Zeus RelayCommand Ver. 1.0

Manual

Zeus RelayCommand version 1.0

Κλάσεις: RelayCommand.

Inherits: -

Implements: System.Windows.Input.ICommand interface.

Namespace: Zeus.WPF.Classes.Commands

Assembly: ZeusRelayCommand (in ZeusRelayCommand.dll)

Dependencies: -

Περιγραφή

Η κλάση **Zeus RelayCommand** έχει δημιουργηθεί για να βοηθήσει στη χρήση των **commands** σε μία WPF εφαρμογή, η οποία σχεδιάζεται με βάση το **MVVM (Model-View-ViewModel)** πρότυπο.

Για παράδειγμα, μπορούμε να δημιουργήσουμε ένα RelayCommand με όνομα AddCustomer το οποίο θα τεθεί ως ιδιότητα σε μία κλάση ViewModel (π.χ. CustomersViewModel.vb). Μέσα στην κλάση του ViewModel υλοποιούμε τις μεθόδους **CanExecute** (*πότε θα είναι δυνατή η προσθήκη του πελάτη, π.χ αν δεν δοθεί το επώνυμό του δεν θα είναι ενεργό το command*) και **Execute** (*πώς θα γίνει η προσθήκη του πελάτη π.χ. σε μία Βάση Δεδομένων*) που συνοδεύουν ένα ICommand. Στη συνέχεια, σε ένα WPF window με όνομα CustomersView.xaml συνδέουμε ένα button "Add Customer", μέσω binding έκφρασης, με την ιδιότητα AddCustomer του ViewModel.

Βεβαίως, αυτή είναι μία γενική επεξήγηση της χρήσης του RelayCommand και για να το χρησιμοποιήσει κάποιος πρέπει να γνωρίζει τις βασικές αρχές του σχεδιαστικού προτύπου MVVM.

Ο ορισμός της κλάσης

Η κλάση έχει οριστεί ως εξής:

- Για την **κλάση RelayCommand**:

  Σύνταξη:

  VB:

```
Public Class RelayCommand
       Implements ICommand
```

  XAML Object Element Usage:

  Δεν χρησιμοποιείται σε XAML κώδικα.

Διανομή

Κατά τη διανομή, στο φάκελο της εφαρμογής σας πρέπει να αντιγράψετε το **assembly αρχείο** **ZeusRelayCommand.dll**.

Επικοινωνία

Για οποιαδήποτε πληροφορία ή διευκρίνηση παρακαλώ επικοινωνήστε στο :

mouratx@yahoo.com ή mouratx@hotmail.com

Χρήστος Μουρατίδης,
Πειραιάς, Σεπτέμβριος 2018

Υ.Γ. Μπορείτε να επικοινωνήσετε μαζί μου για να προμηθευτείτε το **βιβλίο** μου **"Μάθετε το WPF με τη Visual Basic" (1.333 σελίδες, Αυτοέκδοση 2018).**

Παράδειγμα χρήσης

Θα υποθέσουμε ότι έχουμε ένα συνδρομητικό σύστημα, μέρος του οποίου είναι η καταχώρηση στοιχείων μελών (members). Για κάθε μέλος θα κρατάμε τα στοιχεία: Επώνυμο, Όνομα, Ποσό συνδρομής και Ημερομηνία λήξης συνδρομής. Η **αποθήκευση**, για λόγους απλότητας, θα γίνεται σε μία δομή συλλογής στοιχείων (**ObservableCollection** στο WPF).

Θα χρειαστεί να γίνουν τα εξής:

- Να δημιουργηθεί το **μοντέλο του συστήματος** (**Model**). Στην προκειμένη περίπτωση, θα περιλαμβάνει μόνο την κλάση μέλους, με όνομα **Member**. Το αρχείο της κλάσης θα έχει όνομα **Member.vb**.

- Να δημιουργηθεί **φόρμα εμφάνισης των μελών και καταχώρησης των στοιχείων ενός μέλους** (**View**). Στην προκειμένη περίπτωση, θα είναι ένα WPF παράθυρο με όνομα **MembersView.xaml**.

- Να δημιουργηθεί **ο "μεσάζων", δηλαδή η κλάση ViewModel**. Εδώ ορίζονται όλοι οι τύποι δεδομένων και καθώς και άλλες δομές που θα χρειαστούν για την εμφάνιση των δεδομένων στο MembersView καθώς και την αντίστροφη πορεία, δηλαδή την αποθήκευση στο ObservableCollection (σε πραγματικές συνθήκες θα ήταν σε Βάση Δεδομένων). Το όνομα αυτής της ενδιάμεση κλάσης θα είναι **MembersViewModel** και θα αποθηκευτεί στο ομώνυμο αρχείο κλάσης **MembersViewModel.vb**. Εδώ, θα ορίσουμε τα RelayCommands για την έναρξη της προσθήκης/διόρθωσης και διαγραφή ενός μέλους καθώς και για την αποθήκευση/ακύρωση της τρέχουσας προσθήκης/διόρθωσης των στοιχείων μέλους.

Στις επόμενες σελίδες υλοποιούμε τα παραπάνω.

Η κλάση Member

Πρώτα **θα ορίσουμε την κλάση για την οντότητα του Μέλους**.

Για κάθε μέλος θα κρατάμε τα στοιχεία:

- Επώνυμο (LastName),
- Όνομα (FirstName),
- Ποσό συνδρομής (SubscriptionAmount) και
- Ημερομηνία λήξης συνδρομής (SubscriptionExpires)

Το **αρχείο της κλάσης** ονομάζεται **Member.vb**.

```
Imports System.ComponentModel

'*******************************************************************************
'***********  The class for the Member's data. Part of the Model  ***************
'*******************************************************************************

Public Class Member
    Inherits NotifyPropertyChangedBase

    'Private backing fields.
    Private _lastName As String
    Private _firstName As String
    Private _subscriptionAmount As Decimal
    Private _subscriptionExpires As Nullable(Of Date)

    'CLR properties - data fields.
    Public Property LastName As String
        Get
            Return _lastName
        End Get
        Set(ByVal value As String)
            _lastName = value
            OnPropertyChanged(Me, New PropertyChangedEventArgs("LastName"))
        End Set
    End Property

    Public Property FirstName As String
        Get
            Return _firstName
        End Get
        Set(ByVal value As String)
            _firstName = value
            OnPropertyChanged(Me, New PropertyChangedEventArgs("FirstName"))
        End Set
    End Property

    Public Property SubscriptionAmount As Decimal
        Get
            Return _subscriptionAmount
        End Get
        Set(ByVal value As Decimal)
            _subscriptionAmount = value
            OnPropertyChanged(Me, New PropertyChangedEventArgs("SubscriptionAmount"))
        End Set
```

```
    End Property

    Public Property SubscriptionExpires As Nullable(Of Date)
        Get
            Return _subscriptionExpires
        End Get
        Set(ByVal value As Nullable(Of Date))
            _subscriptionExpires = value
            OnPropertyChanged(Me, New PropertyChangedEventArgs("SubscriptionExpires"))
        End Set
    End Property

    'Constructor.
    Public Sub New()

        _lastName = String.Empty
        _firstName = String.Empty
        _subscriptionAmount = 0
        _subscriptionExpires = Date.Now.AddDays(360)   'a commercial year later.

    End Sub

    Public Overrides Function ToString() As String

        Return String.Format("{0}  {1}", _lastName, _firstName)

    End Function

End Class
```

