Zeus ChronoControls Ver. 1.0

Manual

Πίνακας περιεχομένων

Zeus ChronoControls version 1.0

Κλάσεις: Chronometer

Inherits: Για το Chronometer: System.Windows.Controls.TextBlock

Namespace: Zeus.WPF.Controls.ChronoControls

Assembly: ZeusChronoControls (in ZeusChronoControls.dll)

Dependencies: -

Περιγραφή

Το **Zeus ChronoControls** είναι μία βιβλιοθήκη από controls που σχετίζονται με τη διαχείριση χρόνου. Στην παρούσα έκδοση, περιέχεται το **Chronometer**, ένα εξειδικευμένο TextBlock που **εμφανίζει ένα χρονόμετρο αντίστροφης μέτρησης (countdown timer), σε μορφή hh:mm:sss,** το οποίο θα αποτελέσει χρήσιμο control στη δημιουργία εκπαιδευτικών εφαρμογών, quizzes κλπ.

Παρακάτω, παρατίθενται τα **ChronoControls**.

	Chronometer	Εμφανίζει ένα **χρονόμετρο αντίστροφης μέτρησης (countdown timer), σε μορφή hh:mm:sss.**

Ο ορισμός της κλάσης

Η κλάση έχει οριστεί ως εξής:

- Για την **κλάση Chronometer** :

  Σύνταξη:

  VB:

```
Public Class Chronometer
             Inherits TextBlock
```

  XAML Object Element Usage:

  Εισαγωγή namespace:

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

  Χρήση:

```
<zeus:Chronometer ... />
```

Διανομή

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

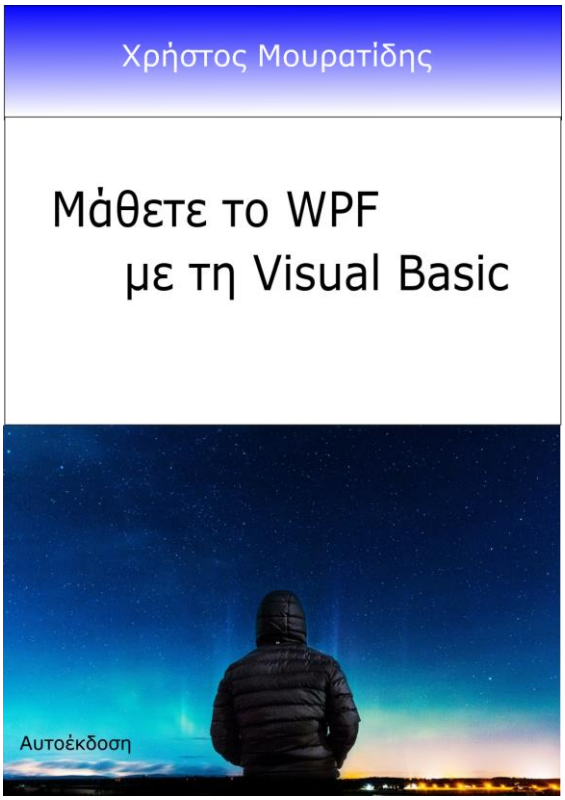
Επικοινωνία

Για οποιαδήποτε πληροφορία ή διευκρίνηση παρακαλώ επικοινωνήστε στο :

mouratx@yahoo.com ή mouratx@hotmail.com

Χρήστος Μουρατίδης,
Πειραιάς, Μάρτιος 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).



Chronometer

Ένα **εξειδεικευμένο TextBlock** που **εμφανίζει ένα χρονόμετρο αντίστροφης μέτρησης (countdown timer), σε μορφή hh:mm:sss.**

Καθορίζουμε έναρξη (ιδιότητα StartTime) και αν επιθυμούμε και CriticalTime σε περίπτωση που θέλουμε να εφιστήσουμε την προσοχή όταν η ώρα ξεπεράσει ένα κρίσιμο όριο. Μέσω ενός custom style μπορούμε να προσδιορίσουμε την εμφάνιση του Chronometer σε normal και critical mode. Όσον αφορά την μορφή, μπορούμε να προσδιορίσουμε αν θα εμφανίζεται η ώρα (μορφή hh:mm:ss) ή όχι (μορφή mm:ss) μέσω της ιδιότητας ShowHour. Το ίδιο μπορούμε να κάνουμε και με τα secs (ιδιότητα ShowSecond).

