
BELEG

Körper und Formenrechner

Frau Juliet Petzold

im Rahmen des Moduls C#

Seminargruppe: IF18wS-B

Email: jpetzol1@hs-mittweida.de

Betreuer: Philipp Engler

Mittweida, Sommersemester 2020

Inhaltsverzeichnis

Inhaltsverzeichnis	0
1 Ideenfindung und Aufgabenstellung	2
1.1 <i>Aufgabenstellung.....</i>	<i>2</i>
1.2 <i>Ideenfindung</i>	<i>2</i>
2 Lösungskonzeption / Ansatz zur Umsetzung.....	4
2.1 <i>Funktionalität.....</i>	<i>4</i>
3 Programmentwurf	5
3.1 <i>Methoden</i>	<i>5</i>
3.2 <i>Programmgerüst.....</i>	<i>5</i>
4 Details zur Implementierung.....	7
4.1 <i>Programmierer Tagebuch.....</i>	<i>7</i>
4.1.1 Tag 1: Umsetzen des Grundgerüsts	7
4.1.2 Tag 2: Erstellen der Rechner- „Schablone“	7
4.1.3 Tag 3: Get und Set Methoden für die Berechnung.....	8
4.1.4 Tag 4: Die Formeln.....	8
4.1.5 Tag 5: Fehlerbehebung	8
4.1.6 Tag 6: Vorbereiten der Klassen	9
4.1.7 Tag 7: Serialisierung in XML.....	9
4.1.8 Tag 8: Letzte Optimierungen	11
4.1.9 Tag 9: Fertigstellung des Programms.....	11
4.1.10 Tag 10: Verpacken aller Projektbestandteile	11
5 Problemanalyse.....	12
5.1 <i>Nutzung einer ComboBox</i>	<i>12</i>
5.2 <i>Anlegen von Berechnungsvorlagen.....</i>	<i>12</i>
5.3 <i>Methode zum Berechnen</i>	<i>13</i>
5.4 <i>Serialisierung in XML</i>	<i>13</i>
5.5 <i>Bedienungsprobleme</i>	<i>14</i>
6 Fazit.....	15
7 Bedienungsanleitung	17

Inhaltsverzeichnis	1
7.1	<i>Start des Programms</i> 17
7.2	<i>Eingabe der Daten und Berechnung</i> 18
7.3	<i>Export in XML</i> 19
Literaturverzeichnis	21
Selbstständigkeitserklärung	22

1 Ideenfindung und Aufgabenstellung

Für diesen Beleg sollte ein Programm angefertigt werden, welches einige wichtige Kriterien erfüllen muss. Grundlegend wichtig dabei ist, dass das Programm funktionsfähig und für den Benutzer verständlich ist. Dazu muss das Programm die erlernten Methoden der Programmiersprache C# aufgreifen. In den folgenden Abschnitten werden die Aufgabenstellung, welche dem Projekt zugrunde liegt, sowie der Prozess der Ideenfindung behandelt.

1.1 Aufgabenstellung

Das erstellte Projekt muss in einem geeigneten Umfang von in dieser Veranstaltung vorgestellten Technologien beziehungsweise Funktionalitäten (Dienste, SQL Server Anbindung, asynchrone Programmierung, Kommunikation zwischen Programmen, Steuerung von Microsoft Office, ...) gebrauch machen. Eine Benutzeroberfläche ist ebenfalls zu erstellen. Dazu sollte bevorzugt WPF verwendet werden.¹

1.2 Ideenfindung

Um überhaupt eine Idee entwickeln zu können, muss zuerst die Aufgabenstellung mit den gestellten Anforderungen an das Programm analysiert werden. Dadurch wird klar, welche Idee sich zur Umsetzung eignet.

Zu allererst geht es darum einen geeigneten Umfang zu wählen. Es ist angebracht das Projekt nicht zu kurz (bspw. <100 Zeilen Code) aber auch nicht übermäßig abstrakt zu gestalten und zu bearbeiten. Dabei ist das Anwenden der in den Veranstaltungen vorgestellten Technologien in ausreichender Tiefe wichtig. Dieser Punkt spielte eine Rolle bei der Ideenfindung, da das Projekt erlauben sollte gewisse Funktionalitäten (Dienste, SQL-Server-Anbindung, Microsoft Office Steuerung, ...) zu nutzen. Das erstellte Projekt muss ebenfalls eine Benutzeroberfläche haben, welche unter Verwendung von WPF erstellt werden soll.

Vorangegangene Punkte schränkten die Vielfalt der Möglichkeiten ein. Die ersten Ideen waren von weniger praktischem Nutzen, eher gingen sie in die Richtung von kleinen Spielereien. Schnell wurde jedoch klar, dass es dabei weder eine Anwendung von Funktionalitäten, wie bereits vorher erwähnt, noch einen geeigneten Umfang haben wird. Letztendlich

¹ M.Sc. Philipp Engler

fiel die Entscheidung auf einen Taschenrechner. Dabei sollte es kein einfacher Taschenrechner sein, welcher nur die einfachsten Rechenoperationen durchführen kann. Stattdessen sollte dieser Funktionen umfassen, welche weitgreifend sind. Deshalb wurde der Taschenrechner kurzerhand zu einem Körper- und Formenrechner. Der Plan dafür ist, dass es möglich sein soll anhand von einigen Eingaben von Größen viele weitere Größen auszurechnen. Um eine spezielle Funktionalität einzubinden, fiel die Entscheidung nach Rücksprache mit dem Betreuer auf die Serialisierung und den Export der Daten in XML.

