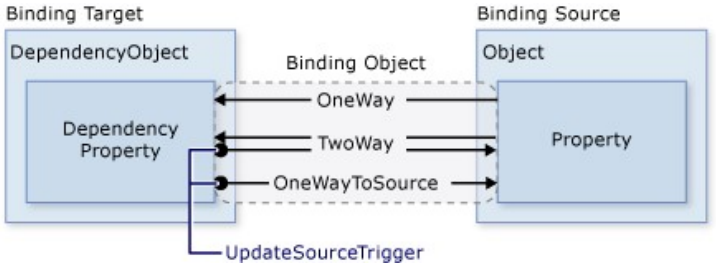
Databinding

Lega *UIElement* all'informazione

- *Source*: dato
- *Target*: elemento



Sorgente: *ElementName*, *Source*, *DataContext*, *RelativeSource*

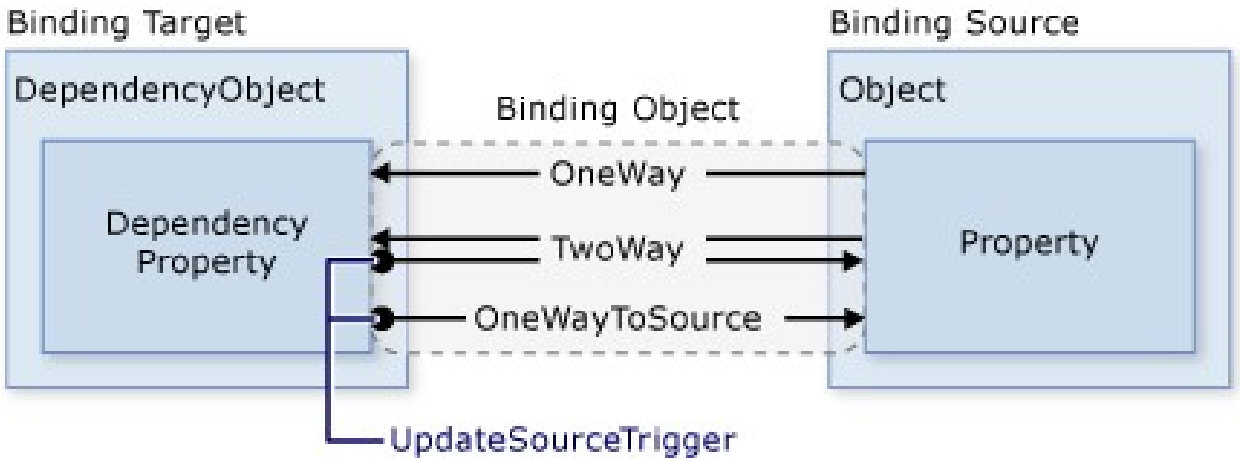
UpdateSourceTrigger: *LostFocus*, *PropertyChanged*, *Explicit*

Path: multilivello, array, index

Direzioni e trigger variano a seconda del controllo/proprietà

icubed

Databinding



Target

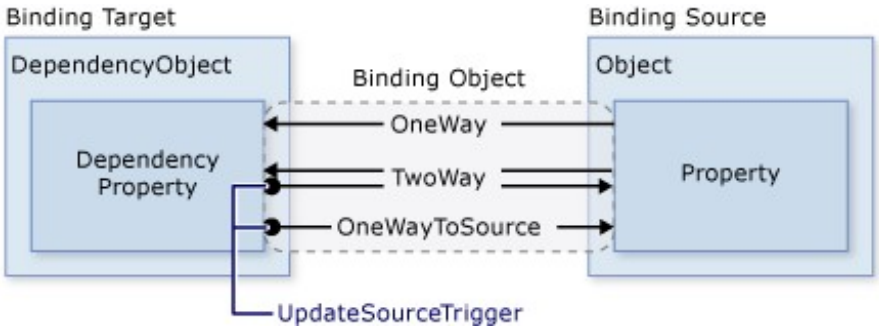
Target Property – spesso definita dependency property

Oggetto sorgente

Proprietà

icubed

Databinding



La direzione del binding indica da chi parte la modifica e come questa viene attuata.

One way – quando cambia il valore dell'oggetto sorgente viene modificato quello dell'oggetto target. Si preferisce quando il valore da mostrare a video è solamente read-only

Two way – l'aggiornamento si ottiene in entrambe le direzioni

One way To Source – Aggiorna la proprietà solo quando quella di target cambia.

One Time – Binding che si scatena solo una volta e non si propaga.

Databinding

Quando viene scatenato il databinding?