Δείγματα:

01:30	01:00:00	00:24
Μορφή **mm:ss**	Μορφή **hh:mm:ss**	Όταν έχει φτάσει σε **critical time**.

Σύνταξη:

VB:

```
Public Class Chronometer
             Inherits TextBlock
```

XAML Object Element Usage:

Εισαγωγή namespace:

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

Χρήση:

```
<zeus:Chronometer ... />
```

Παράδειγμα:

Στο παράδειγμα που ακολουθεί, τοποθετούμε το Chronometer control στο πάνω μέρος του παραθύρου. Πιο κάτω, έχουμε βάλει μέσα σε ένα StackPanel μία σειρά από buttons για τον έλεγχο του χρονομέτρου (έναρξης, προσωρινή παύση, συνέχιση και σταμάτημα). Με το πάτημα αυτών καλείται και η αντίστοιχη μέθοδος του Chronometer. Ως **ώρα έναρξης τίθεται η 1:30 (ενάμιση λεπτό) και κρίσιμη ώρα τα 29 secs**. Επίσης, στα **Window.Resources**, έχουμε δημιουργήσει ένα **custom style για το Chronometer** στο οποίο ορίζουμε κάποια χαρακτηριστικά εμφάνισης σε normal mode αλλά και σε critical mode. Ειδικά, στο **critical mode**, **εφαρμόζεται κόκκινο χρώμα και ένα animation που προσομοιώνει το αναβόσβημα (flash)**.

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.ChronoControls;assembly=ZeusChronoContr
ols"
  mc:Ignorable="d"
  Title="ChronoControls Sample Project" Height="279.374" Width="575.938"
 WindowStartupLocation="CenterScreen">

<Window.Resources>

  <!-- Style for Buttons-->
  <Style TargetType="{x:Type Button}">
      <Setter Property="FontSize" Value="18"/>
      <Setter Property="Width" Value="100"/>
      <Setter Property="Padding" Value="10" />
  </Style>

  <!-- Style for Chronometer.
       When is entered to critical time it becomes red and flashes.-->
  <Style  TargetType="{x:Type zeus:Chronometer}">

      <Setter Property="FontSize" Value="32"/>
      <Setter Property="Padding" Value="10" />

      <Style.Triggers>

          <Trigger Property="HasCriticalTimeReached" Value="True">

              <Setter Property="Foreground" Value="Red"/>
              <Trigger.EnterActions>
                  <BeginStoryboard>
                      <Storyboard>
                          <DoubleAnimation Storyboard.TargetProperty="Opacity"
                                           AutoReverse="True"
                                           RepeatBehavior="Forever"
                                           From="0.0" To="1.0" Duration="0:0:0.5"/>
                      </Storyboard>
                  </BeginStoryboard>
              </Trigger.EnterActions>