Ein letzter Punkt, welcher noch zur Ideenfindung zählen sollte, wäre die Überlegung der Formen und Körper, welche im Programm zur Auswahl stehen sollen. Dafür hat man sich ein Tafelwerk aus Schulzeiten zur Hand genommen und folgende Auswahl getroffen:

Rechteck, Quadrat, Parallelogramm, Trapez, Dreieck, Kreis, Quader, Würfel, Prisma, Kugel, Pyramide, Tetraeder, Kegel, Pyramidenstumpf, Kegelstumpf, Torus (Kreisring).

Da nicht jeder dieser Formen bzw. Körper konventionell ist und demnach zugängliche Berechnungsformeln für vor allem Volumen und Oberflächeninhalt haben, wurden letztendlich einige herausgestrichen. Die endgültige Auswahl fiel auf folgende Formen und Körper:

Rechteck, Quadrat, Parallelogramm, Trapez, Dreieck, Kreis, Quader, Würfel, Kugel, Pyramide und Kegel.

2 Lösungskonzeption / Ansatz zur Umsetzung

2.1 Funktionalität

Das Projekt wird für die verschiedensten Berechnungen und Formeln einige Methoden enthalten. Darauf folgen (für den Benutzer nicht sichtbar) passende Aufrufe der Methoden mit entsprechenden Parametern und letztendlich eine Ausgabe. Die Aufrufparameter der Methoden sollen die Eingaben des Nutzers sein. So soll gewährleistet werden, dass alle Berechnungen möglich sind. Am Ende steht der Gedanke, dass das Programm mit dem Benutzer kommunizieren soll, ihn also eindeutig zur Eingabe von individuellen Größen auffordern soll und am Ende die Ergebnisse der möglichen Berechnungen anzeigt. Generell soll der Nutzer leicht verstehen, was das Programm von ihm verlangt und was es damit macht. Der Benutzer ist nicht dazu gezwungen alle Felder die „aktiviert“ sind auszufüllen. Es müssen lediglich die farblich hinterlegten Textfelder ausgefüllt werden. Ein reales Beispiel dazu wäre, dass ein Viereck kein Volumen hat, da es eine Fläche und kein Körper ist. Da diese simple Berechnungsfunktion allerdings noch nicht besonders umfangreich ist und auch noch keine spezielle Technologie oder Funktionalität beinhaltet galt es nun etwas passendes dafür zu finden. Bezugnehmend auf den bereits vergangenen Prozess der Ideenfindung fiel die Entscheidung deswegen auf die Serialisierung und den Export in XML. Dafür wurden Pfade des Programmiers benutzte, die auf dem Computer zum Desktop führen. Es soll eine Möglichkeit geben den Benutzer entscheiden zu lassen, ob er möchte, dass die (falls) bereits existierende Datei überschrieben werden soll oder nicht.

3 Programmmentwurf

In diesem Kapitel werden alle Vorbereitungen für die Implementierung des Programmes festgehalten. Dabei ist zu bemerken, dass der Entwurf lediglich eine Orientierung für den Programmierer ist. Es kann also durchaus sein, dass das finale Produkt leicht verändert sein wird.

3.1 Methoden

Um die Auswahl der Form bzw. des Körpers sinnvoll unterscheiden zu können, ist bereits am Anfang klar, dass für die Programmierung mehrere case-switches verwendet werden müssen. Dies erlaubt auch eine Unterscheidung zwischen den jeweils zu nutzenden Formeln für die Berechnung der Größen. Um eine Berechnung durchführen zu können, ist eine Methode mit dem Namen „berechne“ erforderlich. Diese kennt jede Formel für jede Form bzw. Körper und durch die case-switches wird entschieden, welche für die entsprechende Auswahl genutzt werden müssen. Zum Schluss muss es noch eine Methode geben, welche die Daten in für einen Export in XML serialisiert und eine Datei erstellt. Sie wird „exportiere“ heißen.


3.2 Programmgerüst

Im weiteren Verlauf der Ideenfindung und Lösungskonzeption entstanden zunächst handschriftliche Skizzen zum Programm. Diese werden an dieser Stelle allerdings nicht weiter vertiefend beschrieben. Mit der Abgabe des Belegthemas fertigte man aus den anfangs handschriftlichen Skizzen folgende grafische und digitalisierte Skizzen zum Aufbau des Programms an. Später jedoch entschied man sich stellenweise gegen einige der hier gezeigten Designelemente oder fügte zusätzlich noch welche hinzu.

Überschrift: Formen- und Körperrechner

Drop-Down-Menü zur Auswahl der Formen bzw. Körper

Anleitung



Überschrift: Formen- und Körperrechner

Eingabefeld Wert a

Eingabefeld Wert b

Eingabefeld Wert c

Eingabefeld Wert d

Eingabefeld Wert ...

Drop-Down-Menü:
Was soll berechnet werden?

Button: "Berechne"

Überschrift: Formen- und Körperrechner

Der berechnete Wert beträgt:

evtl. Bild des Objektes

Button:
Zurück zum Hauptmenü

Abbildung 1: Programmgerüst

4 Details zur Implementierung

Im folgenden Abschnitt wird die genaue Implementierung des Programmcodes ähnlich wie ein Tagebuch beschrieben. Dabei wird jeder Tag, an dem das Programm bearbeitet worden ist, einzeln aufgelistet und die wichtigsten Schritte sowie aufgetretene Probleme betrachtet. Falls es etwas auftritt das nähere Erläuterungen erfordert, dann wird ggf. ein Screenshot in den Eintrag eingefügt.