Η κλάση MembersViewModel

Παραθέτουμε ολόκληρη την **κλάση MembersViewModel**, η οποία αποθηκεύεται στο **αρχείο MembersViewModel.vb.** Τα υπερφωτισμένα σημεία αφορούν το RelayCommand:

```
Imports System.Collections.ObjectModel
Imports System.ComponentModel
Imports Zeus.WPF.Classes.Commands

''' <summary>
'''The ViewModel class that provides data for the MembersView.xaml.
''' </summary>
Public Class MembersViewModel
           Inherits NotifyPropertyChangedBase

    'The internal collection that stores the members data.
    Private _membersObservableCollection As ObservableCollection(Of Member)

    Private currentMember As New Member
    Private backupcurrentMember As New Member

#Region "Command Properties"

    Public Property AddMemberCommand As New RelayCommand(New Action(Of Object)
                                            (AddressOf AddMemberCommandExecute), _
                 New Func(Of Object, Boolean)(AddressOf AddMemberCommandCanExecute))

    Public Property EditMemberCommand As New RelayCommand(New Action(Of Object)
                                             (AddressOf EditMemberCommandExecute), _
                 New Func(Of Object, Boolean)(AddressOf EditMemberCommandCanExecute))

    Public Property DeleteMemberCommand As New _
           RelayCommand(New Action(Of Object)(AddressOf DeleteMemberCommandExecute), _
                New Func(Of Object, Boolean)(AddressOf DeleteMemberCommandCanExecute))

    Public Property SaveMemberDetailsCommand As New _
        RelayCommand(New Action(Of Object)(AddressOf SaveMemberDetailsCommandExecute),_
             New Func(Of Object, Boolean)(AddressOf SaveMemberDetailsCommandCanExecute))

    Public Property CancelEditMemberDetailsCommand As New _
                                                    RelayCommand(New Action(Of Object)
                                      (AddressOf CancelEditMemberDetailsCommandExecute))

#End Region

    'Private backing fields for properties.

    'for Sorting, Grouping the _membersObservableCollection.
    Private _membersListCollectionView As ListCollectionView

    'indicating that the process is in add/edit a member.
    'This affects some elements in the UI.
    Private _isEditMode As Boolean

   'indicating that the viewmodel has errors, so not to save the data.
    Private _hasErrors As Boolean

#Region "Properties"
```

```
    Public Property MembersListCollectionView() As ListCollectionView
        Get
            Return _membersListCollectionView
        End Get
        Set(ByVal value As ListCollectionView)
            _membersListCollectionView = value
            OnPropertyChanged(Me, _
                              New PropertyChangedEventArgs("MembersListCollectionView"))
        End Set

    End Property

    Public Property IsEditMode As Boolean
        Get
            Return _isEditMode
        End Get
        Set(value As Boolean)
            _isEditMode = value
            OnPropertyChanged(Me, New PropertyChangedEventArgs("IsEditMode"))
        End Set
    End Property

    Public Property HasErrors As Boolean
        Get
            Return _hasErrors
        End Get
        Set(value As Boolean)
            _hasErrors = value
            OnPropertyChanged(Me, New PropertyChangedEventArgs("HasErrors"))
        End Set
    End Property

    Public Property FirstViewElement As FrameworkElement

#End Region

#Region "Constructor"

    Public Sub New()

        _membersObservableCollection = New ObservableCollection(Of Member)
        _membersListCollectionView = _
             CType(CollectionViewSource.GetDefaultView(_membersObservableCollection), _
                                                                    ListCollectionView)

        'Sorting the list by LastName ASC, FirstName ASC.
        With _membersListCollectionView
            .SortDescriptions.Add(New SortDescription("LastName", _
                                                      ListSortDirection.Ascending))
            .SortDescriptions.Add(New SortDescription("FirstName", _
                                                      ListSortDirection.Ascending))
        End With

        _isEditMode = False
        _hasErrors = False

    End Sub

#End Region
```

```
#Region "Command handlers"

    '--- Add Member
    Private Function AddMemberCommandCanExecute(parameter As Object) As Boolean

        Return Not IsEditMode  'disable if it is in add/edit mode.

    End Function

    Private Sub AddMemberCommandExecute(parameter As Object)

        currentMember = _membersListCollectionView.AddNew()
        IsEditMode = True

        'Set focus to the first element in View.
        FirstViewElement.Focus()

    End Sub

    '--- Edit Member
    Private Function EditMemberCommandCanExecute(parameter As Object) As Boolean

       'disable if it is in add/edit mode and there are members.
        Return (Not IsEditMode) And (_membersListCollectionView.Count > 0)

    End Function

    Private Sub EditMemberCommandExecute(parameter As Object)

        _membersListCollectionView.EditItem(_membersListCollectionView.CurrentItem)
        currentMember = _membersListCollectionView.CurrentEditItem
        BackupMemberData(currentMember, backupcurrentMember)

        IsEditMode = True

        'Set focus to the first element in View.
        FirstViewElement.Focus()

    End Sub

    '--- Delete Member
    Private Function DeleteMemberCommandCanExecute(parameter As Object) As Boolean

      'disable if it is in add/edit mode and there are members.
        Return (Not IsEditMode) And (_membersListCollectionView.Count > 0)

    End Function

    Private Sub DeleteMemberCommandExecute(parameter As Object)

        With _membersListCollectionView

            If .CurrentItem IsNot Nothing Then

                If MessageBox.Show(String.Format("Delete the member : {0}?", _
                                                 .CurrentItem.ToString), "Confirmation", _
                                   MessageBoxButton.YesNo, MessageBoxImage.Question, _
                                      MessageBoxResult.No) = MessageBoxResult.Yes Then
                    .Remove(.CurrentItem)
```

```
                End If

            End If

        End With

    End Sub

    '--- Save Member details
    Private Function SaveMemberDetailsCommandCanExecute(parameter As Object) As Boolean

        Return If(currentMember.LastName.Length > 0, True, False)

    End Function

    Private Sub SaveMemberDetailsCommandExecute(parameter As Object)

        'Check for errors in the viewmodel input data.
        If HasErrors Then

            MessageBox.Show("The form has errors. Please, correct them and retry.", _
                                                                    "Save member", _
                                        MessageBoxButton.OK, MessageBoxImage.Exclamation)
        Else
            'and if there are not errors then proceed to save the data.
            With _membersListCollectionView
                If .IsAddingNew Then .CommitNew() Else .CommitEdit()
            End With

            _membersListCollectionView.Refresh()
            IsEditMode = False

        End If


    End Sub

    '--- Cancel edit Member details
    Private Sub CancelEditMemberDetailsCommandExecute(parameter As Object)

        With _membersListCollectionView

            If .IsAddingNew Then

                    .CancelNew()

               Else

                   RestoreMemberData(backupcurrentMember, currentMember)
                                        '.CancelEdit() is not working.
               endif

        End With

        IsEditMode = False

    End Sub

#End Region
```

```
#Region "Public methods"

    ''' <summary>
    ''' Creates initially some members data.
    ''' </summary>
    Public Sub InitializeSomeMembersData()

        With _membersObservableCollection

            .Add(New Member With {.FirstName = "Christos", .LastName = "Mouratidis", _
                        .SubscriptionAmount = 35, .SubscriptionExpires = #9/20/2019#})

            .Add(New Member  With {.FirstName = "Panagiotis", .LastName = "Varzakakos", _
                        .SubscriptionAmount = 25, .SubscriptionExpires = #9/26/2019#})

            .Add(New Member  With {.FirstName = "Maria", .LastName = "Theologou", _
                        .SubscriptionAmount = 35, .SubscriptionExpires = #11/20/2019#})

            .Add(New Member  With {.FirstName = "Sofia", .LastName = "Mellisanidi", _
                        .SubscriptionAmount = 45, .SubscriptionExpires = #9/20/2020#})

        End With

    End Sub

    ''' <summary>
    ''' Go to the fisrt member.
    ''' </summary>
    Public Sub MoveToFirstMember()

        _membersListCollectionView.MoveCurrentToFirst()

    End Sub

#End Region

#Region "Private methods"

    'The next 2 methods are useful for the edit/cancel process.
    Private Sub BackupMemberData(sourceMember As Member, backupMember As Member)

        With backupMember
            .LastName = sourceMember.LastName
            .FirstName = sourceMember.FirstName
            .SubscriptionAmount = sourceMember.SubscriptionAmount
            .SubscriptionExpires = sourceMember.SubscriptionExpires
        End With

    End Sub

    Private Sub RestoreMemberData(backupMember As Member, member As Member)

        With member
            .LastName = backupMember.LastName
            .FirstName = backupMember.FirstName
            .SubscriptionAmount = backupMember.SubscriptionAmount
            .SubscriptionExpires = backupMember.SubscriptionExpires
        End With

    End Sub
```

```
#End Region

End Class
```

