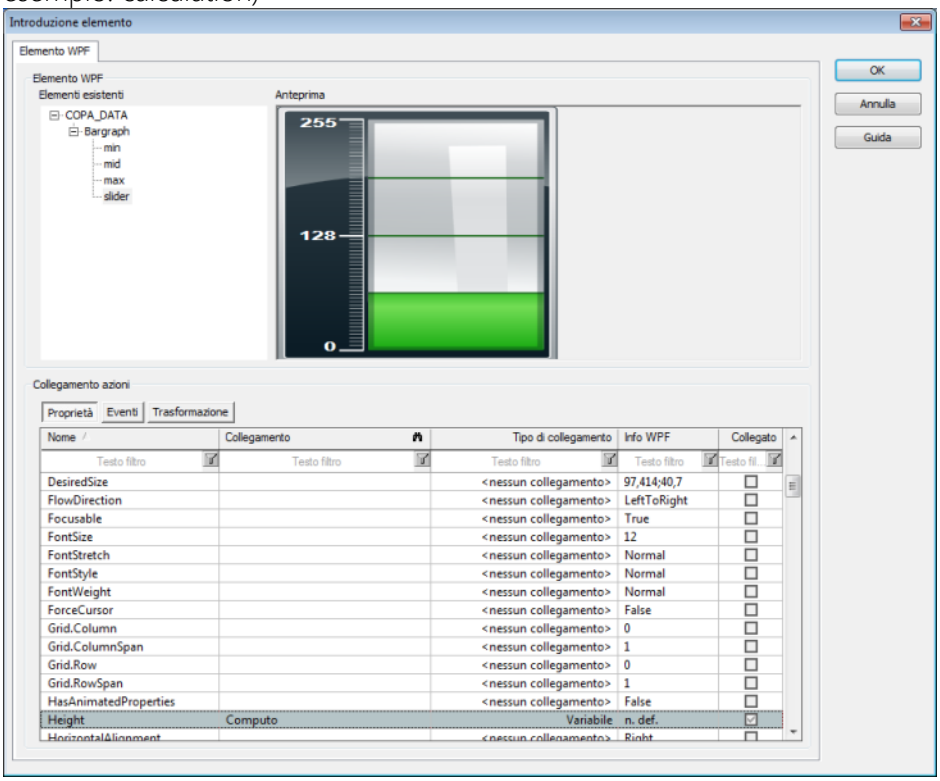
CONFIGURARE IL BARGRAF

1. Cliccare sull'elemento WPF e selezionare la proprietà **Konfiguration** (gruppo "Collegamenti WPF").
2. La finestra di configurazione mostra un'anteprima del file XAML selezionato.

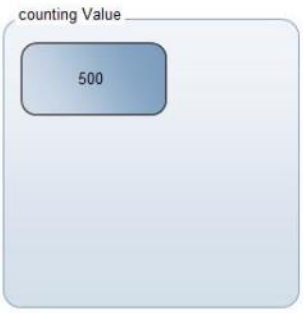
3. Selezionare il valore minimo, quello medio e quello massimo e collegare la variabile corrispondente nella proprietà **Content**.



4. Selezionare lo **Slider** e collegare la proprietà **Value** alla variabile matematica (nel nostro esempio: calculation)



5. Verificare la configurazione a Runtime:



6.7.2 Implementare un pulsante come WPF-XAML in zenon.

Struttura dell´esempio:

- Creazione di un pulsante (A pagina: 12) in Microsoft Expression Blend
- Integrazione in zenon
- Collegamento a una variabile e una funzione.
- Adattamento del pulsante alla dimensione dell'elemento.
- Creare un pulsante

Creare in primo luogo un pulsante come descritto nel capitolo Creare un pulsante come file XAML con Microsoft Expression Blend (A pagina: 12). Per utilizzare il file XAML in zenon, aggiungerlo nella cartella **File/Grafica** dell'albero del progetto .

INTEGRARE IL PULSANTE

Nota: la seguente descrizione utilizza un progetto di zenon con i seguenti contenuti:

- Un´immagine vuota come immagine di partenza
- Una variabile interna **int** di tipo **Int**