4.1 Programmierer Tagebuch

4.1.1 Tag 1: Umsetzen des Grundgerüsts

Um diesen ersten Schritt zu gehen erforderte es einige Vorbereitung. Zu allererst musste die Programmierumgebung eingerichtet werden. Dazu wurden Visual Studio 2019 und einige weitere kleine nicht-nennenswerte Erweiterungen dazu installiert. Da die Umgebung bereits in den Lehrveranstaltungen verwendet wurde, fiel die anfängliche Installation bereits weg. Ein neues Projekt wurde dementsprechend konfiguriert und erstellt. Um die Designelemente des Grundgerüsts in das Programm zu übernehmen, wurden erst einmal mehrere Windows (MainWindow, Window2, ...) mit entsprechenden Elementen erstellt. Die passende Benennung der Elemente folgte darauf.

4.1.2 Tag 2: Erstellen der Rechner- „Schablone“

Die Problematik dieses Tages stellte sich sehr früh heraus. Die Idee ein Drop-Down Menü für die Formen bzw. Körper zu haben war in der Theorie sehr gut und übersichtlich, doch die praktische Umsetzung im Programm stellte sich als schwierig heraus. Dazu gehörte nicht unwesentlich die verwirrende Methode um „Items“ in das Menü einzufügen. Es dauerte einige Zeit, um die nötigen Informationen darüber herauszufinden. Das Drop-Down Menü wurde letztendlich um alle geplanten Formen und Körper erweitert und deren zugewiesene ID's geben Informationen über benötigte Werte und Größen an das nächste Window weiter. Dafür mussten die ID's selbst als globale Variable definiert werden. Hier wurden sie ganz einfach als „Globals“ implementiert. Durch Weitergabe dieser Informationen an weitere Windows werden unbenutzbare bzw. nicht benötigte Größen ausgegraut und man kann auch keine Daten darin eingeben.

4.1.3 Tag 3: Get und Set Methoden für die Berechnung

Wie der Titel des Abschnittes bereits verrät, wurden an diesem Tag der Bearbeitung alle Getter und Setter Methoden für die anschließende Berechnung erstellt. Diese werden gebraucht um alle nötigen Größen bzw. Einträge aus der „Schablone“ zu erhalten und ggf. mögliche Berechnungen auszuführen und dessen Ergebnisse einzutragen. Beispielhaft dafür ist die Fläche des Rechtecks. Zuerst muss im Drop-Down Menü das Rechteck gewählt werden. Danach werden in der „Schablone“ gewünschte Zahlen/Größen für die Länge und die Breite eingegeben. Zum Schluss muss der Button „Berechne“ betätigt werden, ab diesem Punkt arbeitet das Programm selbst. Mit Get-Methoden werden für die Berechnung die eingegebene Länge und die Breite genommen um dann wird mit einer Formel ($\text{Länge} \times \text{Breite}$) die Fläche berechnet. Allerdings müssen diese Größen zuerst in den Datentyp integer konvertiert werden, was mit den Set-Methoden erfolgt. Das Ergebnis wird dann in die TextBox für „Fläche“ eingetragen.

4.1.4 Tag 4: Die Formeln

Dieser Tag der Programmentwicklung war der bisher bei Weitem an schwierigsten. Nicht etwa, weil die Formeln komplex und unübersichtlich waren, sondern weil hier die Gefahr der Verwechslung oder des Durcheinanderkommens sehr hoch war. Mithilfe eines Tafelwerkes waren die Formeln für z.B. Kugelberechnung schnell gefunden, doch C# machte es dem Programmierer nicht leicht. Es mussten Math-Methoden für Potenzen (`Math.Pow`) und Wurzelberechnung (`Math.Sqrt`) gefunden werden. Des Weiteren wurde für die Konstante Pi `Math.PI` benutzt. Aufgrund der Berechnungsschablonen wurde bereits festgelegt, welche Werte die Methode für die Berechnung übergeben bekommt. Deswegen mussten also einige Formeln umgestellt oder kleine Nebenrechnungen durchgeführt werden.

4.1.5 Tag 5: Fehlerbehebung

Bis zum jetzigen Zeitpunkt sollte das Programm mit seinen Grundzügen funktionsfähig sein. Die Formeln für alle „Schablonen“ wurden erstellt und eingetragen - mit den richtigen Eingaben (die farbig markierten Felder) ist auch eine Berechnung für alle möglichen Größen möglich. Nun geht es darum, die übrig gebliebenen Fehler zu beheben. Das Drop-Down Menü machte bisher immer einige Probleme. Es funktionierte nicht ganz, da es immer das zuvor ausgewählte Element nutzte. Der Fehler ließ sich schnell beheben indem einfach 2 Zeilen im Code ausgetauscht wurden. Die Reihenfolge war schlichtweg falsch. Des Weiteren ergab sich eine kleine Unstimmigkeit des Designs, da das Menü dauerhaft ausgeklappt war. Trotz Vereinfachung durch WPF, welche es erlaubte einen Haken bei „`IsDropDownOpen`“ zu setzen oder zu entfernen veränderte sich nichts. Deswegen wurde im xaml-Code die Zeile „`IsDropDownOpen = „false“`“ hinzugefügt. Damit ist das Programm voll funktionsfähig. Als nächstes befasst sich der Programmierer mit der Serialisierung in XML.

4.1.6 Tag 6: Vorbereiten der Klassen

Beim Betrachten der bereits existierenden 10 case's entschied sich der Programmierer dazu eine einzige Klasse zu erstellen. Die Idee dahinter war, dass die Klasse doch jede Größe enthalten kann, aber die Objekte individuell diese Größen nutzen (oder eben nicht). Wie bereits vorher im Beleg beschrieben wurde, werden dann für ungenutzte Größen einfach stellvertretend eine 0 eingetragen. Testweise wurde ein Objekt mit dem Namen „rechteck“ für ein Rechteck erstellt. Es hat Werte für folgende Größen: Länge, Breite, Fläche, Umfang, Diagonale. Für alle anderen Werte wurde eine 0 eingetragen. Die gleiche Vorgehensweise muss also auch noch für alle anderen Formen und Körper angewendet werden, um Objekte für diese zu erstellen. Dabei fiel dem Programmierer auf, dass es Kommazahlen als Lösungen gibt und dass der Benutzer auch Kommazahlen eingeben könnte. Dafür wurden nun alle Datentypen von int in double geändert. Damit die Zahlen nicht zu groß werden, wurden sie mit Math.Round auf 3 Stellen nach dem Komma gerundet.

