



Spruchsammler

Inhalt

Aufgabe.....	2
Format der Datei "Sprüche.txt"	2
Benutzerführung.....	3
Oberfläche beim ersten Start des Programms	3
Spruchtext und Autor wurden eingegeben	3
Spruch wurde gespeichert	4
Mehrere Sprüche können eingegeben werden.....	4
Technische Anmerkungen.....	5
Aktivierung und Deaktivierung der Buttons	5
Nützliche Ereignistypen	5
Lesen und Schreiben von Textdateien.....	5
Lösung (1. Version)	6
MainWindow.xaml.....	6
MainWindow.xaml.cs	7
Spruch.cs.....	9
Lösung (2. Version)	10
MainWindow.xaml.....	10
SpruchListeDatei.cs	12

Aufgabe

Entwickeln Sie ein Programm zum Sammeln von Sprüchen. Ein Spruch besteht aus einem Text und einem Autor. Zur Speicherung eines Spruchs im Programm gibt es eine Klasse 'Spruch' mit den Properties 'Text' und 'Autor'. Diese Klasse besitzt auch eine 'ToString' Methode.

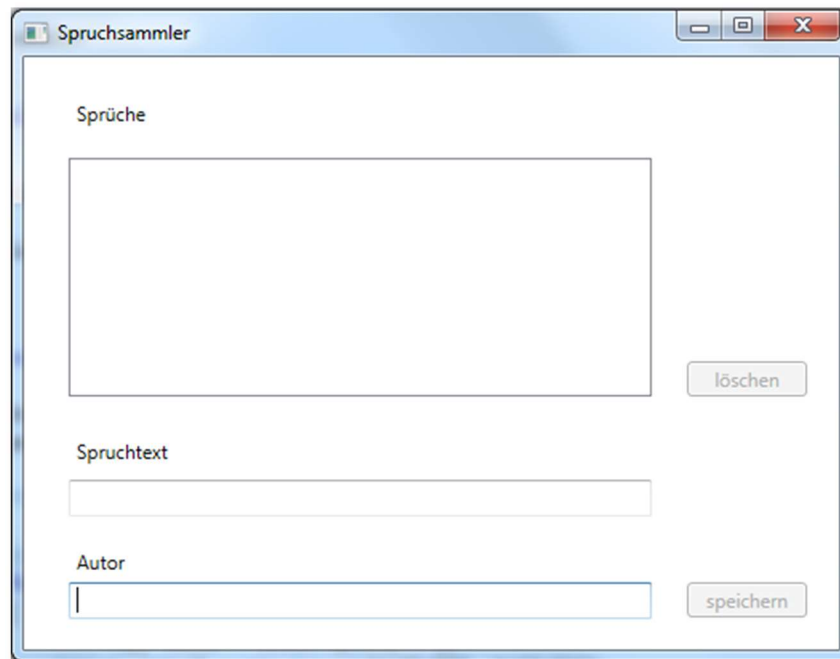
Die Sprüche werden langfristig in der Datei 'Sprüche.txt' gespeichert und von dort beim Programmstart wieder eingelesen.

Format der Datei "Sprüche.txt"

```
Wer sich nicht wehrt, der lebt verkehrt.:Wolf Biermann  
Alles neu macht der Mai.:Volksmund
```

