Oplossing 10.9

StartWindow.xaml

```
<Window x:Class="Oef10_9_MeerdereFormulieren.StartWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Start" Height="300" Width="300">
    <Grid>
        <Button Content="Start Wekker App" HorizontalAlignment="Left"
                Margin="90,115,0,0" VerticalAlignment="Top" Width="125"
                Name="startButton" Click="startButton_Click"/>

    </Grid>
</Window>
```

StartWindow.xaml.cs

```
public partial class StartWindow : Window
{
        public StartWindow()
        {
            InitializeComponent();
        }

        private void startButton_Click(object sender, RoutedEventArgs e)
        {
            MainWindow w = new MainWindow();
            w.Show();
        }
}
```

MainWindow.xaml

```
<Window x:Class="Oef10_9_MeerdereFormulieren.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Wekker" Height="350" Width="525">
    <Grid>
        <Label Content="HH:MM:SS" HorizontalAlignment="Left"
               Margin="20,13,0,0" VerticalAlignment="Top"
               Name="clockLabel" FontSize="100"/>
        <TextBlock HorizontalAlignment="Left" Margin="103,248,0,0"
                   Text="Alarmtijd: " VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23"
                 Margin="183,247,0,0" VerticalAlignment="Top" Width="120"
                 Name="alarmTextBox" />
        <Button Content="Set" HorizontalAlignment="Left"
                Margin="321,247,0,0" VerticalAlignment="Top" Width="75"
                Name="setButton" Click="setButton_Click"/>
    </Grid>
```

```
</Window>
```

MainWindow.xaml.cs

```
public partial class MainWindow : Window
{
        private Wekker wekker;
        private Color color1;
        private Color color2;
        private SolidColorBrush brush;

        private DispatcherTimer clockTimer = new DispatcherTimer();
        private DispatcherTimer alarmTimer = new DispatcherTimer();

        public MainWindow()
        {
            InitializeComponent();
            wekker = new Wekker();
            clockTimer.Start();
            color1 = Colors.White;
            brush = new SolidColorBrush(color1);
            clockLabel.Background = brush;
            color2 = Colors.Tomato;

            clockTimer.Interval = TimeSpan.FromSeconds(1);
            alarmTimer.Interval = TimeSpan.FromMilliseconds(300);

            clockTimer.Tick += clockTimer_Tick;
            alarmTimer.Tick += alarmTimer_Tick;
        }

        void alarmTimer_Tick(object sender, EventArgs e)
        {
            if (brush.Color == color1)
            {
                brush.Color = color2;
                //System.Media.SystemSounds.Beep.Play();
            }
            else
            {
                brush.Color = color1;
            }
        }

        void clockTimer_Tick(object sender, EventArgs e)
        {
            clockLabel.Content = wekker.CurrentTime;
            if (wekker.IsAlarmPassed())
            {
                Console.WriteLine("Alarm passed");
                alarmTimer.Start();
            }
            else
            {
                Console.WriteLine("Alarm NOT passed");
            }
            if (wekker.ShouldStopBeeping())
            {
                Console.WriteLine("ShouldStopBeeping");
```

```
                alarmTimer.Stop();
                wekker.Reset();
                alarmTextBox.Text = "";
                brush.Color = color1;
            }
        }

        private void setButton_Click(object sender, RoutedEventArgs e)
        {
            wekker.AlarmTime = alarmTextBox.Text;
        }
}
```

Wekker.cs

```
public class Wekker
{
        // tijd waarop de wekker zal afgaan
        private DateTime alarmTime;

        // aantal seconden dat de wekker zal afgaan
        private int beepTime;

        public Wekker()
        {
            beepTime = 10;
            alarmTime = DateTime.MinValue; // 1/1/0001 0:00:00
        }

        public bool IsAlarmPassed()
        {
            if (alarmTime == DateTime.MinValue)
            {
                return false;
            }
            else
            {
                return DateTime.Now > alarmTime;
            }
        }

        public bool ShouldStopBeeping()
        {
            if (IsAlarmPassed())
            {
                return alarmTime.AddSeconds(beepTime) < DateTime.Now;
            }
            else
            {
                return false;
            }
        }

        public void Reset()
        {
            alarmTime = DateTime.MinValue;
        }
```

```
        public string AlarmTime
        {
            set
            {
                alarmTime = Convert.ToDateTime(value);
            }
        }

        public int BeepTime
        {
            get { return beepTime; }
            set { beepTime = value; }
        }

        public string CurrentTime
        {
            get { return DateTime.Now.ToLongTimeString(); }
        }
    }
```