4.1.7 Tag 7: Serialisierung in XML

Basierend auf den Vorlesungen und Lehrveranstaltungen waren die grundlegenden Funktionsweisen des Serialisierens bereits bekannt. Allerdings war es wichtig, die jeweiligen Pfade zu den Dateien anzupassen sowie die Namen. Die erstellten Objekte haben bereits alle nötigen Werte. Nun geht es darum, die .xml-Dateien zu erstellen, die Daten hineinzuschreiben und letztendlich zu speichern. Dabei ist es wichtig zu erwähnen, dass die Pfade zu den Dateien nur auf dem Computer des Programmierers funktionieren. Im folgenden Screenshot ist dies zu sehen.

```
XmlTextWriter xmlWriter0 = new XmlTextWriter("C:/Users/julie/Desktop/rechteck.xml", Encoding.UTF8);
XmlSerializer xmlSerializer0 = new XmlSerializer(typeof(Form));
xmlSerializer0.Serialize(xmlWriter0, rechteck);
```

Abbildung 2: Screenshot Dateipfad

Wenn man nun in dem Rechner die Zahlen „5“ im Feld für die Länge eingibt, und „2“ für die Breite, dann kommt nach der Berechnung und dem Export folgende Größen und .xml-Datei heraus:

```

<?xml version="1.0" encoding="UTF-8"?>
- <Form xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Breite>2</Breite>
  <Laenge>5</Laenge>
  <Diagonale>5.385</Diagonale>
  <Flaeche>10</Flaeche>
  <Umfang>14</Umfang>
  <Hoehe>2</Hoehe>
  <Radius>0</Radius>
  <Durchmesser>0</Durchmesser>
  <Kathete>0</Kathete>
  <Ankathete>0</Ankathete>
  <Hypotenuse>0</Hypotenuse>
  <Oberflaeche>0</Oberflaeche>
  <Volumen>0</Volumen>
  <Laenge2>0</Laenge2>
  <Breite2>0</Breite2>
</Form>

```

Abbildung 3: Screenshot XML-Dokument

Des Weiteren wurde angepasst, dass man im Programm erst exportieren kann, sobald man etwas berechnet hat. Dafür wurde der Button für den Export deaktiviert, welcher sich mit dem Betätigen des Berechnen-Buttons aktiviert. Dies müsste ebenfalls für den Button „Berechne“ existieren, da man ihn ebenfalls auch ohne Eintragungen betätigen könnte, was unweigerlich zu einem Absturz des Programmes führt. Eine weitere Neuerung im Programmablauf ist, dass eine Abfrage stattfindet, ob die zu erstellende Datei bereits existiert

```

case 0:
//Rechteck
Form rechteck = new Form(getBreite(), getLaenge(), getDiagonale(), getFlaeche(), getUmfang(), getBreite(), 0, 0, 0, 0, 0, 0, 0, 0);

if (File.Exists("C:/Users/julie/Desktop/rechteck.xml") == false)
{
    XmlTextWriter xmlWriter0 = new XmlTextWriter("C:/Users/julie/Desktop/rechteck.xml", Encoding.UTF8);
    XmlSerializer xmlSerializer0 = new XmlSerializer(typeof(Form));
    xmlSerializer0.Serialize(xmlWriter0, rechteck);
}

else
{
    MessageBoxResult m = MessageBox.Show("Die Datei existiert bereits. Überschreiben?", "Warnung", MessageBoxButton.YesNo);
    if (m == MessageBoxResult.Yes)
    {
        File.Delete("C:/Users/julie/Desktop/rechteck.xml");
        XmlTextWriter xmlWriter0 = new XmlTextWriter("C:/Users/julie/Desktop/rechteck.xml", Encoding.UTF8);
        XmlSerializer xmlSerializer0 = new XmlSerializer(typeof(Form));
        xmlSerializer0.Serialize(xmlWriter0, rechteck);
    }
    else if (m == MessageBoxResult.No)
    {
        //blank
    }
}

break;

```

Abbildung 4: Screenshot "Datei existiert"-Abfrage

oder nicht. Das sieht für das benutzte Beispiel von vorher folgendermaßen aus:

Diese Vorgehensweise für die Erstellung der Objekte und den Export in .xml-Dateien wird nun für jede Form und jeden Körper wiederholt und in einem case-switch zusammengefasst. Dass dies einiges an Schreibaarbeit ist, und eventuell nicht die optimale Lösung, das ist dem Programmierer aufgefallen, als er bereits fertig war.