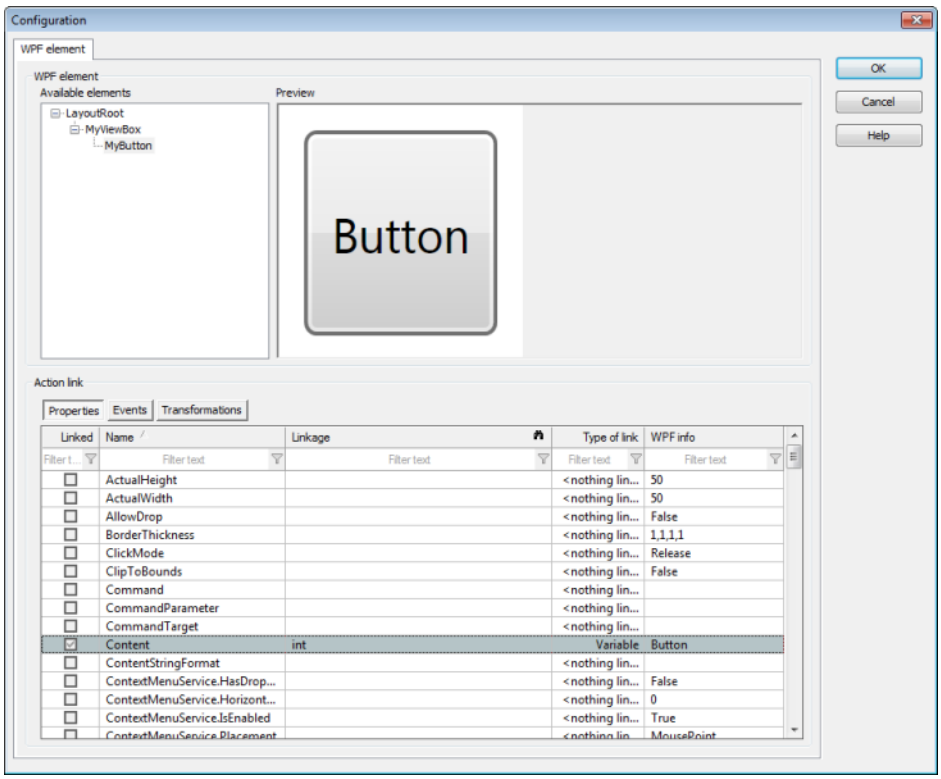
- Una funzione **Funzione_0** di tipo **Setta/modifica valore impostabile** con:
  - Opzione **Direttamente su Hardware** attiva
  - Il valore impostabile è stato configurato con il valore *45*.

Per integrare il pulsante:

1. Anzitutto aprire l´immagine vuota
2. Posizionare un **elemento WPF** (A pagina: 50) nell´immagine
3. Selezionare **XAML-Datei** nella finestra delle proprietà
4. Selezionare il file XAML (per es.: **MyButton.xaml)** e chiudere la finestra di dialogo.
5. Selezionare la proprietà **Konfiguration**

CONFIGURARE IL PULSANTE

La finestra di configurazione mostra un'anteprima del file XAML selezionato. L'albero elenca tutti gli elementi nominati nel file XAML:



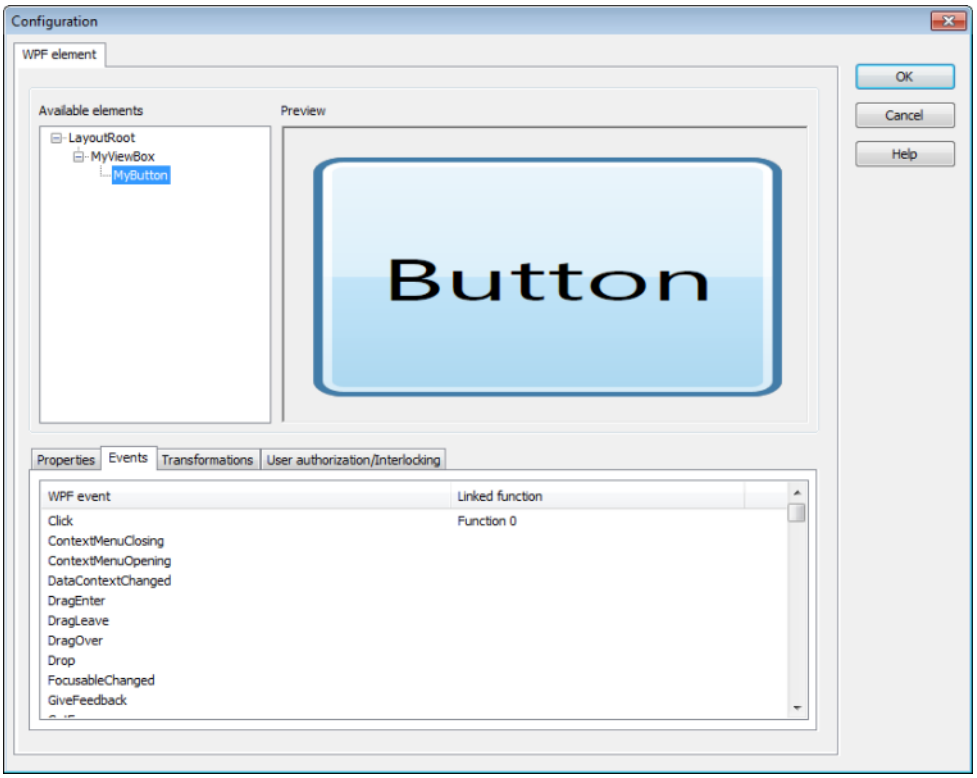
1. Selezionare il pulsante WPF, che si trova in **LayoutRoot->MyViewBox->MyButton**
2. Nella scheda **Proprietà**, cercare l´inserimento**Content**; questo contiene il testo del pulsante.
3. Cliccare nella colonna **Tipo di collegamento**
4. Nel menù a tendina, selezionare la voce **Variabile**.

5. Cliccare nella colonna **Collegamento**.
6. Si apre la finestra di selezione delle variabili.
7. Selezionare la variabile *int* per collegarla alla proprietà **Content**.

