
Volumenberechnung mit WPF

Objektbasiert programmieren nach Vorgabe

Inhaltsverzeichnis

1	Ziel	2
2	Ausgangslage	2
3	Aufgabenstellung	4
3.1	Generelle Hinweise zur Umsetzung	4
4	Hinweise zur Umsetzung der Problemstellung im Programmcode	4
4.1	Verwendung einer ComboBox	4
4.2	Sichtbarkeit der Steuerelemente festlegen	5
5	Geforderte Lösungsbereiche	7
6	Hilfsmittel	7
7	Zeitbedarf	7
8	Abbildungsverzeichnis	7

1 Ziel

Der Lernende macht weitere Erfahrungen wie eine Benutzerschnittstelle für einen einfachen Volumenberechner für verschiedene geometrische Körper und Grösseneinheiten erstellt werden kann. Ziel dieser Aufgabe ist es, das bereits durch diverse Aufgaben erworbene Wissen zu vertiefen und neue weiterführende Aspekte des GUI-Designs wie z.B. die „ComboBox“ kennenzulernen. Dieses Dokument hat aber nicht den Anspruch der Vollständigkeit und der perfekten Umsetzung des Problems. In der professionellen Umsetzung würde man das MVC-Prinzip einsetzen. Da dies aber zum jetzigen Zeitpunkt noch nicht optimal erklärt werden kann (Grundwissen fehlt) wird dieser Aspekt zu einem späteren Zeitpunkt nachgeholt und vertieft werden.

2 Ausgangslage

In dieser Aufgabe wollen wir einen Volumenberechner für diverse geometrische Körper und Grösseneinheiten erstellen. Damit die Aufgabe an Übersichtlichkeit nicht verliert, werden wir uns auf die fünf meistgebrauchten geometrischen Körper beschränken.

Es sind dies:

- Würfel
- Quader
- Pyramide
- Kegel
- Kugel

Als Einheiten für die Grössenangabe sollen die folgende Möglichkeiten bestehen:

- Millimeter
- Zentimeter
- Dezimeter
- Meter
- Kilometer

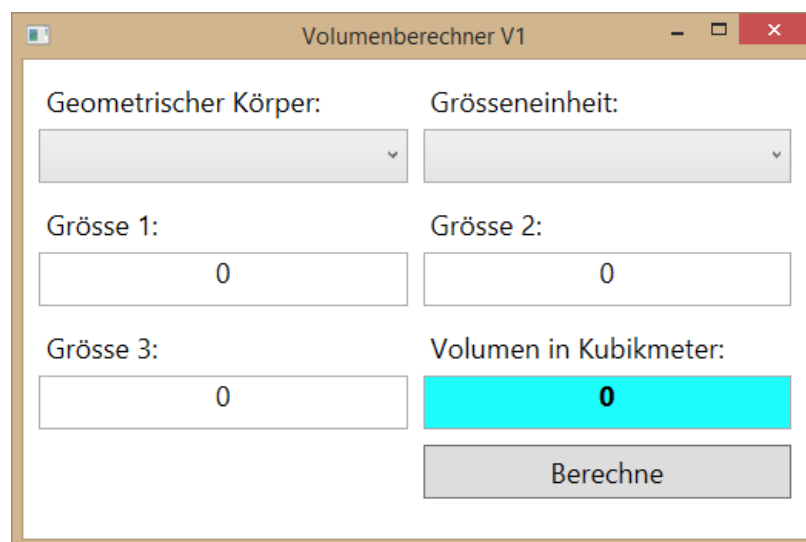


Abb. 1: Ansicht des Volumenberechners beim Start

Wie Sie aus obenstehendem Bild erkennen können, handelt es sich um eine Anwendung mit zwei „ComboBox“-Steuerelementen. Das erste dieser Steuerelemente mit der Bezeichnung: „Geometrischer Körper“ wird für die Auswahl der geometrischen Figur verwendet. Das Zweite für die Auswahl der Eingabeeinheit.

Im Weiteren erkennen wir 3 Eingabefelder für die Angabe der Grössen der gewählten geometrischen Figur. Hier ist zu beachten, dass für die Volumenberechnung nicht immer alle 3 Grössenfelder benötigt werden. Um dem Anwender die Eingabe der Werte zu vereinfachen, sollen nur immer diejenigen Grössenfelder sichtbar sein, welche auch benötigt werden. Im Weiteren sollen die Labels der einzelnen Grössenfelder mit dem richtigen Begriff versehen werden.