4.1.8 Tag 8: Letzte Optimierungen

Da der Exportiere Knopf nur funktionieren soll, nachdem eine Berechnung erfolgt ist, war noch einiges an Optimierungen nötig. Dabei wurde das Programm dahingehend umgeschrieben, dass z.B. die Berechne-Methode erst ausgeführt werden kann, wenn die Eingabe vollständig ist. Dafür wurden if-Anweisungen verwendet. Sobald der Nutzer nichts eingegeben hat, die Felder also leere Strings enthalten, wird der Inhalt als „0“ behandelt. Wenn der Inhalt „0“ ist, dann existiert er für das Programm nicht, wie bereits vorher erwähnt. Lässt der Nutzer wiederum die vorgegebenen Texte in den Boxen, bspw. „Länge“, dann wird der Inhalt als „-1“ behandelt. Das bedeutet, dass der Inhalt fehlerhaft ist, da das Programm nur mit Zahlen rechnen kann und nicht mit Strings oder einzelnen Buchstaben. Für diese Fälle, in denen die Berechnung nicht durchgeführt wird, wird auch der Knopf für den Export nicht aktiviert. Diese hilfreiche Zusatzfunktion bzw. Absicherung stellt auch sicher, dass alle rötlich hinterlegten Textfelder ausgefüllt werden, da ansonsten keine Berechnung möglich wäre bzw. die ausgeführte Berechnung fehlerhaft ist. Dadurch wird verhindert, dass das Programm durch „0“ teilt oder negative Ergebnisse angezeigt werden.

4.1.9 Tag 9: Fertigstellung des Programms

Der 9. Bearbeitungstag war der letzte Tag, an dem der Code editiert bzw. eingesehen wurde. Es wurden zusätzlich Kommentare eingefügt, um das Verständnis des Codes zu verbessern. Des Weiteren wurde ein Icon in die obere linke Ecke des Anwendungsfensters hinzugefügt. Zum Schluss muss das Projekt als ausführbare Datei exportiert und alle genutzten Ordner und Dateien für die Abgabe verpackt werden. Die ausführbare Datei ist als .exe in der Abgabe zu finden und der Projektordner als .zip sowie .rar Archiv.

4.1.10 Tag 10: Verpacken aller Projektbestandteile

Um ein vollständiges und funktionstüchtiges Projekt abzugeben, musste noch einiges getan werden. Zur Abgabe gehören folgende Dinge: die in Punkt 4.1.9 erwähnten Projektordner in .zip und .rar, eine ausführbare Datei in .exe, eine README.txt Datei um Details zur Kompilierung und zur Ausführung mitzuteilen und zum Schluss der schriftliche Beleg in .docx und .pdf Format. Die Abgabe von all diesen Dingen erfolgt mit einem Archiv in dem alles hier genannte zu finden ist.

5 Problemanalyse

In den folgenden Abschnitten werden einzelne Probleme, die während der Programmierung aufgetreten sind, behandelt. Dabei werden insbesondere auf die grundlegenden Anforderungen des Programmes eingegangen. Es werden nur auf schwerwiegende Probleme eingegangen, da diese gelöst werden müssen, um die Funktionalität des Programmes zu gewährleisten. Kleinere Probleme wie z.B. optische Unschönheiten oder die notwendige Änderung einiger anfangs geplanter Funktionen werden in einem späteren Kapitel näher beschrieben und beleuchtet (s. 5.1).

5.1 Nutzung einer ComboBox

Da zur Auswahl der Formen bzw. Körper ein sogenanntes Drop-Down-Menü geplant war, ist es nötig gewesen sich mit diesem auseinanderzusetzen. Die einzelnen Auswahlmöglichkeiten, also die „Items“ in der Combobox werden mit ID's identifiziert. Sie beginnen bei 0 und werden aufsteigend gezählt. Nach dem Hinzufügen jeder Form und jedes Körpers gibt es also insgesamt 11 Auswahlmöglichkeiten. Um damit weiterzuarbeiten muss man sich vergewissern, dass man nicht mit den Zählweisen durcheinanderkommt. Die zugewiesenen ID's werden später gebraucht, um daraus abzuleiten welche Größen (Länge, Höhe, ...) zur Berechnung benötigt werden. Da in dem Programm mit verschiedenen Windows also Fenstern gearbeitet wird, werden die ID's der Auswahlmöglichkeiten eigens als globale Variablen gespeichert und angelegt. Ansonsten wäre es nicht möglich die ID's fensterübergreifend zu verwenden. Anhand der getroffenen Auswahl soll es später möglich sein case-switches durchzuführen bzw. deren Konzept anzuwenden.

5.2 Anlegen von Berechnungsvorlagen

Was hier als „Berechnungsvorlagen“ bezeichnet worden ist, sind die bereits im vorangegangenen Abschnitt erwähnten Ableitungen aus den gewählten ID's. Am einfachsten lässt sich dies mit einem Beispiel erklären. Im Hauptmenü wählt man also das Quadrat aus. Um eine Berechnung darin möglich zu machen, braucht man mindestens die Größe einer einzigen Seite a . Das Quadrat hat im Programm die ID mit der Nummer 1, auch wenn es die 2. Auswahlmöglichkeit ist. Da die Zählweise mit 0 beginnt, liegt die Begründung dafür als auf der Hand. Im Berechnungsfenster befinden sich alle möglichen Größen, die für alle Formen und Körper benutzt oder berechnet werden können. Für das benutzte Beispiel, also dem Quadrat, sind allerdings die meisten Dinge ausgegraut und können demzufolge nicht beschrieben oder benutzt werden. Da das Beispiel eine simple Fläche ist, besitzt diese beispielsweise kein Volumen, wie es ein Körper hätte. Die dringend notwendigen Größen

sind farblich markiert, da ohne diese jegliche Berechnung nicht möglich wären. Dieser Vorgang ist für jede Form oder Fläche anders. Um es einfacher zu beschreiben wurden diese Vorlagen als Schablonen/Templates beschrieben.

