Zeus ButtonControls Ver. 1.0

Manual

Πίνακας περιεχομένων

ZEUS BUTTONCONTROLS VERSION 1.0 1
Ο ΟΡΙΣΜΟΣ ΤΗΣ ΚΛΑΣΗΣ 2
ΔΙΑΝΟΜΗ 3
ΕΠΙΚΟΙΝΩΝΙΑ 4
IMAGEBUTTON 5
ΑΠΑΡΙΘΜΗΣΕΙΣ 8
ImagePositionEnum *9*
ΙΔΙΟΤΗΤΕΣ 10
CornerRadius *11*
ImageHeight *12*
ImagePosition *14*
ImageSource *15*
ImageWidth *16*
ShowImage *18*
ShowText *20*
SpaceBetween *22*
Text *24*
STYLES ΚΑΙ TEMPLATES 25
Parts και States *26*
Το default Style και ControlTemplate *27*
Ένα προσαρμοσμένο Style και ControlTemplate *31*
ΤΕΛΟΣ MANUAL 35

Zeus ButtonControls version 1.0

Κλάσεις: ImageButton

Inherits: System.Windows.Controls.Button

Namespace: Zeus.WPF.Controls.ButtonControls

Assembly: ZeusButtonControls (in ZeusButtonsControls.dll)

Dependencies: -

Περιγραφή

Η βιβλιοθήκη **Zeus ButtonsControls** περιλαμβάνει, προς το παρόν, το **ImageButton** control που είναι ένα κλασσικό button που περιλαμβάνει **Εικόνα** (ImageSource) και **Κείμενο** (Text). Μπορούμε να προσδιορίσουμε τη θέση της εικόνας (αριστερά (default), πάνω, δεξιά η κάτω) σε σχέση με το κείμενο (ImagePosition). Επίσης, να θέσουμε το μέγεθος της εικόνας (ImageWidth, ImageHeight) καθώς και την απόσταση μεταξύ της εικόνας και του κειμένου (SpaceBetween). Αν επιθυμούμε, μπορούμε να ορίσουμε στρογγυλεμένες άκρες (CornerRadius)(*εμμέσως, δημιουργούμε και στρογγυλά buttons*). Τέλος, υπάρχει η δυνατότητα να εμφανίσουμε ή όχι την εικόνα ή/και το κείμενο (ShowImage, ShowText).

Να σημειώσουμε ότι, το **ImageButton** έχει προκαθορισμένο background για την MouseOver και Pressed κατάσταση, παρόμοια όπως και το κλασσικό button. Αν θέλουμε να προσδιορίσουμε διαφορετικό background σε αυτές τις καταστάσεις θα πρέπει να φτιάξουμε ένα custom Style με συγκεκριμένο ControlTemplate. Στο τέλος της ενότητας του ImageButton δίνουμε ένα παράδειγμα για το πόσο εύκολα μπορεί να γίνει και να βασιστείτε πάνω σε αυτό για να δημιουργήσετε παρεμφερή custom styles.

Παρακάτω, παρατίθενται τα **ButtonControls**.

	ImageButton	Ένα button που έχει ως περιεχόμενο μία εικόνα και ένα κείμενο.

Ο ορισμός της κλάσης

Η κλάση έχει οριστεί ως εξής:

- Για την **κλάση ImageButton** :

  Σύνταξη:

  VB:

```
<TemplateVisualState(Name:="Normal", GroupName:="CommonStates"),
TemplateVisualState(Name:="MouseOver", GroupName:="CommonStates"),
TemplateVisualState(Name:="Pressed", GroupName:="CommonStates"),
TemplateVisualState(Name:="Disabled", GroupName:="CommonStates"),
TemplateVisualState(Name:="Focused", GroupName:="FocusStates"),
TemplateVisualState(Name:="Unfocused", GroupName:="FocusStates"),
TemplatePart(Name:="PART_Container", Type:=GetType(StackPanel)),
TemplatePart(Name:="PART_Image", Type:=GetType(Image)),
TemplatePart(Name:="PART_Text", Type:=GetType(TextBlock))>
Public Class ImageButton
    Inherits Button
```

  XAML Object Element Usage:

  Εισαγωγή namespace:

```
xmlns:zeus="clr-namespace:Zeus.WPF.Controls.ButtonControls;assembly=ZeusButtonContr
ols"
```