              <Trigger.ExitActions>
                  <BeginStoryboard>
                      <Storyboard>
                          <DoubleAnimation Storyboard.TargetProperty="Opacity"
```

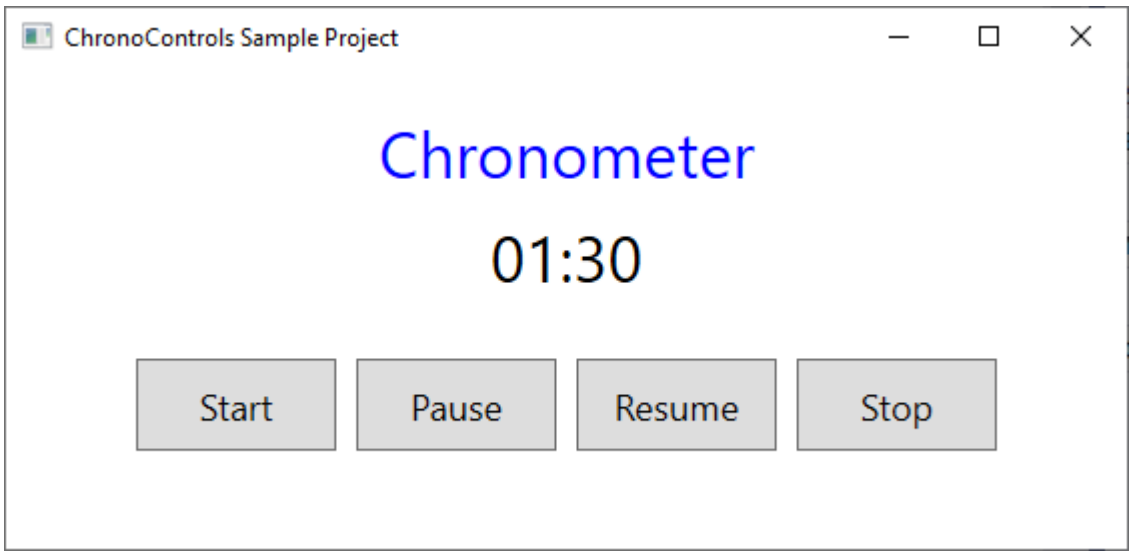
```
                                                To="1.0" Duration="0:0:00"/>
                            </Storyboard>
                        </BeginStoryboard>
                    </Trigger.ExitActions>

                </Trigger>

            </Style.Triggers>

        </Style>

    </Window.Resources>

    <Grid Margin="20">

        <StackPanel HorizontalAlignment="Center">

            <!--Header-->
            <TextBlock Text="Chronometer" FontSize="32" HorizontalAlignment="Center"
                       Foreground="blue" />

            <!-- Chromometer-->
            <zeus:Chronometer Name="chrono" HorizontalAlignment="Center"
                              StartTime="0:1:30"  CriticalTime="00:00:25"
                              TimeChanged="chrono_TimeChanged"
                              CriticalTimeReached="chrono_CriticalTimeReached"
                              TimeEnded="chrono_TimeEnded"/>

            <!-- Buttons -->
            <StackPanel Orientation="Horizontal" Margin="0,20,0,0"
                         ButtonBase.Click="StackPanelButtons_Click" >
                <Button Name="btnStart" Content ="Start"  />
                <Button Name="btnPause" Content ="Pause" Margin="10,0,0,0" />
                <Button Name="btnResume" Content="Resume" Margin="10,0,0,0" />
                <Button Name="btnStop" Content="Stop" Margin="10,0,0,0" />
            </StackPanel>

        </StackPanel>

    </Grid>

</Window>
```

VB:

```
Imports Zeus.WPF.Controls.ChronoControls

Class MainWindow

'When the time changes.
Private Sub chrono_TimeChanged(sender As Object, e As TimeChangedEventArgs)

     'Do something here. e.CurrentTime has the current time.

End Sub

'When critical time is reached.
Private Sub chrono_CriticalTimeReached(sender As Object, e As TimeChangedEventArgs)

     'Do something here. e.CurrentTime has the current time.
```

```
    End Sub

    'When the time ends.
    Private Sub chrono_TimeEnded(sender As Object, e As RoutedEventArgs)

          btnStart.IsEnabled = True

    End Sub

    'When a button is clicked.
    Private Sub StackPanelButtons_Click(sender As Object, e As RoutedEventArgs)

          Dim btn As Button = TryCast(e.OriginalSource, Button)

          If btn IsNot Nothing Then

               If btn Is btnStart Then
                     chrono.StartChrono()
                     btn.IsEnabled = False

               ElseIf btn Is btnPause Then
                     chrono.PauseChrono()