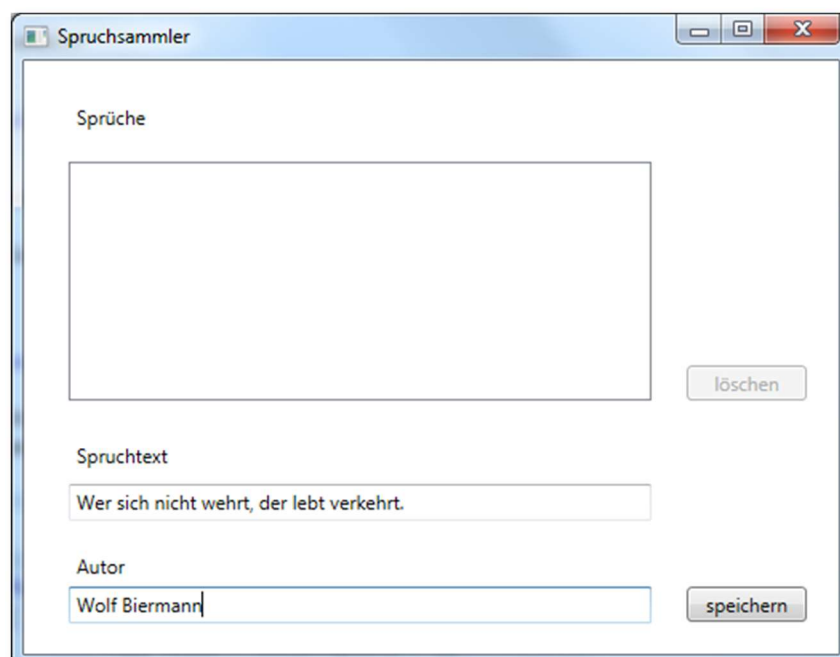
Benutzerführung

Oberfläche beim ersten Start des Programms



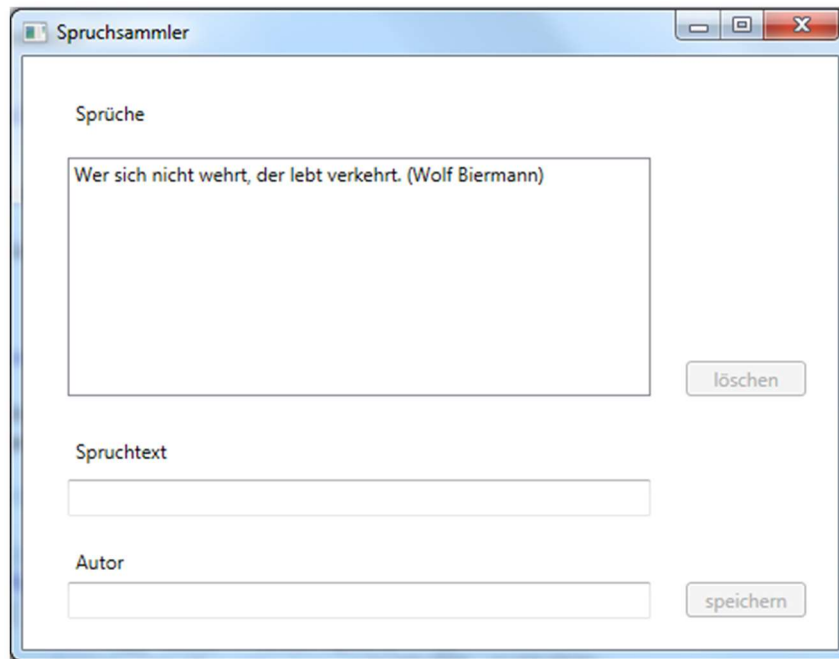
The screenshot shows the 'Spruchsammler' application window. It has a title bar with standard Windows window controls. The main area is titled 'Sprüche' and contains a large empty rectangular box for a quote. To the right of this box is a 'löschen' button. Below the box is a 'Spruchtext' label followed by an empty text input field. Below that is an 'Autor' label followed by an empty text input field. To the right of the 'Autor' field is a 'speichern' button.