  Χρήση:

```
<zeus:ImageButton ... />
```

Διανομή

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

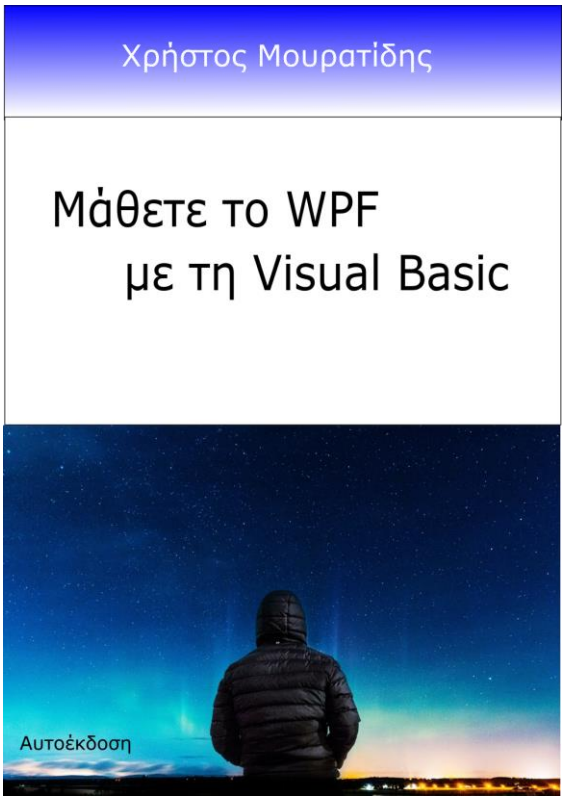
Επικοινωνία

Για οποιαδήποτε πληροφορία ή διευκρίνηση παρακαλώ επικοινωνήστε στο :

mouratx@yahoo.com ή mouratx@hotmail.com

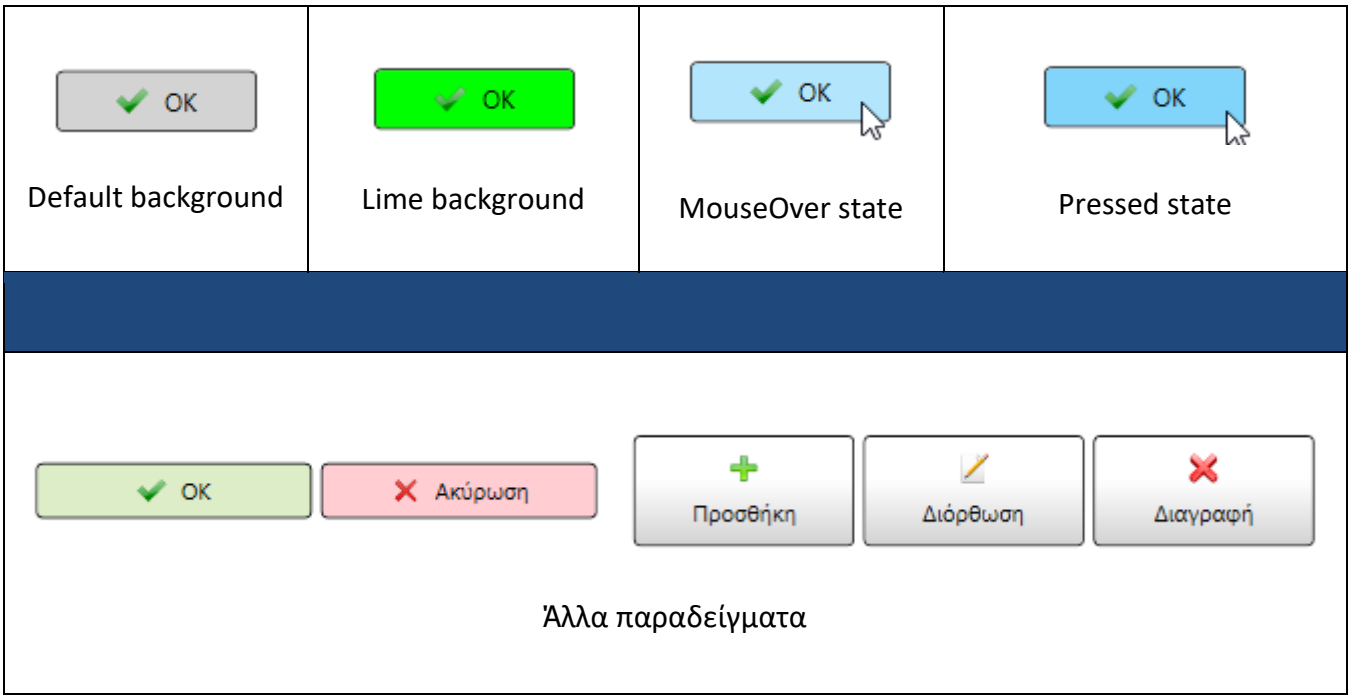
Χρήστος Μουρατίδης,
Πειραιάς, Μάιος 2020

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).**



ImageButton

Ένα **Button control που περιέχει εικόνα και κείμενο**.



Σύνταξη:

VB:

```
<TemplateVisualState(Name:="Normal", GroupName:="CommonStates"),
TemplateVisualState(Name:="MouseOver", GroupName:="CommonStates"),
TemplateVisualState(Name:="Pressed", GroupName:="CommonStates"),
TemplateVisualState(Name:="Disabled", GroupName:="CommonStates"),
TemplateVisualState(Name:="Focused", GroupName:="FocusStates"),
TemplateVisualState(Name:="Unfocused", GroupName:="FocusStates"),
TemplatePart(Name:="PART_Container", Type:=GetType(StackPanel)),
TemplatePart(Name:="PART_Image", Type:=GetType(Image)),
TemplatePart(Name:="PART_Text", Type:=GetType(TextBlock))>
Public Class ImageButton
    Inherits Button
```

XAML Object Element Usage:

Εισαγωγή namespace:

```
xmlns:zeus="clr-namespace:Zeus.WPF.Controls.ButtonControls;assembly=ZeusButtonContr
ols"
```

Χρήση:

```
<zeus:ImageButton ... />
```

Παράδειγμα:

Στο παράδειγμα που ακολουθεί, τοποθετούμε το δύο **ImageButtons** που λειτουργούν ως **dialog buttons OK** και **Ακύρωση**. Έχουμε, επίσης, δημιουργήσει και δύο custom styles ειδικά για OK και **Cancel** buttons, ώστε να είναι ομοιόμορφα παντού στην εφαρμογή μας. Να σημειώσουμε ότι στα Window.Resources έχουμε, επίσης, δημιουργήσει και δύο custom styles ειδικά για OK και Cancel buttons. Στον υποφάκελο Images της εφαρμογής υπάρχουν τα αρχεία εικόνων (τσεκ και x images).

XAML:

```
<Window x:Class="MainWindow"
  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:zeus="clr-namespace:Zeus.WPF.Controls.ButtonControls;assembly=ZeusButtonCont
                                                                          rols"
  mc:Ignorable="d"
  Title="Button Controls Sample Project" Height="500.52" Width="425.78"
   WindowStartupLocation="CenterScreen"
   Loaded="Window_Loaded" >

<Window.Resources>

    <!-- Dialog Buttons Base  Style -->
    <Style x:Key="DialogImageButtonStyle" TargetType="{x:Type zeus:ImageButton}">

        <Setter Property="Width" Value="150" />
        <Setter Property="Height" Value="30" />

        <Style.Triggers>
            <Trigger Property="IsMouseOver" Value="True">
                <Setter Property="FontWeight" Value="DemiBold" />
                <Setter Property="BorderThickness" Value="1,1,1,1.5" />
            </Trigger>
        </Style.Triggers>

    </Style>

    <!-- Dialog  OK Style -->
    <Style x:Key="DialogOKButtonStyle" TargetType="{x:Type zeus:ImageButton}"
           BasedOn="{StaticResource DialogImageButtonStyle}">

        <Setter Property="ImageSource"
             Value="/ButtonControls Sample Project;component/Images/OK_32x25.png" />
        <Setter Property="Background" Value="#DCEDC8" />
```

```
        </Style>

        <!-- Dialog Cancel Style -->
        <Style x:Key="DialogCancelButtonStyle" TargetType="{x:Type zeus:ImageButton}"
               BasedOn="{StaticResource DialogImageButtonStyle}">

            <Setter Property="ImageSource"
           Value="/ButtonControls Sample Project;component/Images/Cancel_32x32.png" />
            <Setter Property="Background" Value="#ffcdd2" />
        </Style>

    </Window.Resources>

    <StackPanel Orientation="Horizontal" TextElement.FontSize="12" Margin="5">

           <zeus:ImageButton Text="OK" Style="{StaticResource DialogOKButtonStyle}" />
           <zeus:ImageButton Text="Ακύρωση" Margin="5,0,0,0"
                     Style="{StaticResource DialogCancelButtonStyle}"/>

    </StackPanel>

</Window>
```

 ImageButtons	 MouseOver στο OK button.

Απαριθμήσεις

Όνομα	Περιγραφή
ImagePositionEnum	Περιλαμβάνει τις **τιμές για την θέση της εικόνας μέσα στο button.**

ImagePositionEnum

Περιλαμβάνει τις **τιμές για την θέση της εικόνας μέσα στο button.**

Σύνταξη:

VB:

```
Public Enum ImagePositionEnum
```

Μέλη:

Τιμή	Περιγραφή
Left	Αριστερά από το κείμενο.
Top	Πάνω από το κείμενο.
Right	Δεξιά από το κείμενο.
Bottom	Κάτω από το κείμενο.

Ιδιότητες

Όνομα	Περιγραφή
CornerRadius	Καθορίζει τη **στρογγυλότητα των άκρων**, **τύπου ConrerRadius**.
ImageHeight	Καθορίζει το **ύψος της εικόνας, τύπου Double.**
ImagePosition	Καθορίζει τη **θέση της εικόνας σε σχέση με το κείμενο, τύπου ImagePositionEnum.**
ImageSource	Καθορίζει την **πηγή της εικόνας, τύπου ImageSource.**
ImageWidth	Καθορίζει το **πλάτος της εικόνας, τύπου Double.**
ShowImage	Καθορίζει **αν θα εμφανίζεται η εικόνα. Είναι τύπου Boolean.**
ShowText	Καθορίζει **αν θα εμφανίζεται το κείμενο. Είναι τύπου Boolean.**
SpaceBetween	Καθορίζει την **απόσταση μεταξύ εικόνας και κειμένου, τύπου Double.**
Text	Καθορίζει το **κείμενο, τύπου String.**

CornerRadius

Καθορίζει τη **στρογγυλότητα των άκρων**, **τύπου ConrerRadius**.