               ElseIf btn Is btnResume Then
                     chrono.ResumeChrono()

               ElseIf btn Is btnStop Then
                     chrono.StopChrono()
                     btn.IsEnabled = True

               End If

           End If

      End Sub


End Class
```

Ένα στιγμιότυπο μπορούμε να δούμε στην παρακάτω εικόνα:


ChronoControls Sample Project
Chronometer
01:30
Start
Pause
Resume
Stop

Ιδιότητες

Όνομα	Περιγραφή
CriticalTime	Καθορίζει την **κρίσιμη ώρα**, δηλαδή την ώρα μετά την οποία χρειάζεται ιδαίτερη προσοχή. Είναι τύπου **TimeSpan**.
HasCriticalTimeReached	**Επιστρέφει True αν το ρολόι έφτασε στην κρίσιμη ώρα**. Είναι τύπου **Boolean**.
ShowHour	Καθορίζει **αν το ρολόι θα εμφανίζει το τμήμα της ώρας**. Είναι τύπου **Boolean**.
ShowSecond	Καθορίζει **αν το ρολόι θα εμφανίζει το τμήμα των δευτερολέπτων**. Είναι τύπου **Boolean**.
StartTime	Καθορίζει την **ώρα εκκίνησης**. Είναι τύπου **TimeSpan**.

CriticalTime

Καθορίζει την **κρίσιμη ώρα**, δηλαδή την ώρα μετά την οποία χρειάζεται ιδιαίτερη προσοχή. Είναι τύπου **TimeSpan**.

Σύνταξη:

VB:

```
Public Property CriticalTime As TimeSpan
```

Τύπος: System.TimeSpan

Προσδιορίζουμε την κρίσιμη ώρα, ως TimeSpan. Η default τιμή είναι TimeSpan(0, 0, 30) η σε Xaml "00:00:30". Δηλαδή 30 secs.

Dependency Property Information:

Identifier field: `CriticalTimeProperty`

Παράδειγμα:

Στο επόμενο παράδειγμα, θέτουμε χρόνο εκκίνησης (StartTime) τα 2 λεπτά και κρίσιμο χρόνο (CriticalTime) τα 20 secs.:

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
chrono.StartTime = New TimeSpan(0,2,0)
chrono.CriticalTime = New TimeSpan(0,0,20)
```

HasCriticalTimeReached

Επιστρέφει True αν το ρολόι έφτασε στην κρίσιμη ώρα. Είναι τύπου **Boolean**.

Σύνταξη:

VB:

```
Public Property HasCriticalTimeReached As Boolean
```

Τύπος: System.Boolean

Επιστρέφει True, αν έφτασε την κρίσιμη ώρα. Η default τιμή False.

Dependency Property Information:

Identifier field: `HasCriticalTimeReachedProperty`

Παρατηρήσεις:

Η **ιδιότητα HasCriticalTimeReached** είναι χρήσιμη για να ορίσουμε ένα **custom style με trigger** ώστε όταν έχει φτάσει στην κρίσιμη ώρα να εφαρμόζεται στο control.

Παράδειγμα:

Στο επόμενο παράδειγμα, ορίζουμε ένα απλό custom style όπου το χρώμα γίνεται κόκκινο όταν το ρολόι φτάσει την κρίσιμη ώρα:

XAML:

```
<!-- Style for Chronometer. When is entered to critical time it becomes red.-->
 <Style  TargetType="{x:Type zeus:Chronometer}">

     <Setter Property="FontSize" Value="32"/>
     <Setter Property="Padding" Value="10" />

     <Style.Triggers>

         <Trigger Property="HasCriticalTimeReached" Value ="True">

             <Setter Property="Foreground" Value="Red"/>

         </Trigger>

     </Style.Triggers>

 </Style>
```

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

ShowHour

Καθορίζει **αν το ρολόι θα εμφανίζει το τμήμα της ώρας**. Είναι τύπου **Boolean**.