Spruchtext und Autor wurden eingegeben



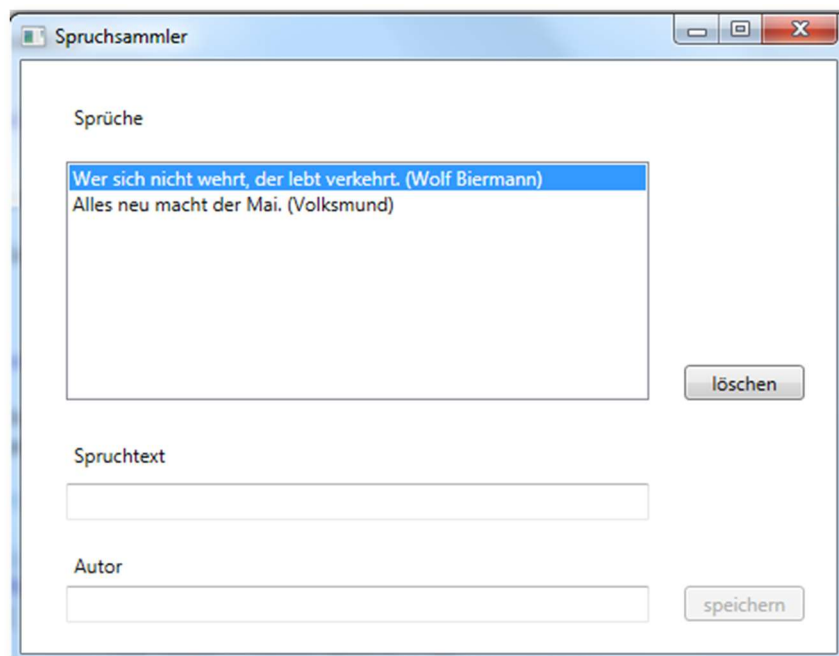
The screenshot shows the 'Spruchsammler' application window with the same layout as before. The 'Sprüche' box is still empty. The 'Spruchtext' input field now contains the text 'Wer sich nicht wehrt, der lebt verkehrt.' The 'Autor' input field now contains the text 'Wolf Biermann'. The 'löschen' and 'speichern' buttons remain in their respective positions.

Spruch wurde gespeichert



The screenshot shows a window titled "Spruchsammler". Inside, there is a section labeled "Sprüche" containing a text box with the text "Wer sich nicht wehrt, der lebt verkehrt. (Wolf Biermann)". To the right of this text box is a button labeled "löschen". Below the "Sprüche" section are two input fields: "Spruchtext" and "Autor". To the right of the "Autor" field is a button labeled "speichern".

Mehrere Sprüche können eingegeben werden



The screenshot shows the same "Spruchsammler" window. The "Sprüche" section now contains two lines of text: "Wer sich nicht wehrt, der lebt verkehrt. (Wolf Biermann)" and "Alles neu macht der Mai. (Volksmund)". The "löschen" button is still present to the right. The "Spruchtext" and "Autor" input fields and the "speichern" button remain at the bottom.

Technische Anmerkungen

Aktivierung und Deaktivierung der Buttons

Wie die Abbildungen der Benutzeroberfläche zeigen, sind beide Buttons immer nur dann aktiviert, wenn sie auch sinnvoll benutzbar sind.

- Wenn ein Eintrag in der Listbox ausgewählt ist, ist der "löschen"-Button aktiviert und sonst nicht.
- Wenn in beiden Eingabefeldern Texte eingegeben sind, ist der "speichern"-Button aktiviert und sonst nicht.

Steuerelemente vom Typ 'Button' besitzen die bool-Property 'IsEnabled' um sie zu aktivieren bzw. zu deaktivieren.

Nützliche Ereignistypen

Steuerelement	Ereignisname	Auslösendes Ereignis
Button	Click	<ul style="list-style-type: none">• Mausklick auf Button oder• fokussieren und ENTER
Window	Loaded	Das Fenster ist geladen.
Window	Closed	Das Fenster wurde geschlossen.
Listbox	SelectionChanged	Ein Eintrag wurde neu ausgewählt.
TextBox	TextChanged	Der Inhalt des Eingabefeldes wurde geändert.

Lesen und Schreiben von Textdateien

Lesen:

```
string[] File.ReadAllLines(string dateiname)
```

Schreiben:

```
File.WriteAllLines(string dateiname, string[] zeilen)
```

Lösung (1. Version)

In der ersten Version der Lösung wird in der Fensterklasse auf die Datei 'Sprüche.txt' zugegriffen.