Παρατηρήσεις:

- Για την αποθήκευση των δεδομένων χρησιμοποιούμε ένα εσωτερικό **ObservableCollection** που αποθηκεύει στοιχεία τύπου Member. Ένα αντικείμενο ObservableCollection ενημερώνει το UI (στην προκειμένη περίπτωση το MembersView.xaml) σε κάθε αλλαγή (προσθήκη/ διαγραφή των αντικειμένων Member).

- Ένα **ListCollectionView** αποτελεί ένα χρησιμότατο αντικείμενο για να θέσουμε μία "οπτική" στη συλλογή μας. Παρέχει, μεταξύ άλλων, τη δυνατότητα για **ταξινόμηση** των στοιχείων της συλλογής μας. Αυτό θα "βλέπει" το UI, μέσω της ιδιότητας MembersListCollectionView.

- Χρειαζόμαστε την ιδιωτική μεταβλητή **currentMember**, η οποία κρατάει τα στοιχεία του τρέχοντος μέλους. Η **backupcurrentMember** είναι χρήσιμη για να κρατάει την προηγούμενη "έκδοση" του μέλους σε περίπτωση που ο χρήστης ακυρώσει μία διαδικασία διόρθωσης (cancel edit). Προς το τέλος βρίσκουμε τις δύο private **μεθόδους BackupMemberData** και **RestoreMemberData** που αφορούν τη διαδικασία backup/restore του τρέχοντος αντικειμένου Member.

- Τα υπερφωτισμένα σημεία αφορούν το πώς χρησιμοποιούμε την κλάση RelayCommand. Δηλώνουμε **5 ιδιότητες τύπου RelayCommand** για: την έναρξη της προσθήκης ενός νέου μέλους (**AddMemberCommand**), διόρθωσης των στοιχείων του τρέχοντος μέλους (**EditMemberCommand**), διαγραφής (**DeleteMemberCommand**), αποθήκευσης (**SaveMemberDetailsCommand**) και ακύρωσης προσθήκης/διόρθωσης (**CancelEditMemberDetailsCommand**).

- Κατά την **αρχικοποίηση ενός RelayCommand (New)** μπορούμε να προσδιορίσουμε **2 παραμέτρους**:

    - μία **διαδικασία Execute**
    - μία **συνάρτηση CanExecute**.

    Αυτές υλοποιούνται παρακάτω (στο τμήμα "Command handlers"). Η **διαδικασία Execute περιέχει κώδικα που υλοποιεί τις ενέργειες που πρέπει να γίνουν** και η **συνάρτηση CanExecute περιέχει κώδικα ελέγχου που προσδιορίζει πότε θα είναι ενεργό το command**. Η CanExecute δεν είναι υποχρεωτικό να οριστεί ως παράμετρος στο RelayCommand, οπότε θεωρείται ότι αυτόματα επιστρέφει True, δηλαδή το command θα είναι πάντα ενεργό. Το τελευταίο RelayCommand (CancelEditMemberDetailsCommand) έχει μόνο παράμετρο Execute.

    Για παράδειγμα, για το **EditMemberCommand** ορίζεται η **παράμετρος EditMemberCommandExecute** και στην υλοποίησή της παρακάτω εκτελούμε τις εντολές έναρξης της διόρθωσης του τρέχοντος μέλους (κρατώντας παράλληλα ως backup τα τρέχοντα στοιχεία). Επίσης, ορίζεται και η παράμετρος **EditMemberCommandCanExecute** και στην υλοποίησή της παρακάτω ελέγχουμε αν είναι ήδη σε κατάσταση προσθήκης/διόρθωσης και αν υπάρχουν αντικείμενα

Member στη λίστα. Αν ο έλεγχος επιστρέψει True τότε το command θα είναι ενεργό αλλιώς ανενεργό. Αυτό επηρεάζει το UI που συνδέεται με αυτό το command (π.χ. ένα button να είναι enabled/disabled).