EVENTI

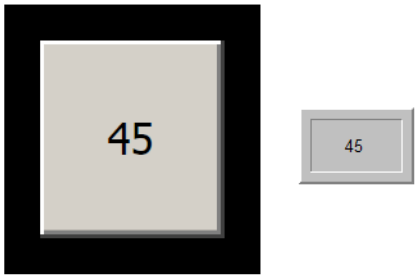
Per assegnare anche eventi:

1. Selezionare direttamente la scheda "Eventi"



2. Cercare la voce "Click"; questo evento viene attivato dall'elemento WPF non appena si fa clic sul pulsante
3. Cliccare nella colonna **Tipo di collegamento**
4. Nel menù a tendina, selezionare la voce **Funzione**.
5. Cliccare nella colonna **Collegamento**.
6. Si apre la **finestra di selezione funzioni**.
7. Selezionare **Funzione_0**
8. Confermare le modifiche con **OK**.
9. Aggiungere nell´immagine anche un **Elemento "Valore numerico"**.
10. Collegare anche questo **Valore numerico** alla variabile *int*.
11. Compilare i file Runtime e avviare il Runtime.

A Runtime viene visualizzato l´**elemento WPF**; il testo del pulsante è *0*. Non appena si fa clic sul pulsante, viene attivato l´evento **Click** e viene eseguita la funzione **Imposta valore**. Il valore *45* viene inviato direttamente all'hardware e sia il **Valore numerico**, che il **pulsante** indicano il valore *45* .

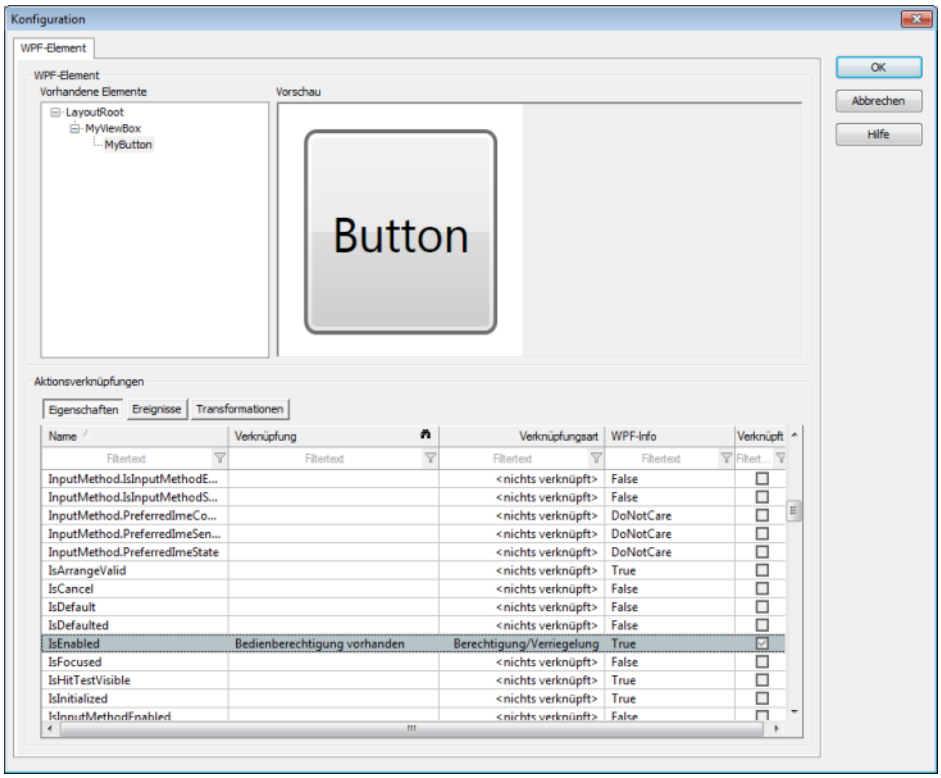


Se si imposta un valore di *30* utilizzando l'elemento **Valore numerico**, questo valore viene anche adottato dall´**elemento WPF**.

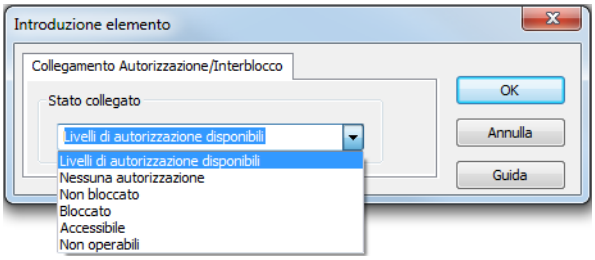
AUTORIZZAZIONE OPERATIVA

Come nel caso del **Valore numerico**, un elemento **WPF** può essere bloccato (simbolo del lucchetto) oppure rilasciato in dipendenza di un´autorizzazione operativa. Impostare il livello di autorizzazione per l'**elemento WPF** su *1* e creare un utente **Test** con il **livello di autorizzazione 1**. Creare poi le funzioni **Login con dialogo** e **Logout**. Collegare queste due funzioni con 2 nuovi pulsanti di testo nell´immagine.