Σύνταξη:

VB:

```
Public Property CornerRadius As CornerRadius
```

Τύπος: System.Windows.CornerRadius

Προσδιορίζουμε μία τιμή CornerRadius που θα καθορίζει πόσο στρογγυλεμένες θα είναι οι άκρες του button. Η default τιμή είναι "3" (ομοιόμορφα σε όλες τις γωνίες).

Dependency Property Information:

Identifier field: `CornerRadiusProperty`

Παράδειγμα:

Στο επόμενο παράδειγμα, καθορίζουμε μία αρκετά στρογγυλή τιμή στο ImageButton:

XAML:

```
<zeus:ImageButton Name="btnOK" Text="OK"
                  Style="{StaticResource DialogOKButtonStyle}"
                  CornerRadius="20" />
```

VB:

```
Imports Zeus.WPF.Controls.ButtonControls

btnOK.CornerRadius = New CornerRadius(20)
```

ImageHeight

Καθορίζει το **ύψος της εικόνας, τύπου Double.**

Σύνταξη:

VB:

```
Public Property ImageHeight As Double
```

Τύπος: System.Double

Προσδιορίζουμε μία τιμή Double για το ύψος της εικόνας του button. Η default τιμή είναι 16 (pixels).

Dependency Property Information:

Identifier field: `ImageHeightProperty`

Παρατηρήσεις:

Ίσως χρειαστεί να μεταβάλλουμε το ύψος ή/και το πλάτος (Height / Width) του ImageButton για να δούμε την επίδραση της ImageHeight.

Παράδειγμα:

Στο επόμενο παράδειγμα, καθορίζουμε το πλάτος και ύψος του ImageButton σε 32x32 pixels.

XAML:

```
<zeus:ImageButton Name="btnOK" Text="OK"
                  Style="{StaticResource DialogOKButtonStyle}"
                  ImageHeight="32" ImageWidth="32"
                  Height="50" FontSize="22"
                  SpaceBetween="12"/>
```

VB:

```
Imports Zeus.WPF.Controls.ButtonControls

btnOK.ImageHeight = 32
btnOK.ImageWidth = 32
```

 16x16	 32x32

ImagePosition

Καθορίζει τη **θέση της εικόνας σε σχέση με το κείμενο, τύπου ImagePositionEnum.**

Σύνταξη:

VB:

```
Public Property ImagePosition As ImagePositionEnum
```

Τύπος: Zeus.WPF.Controls.ButtonControls.ImageButton.ImagePositionEnum

Προσδιορίζουμε μία τιμή από την απαρίθμηση ImagePositionEnum για την θέση της εικόνας, σε σχέση με το κείμενο, μέσα στο button.
Η default τιμή είναι ImagePositionEnum.Left.

Dependency Property Information:

Identifier field: `ImagePositionProperty`

Παρατηρήσεις:

Ίσως χρειαστεί να μεταβάλλουμε το ύψος ή/και το πλάτος (Height / Width) του ImageButton για να δούμε την επίδραση της ImagePosition.

Παράδειγμα:

Στο επόμενο παράδειγμα, καθορίζουμε ότι η εικόνα θα είναι πάνω από το κείμενο:

XAML:

```
<zeus:ImageButton Name="btnAdd" Text="Προσθήκη"
                  ImageSource="Images/Add_32x32.png"
                  ImagePosition="Top"
                  Style="{StaticResource EditingButtonStyle}" />
```

VB:

```
Imports Zeus.WPF.Controls.ButtonControls

btnAdd.ImagePosition = ImageButton.ImagePositionEnum.Top
```

ImageSource

Καθορίζει την **πηγή της εικόνας, τύπου ImageSource.**

Σύνταξη:

VB:

```
Public Property ImageSource As ImageSource
```

Τύπος: System.Windows.Media.ImageSource
Προσδιορίζουμε ένα αντικείμενο ImageSource για την πηγή της εικόνας.
Η default τιμή είναι η εικόνα  .

Dependency Property Information:

Identifier field: `ImageSourceProperty`

Παράδειγμα:

Στο επόμενο παράδειγμα, καθορίζουμε ως πηγή της εικόνας το αρχείο **Add_32x32.png**, το οποίο βρίσκεται ήδη στον υποφάκελο **Images** της εφαρμογής:

XAML:

```
<zeus:ImageButton Name="btnAdd" Text="Προσθήκη"
                  ImageSource="Images/Add_32x32.png"
                  ImagePosition="Top"
                  Style="{StaticResource EditingButtonStyle}" />
```

VB:

```
Imports Zeus.WPF.Controls.ButtonControls

btnAdd.ImageSource = New BitmapImage(New Uri("/Images/Add_32x32.png", _
                                             UriKind.Relative))
```

ImageWidth

Καθορίζει το **πλάτος της εικόνας, τύπου Double.**

Σύνταξη:

VB:

```
Public Property ImageWidth As Double
```

Τύπος: System.Double

Προσδιορίζουμε μία τιμή Double για το πλάτος της εικόνας του button. Η default τιμή είναι 16 (pixels).

Dependency Property Information:

Identifier field: `ImageWidthProperty`

Παρατηρήσεις:

Ίσως χρειαστεί να μεταβάλλουμε το ύψος ή/και το πλάτος (Height / Width) του ImageButton για να δούμε την επίδραση της ImageWidth.