Να σημειώσουμε ότι, στην υλοποίηση των Execute και CanExecute μπορούμε να περάσουμε επιπρόσθετη πληροφορία μέσω της παραμέτρου **parameter**. Για παράδειγμα, σε ένα button θα μπορούσαμε να θέσουμε στην ιδιότητα **CommandParameter** την τιμή που θέλουμε:

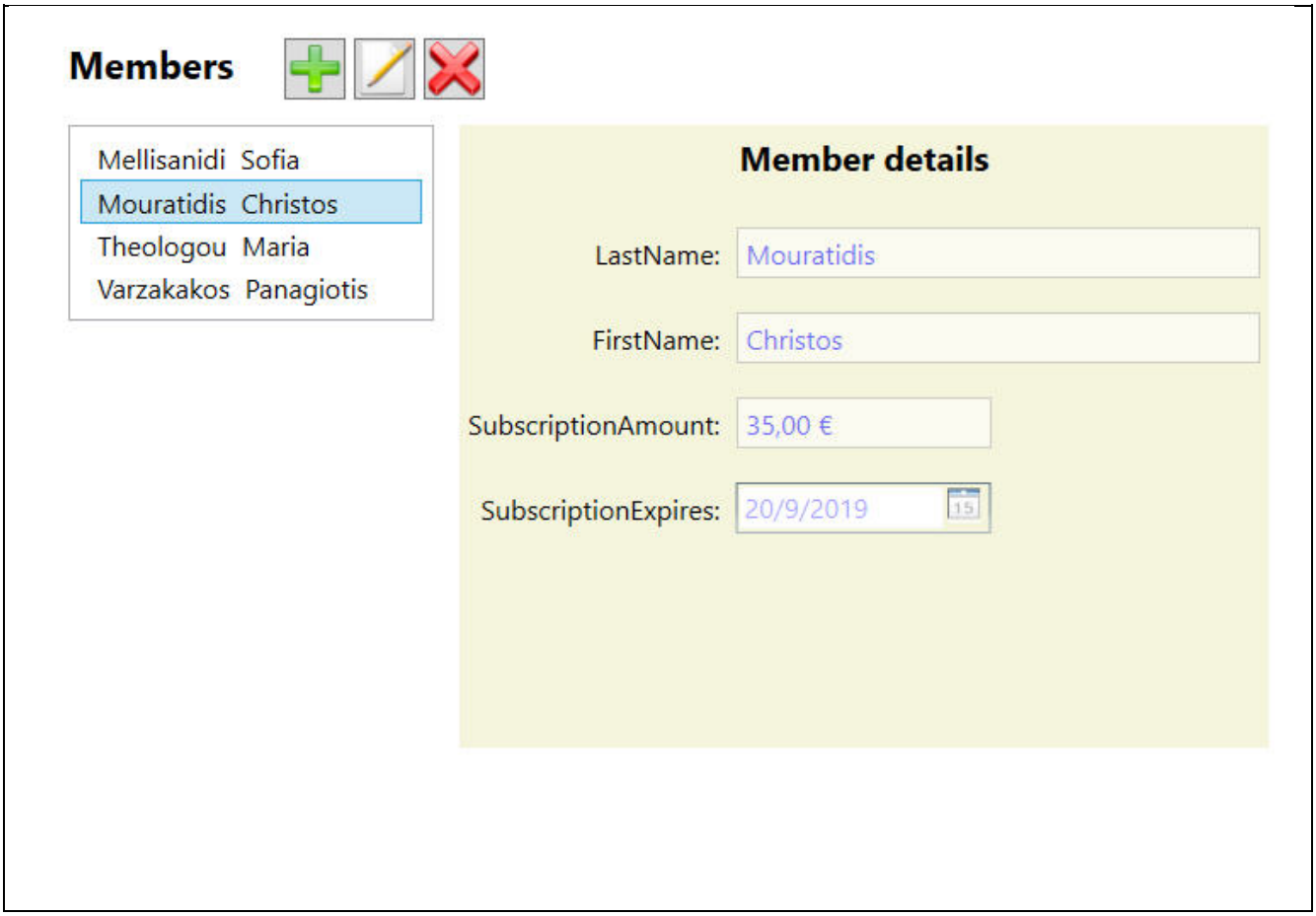
```
<Button Command="{Binding EditMemberCommand }" CommandParameter="Edit"
        ToolTip="Edit a member" ... />
```

- Τέλος, υπάρχουν **2 ιδιότητες** που λειτουργούν ως **μεταβλητές-σημαίες**:
  - η **IsEditMode**, καθορίζει **αν η φόρμα βρίσκεται σε κατάσταση προσθήκης/διόρθωσης ενός μέλους**. Η τιμή αυτής, επηρεάζει κάποια UI elements στο View (για παράδειγμα, αν είναι True, τότε η λίστα των μελών γίνεται disabled μέχρι να ολοκληρωθεί με save ή cancel η καταχώρηση των στοιχείων του μέλους).
  - η **HasErrors**, καθορίζει **αν η φόρμα περιέχει λάθη**. Αν είναι True, τότε δεν επιτρέπεται η αποθήκευση μέχρι να διορθωθεί το λάθος. Για παράδειγμα, αν έχει δοθεί λανθασμένη τιμή στο πεδίο SubscriptionAmount τότε η HasError θα είναι True.

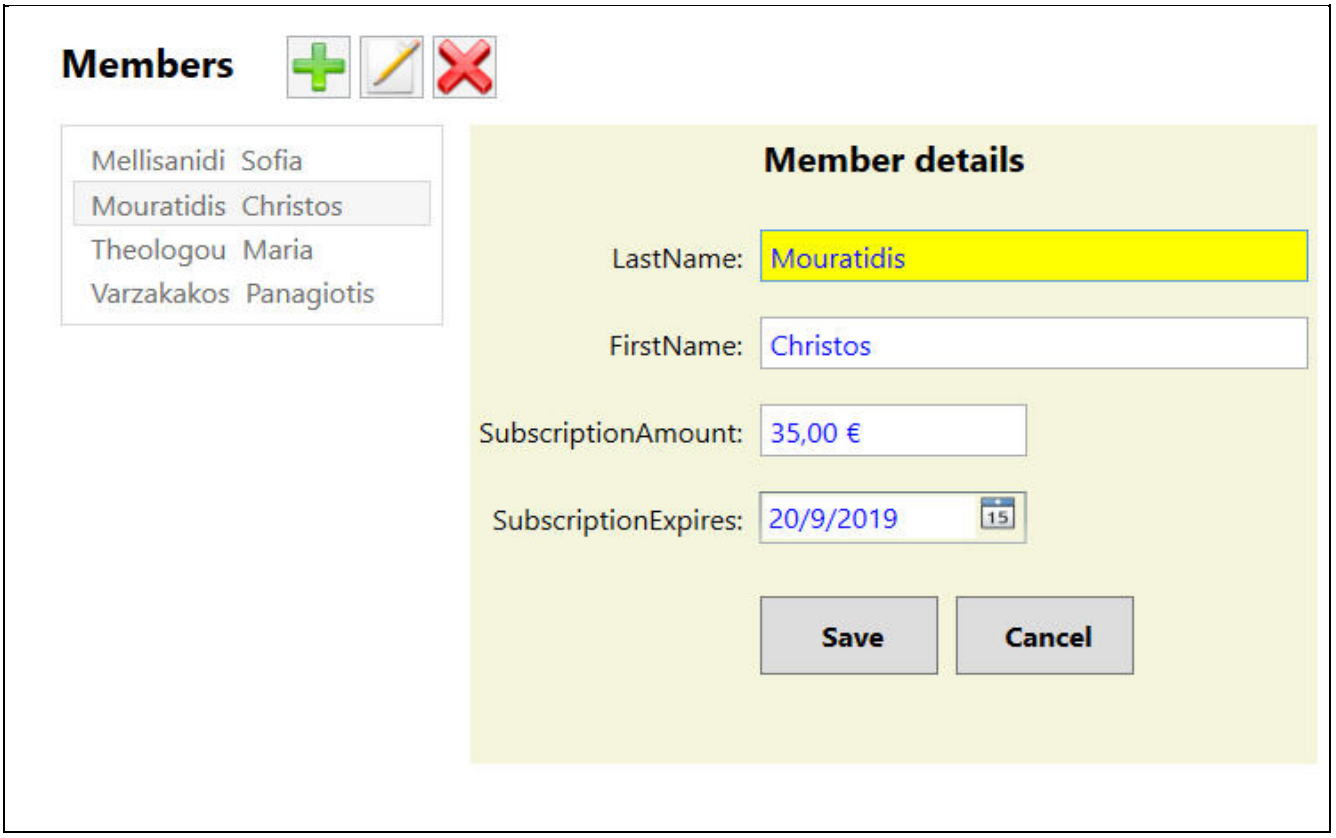
Η κλάση MembersView

Η **κλάση MembersView** αποτελεί το View κομμάτι για την διαχείριση των μελών. Πρόκειται για ένα παράθυρο, δηλαδή ένα αντικείμενο **Window**.