MainWindow.xaml

```
<Window x:Class="Spruchsammler.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"
        mc:Ignorable="d"
        Title="Spruchsammler" Height="480.657" Width="586.496"
        Loaded="Window_Loaded" Closed="Window_Closed">
    <Grid>
        <Label Content="Sprüche" HorizontalAlignment="Left" Height="30"
            Margin="23,25,0,0" VerticalAlignment="Top" Width="106"/>
        <Label Content="Spruchtext" HorizontalAlignment="Left" Height="30"
            Margin="23,268,0,0" VerticalAlignment="Top" Width="106"/>
        <Label Content="Autor" HorizontalAlignment="Left" Height="30"
            Margin="23,353,0,0" VerticalAlignment="Top" Width="106"/>
        <TextBox Name="tbxAutor" HorizontalAlignment="Left" Height="24"
            Margin="23,388,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
            Width="384" Grid.ColumnSpan="2" TextChanged="Tbx_TextChanged"/>
        <TextBox Name="tbxSpruchtext" HorizontalAlignment="Left" Height="24"
            Margin="23,303,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
            Width="384" Grid.ColumnSpan="2" TextChanged="Tbx_TextChanged"/>
        <Button Name="btnSpeichern" Content="speichern" HorizontalAlignment="Left"
            Height="24" Margin="445,388,0,0" VerticalAlignment="Top"
            Width="105" Click="BtnSpeichern_Click"/>
        <ListBox Name="lbxSprüche" HorizontalAlignment="Left" Height="191"
            Margin="23,60,0,0" VerticalAlignment="Top" Width="384"
            Grid.ColumnSpan="2"
            SelectionChanged="LbxSprüche_SelectionChanged"/>
        <Button Name="btnLöschen" Content="löschen" HorizontalAlignment="Left"
            Height="24" Margin="445,227,0,0" VerticalAlignment="Top"
            Width="105" Click="BtnLöschen_Click"/>
    </Grid>
</Window>
```

MainWindow.xaml.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Windows;
using System.Windows.Controls;

namespace Spruchsammler
{
    public partial class MainWindow : Window
    {
        private string _dateiname = "../../../Sprüche.txt";
        private List<Spruch> _sprüche = new List<Spruch>();

        public MainWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            if (File.Exists(_dateiname))
            {
                string[] zeilen = File.ReadAllLines(_dateiname);
                foreach (string zeile in zeilen)
                {
                    string[] teile = zeile.Split(':');
                    _sprüche.Add(new Spruch(teile[0], teile[1]));
                }
            }

            btnSpeichern.IsEnabled = false;
            btnLöschen.IsEnabled = false;
            lbxSprüche.ItemsSource = _sprüche;
        }

        private void BtnSpeichern_Click(object sender, RoutedEventArgs e)
        {
            string text = tbxSpruchtext.Text;
            string autor = tbxAutor.Text;
            Spruch spruch = new Spruch(text, autor);
            _sprüche.Add(spruch);
            lbxSprüche.Items.Refresh();
            tbxSpruchtext.Text = "";
            tbxAutor.Text = "";
        }

        private void BtnLöschen_Click(object sender, RoutedEventArgs e)
        {
            Spruch spruch = (Spruch)lbxSprüche.SelectedItem;
            _sprüche.Remove(spruch);
            lbxSprüche.Items.Refresh();
        }
    }
}
```

```
private void Tbx_TextChanged(object sender, TextChangedEventArgs e)
{
    if (tbxSpruchtext.Text != "" && tbxAutor.Text != "")
    {
        btnSpeichern.IsEnabled = true;
    }
    else
    {
        btnSpeichern.IsEnabled = false;
    }
}

private void LbxSprüche_SelectionChanged(object sender,
                                         SelectionChangedEventArgs e)
{
    if (lbxSprüche.SelectedItem != null)
    {
        btnLöschen.IsEnabled = true;
    }
    else
    {
        btnLöschen.IsEnabled = false;
    }
}

private void Window_Closed(object sender, EventArgs e)
{
    List<string> zeilen = new List<string>();
    foreach (Spruch spruch in _sprüche)
    {
        zeilen.Add($"{spruch.Text}:{spruch.Autor}");
    }
    File.WriteAllLines(_dateiname, zeilen);
}
}
```


Spruch.cs