Παράδειγμα:

Στο επόμενο παράδειγμα, καθορίζουμε το πλάτος και ύψος του ImageButton σε 32x32 pixels.

XAML:

```
<zeus:ImageButton Name="btnOK" Text="OK"
                  Style="{StaticResource DialogOKButtonStyle}"
                  ImageHeight="32" ImageWidth="32"
                  Height="50" FontSize="22"
                  SpaceBetween="12"/>
```

VB:

```
Imports Zeus.WPF.Controls.ButtonControls

btnOK.ImageHeight = 32
btnOK.ImageWidth = 32
```

 16x16	 32x32

ShowImage

Καθορίζει **αν θα εμφανίζεται η εικόνα. Είναι τύπου Boolean.**

Σύνταξη:

VB:

```
Public Property ShowImage As Boolean
```

Τύπος: System.Boolean

Προσδιορίζουμε μία τιμή Boolean που καθορίζει αν θα εμφανίζεται ή όχι η εικόνα στο button. Η default τιμή είναι True.

Dependency Property Information:

Identifier field: `ShowImageProperty`

Παρατηρήσεις:

Ίσως χρειαστεί να μεταβάλλουμε το ύψος ή/και το πλάτος (Height / Width) του ImageButton για να δούμε την επίδραση της ShowImage. Επίσης, αν καθορίζουμε τιμή False τότε το κείμενο κεντράρεται αυτόματα οριζόντια και κατακόρυφα στον χώρο του button και δεν λαμβάνεται υπόψη η ιδιότητα SpaceBetween.

Παράδειγμα:

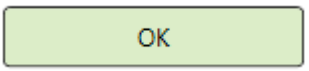
Στο επόμενο παράδειγμα, καθορίζουμε ότι **δεν** θα εμφανίζεται η εικόνα στο ImageButton.

XAML:

```
<zeus:ImageButton Name="btnOK" Text="OK"
                  Style="{StaticResource DialogOKButtonStyle}"
                  ShowImage="False"
```

VB:

```
Imports Zeus.WPF.Controls.ButtonControls

btnOK.ShowImage = False
```

 ShowImage = True	 ShowImage = False

ShowText

Καθορίζει **αν θα εμφανίζεται το κείμενο. Είναι τύπου Boolean.**

Σύνταξη:

VB:

```
Public Property ShowText As Boolean
```

Τύπος: System.Boolean

Προσδιορίζουμε μία τιμή Boolean που καθορίζει αν θα εμφανίζεται ή όχι το κείμενο στο button. Η default τιμή είναι True.

Dependency Property Information:

Identifier field: `ShowTextProperty`

Παρατηρήσεις:

Ίσως χρειαστεί να μεταβάλλουμε το ύψος ή/και το πλάτος (Height / Width) του ImageButton για να δούμε την επίδραση της ShowText.

Παράδειγμα:

Στο επόμενο παράδειγμα, καθορίζουμε ότι δεν θα εμφανίζεται το κείμενο στο ImageButton.

XAML:

```
<zeus:ImageButton Name="btnAdd" Text="Προσθήκη"
                  ImageSource="Images/Add_32x32.png"
                  ImagePosition="Top"
                  ShowText ="False"
                  Width="40" Height="40"
                  Style="{StaticResource EditingButtonStyle}" />
```

VB:

```
Imports Zeus.WPF.Controls.ButtonControls

btnOK.ShowText = False
```

 ShowText = True	 ShowText = False *(με προσαρμογή της Width και Height)*

SpaceBetween

Καθορίζει την **απόσταση μεταξύ εικόνας και κειμένου, τύπου Double.**

Σύνταξη:

VB:

```
Public Property SpaceBetween As Double
```

Τύπος: System.Double

Προσδιορίζουμε μία τιμή Double που καθορίζει την απόσταση μεταξύ της εικόνας και κειμένου. Η default τιμή είναι 8 (pixels).

Dependency Property Information:

Identifier field: `SpaceBetweenProperty`

Παρατηρήσεις:

Ίσως χρειαστεί να μεταβάλλουμε το ύψος ή/και το πλάτος (Height / Width) του ImageButton για να δούμε την επίδραση της SpaceBetween.

Παράδειγμα:

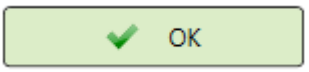
Στο επόμενο παράδειγμα, καθορίζουμε την απόσταση σε 15 pixels στα δύο ImageButtons , όπου στο ένα η εικόνα είναι αριστερά και στο άλλο η εικόνα είναι πάνω:

XAML:

```
<zeus:ImageButton Name="btnOK" Text="OK"
                  Style="{StaticResource DialogOKButtonStyle}"
                  SpaceBetween="15" />

<zeus:ImageButton Name="btnAdd" Text="Προσθήκη"
                  ImageSource="Images/Add_32x32.png" ImagePosition="Top"
                  Style="{StaticResource EditingButtonStyle}"
                  SpaceBetween="15"  />
```

VB:

```
Imports Zeus.WPF.Controls.ButtonControls

btnOK.SpaceBetween = 15
btnAdd.SpaceBetween = 15
```

 SpaceBetween = 8	 SpaceBetween = 15
 SpaceBetween = 8	 SpaceBetween = 15

Text

Καθορίζει το **κείμενο, τύπου String.**

Σύνταξη:

VB:

```
Public Property Text As String
```

Τύπος: System.String

Προσδιορίζουμε μία τιμή String που αποτελεί το κείμενο (caption) του button.
Η default τιμή "Text".