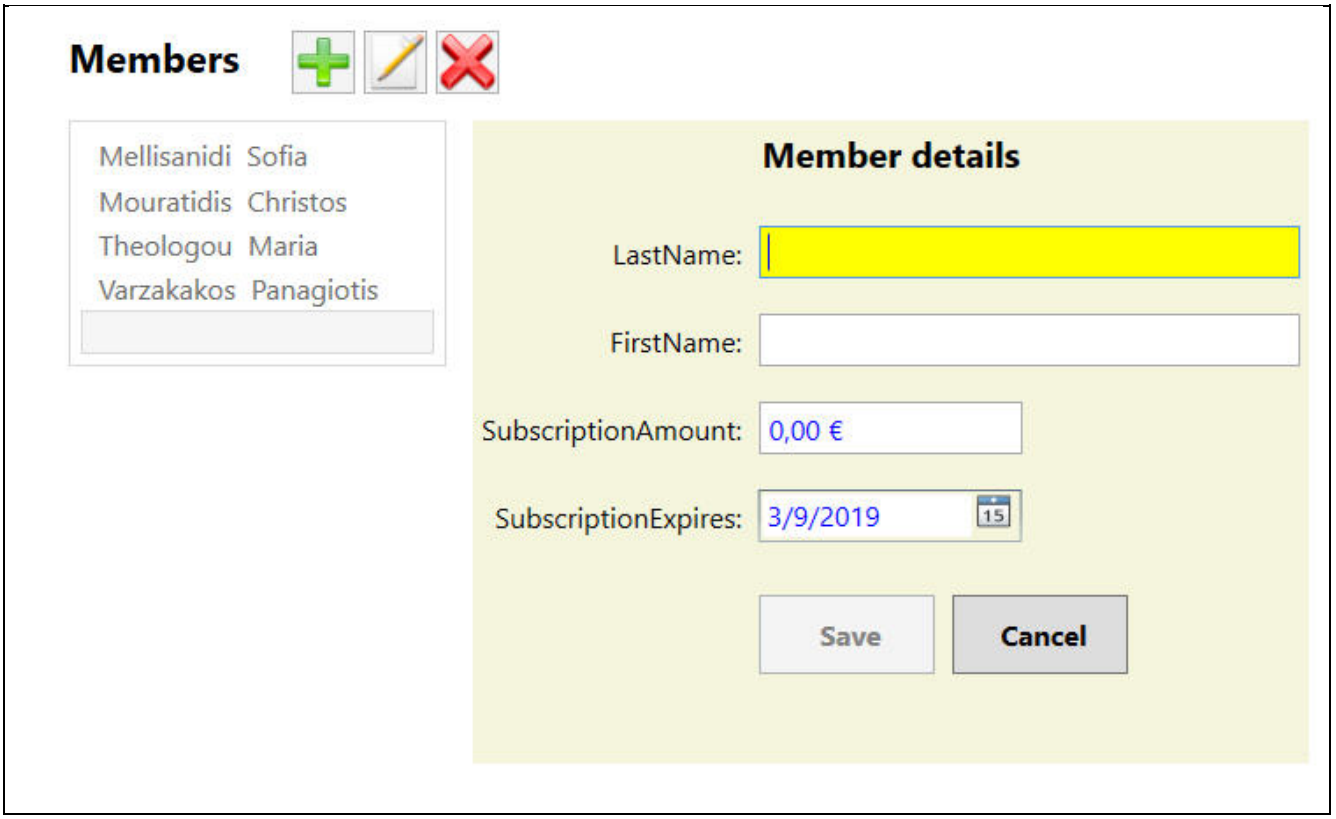
Το παράθυρο αυτό χωρίζεται σε δύο τμήματα: Αριστερά, μία λίστα των καταχωρημένων μελών και δεξιά, τις λεπτομέρειες του επιλεγμένου μέλους. Στο πάνω μέρος, δίπλα στην επικεφαλίδα τοποθετούμε **τρία buttons για την προσθήκη, διόρθωση και διαγραφή μελών. Αυτά τα buttons θα συνδεθούν με τα RelayCommands του ViewModel**.



Όταν πατήσουμε το **button "Edit"** (το εικονίδιο με το μολύβι) τότε η φόρμα μπαίνει σε **edit mode** και ταυτόχρονα εμφανίζονται και τα **buttons "Save"** και **"Cancel"**:

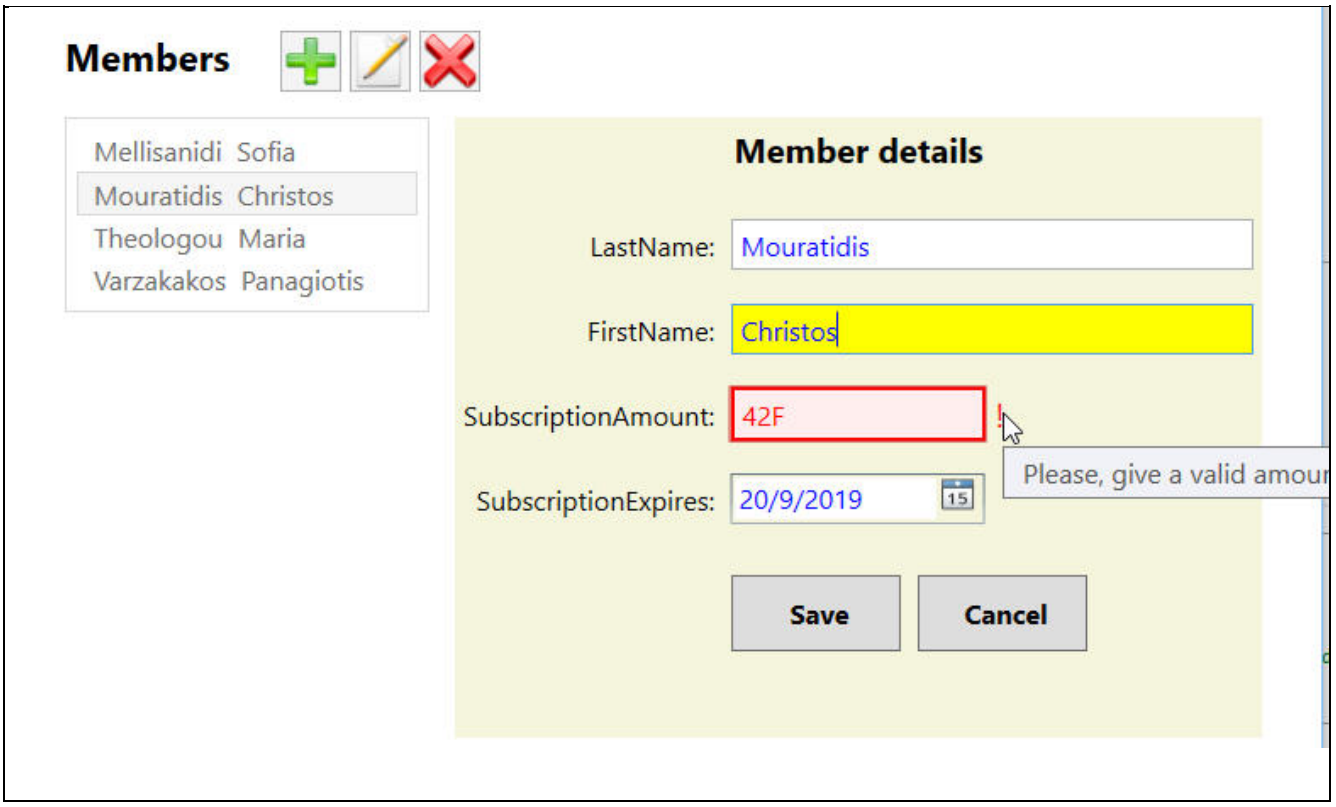


Όταν πατήσουμε το **button "Add"** (το εικονίδιο με το +) τότε η φόρμα μπαίνει σε **edit mode,** τα πεδία είναι κενά ή έχουν τις default τιμές και ταυτόχρονα εμφανίζονται και τα **buttons "Save"** και **"Cancel"**:



Τα **buttons "Save"** και **"Cancel" συνδέονται με αντίστοιχα RelayCommands του ViewModel**. Όσον αφορά το button "Save", έχουμε ορίσει στην CanExecute μέθοδο του αντίστοιχου RelayCommand, την κατάλληλη συνθήκη που ορίζει πότε είναι ενεργό. Στην προκειμένη περίπτωση πρέπει το πεδίο LastName να έχει τουλάχιστον έναν χαρακτήρα. Αν η συνθήκη επιστρέψει False τότε το button "Save" θα είναι αυτόματα disabled.

Επίσης, **αν ο χρήστης δώσει λάθος στο πεδίο SubscriptionAmount** τότε η φόρμα μπαίνει σε **κατάσταση λάθους** και το πεδίο κοκκινίζει με ένα θαυμαστικό δεξιά που όταν ο δείκτης του ποντικιού ίπταται σε αυτό εμφανίζει ένα σχετικό μήνυμα:



Ο χειριστής όλων αυτών των καταστάσεων είναι το ViewModel.