5.3 Methode zum Berechnen

Ursprünglich war es geplant, dass das Programm selbstständig dazu in der Lage ist die Eintragungen zu prüfen und zu schauen, ob bereits eine Berechnung möglich ist. Dazu kam die Idee auf einen EventHandler zu verwenden, der „auslöst“ sobald sich der Inhalt der Textboxen verändert hat. Diese Funktion stellte sich allerdings als sehr schwierig zu programmieren heraus. Eigentlich ist die Idee dieser ständigen Überprüfung sehr gut und würde die Funktionalität und generell den Programmfluss positiv beeinflussen, doch hierfür musste eine andere Lösung gefunden werden.

Ein weiterer Ansatz zur Lösung des benannten Problems wäre eine immerwährende Schleife, in der die Inhalte der Textboxen mit dem Standardinhalt abgeglichen werden, doch auch dies war nicht passend. So wäre es nämlich möglich, dass bereits unvollständige Eingaben zur Berechnung verwendet werden.

Letztendlich entschied man sich dazu einen weiteren Button mit der Bezeichnung „Berechne“ hinzuzufügen. Damit ist es möglich die Eingaben vollständig in die Berechnung einfließen zu lassen. Da der Knopf unendlich oft benutzbar ist, kann man nun sogar, während eine fertige Berechnung in den Boxen steht, neue Werte eingeben und erneut berechnen.

5.4 Serialisierung in XML

Die größte Schwierigkeit was hierbei die Funktionalität der Serialisierung zu verstehen. Dabei muss zuerst eine Klasse erstellt werden. Die Klasse stellt das Grundgerüst des xml-Dokumentes dar. So gehören zu der Klasse alle Größen, die man Eingeben und Berechnen kann (Länge, Breite, ...). Des Weiteren war es nötig die Übergabewerte für die Objekte zu deklarieren. Für die Serialisierung muss zuerst ein Objekt erstellt werden, welches nach der Vorlage der zuvor erstellten Klasse kreiert wird. Alle Werte, die es nicht gibt oder welche nicht berechenbar waren werden mit einer 0 beschrieben. Diese 0 steht dafür, dass es keinen Wert gibt. Die Serialisierung verwendet einen XmlTextWriter und einen XmlSerializer. Der TextWriter schreibt, wie es der Name schon verrät, in die Datei während der Serializer die Daten aus dem Objekt liest und in das passende Format umwandelt.

5.5 Bedienungsprobleme

Eine wichtige Frage, die es sich bei der Erstellung des Programmes und dessen Funktionsweise zu stellen gilt, ist folgende: Was passiert, wenn sich der Nutzer nicht an die Vorgaben des Programmes hält? Diese Frage sollte man sich unbedingt stellen, um Probleme zu beseitigen, welche zu einem vorzeitigen Programmabsturz oder einem Fehlverhalten des Programmes führt. Deshalb muss bei der Programmierung einiges beachtet werden. Durch die farbige Kennzeichnung von wichtigen, unverzichtbaren Eingabefeldern kann grundsätzlich die Gefahr von gemachten Fehlern verringert werden, aber diese Handlung beseitigt nicht die Fehler an sich. Dafür müssen weitere Fehlerbehandlungen im Code hinzugefügt werden. So dürfen zum Beispiel kein String und keine Buchstaben in der Eingabe stehen. Das Programm überprüft, ob die Eingabe verändert wurde und reagiert im Fehlerfall folgendermaßen: Es passiert nichts. Dies soll den Benutzer darüber informieren, dass die Berechnung nicht funktioniert hat bzw. nicht möglich war. Er wird es dementsprechend nochmal versuchen nachdem er den Fehler behoben hat. Ein weiterer Eingabefehler ist, dass der Benutzer ein Feld komplett leer lassen kann. Für das Programm ist eine Berechnung mit „0“, also einem nichtexistierenden Maß, nicht möglich. Es reagiert genau wie im vorher behandelten Fehlerfall: Es passiert nichts.

Um zu verhindern, dass bei diesen Fehlern ein Export der Daten in XML möglich ist, geschieht nun ebenfalls eine Abfrage nach betätigen des „Berechne“-Knopfes. Es wird überprüft, ob die Berechnung erfolgreich ist. Dann wird der Export-Knopf erst aktiviert. Sobald bei der Überprüfung irgendeine Größe „0“ oder „-1“ ist, geschieht demnach weder eine Berechnung, noch wird der „Export in XML“-Knopf aktiviert.

6 Fazit

Abschließend lässt sich die Erstellung des Projektes in einigen Punkten zusammenfassen. Diese werden im folgenden Text aufgegriffen und näher beleuchtet. Zuerst begab man sich in die anfängliche Planungsphase. Dabei war das Ziel eine Idee zu finden, welche sich im vorgegebenen Zeitraum umsetzen lässt und nicht übermäßig kompliziert wird. Die finale Idee, der Körper- und Formenrechner, kam wiedererwarten schnell auf und ließ genügend Raum für weitere Ergänzungen für die Funktionalität oder den Umfang. Zu Beginn waren viel mehr Formen und Körper geplant, welche letztendlich keinen Weg in das finale Programm gefunden haben. Das lag daran, dass die Formeln zu komplex oder die Form bzw. der Körper zu unbedeutend waren, um eine Implementierung in dem Rechner zu finden. Nennenswerte Körper dafür sind z.B. der Pyramidenstumpf, der Kegelstumpf oder der Torus.