Nella finestra di dialogo di configurazione dell´**elemento WPF**, selezionare il pulsante WPF **MyButton** e aprire la scheda **Proprietà**

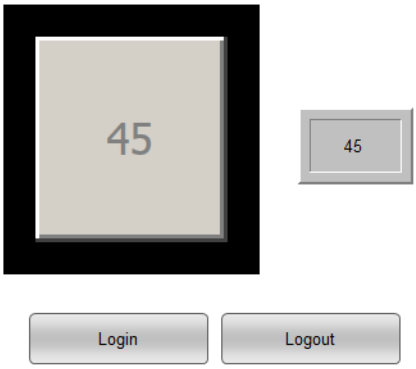


1. Selezionare l´elemento **IsEnabled**
2. Cliccare nella colonna **Tipo di collegamento**
3. Nel menù a tendina, selezionare la voce **Autorizzazione utente/Interblocco**.
4. Cliccare nella colonna **Collegamento**.
5. Nel menù a tendina, selezionare l´opzione *Livelli di autorizzazione disponibili*.



6. Chiudere la finestra di dialogo cliccando su **OK**.

Compilare i file Runtime e assicurarsi di selezionare anche le autorizzazioni operative per il trasferimento. Dopo l'avvio del Runtime, il pulsante WPF nell'immagine è disattivato e non può essere azionato. Se si esegue il Login con l´utente **Test**, il pulsante viene attivato e può essere utilizzato. Non appena si esegue il logout, il pulsante viene bloccato nuovamente.



TRASFORMAZIONE

Per poter utilizzare le trasformazioni, il file XAML deve essere adattato:

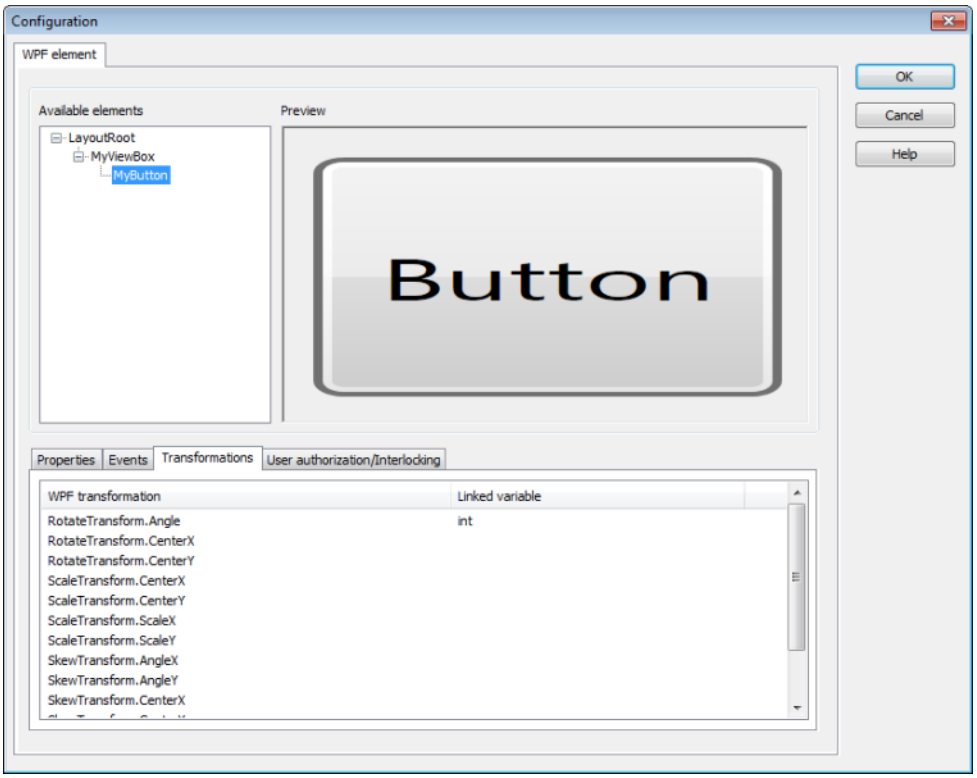
1. Passare al programma **Expression Blend**
2. Selezionare **MyButton**, in modo che le proprietà del pulsante siano visibili nella finestra delle proprietà.

3. Alla voce **Transform** di **RenderTransform**, selezionare l´opzione **Apply relative transform**

Un blocco viene inserito nel file XAML che memorizza le impostazioni di trasformazione a Runtime.