Παραθέτουμε τον κώδικα του **MembersView.xaml.** Τα styles που χρησιμοποιούμε έχουν οριστεί στο Application.xaml και δεν μας ενδιαφέρουν εδώ να τα παρουσιάσουμε. Τα σημεία ενδιαφέροντος για το RelayCommand είναι υπερφωτισμένα:

```
<Window x:Class="MembersView"
        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"
        xmlns:p="clr-namespace:RelayCommand_Sample_Project"
        mc:Ignorable="d"
        Title="RelayCommand Sample Project"
        Height="511.348" Width="792.489" Loaded="Window_Loaded">

    <!-- For global resources (styles etc) goto the Application.xaml -->

    <Window.Resources >

        <!-- The data context for the view.-->
        <p:MembersViewModel x:Key="membersViewModel" />

        <!-- ValueConverters-->
        <p:BooleanToVisibilityConverter x:Key="booleanToVisibilityConverter"/>
        <p:InvertBooleanConverter x:Key="invertBooleanConverter"/>
```

```
    </Window.Resources>

    <!-- Connect keys with specific commands of the viewmodel.-->
    <Window.InputBindings >

        <KeyBinding Key="A" Modifiers="Ctrl"
                    Command="{Binding Source={StaticResource membersViewModel},
                    Path=AddMemberCommand}"/>

        <KeyBinding Key="E" Modifiers="Ctrl"
                    Command="{Binding Source={StaticResource membersViewModel},
                    Path=EditMemberCommand}"/>

        <KeyBinding Key="D" Modifiers="Ctrl"
                    Command="{Binding Source={StaticResource membersViewModel},
                    Path=DeleteMemberCommand}"/>

    </Window.InputBindings>

    <!-- Content.-->
    <Grid Margin="20" DataContext="{StaticResource membersViewModel}">

        <Grid.ColumnDefinitions >
            <ColumnDefinition Width="230"/>
            <ColumnDefinition Width="*"/>
        </Grid.ColumnDefinitions>

        <Grid.RowDefinitions >
            <RowDefinition  Height="Auto"/>
            <RowDefinition  Height="*"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>

        <!-- HEADER -->
        <StackPanel Grid.ColumnSpan="2" Margin="15,0,0,0" Orientation="Horizontal" >

            <!-- Header text -->
            <TextBlock  Text="Members" Style="{StaticResource h1Style}" />

            <!-- Action buttons. Add/Edit/Delete member-->
            <StackPanel Orientation="Horizontal" Margin="30,0,0,0">

                <Button Command="{Binding AddMemberCommand }"  ToolTip="Add a member">
                    <Image Source="/ZeusRelayCommand Sample
                                        Project;component/Images/Add_32x32.png"/>
                </Button>

                <Button Command="{Binding EditMemberCommand }"
                        ToolTip="Edit the selected member" Margin="5,0,0,0">
                    <Image Source="/ZeusRelayCommand Sample
                                        Project;component/Images/Edit_32x32.png"/>
                </Button>

                <Button Command="{Binding DeleteMemberCommand }"
                        ToolTip="Delete the selected member" Margin="5,0,0,0">
                    <Image Source="/ZeusRelayCommand Sample
                                    Project;component/Images/Delete_32x32.png"/>
                </Button>

            </StackPanel>

        </StackPanel>
```

```
<!-- 1st Column -->

<!-- The member's list.-->
<StackPanel Grid.Row="1" Margin="0,15,0,0"  >

    <ListBox Name="lstMembers" IsSynchronizedWithCurrentItem="True" Padding="5"
             Margin="15,0,0,0"
             ItemsSource="{Binding MembersListCollectionView}"
             IsEnabled="{Binding IsEditMode, Mode=OneWay,
                         Converter={StaticResource invertBooleanConverter }}">
    </ListBox>

</StackPanel>

<!-- 2nd Column - member details.-->
<Grid Name="grdMemberDetails" Grid.Row="1" Grid.Column="1" Margin="15"
      Background="Beige"
      DataContext="{Binding ElementName=lstMembers, Path=SelectedItem}"
      Language="el">

    <Grid.ColumnDefinitions >
        <ColumnDefinition Width="Auto" />
        <ColumnDefinition Width="*"/>
    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions >
        <RowDefinition Height="50"/>
        <RowDefinition Height="50"/>
        <RowDefinition Height="50"/>
        <RowDefinition Height="50"/>
        <RowDefinition Height="50"/>
        <RowDefinition Height="85"/>
    </Grid.RowDefinitions>

    <!-- header section -->
    <TextBlock Grid.ColumnSpan="2" Text="Member details"
               HorizontalAlignment="Center" Margin="0,5,0,0"
               Style="{StaticResource h2Style }"/>

    <!-- Here, we could use a usercontrol for the member details. We choose on-site.-->

    <!-- LastName -->
    <TextBlock Grid.Row="1" Text="LastName: " />
    <TextBox Name="txtLastName" Grid.Row="1" Grid.Column="1"
            Text="{Binding LastName, UpdateSourceTrigger=PropertyChanged}"
             IsEnabled="{Binding Source={StaticResource membersViewModel },
                                 Path=IsEditMode, Mode=OneWay}" />

    <!-- FirstName -->
    <TextBlock Grid.Row="2" Text="FirstName: " />
    <TextBox Grid.Row="2" Grid.Column="1" Text="{Binding FirstName}"
             IsEnabled="{Binding Source={StaticResource membersViewModel },
                                 Path=IsEditMode, Mode=OneWay}" />

    <!-- SubscriptionAmount -->
    <TextBlock Grid.Row="3" Text="SubscriptionAmount: " />
    <TextBox Name="txtSubscriptionAmount" Grid.Row="3" Grid.Column="1"
             IsEnabled="{Binding Source={StaticResource membersViewModel },
                                 Path=IsEditMode, Mode=OneWay}"
             Style="{StaticResource NumericTextBoxStyle }"
             Validation.ErrorTemplate="{StaticResource validationErrorTemplate }"
             Validation.Error="txtSubscriptionAmount_Error"  >
```

```
                <Binding Path="SubscriptionAmount" StringFormat="{}{0:C2}"
                         NotifyOnValidationError="True" >
                    <Binding.ValidationRules >
                        <p:ValidateDecimal ErrorMessage="Please, give a valid amount"/>
                    </Binding.ValidationRules>
                </Binding>
            </TextBox>

            <!-- SubscriptionExpires -->
            <TextBlock Grid.Row="4" Text="SubscriptionExpires: " />
            <DatePicker  Grid.Row="4" Grid.Column="1"
                         SelectedDate="{Binding SubscriptionExpires}"
                         IsEnabled="{Binding Source={StaticResource membersViewModel },
                                             Path=IsEditMode, Mode=OneWay}"
                         Height="30">
            </DatePicker>

            <!-- The Save and Cancel buttons. They are vidible only when the members details
                 secion is in EditMode. -->
            <StackPanel Name="stkSaveAndCancelButtons" Grid.Column="1" Grid.Row="5"
                        Orientation="Horizontal"
                        Margin="5,20" TextElement.FontSize="16"
                        TextElement.FontWeight="Bold"
                        Visibility="{Binding Source={StaticResource membersViewModel },
                                             Path=IsEditMode,
                                    Converter={StaticResource booleanToVisibilityConverter},
                                    ConverterParameter=Hidden}">

                <StackPanel.Resources >

                    <!-- The style for the inside buttons-->
                    <Style TargetType="{x:Type Button}">
                        <Setter Property="FontSize" Value="16"/>
                        <Setter Property="FontWeight" Value="Bold"/>
                        <Setter Property="Padding" Value="10"/>
                        <Setter Property="Width" Value="100"/>
                    </Style>

                </StackPanel.Resources>

                <Button Name="btnSave" Content="Save"
                        Command="{Binding Source={StaticResource membersViewModel},
                                          Path=SaveMemberDetailsCommand}"  />

                <Button Name="btnCancel" Content="Cancel"  Margin="10,0,0,0"
                        Command="{Binding Source={StaticResource membersViewModel},
                                          Path=CancelEditMemberDetailsCommand }"/>

            </StackPanel>

        </Grid>

    </Grid>

</Window>
```

Παρατηρήσεις:

- Ορίζουμε ως resource το **αντικείμενο membersViewModel**. Αυτό παρακάτω θα χρησιμοποιηθεί ως **data context** για το **Grid** και οι διάφορες ιδιότητές του θα γίνουν binding

με τις ιδιότητες μερικών UI elements. Για παράδειγμα, το **ListBox** με τη λίστα των μελών, έχει την **ιδιότητα IsEnabled** που γίνεται **binding** με την **ιδιότητα IsEditMode** του **membersViewModel**. Μάλιστα, όταν η IsEditMode είναι True (κατάσταση προσθήκης/διόρθωσης) θέλουμε ή IsEnabled να είναι False (το ListBox μη ενεργό). Γι' αυτό το λόγο, χρησιμοποιούμε επιπλέον κι ένα **ValueConverter** με όνομα **invertBooleanConverter,** τον κώδικα του οποίου παραθέτουμε παρακάτω.

- Συνδέουμε τα **AddMemberCommand**, **EditMemberCommand** και **DeleteMemberCommand** με συγκεκριμένα **πλήκτρα συντόμευσης**. Να σημειώσουμε, όμως, ότι τα πλήκτρα ενεργοποιούνται όταν το παράθυρο έχει την εστίαση.

- Συνδέουμε τα **AddMemberCommand**, **EditMemberCommand, DeleteMemberCommand, SaveMemberDetailsCommands** και **CancelMemberDetailsCommand** με τα σχετικά buttons του παραθύρου, μέσω έκφρασης **binding**. Ο κώδικας υλοποίησής τους βρίσκεται στο MembersViewModel.vb.

Ο κώδικας του MembersView.xaml.vb:

```
Class MembersView

    Private membersViewModel As MembersViewModel

    'errors counter for the validation purpose.
    Private errorsCountOnSubscriptionAmount As Integer

    Private Sub Window_Loaded(sender As Object, e As RoutedEventArgs)

        'Get the view model.
        membersViewModel = CType(Me.FindResource("membersViewModel"), MembersViewModel)

        'and create some sample data.
        membersViewModel.InitializeSomeMembersData()

        'Define that the first element in View is the txtLastnmae.
        'This is useful for focus purpose at the beginning on add/edit proccess.
        membersViewModel.FirstViewElement = txtLastName

        'Move to the first member.
        membersViewModel.MoveToFirstMember()

    End Sub

    'Check for errors in the subscription amount.
    Private Sub txtSubscriptionAmount_Error(sender As Object, _
                                            e As ValidationErrorEventArgs)

        Dim txtBox As TextBox = TryCast(sender, TextBox)

        If txtBox IsNot Nothing Then

            'The standard mechanism for checking the field errors.
            errorsCountOnSubscriptionAmount = _
                                If(e.Action = ValidationErrorEventAction.Added, _
                                           errorsCountOnSubscriptionAmount + 1, _
                                           errorsCountOnSubscriptionAmount - 1)
```

```
            membersViewModel.HasErrors = If(errorsCountOnSubscriptionAmount = 0, _
                                                                    False, True)

        End If

    End Sub

End Class
```

