

Fachinformatiker Anwendungsentwicklung

# ToBuy Einkaufsliste

E2FI1

Yu Zhu

7-1-2018

## Inhaltsverzeichnis

Erklärung .....	2
Projektidee / Projektziel.....	3
Anforderungsanalyse .....	4
Projektverlauf.....	5
Allgemeines .....	5
Implementierung .....	5
Test.....	5
Klassen-Struktur / UML.....	6
Bedienung .....	7
Artikel Hinzufügen .....	7
Problematiken .....	9
Fazit.....	10

## Erklärung

Hiermit erkläre ich, dass ich die vorliegende Projektarbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe.

Die Stelle der Projektarbeit, die andere Quellen im Wortlaut oder dem Sinn nach entnommen wurden sind durch Angaben der Herkunft kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet.

Ulm, den 27. Juni 2018

Yu Zhu



## Quellen

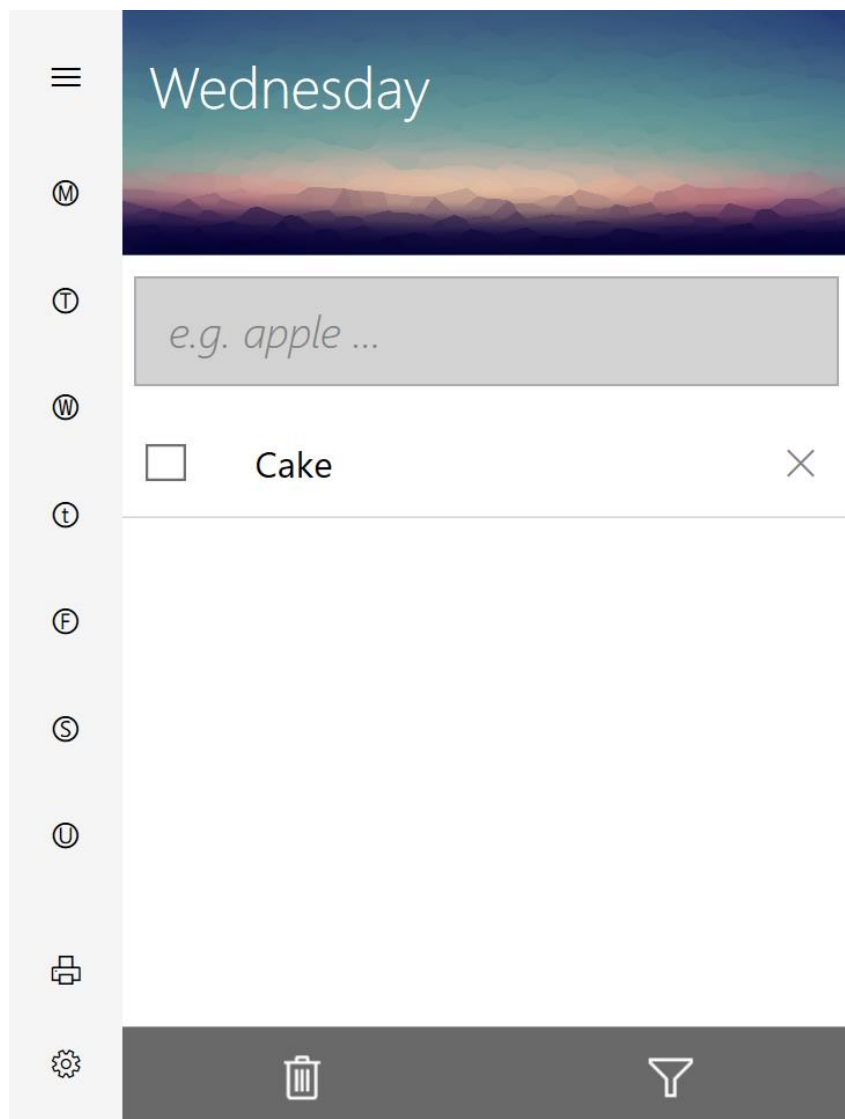
Newtonsoft.Json

- Visual Studio NuGet-Paket

## Projektidee / Projektziel

*ToBuy* ist eine "Todo List" Applikation. Man kann diese benutzen, wenn man einkaufen geht. Die Motivation das Applikation zu entwickeln ist, dass 2018 Neujahrswünsche (eine App für Mama schreiben) erfüllen. C# ist mein erste programmieren Sprache. Das ist auch einen gute Chance C# OOP (Object-oriented programming) programmieren wiederholen. Meine Ausbildungsfirma verwendet meisten .NET Technik. Bei diesem Projekt kann ich WPF Desktop Anwendung Entwicklung lernen. Als Entwickler ist Design Fähigkeit nötig. Das kann ich auch in diesem Projekt üben.

Wenn das Umbenennen abgeschlossen wurde, soll dies folgendermaßen aussehen:



Benutzer kann neue Einkaufens Liste als Wochentage erstellen. Bei jeder Liste kann man neue Artikel hinzufügen. Wenn Benutzer einen Artikel gekauft hat. Er kann sich der Artikel maskieren. Man kann jeden Artikel in einer Liste löschen oder alle Artikel in einer Liste auslassen. Wenn es viele Artikel gibt, man kann die Artikel sortieren. Das ist auch möglich, dass Benutzer die aktuelle Liste ausdrucken.

## Anforderungsanalyse

### LIST BEARBEITEN

- Liste als Hauptelementedarstellung.
- Der User kann einen neuen „ToBuy“ Artikel hinzufügen.
- Der User kann einzelne Artikel von der Liste entfernen.
- Der User kann alle Artikel löschen.
- Der User kann die Liste nach Bedingung sortieren.

### DATEN SPEICHERN

- Der User kann die Daten als JSON Datei in Lokal speichern.
- Wenn die App wieder gestartet wird, können die Daten wieder geladen werden.
- Vor dem Herunterfahren der App, werden die Daten automatisch gespeichert.

### BNUTZERFREUNDLICH

- Zielbenutzer kommt aus China, dieser Umstand muss berücksichtigt werden.
- Bei der PC Applikation, ist das Bearbeiten per Tastatur möglich.

## Projektverlauf

### Allgemeines

#### Mi, 12.06

- Planung des UMLs/Klassen-Struktur
- Vorstellung der Projektidee (genehmigt)

#### Mi, 13.06

- GUI designt
- Klassen erstellen

#### Mi, 15.06

- Icon erstellt und eingebaut
- UI Elemente implementieren

#### Mi, 16.06

- Button und Textbox mit View Model verbinden
- Funktion „Add“ und „Delete“ Item von Liste