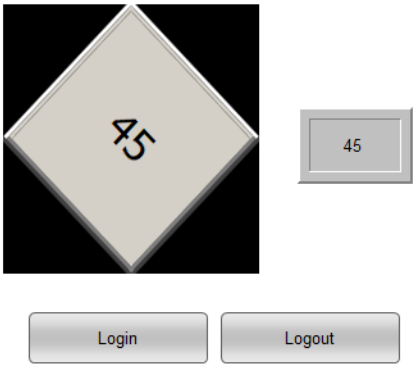
```
<Button.RenderTransform>
    <TransformGroup>
        <ScaleTransform ScaleX="1" ScaleY="1"/>
        <SkewTransform AngleX="0" AngleY="0"/>
        <RotateTransform Angle="0"/>
        <TranslateTransform X="0" Y="0"/>
    </TransformGroup>
</Button.RenderTransform>
```

4. Salvare il file e sostituire nel progetto zenon la vecchia versione con questo nuovo file.
5. Aprire di nuovo la finestra di configurazione dell´ **elemento WPF**:
   a) Selezionare il pulsante **MyButton**.
   b) Passare alla scheda **Trasformazione**



   c) Selezionare l´elemento **RotateTransform.Angle**
   d) Cliccare nella colonna **Tipo di collegamento**
   e) Nel menù a tendina, selezionare la voce **Trasformazioni**.
   f) Cliccare nella colonna **Collegamento**.
   g) Si apre la finestra di selezione delle variabili.
   h) Selezionare la variabile**int** per collegarla alla proprietà **RotateTransform.Angle**.

Compilare i file Runtime e avviare il Runtime. Eseguire il login come utente **Test** e cliccare sul pulsante. Il pulsante riceve il valore *45* e l`**elemento WPF** ruota di 45°.



6.7.3 Implementare DataGrid Control in zenon

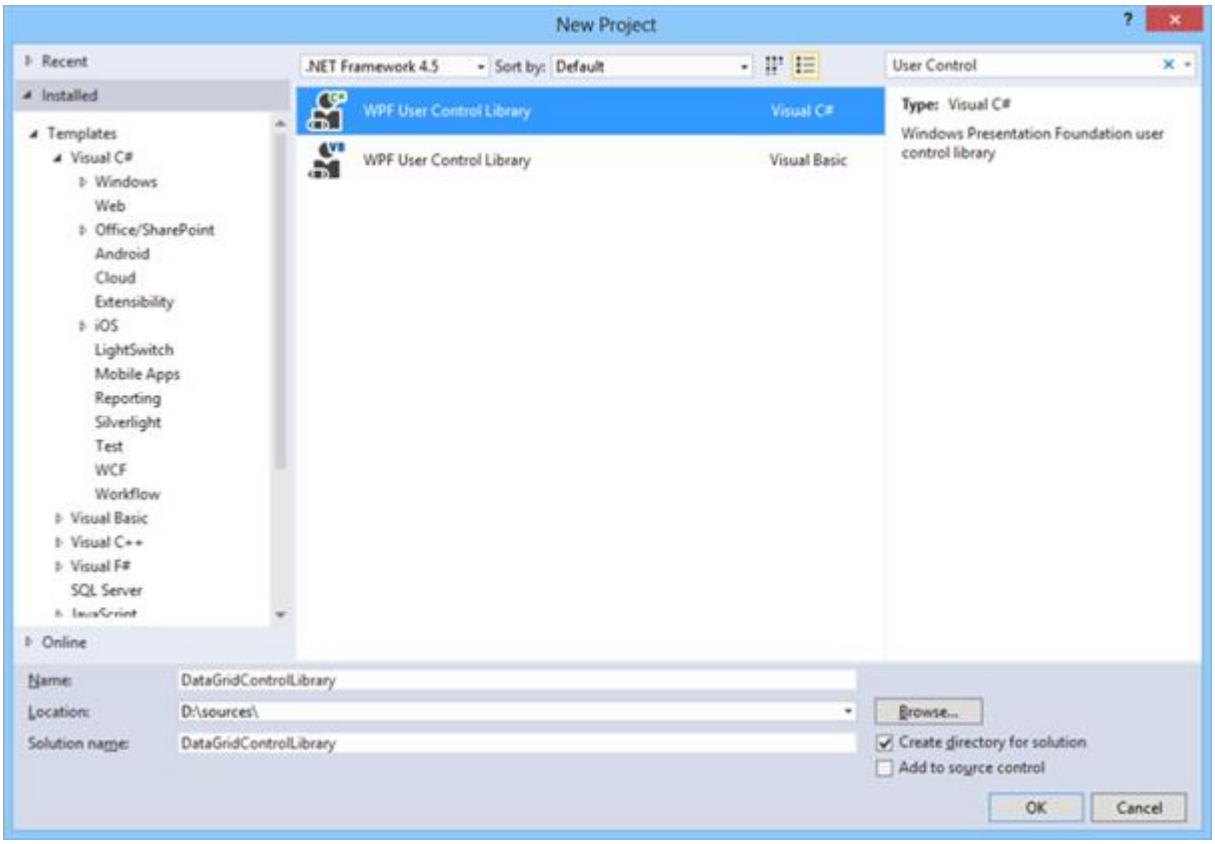
Per creare un controllo DataGrid per zenon è necessario:

- Visual Studio (in questo esempio Visual Studio 2015)

CREARE UN CONTROLLO UTENTE WPF

1. Creare una nuova **Solution** in Visual Studio e un progetto **WPF User Control Library** in .NET Framework versione 4.6.2, o successiva.

   Info: se il modello di progetto corrispondente non appare nell'elenco dei modelli disponibili, può essere aggiunto tramite la ricerca (campo nell'angolo in alto a destra della finestra di dialogo).



Nel nostro esempio, il progetto è denominato **DataGridControlLibrary**.