Dipende!

Molti controlli utilizzano la proprietà PropertyChanged, ovvero non appena la proprietà viene modificata.

Alcuni controlli (ad es. DataText) traggono invece vantaggio da eventi differenti come il LostFocus

Databinding - Creazione

La creazione del collegamento tra sorgente e target può avvenire in due modalità:

- Dichiarativa
- Tramite View

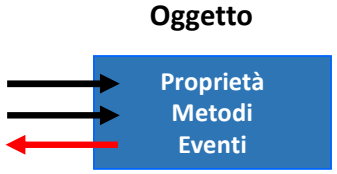
Entrambe utilizzano un particolare oggetto, il DataContext, per stabilire qual è l'origine dei dati.

Eventi

Un **evento** è un membro che permette alla classe di inviare notifiche verso l'esterno

L'evento mantiene una lista di *subscriber* che vengono iterati per eseguire la notifica

Tipicamente sono usati per gestire nelle Windows Forms le notifiche dai controlli all'oggetto container (la Form)

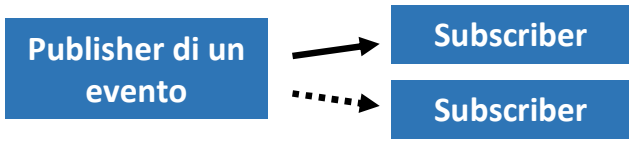


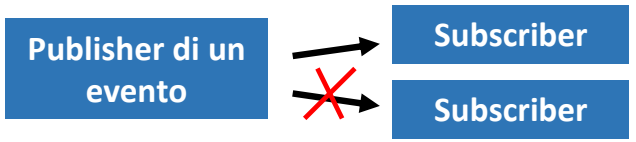
Si parla di:

- *Publisher* Inoltra gli eventi a tutti i subscriber
- *Subscriber* Riceve gli eventi dal publisher

Eventi

Ciascun subscriber deve essere aggiunto alla lista del publisher (*subscribe*) oppure rimosso (unsubscribe).






```
c.MyEvent += f;
```

```
c.MyEvent -= f;
```

icubed

Eventi

RoutedEvent

- Proprietà Handled per inibire la propagazione

Eventi tastiera, mouse e touch

- Classi *Keyboard*, *Mouse*, *Stylus*

I controlli intercettano eventi dei device e rilanciano eventi logici

- Es: *ButtonBase.Click*

Keyboard focus e scope focus

Commanding

Separa UI e logica di esecuzione

ICommand per rappresentare la logica

- *RoutedCommand*: comandi che percorrono il visual tree
  - *MediaCommands*, *ApplicationCommands*, *NavigationCommands*, *ComponentCommands*, ed *EditingCommands*

ICommandSource per invocare i comandi

- *ButtonBase*, *MenuItem*, *KeyBinding*, *MouseBinding*

CommandBinding: per eseguire le logiche

Commanding

ICommand

- *CanExecute*: valuta se è possibile eseguire
- *Execute*: esegure l'operazione

CommandManager: gestisce i comandi

- *InvalidateRequerySuggested*: lancia una rivalutazione dei *CanExecute*
  - Avviene automaticamente sul cambio focus, esecuzione del comando e switch di finestra

CommandParameter: passato di tipo object

CommandTarget: per specificare il destinatario

icubed

Converter

In alcuni casi può essere necessario trasformare alcune tipologie di dati in altre (ad es. le date).

In quel caso la classe che effettuerà la conversione dovrà implementare l'interfaccia IMultiValueConverter che ha i seguenti metodi:

- Convert
- ConvertBack

Si ha la possibilità di specificare dei parametri per gestire la conversione.

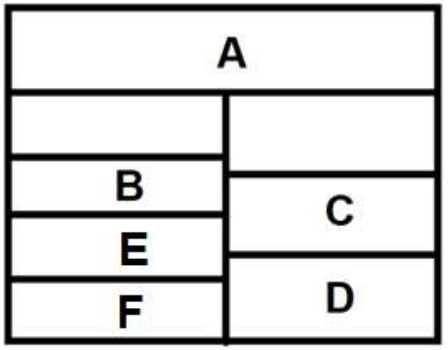
Demo

Eventi UI

Trasformazione applicazione di base

icubed

Esercitazione n. 2



Aggiungere un bottone nel campo E che consenta di aggiungere il prodotto selezionato al «carrello». Il riquadro F dovrà mostrare la somma dei prezzi dei prodotti selezionati.

La visualizzazione di questo campo sarà comandato da un check che lo abiliterà/disabiliterà

icubed