#### Mi, 17.06

- Funktion „Sort“ Items von Liste implementiert
- UI optimieren

#### Mi, 18.06

- Daten in Json Datei speichern
- Dokumentation erstellt

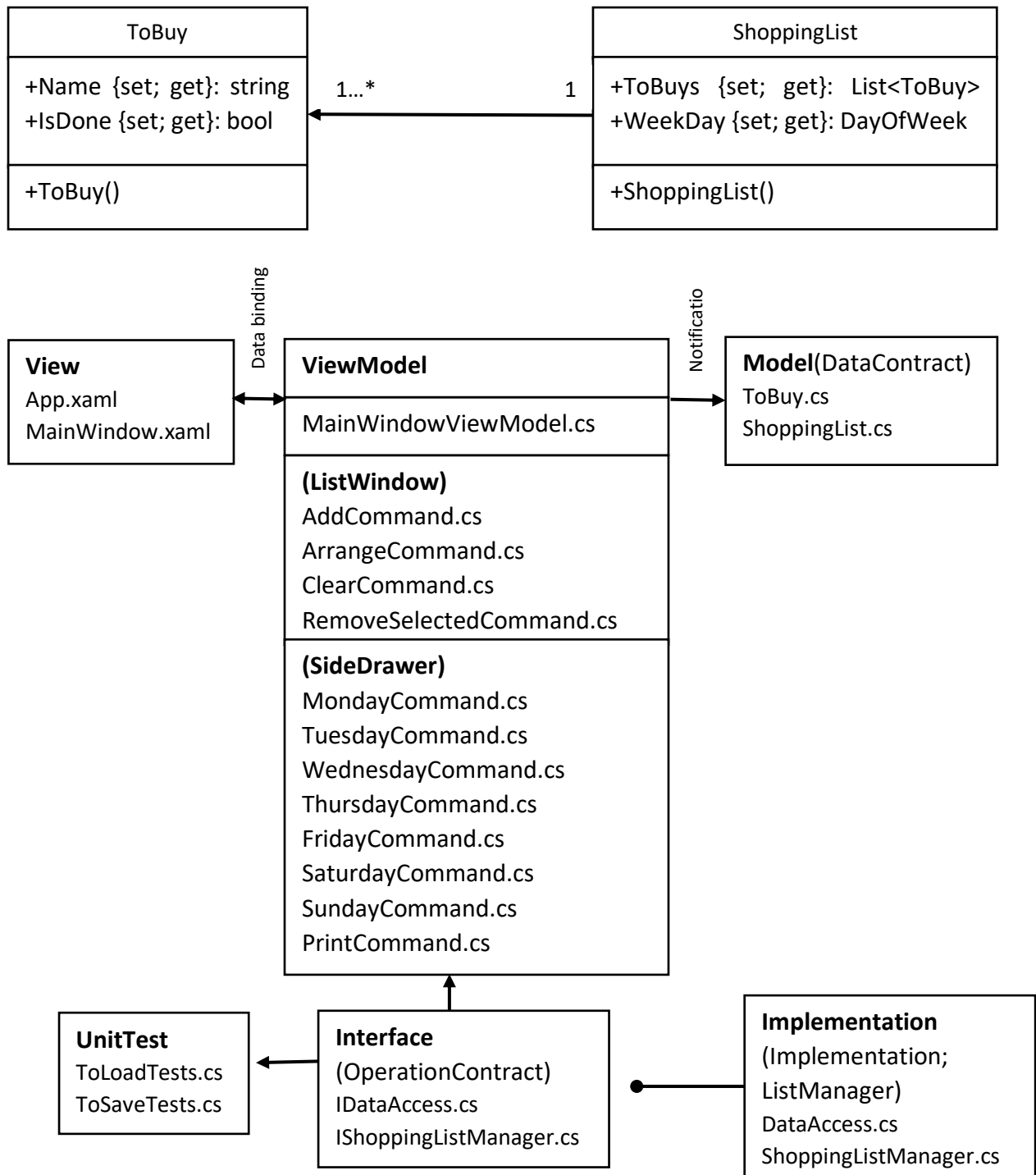
### Implementierung

Bei der Implementierung habe ich als erstes die GUI designt. Ich habe MVVM Pattern verwendet. „ShoppingList“ und „ToBuy“ sind Modal Klassen. Klasse „DataAccess“ benötigt für Daten speichern und laden. Klassen „MainWindowViewModal“ ist mit der grafischen Oberfläche (XAML) verknüpft werden. Jeden Feature zu Beispiel „Add“, „Arrange“, „Remove“, „Clear“, die die Liste Bearbeiten kann. Sie haben eigene Klasse. Damit kann man den Code besser organisieren.

### Test

Während der Entwicklung wurden bereits einige Tests durchgeführt um die Funktion der Methoden sicherzustellen. Die Methoden „ToLoad()“ und „ToSave()“ benötigen für die Daten als Json Datei zu speichern. Dafür gibt es Unit Test durchführen um die Methoden sicherzustellen.