2. Creare una nuova connessione dati in **Server Explorer**,

   In questo esempio, viene utilizzato il database *Northwind*, fornito da Microsoft come database di esempio per il download gratuito.

   Per creare la connessione database:

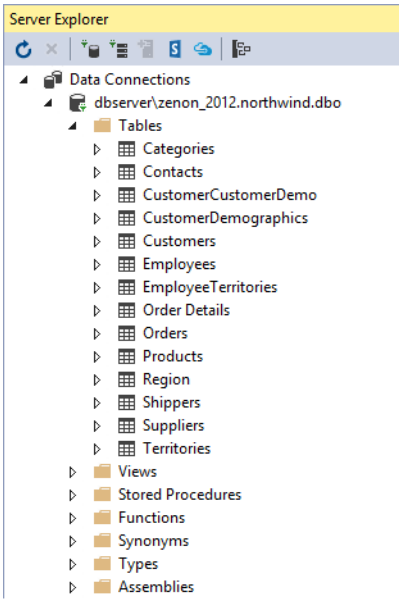
   a) Cliccare con il tasto destro del mouse su **Data Connections**.

   b) Selezionare **Add connection...**.

   c) Selezionare *Microsoft SQL Server (SQLClient)* come **Data source**.

   d) Selezionare il server e il nome del database appropriati.

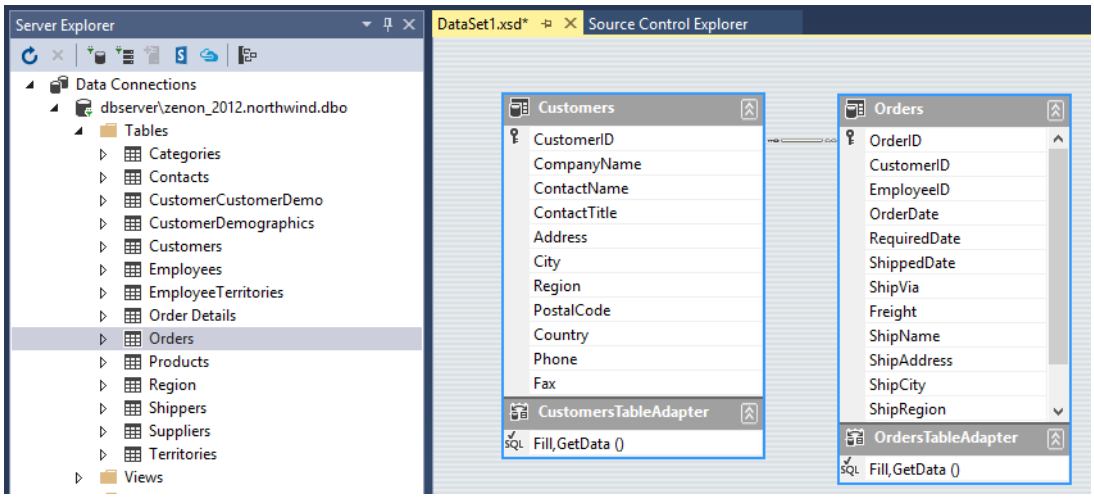
Dopo aver aggiunto la connessione, la finestra Server Explorer dovrebbe avere il seguente aspetto:



Nel passaggio successivo viene creato un nuovo DataSet.

CREAZIONE DI UN DATASET.

1. Cliccare con il tasto destro del mouse sul progetto.
2. Nel menù contestuale, selezionare **Add – New Item...**.
3. Creare un nuovo **DataSet** con il nome *DataSet1*.
4. Fare doppio clic sul DataSet per aprirlo nella finestra di progettazione.
5. Trascinare le tabelle di cui si ha bisogno (in questo esempio *Customers* e *Orders*) nella finestra di progettazione DataSet.



Il prossimo passo consiste nella modifica del file XAML.

CONFIGURAZIONE DEL FILE XAML

1. Se non è già presente, aggiungere il **Namespace** come riferimento alla classenel file XAML:

```
<UserControl x:Class="DataGridControlLibrary.UserControl1"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
             xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
             xmlns:local="clr-namespace:DataGridControlLibrary"
             mc:Ignorable="d"
             d:DesignHeight="300" d:DesignWidth="300">
```

2. Definire le risorse e il DataGrid da utilizzare nel WPF:

```
<UserControl.Resources>
    <local:DataSet1 x:Key="DataSet1"/>
    <CollectionViewSource x:Key="CustomersViewSource" Source="{Binding Path=Customers,
    Source={StaticResource DataSet1}}"/>
</UserControl.Resources>
<Grid DataContext="{StaticResource CustomersViewSource}">
    <DataGrid Name="DataGrid1" DisplayMemberPath="CompanyName" ItemsSource="{Binding}"
    SelectedValuePath="CustomerID" HorizontalAlignment="Stretch" VerticalAlignment="Stretch"/>
</Grid>
```

3. Aprite il file code-behind (**UserControl1.xaml.cs**) e aggiungete nel costruttore le seguenti righe:

```
public UserControl1()
{
    InitializeComponent();
    DataSet1 ds = ((DataSet1)(FindResource("DataSet1")));
    DataSet1TableAdapters.CustomersTableAdapter ta = new
    DataSet1TableAdapters.CustomersTableAdapter();
    ta.Fill(ds.Customers);
    CollectionViewSource CustomersViewSource =
    ((CollectionViewSource)(this.FindResource("CustomersViewSource")));
    CustomersViewSource.View.MoveCurrentToFirst();
}
```

In questo processo:

- Viene ottenuto il DataSet.
- Viene creato un nuovo TableAdapter.
- Il Dataset viene popolato.