Daraus ergeben sich die folgenden Darstellungen des Anwendungsfensters:

Volumenberechner V1

Geometrischer Körper: Würfel

Grösseneinheit: Millimeter

Seitenlänge: 1

Volumen in Kubikmeter: 1E-09

Berechne

Abb. 2: Volumenb.: Würfel, Eingabeeinheit: mm

Volumenberechner V1

Geometrischer Körper: Quader

Grösseneinheit: Zentimeter

Länge: 1

Breite: 2.2

Höhe: 3.5

Volumen in Kubikmeter: 7.7E-06

Berechne

Abb. 3: Volumenb.: Quader, Eingabeeinheit: cm

Volumenberechner V1

Geometrischer Körper: Pyramide

Grösseneinheit: Dezimeter

Seitenlänge: 25

Höhe: 12.9

Volumen in Kubikmeter: 2.6875

Berechne

Abb. 4: Volumenb.: Pyramide, Eingabeeinheit: dm

Volumenberechner V1

Geometrischer Körper: Kegel

Grösseneinheit: Meter

Radius: 30

Höhe: 100

Volumen in Kubikmeter: 94247.7796076938

Berechne

Abb. 5: Volumenb.: Kegel, Eingabeeinheit: m

Volumenberechner V1

Geometrischer Körper: Kugel

Grösseneinheit: Kilometer

Radius: 125.2

Volumen in Kubikmeter: 8220563642.25669

Berechne

Abb. 6: Volumenb.: Kugel, Eingabeeinheit: km

Wie Sie sicherlich festgestellt haben, wurde bei den obigen Darstellungen auch immer die Eingabeeinheit gewechselt. Bitte beachten Sie aber, dass die Einheit bei allen Volumenberechnungen frei wählbar sein soll und nicht für die Volumenberechnung eines bestimmten geometrischen Körpers vorgegeben wird!

Durch die Betätigung des Buttons: „Berechne“ wird das Volumen des gewählten geometrischen Körpers mit der angegebenen Grösseneinheit berechnet und als Resultat mit der Einheit Kubikmeter [m³] ausgegeben.

3 Aufgabenstellung

Programmieren Sie den Volumenrechner nach den bereits angegebenen Forderungen.

Der Volumenberechner selbst soll in der Grösse der Darstellung frei verändert werden können.

3.1 Generelle Hinweise zur Umsetzung

Da diese Aufgabe eine Fortsetzung bereits bestehender Aufgaben ist, sollen jetzt nicht mehr alle Teilbereiche der Umsetzung als Tutorial vorgegeben werden. Sie können aber jederzeit in die Lösung der letzten Aufgaben hineinschauen um diese Aufgabe selbständig lösen zu können.

4 Hinweise zur Umsetzung der Problemstellung im Programmcode

4.1 Verwendung einer ComboBox

Ich zeige Ihnen in diesem Abschnitt wie eine ComboBox verwendet wird. Um die Umsetzung effektiv zeigen zu können, wird die Auswahl des geometrischen Körpers als Beispiel gezeigt. Die gleiche Umsetzungsart kann aber auch für die Auswahl der Grösseneinheit verwendet werden.

Geometrischer Körper:

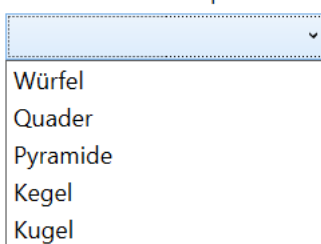


Abb. 7: ComboBox:
Geometrischer Körper

Damit der Benutzer den geometrischen Körper frei wählen kann, wollen wir diesmal ein neues Steuerelement einsetzen. Es handelt sich hierbei um das Steuerelement: **ComboBox**.

Bei einer ComboBox lässt sich in der Default-Einstellung nur immer ein Eintrag auswählen. Diese Default-Einstellung kann aber geändert werden und es können dann auch mehrere Einträge ausgewählt werden. Für unsere Aufgabe belassen wir aber die ComboBox in der Default-Einstellung.

4.1.1 Umsetzung im File: MainWindow.xaml

```
<ComboBox x:Name="cboxGeometrischerKoeper" Grid.Column="1" Grid.Row="2"
          FontSize="18" SelectionChanged="cboxGeometrischerKoeper_SelectionChanged"
/>
```

Abb. 8: Erzeugung einer ComboBox: cboxGeometrischerKoeper im File: MainWindow.xaml

Legende:

- x:Name: Name der ComboBox. Mit diesem Namen wird das Auswahlfeld im Programmcode angesprochen.
- Grid.Column: Angabe in welcher Spalte der RadioButton platziert wird.
- Grid.Row: Angabe in welcher Zeile der RadioButton platziert wird.
- FontSize: Grösse der sichtbaren Buttonbeschriftung.
- SelectionChanged: Funktion welche beim Auswählen des Eintrages durch den Benutzer aufgerufen werden soll.