Σύνταξη:

VB:

```
Public Property ShowHour As Boolean
```

Τύπος: System.Boolean

Αν θέσουμε True, τότε εμφανίζει το τμήμα της ώρας. Φυσικά, αυτό έχει νόημα αν θέσουμε χρόνο εκκίνησης μεγαλύτερη της ώρας. Η default τιμή False.

Dependency Property Information:

Identifier field: `ShowHourProperty`

Παράδειγμα:

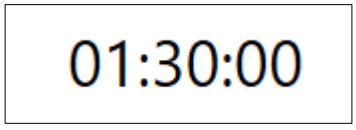
Στο επόμενο παράδειγμα, ορίζουμε χρόνο εκκίνησης την 1,5 ώρα και εμφάνιση του τμήματος της ώρας:

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="1:30:00"  CriticalTime="00:00:20"
                  ShowHour="True"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
chrono.ShowHour = True
```



ShowSecond

Καθορίζει **αν το ρολόι θα εμφανίζει το τμήμα των δευτερολέπτων**. Είναι τύπου **Boolean**.

Σύνταξη:

VB:

```
Public Property ShowSecond  As Boolean
```

Τύπος: System.Boolean

Αν θέσουμε False, τότε δεν εμφανίζει το τμήμα των δευτερολέπτων. Η default τιμή True.

Dependency Property Information:

Identifier field: `ShowSecondProperty`

Παράδειγμα:

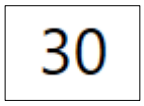
Στο επόμενο παράδειγμα, ορίζουμε χρόνο εκκίνησης τα 30 λεπτά, κρίσιμο χρόνο το 1 λεπτό και όχι εμφάνιση του τμήματος των δευτερολέπτων:

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:30:00"  CriticalTime="00:1:00"
                  ShowSecond="False"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
chrono.ShowSecond = False
```



StartTime

Καθορίζει την **ώρα εκκίνησης**. Είναι τύπου **TimeSpan**.

Σύνταξη:

VB:

```
Public Property StartTime As TimeSpan
```

Τύπος: System.TimeSpan

Προσδιορίζουμε την ώρα έναρξης του χρονομέτρου, ως TimeSpan. Η default τιμή είναι TimeSpan(0, 3, 0) η σε Xaml "00:03:00". Δηλαδή 3 λεπτά.

Dependency Property Information:

Identifier field: `StartTimeProperty`

Παράδειγμα:

Στο επόμενο παράδειγμα, θέτουμε χρόνο εκκίνησης (StartTime) τα 2 λεπτά και κρίσιμο χρόνο (CriticalTime) τα 20 secs.:

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
chrono.StartTime = New TimeSpan(0,2,0)
chrono.CriticalTime = New TimeSpan(0,0,20)
```

Συμβάντα

Όνομα	Περιγραφή
CriticalTimeReached	Συμβαίνει **όταν το ρολόι φτάσει στον κρίσιμο χρόνο.**
TimeChanged	Συμβαίνει **όταν περάσει ένα δευτερόλεπτο.**
TimeEnded	Συμβαίνει **όταν το ρολόι φτάσει στο 0.**
TimeStarted	Συμβαίνει **όταν το ρολόι ξεκινήσει.**

CriticalTimeReached

Συμβαίνει **όταν το ρολόι φτάσει στον κρίσιμο χρόνο.**

Σύνταξη:

VB (ορισμός):

```
Public Custom Event CriticalTimeReached As RoutedEventHandler
```

XAML attribute usage:

```
<zeus:Chronometer CriticalTimeReached ="eventHanlder" ... />
```

Παράδειγμα:

Στο επόμενο παράδειγμα, θέτουμε χρόνο εκκίνησης (StartTime) τα 2 λεπτά και κρίσιμο χρόνο (CriticalTime) τα 20 secs. Επίσης, καθορίζουμε event handlers για τα συμβάντα TimeChanged, CriticalTimeReached και TimeEnded.

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
Imports Zeus.WPF.Controls.ChronoControls

...

 'When the time changes.
 Private Sub chrono_TimeChanged(sender As Object, e As TimeChangedEventArgs)

   'Do something here. e.CurrentTime has the current time.

 End Sub

 'When critical time is reached.
 Private Sub chrono_CriticalTimeReached(sender As Object, e As TimeChangedEventArgs)

   'Do something here. e.CurrentTime has the current time.

 End Sub

 'When the time ends.
 Private Sub chrono_TimeEnded(sender As Object, e As RoutedEventArgs)
```

```
    'Do something here.

End Sub
```

TimeChanged

Συμβαίνει **όταν περάσει ένα δευτερόλεπτο.**

Σύνταξη:

VB (ορισμός):

```
Public Custom Event TimeChanged As RoutedEventHandler
```

XAML attribute usage:

```
<zeus:Chronometer TimeChanged ="eventHanlder" ... />
```

Παράδειγμα:

Στο επόμενο παράδειγμα, θέτουμε χρόνο εκκίνησης (StartTime) τα 2 λεπτά και κρίσιμο χρόνο (CriticalTime) τα 20 secs. Επίσης, καθορίζουμε event handlers για τα συμβάντα TimeChanged, CriticalTimeReached και TimeEnded.

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
Imports Zeus.WPF.Controls.ChronoControls

...

 'When the time changes.
 Private Sub chrono_TimeChanged(sender As Object, e As TimeChangedEventArgs)

   'Do something here. e.CurrentTime has the current time.

 End Sub

 'When critical time is reached.
 Private Sub chrono_CriticalTimeReached(sender As Object, e As TimeChangedEventArgs)

   'Do something here. e.CurrentTime has the current time.

 End Sub
```

```
        'When the time ends.
        Private Sub chrono_TimeEnded(sender As Object, e As RoutedEventArgs)

           'Do something here.

        End Sub
```

TimeEnded

Συμβαίνει **όταν το ρολόι φτάσει στο 0.**

Σύνταξη:

VB (ορισμός):

```
Public Custom Event TimeEnded As RoutedEventHandler
```

XAML attribute usage:

```
<zeus:Chronometer TimeEnded ="eventHanlder" ... />
```

Παράδειγμα:

Στο επόμενο παράδειγμα, θέτουμε χρόνο εκκίνησης (StartTime) τα 2 λεπτά και κρίσιμο χρόνο (CriticalTime) τα 20 secs. Επίσης, καθορίζουμε event handlers για τα συμβάντα TimeChanged, CriticalTimeReached και TimeEnded.

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
Imports Zeus.WPF.Controls.ChronoControls

...

 'When the time changes.
 Private Sub chrono_TimeChanged(sender As Object, e As TimeChangedEventArgs)

   'Do something here. e.CurrentTime has the current time.

 End Sub

 'When critical time is reached.
 Private Sub chrono_CriticalTimeReached(sender As Object, e As TimeChangedEventArgs)

   'Do something here. e.CurrentTime has the current time.

 End Sub
```

```
        'When the time ends.
        Private Sub chrono_TimeEnded(sender As Object, e As RoutedEventArgs)

           'Do something here.

        End Sub
```

TimeStarted

Συμβαίνει **όταν το ρολόι ξεκινήσει.**

Σύνταξη:

VB (ορισμός):

```
Public Custom Event TimeStarted As RoutedEventHandler
```

XAML attribute usage:

```
<zeus:Chronometer TimeStarted ="eventHanlder" ... />
```

Παράδειγμα:

Στο επόμενο παράδειγμα, θέτουμε χρόνο εκκίνησης (StartTime) τα 2 λεπτά και κρίσιμο χρόνο (CriticalTime) τα 20 secs. Επίσης, καθορίζουμε event handlers για τα συμβάντα TimeChanged, CriticalTimeReached και TimeStarted.

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeStarted="chrono_TimeStarted"/>
```

VB:

```
Imports Zeus.WPF.Controls.ChronoControls

...