- Le informazioni sono rese disponibili al DataGrid Control.

Adesso si può compilare la soluzione.

BUILD

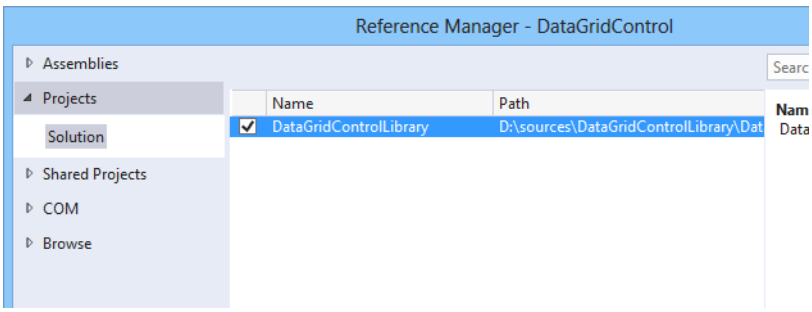
Compilare adesso la solution. La DLL corrispondente (**DataGridControlLibrary.dll**) viene ora creata nella cartella di output del progetto.

Ora è disponibile la DLL che fornisce la funzionalità richiesta.

Tuttavia zenon può solo visualizzare file XAML che non possono essere collegati al file code-behind, quindi è necessario un file XAML aggiuntivo che faccia riferimento alla DLL appena creata.

Procedura:

1. Create anzitutto un altro progetto, ancora una volta del tipo **WPF User Control Library**.
2. In questo esempio, viene chiamato **DataGridControl**.
3. Includere un riferimento al progetto appena creato in questo nuovo progetto.



4. Il file XAML (**UserControl1.xaml**) ha il seguente aspetto:

```
<UserControl x:Class="DataGridControl.UserControl1"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
             xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
             xmlns:local="clr-namespace:DataGridControl"
             mc:Ignorable="d"
             d:DesignHeight="300" d:DesignWidth="300">
  <Grid>

  </Grid>
</UserControl>
```

5. Poiché tutti i contenuti necessari sono contenuti nella DLL creata in precedenza e non è richiesto alcun code behind, cancellare le righe seguenti:

   x:Class="`DataGridControl.UserControl1`"

   xmlns:local="`clr-namespace:DataGridControl`"

6. Cancellare le seguenti righe aggiuntive (per il posizionamento):

   mc:Ignorable="`d`"

   d:DesignHeight="`300`" d:DesignWidth="`300`"

7. Eliminare il file code-behind (**UserControl1.xaml.cs**) in questo progetto.
8. Definire cosa si vuole visualizzare nel file XAML.

   Per fare questo, modificare il file XAML come segue:

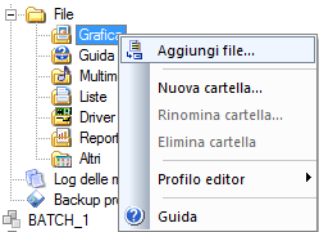
```
<UserControl xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:dataGridLibrary="clr-namespace:DataGridControlLibrary;assembly=DataGridControlLibrary">
<Grid x:Name="Grid1">
    <dataGridLibrary:UserControl1 Name="DataGridControl" HorizontalAlignment="Left"
    VerticalAlignment="Top"/>
</Grid>
</UserControl>
```

   La riga `xmlns:dataGridLibrary="clr-namespace:DataGridControlLibrary;assembly=DataGridControlLibrary"` definisce lo spazio nome **dataGridLibrary** e stabilisce che esso deve usare l´assembly precedentemente creato.
9. Assegnare un nome al Grid.
10. Aggiungere il Control **dataGridLibrary:UserControl1** della nostra libreria e assegnare un nome, perché zenon può modificare solamente quegli oggetti che hanno un nome.
11. Compilare adesso la soluzione.

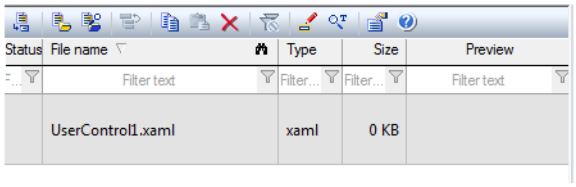
Il passo successivo spiega come inserire la DLL e il file XAML in zenon.