4.1.2 Umsetzung im File: MainWindow.xaml.cs

4.1.2.1 Festlegung des Inhaltes einer ComboBox:

Um den Inhalt einer ComboBox festzulegen, verwendet man am einfachsten eine Liste. In unserem Beispiel könnte einer Liste für die Auswahl des geometrischen Körpers wie folgt aufgebaut werden:

```
List<string> lGeometrischerKoerper = new List<string>();  
lGeometrischerKoerper.Add("Würfel");  
lGeometrischerKoerper.Add("Quader");  
lGeometrischerKoerper.Add("Pyramide");  
lGeometrischerKoerper.Add("Kegel");  
lGeometrischerKoerper.Add("Kugel");
```

Abb. 9: Erzeugung einer Inhaltsliste für die ComboBox: cboxGeometrischerKoerper

4.1.2.2 Zuordnung des Inhaltes (Liste) an eine ComboBox:

```
cboxGeometrischerKoerper.ItemsSource = lGeometrischerKoerper;
```

Abb. 10: Listenzuordnung an die ComboBox: cboxGeometrischerKoerper

4.1.2.3 Abfrage was in einer ComboBox gewählt wurde:

Variante 1: Rückgabe der gewählten Position als Integerwert

```
cboxGeometrischerKoerper.SelectedIndex
```

Abb. 11: Rückgabe des selektierten Wertes der ComboBox: cboxGeometrischerKoerper als Integerwert

Variante 2: Rückgabe der gewählten Position als String-Wert

```
Convert.ToString(cboxGeometrischerKoerper.SelectedValue)
```

Abb. 12: Rückgabe des selektierten Wertes der ComboBox: cboxGeometrischerKoerper als String-Wert

Diese Rückgabewerte können im Verlauf des Programms z.B. mittels einer switch-Anweisung ausgewertet werden.

4.2 Sichtbarkeit der Steuerelemente festlegen

Damit dem Benutzer nur diejenigen Eingabefelder zur Verfügung stehen die er auch benötigt, müssen die Steuerelemente in der Sichtbarkeit verändert werden können.

Um dieser Forderung gerecht zu werden, müssen Sie die Steuerelemente „verstecken“ oder „sichtbar“ machen. Dies geschieht wie folgt und wird steuerungsmässig im File: MainWindow.xaml.cs eingesetzt.

4.2.1 Steuerelemente auf „sichtbar“ stellen:

```
lblGroesse1.Visibility = Visibility.Visible;  
tboxGroesse1.Visibility = Visibility.Visible;
```

Abb. 13: Label: lblGroesse1 und Textbox: tboxGroesse1 werden sichtbar gesetzt

4.2.2 Steuerelemente auf „unsichtbar“ stellen:

```
lblGroesse1.Visibility = Visibility.Hidden;  
tboxGroesse1.Visibility = Visibility.Hidden;
```

Abb. 14: Label: lblGroesse1 und Textbox: tboxGroesse1 werden unsichtbar gesetzt

5 Geforderte Lösungsbereiche

- Erzeugung des GUI-Oberfläche nach Vorgabe
- Einfügen der Ereignismethoden
- Erklärung der Funktionsweise des Codes
- Auflistung von Verbesserungsmöglichkeiten des GUI

6 Hilfsmittel

Visual Studio

7 Zeitbedarf

ca. 90 Minuten

8 Abbildungsverzeichnis

Abb. 1: Ansicht des Volumenberechners beim Start	2
Abb. 2: Volumenb.: Würfel, Eingabeeinheit: mm	3
Abb. 3: Volumenb.: Quader, Eingabeeinheit: cm	3
Abb. 4: Volumenb.: Pyramide, Eingabeeinheit: dm	3
Abb. 5: Volumenb.: Kegel, Eingabeeinheit: m	3
Abb. 6: Volumenb.: Kugel, Eingabeeinheit: km	3
Abb. 7: ComboBox: Geometrischer Körper	4
Abb. 8: Erzeugung einer ComboBox: cboxGeometrischerKoerper im File: MainWindows.xaml	4
Abb. 9: Erzeugung einer Inhaltsliste für die ComboBox: cboxGeometrischerKoerper	5
Abb. 10: Listenzuordnung an die ComboBox: cboxGeometrischerKoerper	5
Abb. 11: Rückgabe des selektierten Wertes der ComboBox: cboxGeometrischerKoerper als Integerwert	5
Abb. 12: Rückgabe des selektierten Wertes der ComboBox: cboxGeometrischerKoerper als String-Wert	5
Abb. 13: Label: lblGroesse1 und Textbox: tboxGroesse1 werden sichtbar gesetzt	5
Abb. 14: Label: lblGroesse1 und Textbox: tboxGroesse1 werden unsichtbar gesetzt	6

Historie

Vers.	Bemerkungen	Verantwortl.	Datum
1.0	- Komplette Aufgabe in C# erstellt	W. Odermatt	05.01.2016

Referenzunterlagen

LfNr.	Titel / Autor / File / Verlag / ISBN	Dokument-Typ	Ausgabe
1	„Einstieg in Visual C# 2013“ / Thomas Theis / Galileo Computing / 978-3-8362-2814-5	Fachbuch	2014
2	“Visual C# 2012” / Andreas Kühnel / Rheinwerk Computing / 978-3-8362-1997-6	Fachbuch	2013
3	“Objektorientiertes Programmieren in Visual C#” / Peter Loos / Microsoft Press / 3-86645-406-6	Fachbuch	2006