Darauffolgend erfolgte das Erstellen erster Entwürfe. Viele davon wurden zuerst auf Papier skizziert, wobei es wichtig war die geplanten Funktionen (z.B. Knöpfe oder das Drop-Down Menü) korrekt zu platzieren und einzubeziehen. Letztendlich wurde der optisch ansprechendste Entwurf digitalisiert und ist nun im Kapitel „Programmgerüst“ zu finden. Zu Beginn der Programmierphase wurde die grafische Oberfläche erstellt. Dabei wurde die Funktionalität zunächst völlig ausgelassen. Der Grundgedanke war, eine übersichtliche Adaption des Programmgerüsts zu erstellen. Letztendlich wurden einige optische Änderungen durchgeführt, welche die Funktionalität nicht einschränken sollten. Bei der Erstellung wurde besonders Augenmerk darauf genommen, was für einen Anwender am besten wäre. Dafür musste man sich in die Situation des Nutzers hineinversetzen und sich überlegen, was am simpelsten zu verstehen wäre. Daraus resultierten die umfassende Bedienungsanleitung und die farblich hervorgehobenen Schaltflächen.

Während des Programmierens gab es öfter Probleme mit Visual Studio, da es bei kleineren Fehlern im Code, die die Lauffähigkeit des Programmes ohnehin beeinflussten, öfter zu Abstürzen kam. Dies führte zu Frustration und Motivationsverlust, da sobald der Garbage-Collector anfang zu viel zu sammeln, ein Absturz bereits abzusehen war. Dazu kam die steigende Prozessorauslastung, die bei Programmierung mit gegebener Hardware sehr begrenzt war. Es wurde mit einem Microsoft Surface Go programmiert, was ein flüssiges Arbeiten nahezu unmöglich machte. Die begrenzte Belastbarkeit des Geräts, die niedrige Prozessorleistung und der geringe Speicherplatz sorgte dafür, dass mit einfachsten Mitteln programmiert werden musste. Somit verhinderte der Programmierer, dass sich Visual Studio zunehmen aufhing, sich „festfuhr“ oder komplett abstürzte.

Zusammenfassend lässt sich trotz allem sagen, dass mit relativ mangelhaften Mitteln, wenn man es als solches bezeichnen kann, ein funktionsfähiges und einigermaßen zufriedenstellendes Programm erstellt worden ist. Es erfüllt die geplanten Funktionen und ist optisch übersichtlich und nicht zu abstrakt und unverständlich. Rückblickend auf die Arbeitsweise und das Vorankommen im Allgemeinen lässt sich sagen, dass einiges am Programm verbesserungswürdig wäre. Da der Schwerpunkt beim Programmieren eher auf die Funktionsfähigkeit und Zugänglichkeit gelegt wurde anstelle auf optische Perfektion, ist dieser Kritikpunkt im Nachhinein vertretbar und verständlich.

7 Bedienungsanleitung

In diesem Kapitel wird einem Benutzer schrittweise die Benutzung des Programmes erklärt. Es werden auch auf alle Möglichkeiten der Bedienung eingegangen. Dabei wird vorausgesetzt, dass der Nutzer grundlegend dazu in der Lage ist einen Computer zu benutzen und auch dazu fähig ist die Eingaben durchzuführen. Er sollte auch wissen, dass man Schaltflächen anklicken kann und was ein Drop-Down Menü ist, also dass es ausklappbar ist.

7.1 Start des Programms

Mit dem Start des Programms wird das Hauptmenü angezeigt. Es gibt 4 Schaltflächen. Die obere davon ist ein ausklappbares Drop-Down-Menü. Der Benutzer muss nun entscheiden, welche Form bzw. welchen Körper er vor sich hat und dessen Größen er berechnen will.

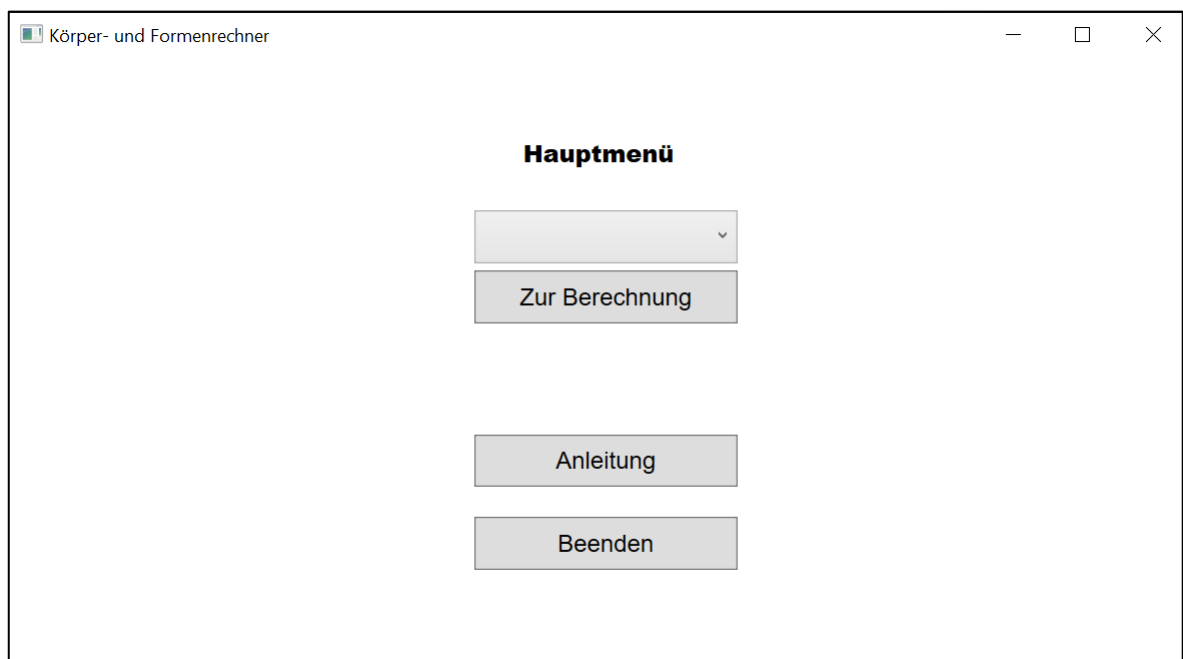


Abbildung 5: Screenshot Hauptmenü

Der Knopf, welcher sich unmittelbar darunter befindet, führt den Benutzer beim Betätigen weiter zu den Eingabefeldern. Er führt den Benutzer nur dann weiter zur Berechnung, wenn er eine Form oder ein Körper im Drop-Down Menü ausgewählt hat. Hat er das nicht, dann passiert nichts. Darunter ist ein Knopf mit der Beschriftung „Anleitung“. Dieser führt den Nutzer zu einer stark vereinfachten Form dieser Bedienungsanleitung.