ValidationRule

Στο **MembersView** χρησιμοποιούμε ένα ValidationRule στο **πεδίο SubscriptionAmount**. Αυτό κάνει έναν τυπικό έλεγχο για το αν εισάγεται μία έγκυρη τιμή τύπου Decimal. Έχει εισαχθεί στο project σε ένα **αρχείο κλάσης** με **όνομα ValidateDecimal.vb** και ο κώδικας είναι ο εξής:

```
Imports System.Globalization

''' <summary>
''' Validates a string object. Valid if it contains a decimal number. The string object cannot
                                                                                    be empty.
''' </summary>
Public Class ValidateDecimal
                Inherits ValidationRule

    ''' <summary>
    ''' An error message when the string value doesn't contain a decimal number.
    ''' </summary>
    ''' <returns>A String message when the validation fails, otherwise nothing.</returns>
    Public Property ErrorMessage As String = "Incorrect value."

    Public Overrides Function Validate(value As Object, cultureInfo As CultureInfo)
                              As ValidationResult

        Dim isValid As Boolean = False

        If value IsNot Nothing AndAlso value.ToString.Trim.Length > 0 Then

            Try
                Dim val As Decimal = Decimal.Parse(value, NumberStyles.Any)
                isValid = True

            Catch
            End Try

        End If

        Return If(isValid, New ValidationResult(True, Nothing), _
                                            New ValidationResult(False, ErrorMessage))

    End Function

End Class
```

Η χρήση του στο MembersView.xaml:

```
        <TextBox Name="txtSubscriptionAmount" Grid.Row="3" Grid.Column="1"
                    IsEnabled="{Binding Source={StaticResource membersViewModel },
                                        Path=IsEditMode, Mode=OneWay}"
                    Style="{StaticResource NumericTextBoxStyle }"
                    Validation.ErrorTemplate="{StaticResource validationErrorTemplate }"
                    Validation.Error="txtSubscriptionAmount_Error"  >

                <Binding Path="SubscriptionAmount" StringFormat="{}{0:C2}"
                         NotifyOnValidationError="True"   >
                    <Binding.ValidationRules >
```

```
                    <p:ValidateDecimal ErrorMessage="Please, give a valid amount"/>
                </Binding.ValidationRules>
            </Binding>

        </TextBox>
```

ValueConverters

Στο **MembersView** χρησιμοποιούμε δύο **ValueConverters**, τα οποία μάλιστα απαιτούνται πολύ συχνά στις εφαρμογές μας όπου έχουμε binding συνδέσεις.

- Το **BooleanToVisibilityConverter**, το οποίο **μετατρέπει μία τιμή Boolean σε μία τιμή Visibility**. Χρησιμοποιείται στην περίπτωση που έχουμε ένα σενάριο όπου ανάλογα με το αν η source τιμή είναι True τότε το element στο View να εμφανίζεται αλλιώς όχι. Μάλιστα στην κλάση που παραθέτουμε πιο κάτω, μπορούμε να περάσουμε ως παράμετρο την τιμή "Hiddden" αν θέλουμε το element να κρυφτεί (με δέσμευση, όμως, του χώρου που καταλαμβάνει) αλλιώς θα γίνει collapsed.

```
<StackPanel Name="stkSaveAndCancelButtons" Grid.Column="1" Grid.Row="5"
            Orientation="Horizontal" Margin="5,20"
            TextElement.FontSize="16" TextElement.FontWeight="Bold"
            Visibility="{Binding Source={StaticResource membersViewModel },
                                 Path=IsEditMode,
                        Converter={StaticResource booleanToVisibilityConverter},
                        ConverterParameter=Hidden}">
```

- Το **InvertBooleanConverter**, το οποίο **αναστρέφει μία Boolean τιμή**. Στην περίπτωσή μας, η λίστα των μελών (ListBox) απενεργοποιείται όταν η φόρμα των λεπτομερειών μέλους μπαίνει σε κατάσταση προσθήκης/διαγραφής. Τεχνικά μιλώντας, όταν η ιδιότητα IsEditMode του αντικειμένου MembersViewModel γίνεται True τότε η ιδιότητα IsEnabled του ListBox γίνεται False.

```
<ListBox Name="lstMembers" IsSynchronizedWithCurrentItem="True" Padding="5"
         Margin="15,0,0,0"
        ItemsSource="{Binding MembersListCollectionView}"
        IsEnabled="{Binding IsEditMode, Mode=OneWay,
                            Converter={StaticResource invertBooleanConverter }}">
```

Παραθέτουμε τους **VB κώδικες των κλάσεων** που βρίσκονται στα ομώνυμα **αρχεία BooleanToVisibilityConverter.vb** και **InvertBooleanConverter.vb**.

```
<ValueConversion(GetType(Boolean), GetType(Visibility))>
Public Class BooleanToVisibilityConverter
                        Implements IValueConverter

    Public Function Convert(ByVal value As Object, ByVal targetType As System.Type, _
                            ByVal parameter  As Object, _
                            ByVal culture As System.Globalization.CultureInfo) As Object _
                                   Implements System.Windows.Data.IValueConverter.Convert

        Try

            Dim boolValue As Nullable(Of Boolean) = System.Convert.ToBoolean(value)

            If boolValue Is Nothing Then
                Return Nothing
            Else
                If boolValue = True Then
```

```
                    Return Visibility.Visible
                Else
                    If parameter IsNot Nothing Then
                        Return If(parameter.ToString = "Hidden", Visibility.Hidden, _
                                                                 Visibility.Collapsed)
                    Else
                        Return Visibility.Collapsed
                    End If

                End If
            End If

        Catch ex As Exception
            Return Nothing
        End Try

    End Function


    Public Function ConvertBack(value As Object, targetType As Type, parameter As Object, _
                                culture As Globalization.CultureInfo) As Object _
                                                    Implements IValueConverter.ConvertBack

        Throw New NotSupportedException ()

    End Function

End Class


'-------------------------------------------------------------------------------------------

<ValueConversion(GetType(Boolean), GetType(Boolean))>
Public Class InvertBooleanConverter
                     Implements IValueConverter

    Public Function Convert(ByVal value As Object, ByVal targetType As System.Type, _
                            ByVal parameter  As Object, _
                            ByVal culture As System.Globalization.CultureInfo) As Object _
                                    Implements System.Windows.Data.IValueConverter.Convert

        Try
            Dim booleanValue As Boolean = System.Convert.ToBoolean(value)

            'Reverse.
            Return Not booleanValue

        Catch ex As Exception
            Return True
        End Try

    End Function


    Public Function ConvertBack(value As Object, targetType As Type, parameter As Object, _
                                culture As Globalization.CultureInfo) As Object _
                                                Implements IValueConverter.ConvertBack

        Throw New NotSupportedException ()

    End Function
```

```
End Class
```