 'When the time changes.
 Private Sub chrono_TimeChanged(sender As Object, e As TimeChangedEventArgs)

   'Do something here. e.CurrentTime has the current time.

 End Sub

 'When critical time is reached.
 Private Sub chrono_CriticalTimeReached(sender As Object, e As TimeChangedEventArgs)

   'Do something here. e.CurrentTime has the current time.

 End Sub
```

```
        'When the time starts.
        Private Sub chrono_TimeStarted(sender As Object, e As RoutedEventArgs)

           'Do something here.

        End Sub
```

Μέθοδοι

Όνομα	Περιγραφή
PauseChrono()	**Παύει προσωρινά τη λειτουργία του χρονομέτρου.**
ResumeChrono()	**Συνεχίζει τη λειτουργία του χρονομέτρου μετά από παύση.**
StartChrono()	**Ξεκινάει το χρονόμετρο από την αρχή.**
StopChrono()	**Σταματάει το χρονόμετρο και το μηδενίζει.**

PauseChrono

Παύει προσωρινά τη λειτουργία του χρονομέτρου.

Σύνταξη:

VB :

```
Public Sub PauseChrono()
```

Παράδειγμα:

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
chrono.PauseChrono()
```

ResumeChrono

Συνεχίζει τη λειτουργία του χρονομέτρου μετά από παύση.

Σύνταξη:

VB :

```
Public Sub ResumeChrono()
```

Παράδειγμα:

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
chrono.ResumeChrono()
```

StartChrono

Ξεκινάει το χρονόμετρο από την αρχή.

Σύνταξη:

VB :

```
Public Sub StartChrono()
```

Παράδειγμα:

XAML:

```
<zeus:Chronometer Name="chrono"  StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
chrono.StartChrono()
```

StopChrono

Σταματάει το χρονόμετρο και το μηδενίζει.

Σύνταξη:

VB :

```
Public Sub StopChrono()
```

Παράδειγμα:

XAML:

```
<zeus:Chronometer Name="chrono"  StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged"
                  CriticalTimeReached="chrono_CriticalTimeReached"
                  TimeEnded="chrono_TimeEnded"/>
```

VB:

```
chrono.StopChrono()
```

TimeChangedEventArgs

Η κλάση περιέχει τα δεδομένα του RoutedEvent για την τρέχουσα ώρα. Χρησιμοποιείται ένα αντικείμενο αυτής ως δεύτερη παράμετρος στους handlers για τα συμβάντα CriticalTimeReached και TimeChanged.

Σύνταξη:

VB:

```
Public Class TimeChangedEventArgs
                Inherits RoutedEventArgs
```

Ιδιότητες

Όνομα	Περιγραφή
CurrentTime	Επιστρέφει την **τρέχουσα ώρα**. Είναι τύπου **TimeSpan**.

CriticalTime

Επιστρέφει την **τρέχουσα ώρα**. Είναι τύπου **TimeSpan**.

Σύνταξη:

VB:

```
Public Property CurrentTime As TimeSpan
```

Τύπος: System.TimeSpan

Η default τιμή είναι TimeSpan(0, 0, 0).

Dependency Property Information:

No dependency property.

Παράδειγμα:

Στο επόμενο παράδειγμα, θέτουμε για το Chronometer, χρόνο εκκίνησης (StartTime) τα 2 λεπτά και κρίσιμο χρόνο (CriticalTime) τα 20 secs. Επίσης, καθορίζουμε event handler για το συμβάν TimeChanged:

XAML:

```
<zeus:Chronometer Name="chrono" StartTime="0:2:00"  CriticalTime="00:00:20"
                  TimeChanged="chrono_TimeChanged" ... />
```

VB:

```
Imports Zeus.WPF.Controls.ChronoControls

...

 'When the time changes.
 Private Sub chrono_TimeChanged(sender As Object, e As TimeChangedEventArgs)

   'Do something here. e.CurrentTime has the current time.

 End Sub
```

Τέλος Manual