Dependency Property Information:

Identifier field: `TextProperty`

Παρατηρήσεις:

Ίσως χρειαστεί να μεταβάλλουμε το ύψος ή/και το πλάτος (Height / Width) του ImageButton για να δούμε την επίδραση της Text.

Παράδειγμα:

Στο επόμενο παράδειγμα, καθορίζουμε το κείμενο "OK" για το ImageButton:

XAML:

```
<zeus:ImageButton Name="btnOK" Text="OK"
                  Style="{StaticResource DialogOKButtonStyle}" />
```

VB:

```
Imports Zeus.WPF.Controls.ButtonControls

btnOK.Text = "OK"
```

Styles και Templates

- Parts και States
- Το default Style και ControlTemplate
- Ένα προσαρμοσμένο Style και ControlTemplate

Parts και States

Το default ControlTemplate περιλαμβάνει κάποια **part names** και **visual states**. Μπορείτε να τροποποιήσετε το default ControlTemplate ώστε να δώσετε στο control μία μοναδική εμφάνιση.

Ο παρακάτω πίνακας εμφανίζει τα **part names** του **ImageButton**:

Part	Τύπος	Περιγραφή
PART_Container	StackPanel	Το layout container panel, το οποίο περιέχει τα PART_Image και PART_Text.
PART_Image	Image	Το Image control, το οποίο περιέχει την εικόνα.
PART_Text	TextBlock	Το TextBlock control, το οποίο περιέχει το κείμενο.

Ο παρακάτω πίνακας εμφανίζει τα **visual states** του **ImageButton**:

VisualState	VisualStateGroup	Περιγραφή
Normal	CommonStates	Το default state.
MouseOver	CommonStates	Το state όταν ο δείκτης του ποντικιού είναι πάνω από το control.
Pressed	CommonStates	Το state όταν γίνεται κλικ στο control.
Disabled	CommonStates	Το state όταν το control είναι disabled.
Focused	FocusStates	Το state όταν το control έχει το focus.
Unfocused	FocusStates	Το state όταν το control δεν έχει το focus.

Το default ControlTemplate έχει καθορισμένο μόνο το Disabled state. Μπορείτε να δημιουργήσετε ένα custom ControlTemplate για να το αλλάξετε ή/και για να καθορίσετε τα υπόλοιπα.