```
using System;

namespace Spruchsammler
{
    class Spruch
    {
        private string _text;
        private string _autor;

        public Spruch(string text, string autor)
        {
            _text = text;
            _autor = autor;
        }

        public string Text
        {
            get
            {
                return _text;
            }
        }

        public string Autor
        {
            get
            {
                return _autor;
            }
        }

        public override string ToString()
        {
            return $"{_text} ({_autor})";
        }
    }
}
```

Lösung (2. Version)

In der zweiten Version ist der Zugriff auf die Datei 'Sprüche.txt' in eine eigene Klasse ausgelagert. Die Dateien 'MainWindow.xaml' und 'Spruch.cs' sind in beiden Versionen gleich und werden deshalb nicht noch einmal aufgelistet.

MainWindow.xaml

```
using System;
using System.Collections.Generic;
using System.Windows;
using System.Windows.Controls;

namespace Spruchsammler
{
    public partial class MainWindow : Window
    {
        private SpruchListeDatei _datei;
        private List<Spruch> _sprüche;

        public MainWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            _datei = new SpruchListeDatei("../../../Sprüche.txt");
            _sprüche = _datei.Laden();
            lbxSprüche.ItemsSource = _sprüche;
            btnSpeichern.IsEnabled = false;
            btnLöschen.IsEnabled = false;
        }

        private void BtnSpeichern_Click(object sender, RoutedEventArgs e)
        {
            string text = tbxSpruchtext.Text;
            string autor = tbxAutor.Text;
            Spruch spruch = new Spruch(text, autor);
            _sprüche.Add(spruch);
            lbxSprüche.Items.Refresh();
            tbxSpruchtext.Text = "";
            tbxAutor.Text = "";
        }

        private void BtnLöschen_Click(object sender, RoutedEventArgs e)
        {
            Spruch spruch = (Spruch)lbxSprüche.SelectedItem;
            _sprüche.Remove(spruch);
            lbxSprüche.Items.Refresh();
        }
    }
}
```

```
private void Tbx_TextChanged(object sender, TextChangedEventArgs e)
{
    if (tbxSpruchtext.Text != "" && tbxAutor.Text != "")
    {
        btnSpeichern.IsEnabled = true;
    }
    else
    {
        btnSpeichern.IsEnabled = false;
    }
}

private void LbxSprüche_SelectionChanged(object sender,
                                         SelectionChangedEventArgs e)
{
    if (lbxSprüche.SelectedItem != null)
    {
        btnLöschen.IsEnabled = true;
    }
    else
    {
        btnLöschen.IsEnabled = false;
    }
}

private void Window_Closed(object sender, EventArgs e)
{
    _datei.Speichern(_sprüche);
}
}
```

SpruchListeDatei.cs

```
using System;
using System.Collections.Generic;
using System.IO;

namespace Spruchsammler
{
    class SpruchListeDatei
    {
        private string _dateiname;

        public SpruchListeDatei(string dateiname)
        {
            _dateiname = dateiname;
        }

        public List<Spruch> Laden()
        {
            List<Spruch> sprüche = new List<Spruch>();
            if (File.Exists(_dateiname))
            {
                string[] zeilen = File.ReadAllLines(_dateiname);
                foreach(string zeile in zeilen)
                {
                    string[] teile = zeile.Split(':');
                    sprüche.Add(new Spruch(teile[0], teile[1]));
                }
            }
            return sprüche;
        }

        public void Speichern(List<Spruch> sprüche)
        {
            List<string> zeilen = new List<string>();
            foreach(Spruch spruch in sprüche)
            {
                zeilen.Add($"{spruch.Text}:{spruch.Autor}");
            }
            File.WriteAllLines(_dateiname, zeilen);
        }
    }
}
```