## Klassen-Struktur / UML

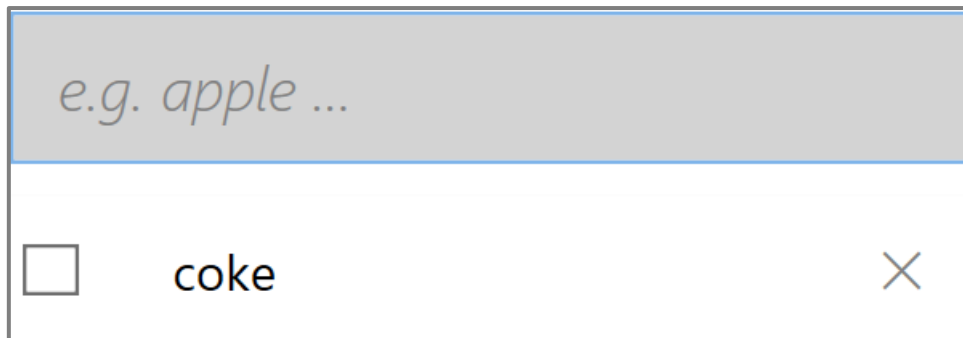


Das Programm setzt sich aus zwei Klassen zusammen. Einmal die Klasse „ToBuy“, welche die Info des Einkaufens Artikel zu bezeichnen. Die Klasse „ShoppingList“ enthält mehrer Elemente des Type “ToBuy”. Die Klasse „ShoppingList“ erbt von Klasse “ToBuy”. Die Struktur das Programm ist bei MVVM Pattern zugeordnet. „View“ ist mit „ViewModel“ verbunden. Dabei reflektieren die kompletten Funktionen der zwei Klassen auf die grafische Oberfläche. „ViewModel“ kommuniziert hmit „Model“.

## Bedienung

### Artikel Hinzufügen

Benutzer kann neuer Artikel Hinzufügen. Aus besser User Experience Gründen. Ein Button für Artikel Hinzufügen ist unnötig. Benutzer kann direkt mit der Tastatur (Button Enter) bearbeiten. Die neuen Artikel wird auf Einkaufen Liste zeigen.



The image shows a user interface for adding a new item to a shopping list. It consists of a text input field with a light gray background and a blue border. Inside the field, the placeholder text "e.g. apple ..." is written in a light gray, italicized font. Below the input field is a white rectangular area. On the left side of this area is a small, empty square checkbox. To the right of the checkbox is the text "coke" in a black, sans-serif font. On the far right of this white area is a small, gray "X" icon, which serves as a delete button for the item.

### Artikel markieren und löschen

Man kann mit jedem Artikel bearbeiten. Wenn Benutzer den Button "X" drücken, welche nach dem Artikel ist. Die Artikel werden gelöscht. Wenn Benutzer die Checkbox markieren, das bedeutet die Artikel schon gekauft hat.

### Liste löschen und sortieren

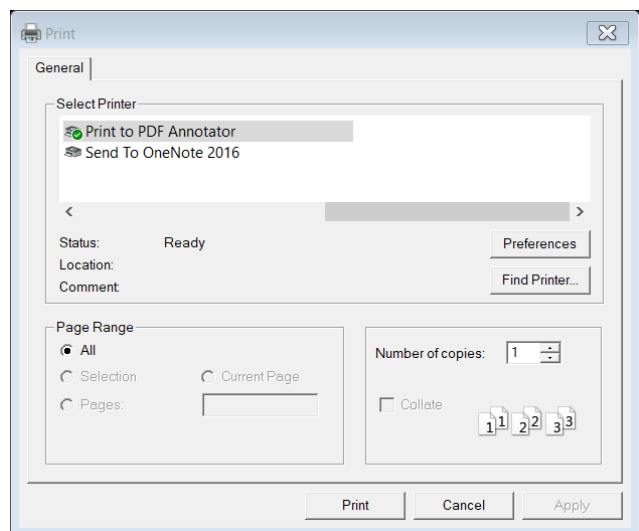
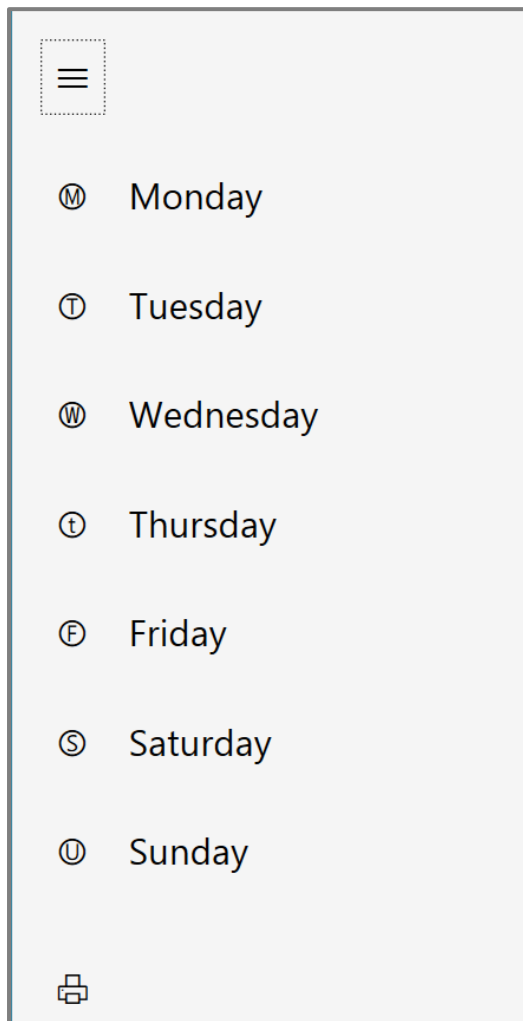
Man kann auch alle Artikel eine Liste einmal löschen. Das braucht man die Mülleimer Button drücken. Wenn man die sortieren Button drücken, alle Artikel in eine Liste wird sortiert. Die Artikel, die noch nicht markiert hat. Sie werden zuerst in der Liste zeigen.