Το default Style και ControlTemplate

Ο XAML κώδικας για το **default Style** και **ControlTemplate** φαίνεται παρακάτω. Μπορείτε να βασιστείτε σε αυτόν για να δημιουργήσετε μία μικρή ή μεγάλη παραλλαγή του δικού σας custom Style και ControlTemplate:

```
xmlns:z="clr-namespace:Zeus.WPF.Controls.ButtonControls"

  <!--============================================================ -->
  <!-- Default Style for ImageButton -->
  <!--============================================================ -->
  <Style TargetType="{x:Type z:ImageButton}" BasedOn="{StaticResource {x:Type Button}}">

        <!-- Resources -->
        <Style.Resources>

            <z:BooleanToVisibilityConverter x:Key="booleanToVisibilityConverter" />

            <!-- Default Style for Image -->
            <Style x:Key="ButtonImageDefaultStyle" TargetType="{x:Type Image }">

                <Setter Property="Source"
                    Value="/ZeusButtonControls;component/Images/SampleImage_32x32.png"/>
                <Setter Property="Width" Value="16"/>
                <Setter Property="Height" Value="16"/>

            </Style>

            <!-- Default Style for TextBlock -->
            <Style x:Key="TextDefaultStyle" TargetType="{x:Type TextBlock }">

                <Setter Property="Margin" Value="8,0,0,0"/>

            </Style>

            <!-- Focus rectangle style documentation color: #60000000-->
            <Style x:Key="ButtonFocusVisualStyle">

                <Setter Property="Control.Template">

                    <Setter.Value>

                        <ControlTemplate>
                            <Border>
                                <Rectangle Margin="2"   StrokeThickness="1.2"
                                           Stroke="Black"  StrokeDashArray="1 2" />
                            </Border>
                        </ControlTemplate>

                    </Setter.Value>

                </Setter>

            </Style>

        </Style.Resources>
```

```
<!-- Setters -->
<Setter Property="Background" Value="LightGray"/>
<Setter Property="BorderBrush" Value="Black"/>
<Setter Property="BorderThickness" Value="0.9"/>
<Setter Property="FocusVisualStyle"
                              Value="{StaticResource ButtonFocusVisualStyle}"/>

<!-- ControlTemplate -->
<Setter Property="Template">

    <Setter.Value>

        <ControlTemplate TargetType="{x:Type z:ImageButton}">

            <Border Name="border" Background="{TemplateBinding Background}"
                    BorderBrush="{TemplateBinding BorderBrush}"
                    BorderThickness="{TemplateBinding BorderThickness}"
                    Padding="{TemplateBinding Padding}"
                    CornerRadius="{TemplateBinding CornerRadius}" >

                <!-- Visual States -->
                <VisualStateManager.VisualStateGroups >

                    <VisualStateGroup Name="CommonStates">

                        <VisualState Name="Normal"/>
                        <VisualState Name="MouseOver" />
                        <VisualState Name="Pressed"/>
                        <VisualState Name="Disabled">
                            <Storyboard >
                                <DoubleAnimation Storyboard.TargetName="border"
                                                 Storyboard.TargetProperty="Opacity"
                                                       To="0.4" Duration="0:0:0.1"/>
                            </Storyboard >
                        </VisualState>

                    </VisualStateGroup>

                    <VisualStateGroup Name="FocusStates">

                        <VisualState Name="Focused" />
                        <VisualState Name="Unfocused" />

                    </VisualStateGroup>

                </VisualStateManager.VisualStateGroups>

                <!-- Content -->
                <Grid Background="Transparent">

                    <StackPanel Name="PART_Container" Orientation="Horizontal"
                                IsEnabled="{TemplateBinding IsEnabled }"
                                Background="Transparent"
                                HorizontalAlignment="Center"
                                VerticalAlignment="Center">

                        <Image Name="PART_Image"  VerticalAlignment="Center"
                               HorizontalAlignment="Center"
                               Style="{StaticResource ButtonImageDefaultStyle}"
                               Source="{TemplateBinding ImageSource }"
                               Width="{TemplateBinding ImageWidth}"
```

```
                                    Height="{TemplateBinding ImageHeight}"
                                   Visibility="{TemplateBinding ShowImage,
                                Converter={StaticResource booleanToVisibilityConverter}}"/>

                                  <TextBlock Name="PART_Text" VerticalAlignment="Center"
                                             HorizontalAlignment="Center"
                                            Text="{TemplateBinding Text}"
                                            Style="{StaticResource TextDefaultStyle}"
                                           Visibility="{TemplateBinding ShowText,
                                Converter={StaticResource booleanToVisibilityConverter}}"/>

                                </StackPanel>

                            </Grid>

                        </Border>

                        <!-- ControlTemplate Triggers -->
                        <ControlTemplate.Triggers>

                            <Trigger Property="IsMouseOver" Value="True">
                                <Setter Property="Background" TargetName="border"
                                        Value="#B3E5FC"/>  <!-- something like light blue -->
                            </Trigger>

                            <Trigger Property="IsPressed" Value="True">
                                <Setter Property="Background" TargetName="border"
                                     Value="#81D4FA"/>  <!-- something like stronger blue -->
                            </Trigger>

                        </ControlTemplate.Triggers>

                    </ControlTemplate>

                </Setter.Value>

            </Setter>

        </Style>
```

- Να σημειώσουμε ότι το default Style χρησιμοποιεί:

    1. Μία default εικόνα όταν αρχικοποιείται, η οποία προέρχεται από το αρχείο **SampleImage_32x32.png** που βρίσκεται στον υποφάκελο **Images**:  . Εσείς μπορείτε να προσθέσετε μία δική σας αρχική εικόνα.

    2. Ένα **custom ValueConverter** για την **ιδιότητα Visibility** των controls, το οποίο μετατρέπει την τιμή True σε Visibility.Visible και την False σε Visibility.Collapsed. Το παραθέτουμε παρακάτω:

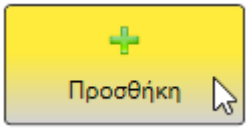
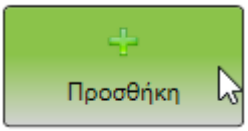
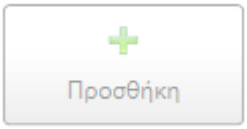
```
<ValueConversion(GetType(Boolean), GetType(Visibility))>
Friend Class BooleanToVisibilityConverter
    Implements IValueConverter

    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
```

Ένα προσαρμοσμένο Style και ControlTemplate

Παρακάτω, δημιουργούμε ένα ελαφρώς **προσαρμοσμένο style**, με όνομα **EditingButtonStyle** όπου **"πειράζουμε" τα visual states MouseOver και Pressed**. Τα υπόλοιπα παραμένουν ως έχουν στο default ControlTemplate.