PASSAGGI IN ZENON

1. Aprire l´Editor di zenon.
2. Passare al nodo *File -> Grafica*.
3. Nel menù contestuale, selezionare **Aggiungi file...**



4. Scegliere il file XAML nella posizione di memorizzazione (**UserControl1.xaml** del progetto **DataGridControl**) e aggiungerlo:

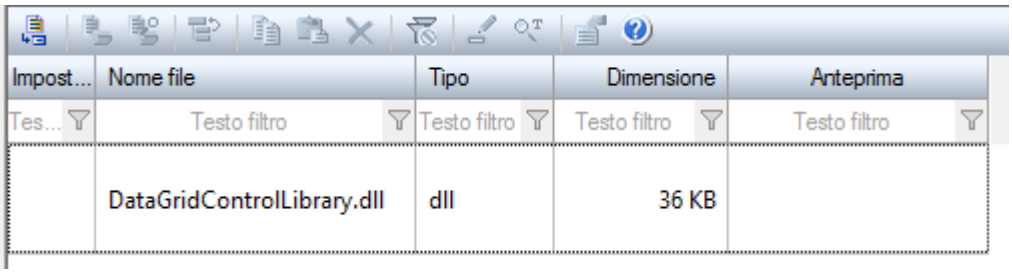


5. Aggiungere la DLL con la funzionalità per il file XAML.

   Procedura:

   a) Nel menù contestuale del nodo File -> Altri, selezionare **Aggiungi file...**.

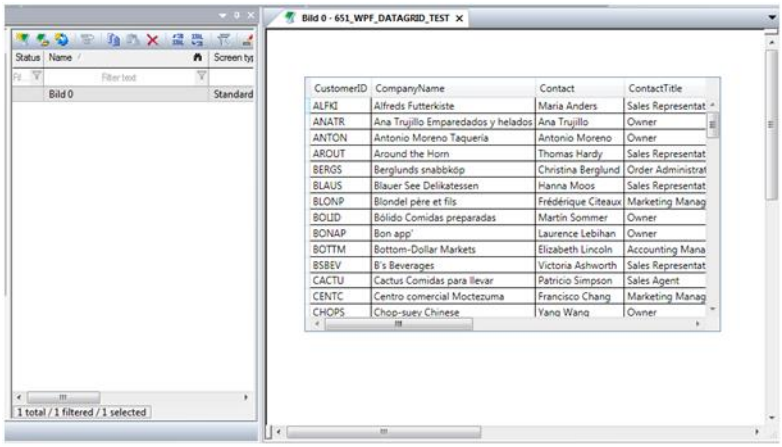
   b) Selezionare il file **DataGridControlLibrary.dll** del primo progetto (**DataGridControlLibrary**).



6. Creare un´immagine di zenon.

7. Aggiungere un elemento WPF e selezionare il file XAML precedentemente implementato.

   Adesso si dovrebbe vedere quanto segue nell´Editor di zenon:



8. Avviare il Runtime di zenon per testare il controllo anche in quella sede.

Suggerimento

Le DLL appartenenti ad un elemento WPF possono anche essere sostituite anche durante il funzionamento dell´Editor. Il referenziamento avviene tramite un link nel file XAML.

Per sostituire una DLL:

- Chiudere tutte le immagini di zenon che utilizzano l'elemento WPF.
- Chiudere tutti i simboli che utilizzano un elemento WPF.
- In Explorer, sostituire la DLL nella cartella *\wpfache* dei file Editor. Questa cartella si trova nella directory SQL, alla voce: *...\PROJECT-GUID\FILES\zenon\custom\wpfcache*

In alternativa alla sostituzione tramite Explorer, è anche possibile sostituire il file direttamente nell'Editor di zenon. Procedura:

- Nelle impostazioni di progetto di Visual Studio, aumentare la versione del file della DLL.
- Ricostruire la DLL.
- Chiudere tutte le immagini di zenon che utilizzano l'elemento WPF.
- Chiudere tutti i simboli che utilizzano un elemento WPF.
- Nell´Editor di zenon, cancellare la DLL dalla cartella *\File\Altri* e incollare il file con il numero di versione aumentato.

6.8 Gestione degli errori

INSERIMENTI NEI FILE LOG.

Voce	Livello	Significato
Xaml file found in %s with different name, using default!	Avviso	Il nome del file collection e quello del file XAML in esso contenuto non corrispondono. Per evitare dei conflitti interni, nella cartella viene usato il file con il nome del file collection e la desinenza **.xaml**.
no preview image found in %s	Avviso	Il file collection non contiene un'immagine di anteprima valida (**preview.png** oppure **[Nome del file XAML]**.png). Perciò non può essere visualizzata nessuna anteprima.
Xaml file in %s not found or not unique!	Errore	Il file collection non contiene nessun file XAML, oppure più file con la desinenza **.xaml**; non può essere usato.

Voce	Livello	Significato
Could not remove old assembly %s	Avviso	Nella cartella di lavoro si trova un assembly che deve essere sostituito con una nuova versione, ma non può essere cancellato.
Could not copy new assembly %s	Errore	Per un assembly nella cartella di lavoro è disponibile una nuova versione, che però non può essere copiata nella cartella stessa. Possibile causa: per es. il vecchio Assembly è ancora caricato. La vecchia versione continua ad essere usata, la nuova non può essere utilizzata.
file exception in %s	Errore	Si è verificato un errore file nel momento di accedere al file collection.
Generic exception in %s	Errore	Si è verificato un errore generale al momento di accedere al file collection.