### Sidebar

Oben links befindet sich eine „Hamburg Button“. Klickt man auf diesem Button und die Seiten Bereich wird sich nach rechts erstrecken. In diesem Bereich enthält eine Einkaufslisten Button von Montag bis Sonntag sowie eine Druck Button.





### Wochentag Button und Druck Button

Tippen auf die verschiedenen Wochentags-Buttons und die Einkaufsliste wechselt zwischen dem verschiedenen Wochentag. Daher können Benutzer separate Einkaufslisten für verschiedene Wochentags hinzufügen. Klicken auf dem Drucken-Button, um die Windows-Druckeinstellungen Fenster zu öffnen. Der Benutzer kann die aktuelle Einkaufsliste ausdrucken.

## Problematiken

Nachdem die Anwendungsentwicklung abgeschlossen ist, läuft die Anwendung ordnungsgemäß. Bei der Entwicklung dieser Anwendung wurde mir klar, dass aus Nutzersicht eines der wichtigsten Merkmale einer Einkaufsliste ist, dass sie leicht und tragbar ist. Da es sich bei der Software jedoch um eine WPF-Anwendung handelt, können nur Windows-PCs diese Anwendung ausführen. Personal Computer sind groß und nicht für Benutzer geeignet, um auf den Markt zu bringen. Die Lösung besteht darin, einen Druckknopf für den Benutzer vorzubereiten, und der Benutzer kann die Einkaufsliste drucken und sie zur Verwendung auf den Markt bringen. Aber diese Lösung ist nicht die ideale Lösung.

## Fazit

Das Ergebnis dieses Projekts war relativ zufriedenstellend, es hat auch alle vorgegebenen Ziele erreicht und die grundlegende Funktion einer Einkaufsliste erreicht.

Bei diesem Projekt hatte ich die Möglichkeit, objektorientierte Programmierung auf Basis von C# zu üben und zu lernen, eine WPF-Anwendung in der WPF-Entwicklungsumgebung zu entwickeln. Außerdem habe ich die Möglichkeit, meine eigenen Designfähigkeiten zu trainieren.

Es sind noch ein paar „Nice-to-have“ Features dazu gekommen, die ich persönlich für sehr relevant erachte. Ein „Nice-to-have“ ist die Möglichkeit um z.B. Cross Platform möglichkeit. Die Möglichkeit suchen, die App in eine App Cross Platform umzuwandeln. Das Projekt könnte als Android oder iOS Applikation erweitern zu entwickeln. Die andere Möglichkeit ist als Web Applikation übertragen.