Βλέπουμε δείγματα ενός τέτοιου ImageButton σε διάφορα states:

			
Normal state	MouseOver state	Pressed state	Disabled state

Ο XAML κώδικας για το style είναι ο εξής:

```
xmlns:z="clr-namespace:Zeus.WPF.Controls.ButtonControls"

 <!-- Editing Buttons Base Style -->

<Style x:Key="EditingButtonStyle" TargetType="{x:Type zeus:ImageButton}"
       BasedOn="{StaticResource {x:Type Button}}">

        <!-- Resources -->
        <Style.Resources>

            <p:BooleanToVisibilityConverter x:Key="booleanToVisibilityConverter"/>

            <!-- Default Background Brush for Border Background-->
            <LinearGradientBrush x:Key="BrushBackground"
                                 StartPoint="0.5,0" EndPoint="0.5,1">
                <GradientStop Color="White" Offset="0.4"/>
                <GradientStop Color="LightGray" Offset="1"/>
            </LinearGradientBrush>

            <!-- Default Style for Image -->
            <Style x:Key="ButtonImageDefaultStyle" TargetType="{x:Type Image }">

                <Setter Property="Width" Value="16"/>
                <Setter Property="Height" Value="16"/>

            </Style>

            <!-- Default Style for TextBlock -->
            <Style x:Key="TextDefaultStyle" TargetType="{x:Type TextBlock }">

                <Setter Property="Margin" Value="8,0,0,0"/>

            </Style>

            <!-- Focus rectangle style documentation color: #60000000-->
```

```
        <Style x:Key="ButtonFocusVisualStyle">

            <Setter Property="Control.Template">

                <Setter.Value>

                    <ControlTemplate>
                        <Border>
                            <Rectangle Margin="2"   StrokeThickness="1.2"
                                       Stroke="Black"  StrokeDashArray="1 2" />
                        </Border>
                    </ControlTemplate>

                </Setter.Value>

            </Setter>

        </Style>

    </Style.Resources>

    <!-- Setters -->
    <Setter Property="Background" Value="{StaticResource BrushBackground}"/>
    <Setter Property="BorderBrush" Value="Black"/>
    <Setter Property="BorderThickness" Value="0.9"/>
    <Setter Property="FocusVisualStyle"
                               Value="{StaticResource ButtonFocusVisualStyle}"/>
    <Setter Property="Width" Value ="120"/>
    <Setter Property="Height" Value="60"/>

    <!-- ControlTemplate -->
    <Setter Property="Template">

        <Setter.Value>

            <ControlTemplate TargetType="{x:Type zeus:ImageButton}">

                <Border Name="border" Background="{TemplateBinding Background}"
                        BorderBrush="{TemplateBinding BorderBrush}"
                        BorderThickness="{TemplateBinding BorderThickness}"
                        Padding="{TemplateBinding Padding}"
                        CornerRadius="{TemplateBinding CornerRadius}" >

                    <!-- Visual States -->
                    <VisualStateManager.VisualStateGroups >

                        <VisualStateGroup Name="CommonStates">

                            <VisualState Name="Normal"/>

                            <VisualState Name="MouseOver" >

                                <Storyboard>

                                    <ColorAnimation Storyboard.TargetName="border"
                                Storyboard.TargetProperty="Background.GradientStops[0].Color"
                                To="#FFEB3B" Duration="0:0:0" />    <!-- something like yellow -->

                                </Storyboard>

                            </VisualState>
```

```
                        <VisualState Name="Pressed">

                            <Storyboard>

                                <ColorAnimation Storyboard.TargetName="border"
                    Storyboard.TargetProperty="Background.GradientStops[0].Color"
                      To="#8BC34A" Duration="0:0:0"/> <!-- something like green-->

                            </Storyboard>

                        </VisualState>

                        <VisualState Name="Disabled">

                            <Storyboard >
                                <DoubleAnimation Storyboard.TargetName="border"
                                            Storyboard.TargetProperty="Opacity"
                                                  To="0.4" Duration="0:0:0.1"/>
                            </Storyboard >
                        </VisualState>

                    </VisualStateGroup>

                    <VisualStateGroup Name="FocusStates">

                        <VisualState Name="Focused" />
                        <VisualState Name="Unfocused"/>

                    </VisualStateGroup>

                </VisualStateManager.VisualStateGroups>

                <!-- Content -->
                <Grid Background="Transparent">

                    <StackPanel Name="PART_Container" Orientation="Horizontal"
                                IsEnabled="{TemplateBinding IsEnabled }"
                                Background="Transparent"
                                 HorizontalAlignment="Center"
                                VerticalAlignment="Center">

                        <Image Name="PART_Image"  VerticalAlignment="Center"
                               HorizontalAlignment="Center"
                               Style="{StaticResource ButtonImageDefaultStyle }"
                               Source="{TemplateBinding ImageSource }"
                               Width="{TemplateBinding ImageWidth}"
                               Height="{TemplateBinding ImageHeight}"
                               Visibility="{TemplateBinding ShowImage,
                    Converter={StaticResource booleanToVisibilityConverter}}"/>

                        <TextBlock Name="PART_Text" VerticalAlignment="Center"
                                   HorizontalAlignment="Center"
                                Text="{TemplateBinding Text}"
                                Style="{StaticResource TextDefaultStyle}"
                                Visibility="{TemplateBinding ShowText,
                      Converter={StaticResource booleanToVisibilityConverter}}"/>

                    </StackPanel>

                </Grid>

            </Border>
```

```
            </ControlTemplate>
        </Setter.Value>
    </Setter>
</Style>
```

Το **custom ValueConverter** για την **ιδιότητα Visibility** των controls, το οποίο μετατρέπει την τιμή True σε Visibility.Visible και την False σε Visibility.Collapsed το έχουμε παραθέσει στο τέλος της προηγούμενης ενότητας.

Τέλος Manual