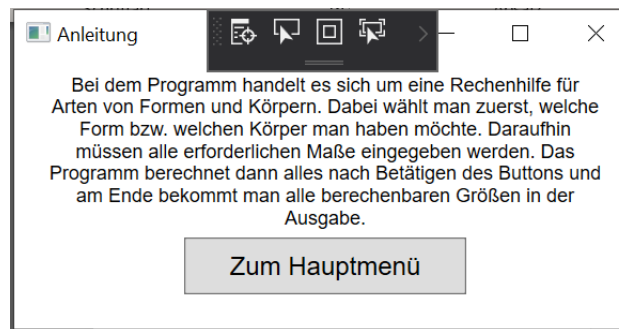


Abbildung 6: Screenshot Anleitung

Der letzte Knopf ist selbsterklärend, er beendet das Programm.

7.2 Eingabe der Daten und Berechnung

Nach der Auswahl der Form bzw. des Körpers im Drop-Down-Menü und nach Betätigen des Knopfes „Zur Berechnung“ hat der Benutzer folgende Anzeige vor sich:

Abbildung 7: Screenshot Berechnung

Hierfür wurde das Standardbeispiel dieses Beleges genommen. Die rosa hinterlegten Flächen sind Felder, welche ausgefüllt werden müssen, um die nicht ausgegrauten Felder zu berechnen. Dafür muss der Benutzer die Bezeichnungen aus den Feldern herauslöschen und eigene Zahlen eintragen. Der Gedanke dahinter war, dass der Benutzer ein Lineal besitzt und die Form oder den Körper direkt vor sich hat, um die Maße auszumessen. Alles was nicht direkt messbar ist, das übernimmt das Programm.

Nach Eingabe der rosa hinterlegten Größen und Betätigen des „Berechne“ Knopfes, wird der Benutzer folgende Anzeige sehen. Die Diagonale, der Umfang und die Fläche wurden anhand von Länge und Breite ausgerechnet. Die Diagonale wurde auf 3 Nachkommazahlen gerundet.

Körper- und Formenrechner

Rechteck

Bitte geben Sie nun die Werte für die Berechnungen ein. Die Maße sollten einheitlich in cm oder m sein.

5	Kathete (a)	10
2	Ankathete (b)	14
5,385	Hypotenuse (c)	Oberfläche
Höhe	Radius	Volumen
Länge 2	Durchmesser	Breite 2

Export in XML Berechne Zurück zum Hauptmenü

Abbildung 8: Screenshot Berechnete Größen

7.3 Export in XML

Nach der Berechnung hat der Benutzer folgende Möglichkeiten: Entweder er möchte seine Werte in einem .xml-Dokument exportieren oder er möchte neue Werte eingeben und erneut berechnen oder er möchte zum Hauptmenü zurückkehren und eine neue Form auswählen oder das Programm beenden. Um die Daten zu exportieren, muss er lediglich die Schaltfläche „Export in XML“ betätigen. Diese Schaltfläche ist erst nutzbar, sobald eine gültige Berechnung stattgefunden hat und alle nicht ausgegrauten Felder einen Wert in sich tragen. Das Programm erstellt nach Betätigen des Knopfes selbstständig eine .xml-Datei mit den berechneten Daten darin. Falls die Datei bereits existiert, wird der Benutzer gefragt, ob er sie überschreiben möchte.

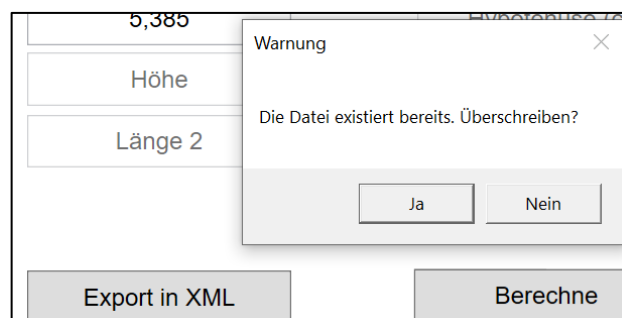


Abbildung 9: Screenshot Abfrage

Anschließend findet der Benutzer eine .xml-Datei, welche auf seinem Desktop erstellt worden ist. (In dem Programm befinden sich lediglich Pfadbeschreibungen des Programmierers, also sollten diese vorher im Falle von durchgeführten Tests manuell geändert werden.)

Literaturverzeichnis

- Becker, F. -M., Boortz, G., Dr. habil. Dietrich, V., Dr. Engelmann, L., Dr. Ernst, C., Dr. habil. Fanghänel, G., Höhne, H., Lenertat, R., Dr. Liesenberg, G., Prof. Dr. habil. Meyer, L., Doz. Dr. habil. Pews-Hocke, C., Dr. Schmidt, G.-D., Dr. habil. Stamm, R., Prof. Dr. habil. Weber, K. (1994). Formeln und Tabellen für die Sekundarstufen I. und II. (5., überarb. Auflage). Berlin, Deutschland: Paetec.

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Frohbürg, den 28.08.2020



Juliet Petzold