|  |  |
| --- | --- |
| HÜ02 | Abgabetermin: 9.1.2023 23:59 |
| Name: Johannes Werner | Aufwand in h: 5 |
|  | Punkte: |

## Chatbot 8(+2) Punkte

Erstellen Sie einen einfachen Chatbot nach dargestelltem Muster im Ablaufdiagramm.

Der Nutzer wird vom Chatbot begrüßt und gleich aufgeklärt, dass er die Unterhaltung mit „bye“ beenden kann. Dann wird auf die Nutzereingabe gewartet.

Liegt die Nutzereingabe vor, wird dir dieser Satz in Einzelteile (sprich Wörter) zerlegt. Die einzelnen Wörter werden harmonisiert (kleingeschrieben).

Dann wird überprüft, ob Einzelwörter in der Datenbasis vorkommen, die als „passende“ Antwort vorliegen. Dazu soll beim Programmstart eine Datei eingelesen werden und der Inhalt dieser Datei in einem generischen Dictionary gespeichert werden. Das Wort vor dem Doppelpunkt stellt das Wort dar, auf welches reagiert werden soll. Nach dem Doppelpunkt ist die Antwort enthalten.

Beispielhafter Inhalt dieser Datei:

hallo:aber Hallo

geht:Was verstehtst du darunter?

essen:Ich habe leider keinen Geschmackssinn

Wenn eine passende Antwort vorliegt, wird diese als Feedback ausgegeben.

Liegt keine passende Antwort vor, wird eine zufällige Antwort vom Chatbot zurückgegeben, um das Gespräch aufrecht zu erhalten. Die Zufällige Antwort kommt aus einer zweiten Datei. Der Inhalt dieser Datei soll in einer generischen Liste gespeichert werden.

Beispielhafter Inhalt dieser Datei:

Oh, wirklich?

Interessant…

Das kann man so sehen.

Erzähle mir mehr davon.

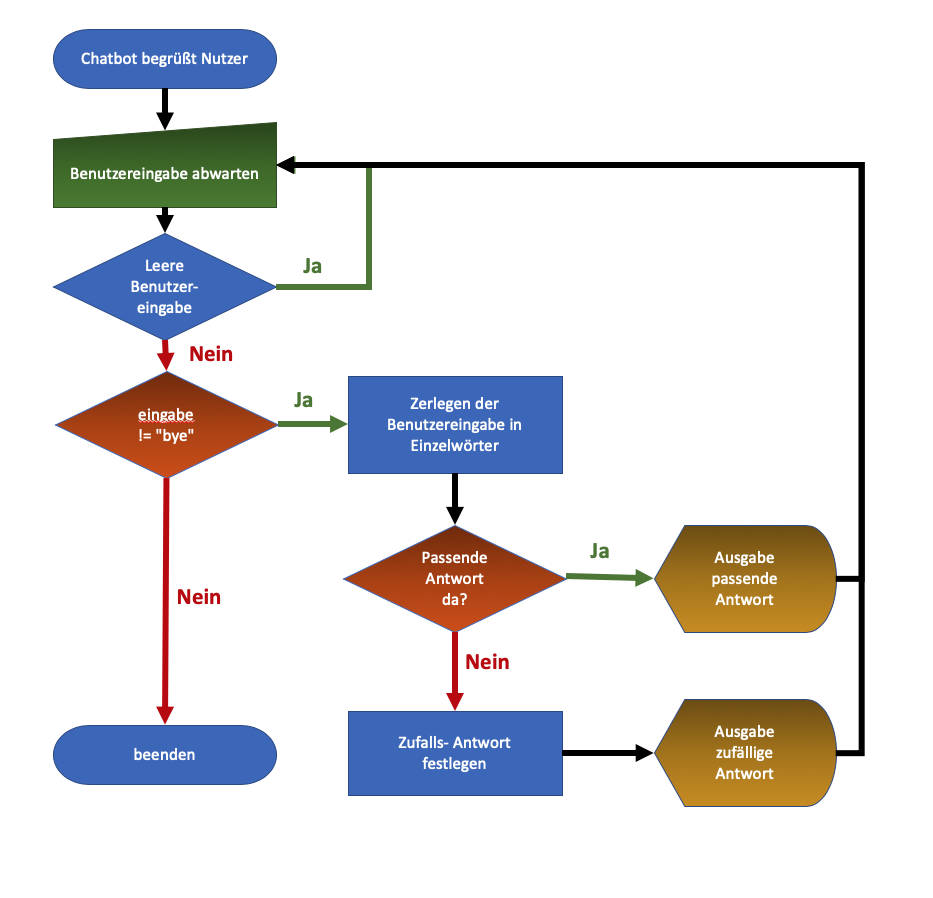
Ich verstehe.

Das verstehe ich jetzt aber nicht.

Der Chatbot soll zweisprachig umgesetzt werden. Definiere dazu eine Enumeration mit den Werten Deutsch und Englisch. Nach der Begrüßung durch den Chatbot soll der Benutzer gefragt werden, in welcher Sprache er kommunizieren möchte. Es sind daher 4 Dateien mit entsprechendem Inhalt notwendig. Verwende für die Unterscheidung der Sprache jeweils die Enumeration – Anmerkung: Ich finde diese Lösung mit Enum unpraktikabel, da mir keine einfach erweiterbare Eingabeüberprüfung eingefallen ist, weshalb ich mich dazu entschieden habe es so zu lösen, wie ich es gemacht habe – ohne Enum.

Erstelle eine Konsolenapplikation mit einer Klasse ChatBot. Eine Methode StartChat soll den Chat starten.

Zusatzaufgabe: Es soll die Möglichkeit geschaffen werden, die Sprache während der Unterhaltung zu wechseln. Überlege eine sinnvolle Vorgangsweise.



# Idea of solution

First, I thought how I want to build my application. Actually, I started working on it before reading the instructions fully. Still, everything was assam but one point, but more on that later. I wanted to build my program very structured just like how I worked at the internship. That meant almost no code duplication and a lot of functions. So is coded everything. After that I fully read the instructions and saw a difference in my implementation to the instructions. The way of handling different languages. The instructions show that we should use an Enum, but I could not find any good and reasonable implementation for this problem. So, I implemented it like I found it best (hopefully someone can show me how to use a Enum in this case). After thinking about this problem for a long time I fixed a few bugs and polished the text files.

# Code

Class ChatBot:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Hue02

{

internal class ChatBot

{

string input = "";

string currentLanguage = "en";

string lastAnswer = "";

List<string> smallTalk = new List<string>();

Dictionary<string, string> languages = new Dictionary<string, string>()

{

{"en", "English"},

{"de", "Deutsch"}

};

Dictionary<string, string> answers = new Dictionary<string, string>();

Dictionary<string, string> commands = new Dictionary<string, string>();

Random rnd = new Random();

/// <summary>

/// Param <param>language</param><para />

/// Availibel languages

/// <list type="bullet">

/// <item>

/// <description>'en' for English</description>

/// </item>

/// <item>

/// <description>'de' for Deutsch</description>

/// </item>

/// </list>

/// </summary>

public ChatBot(string language = "en")

{

this.currentLanguage = language;

}

// main loop for the bot

public void StartChat()

{

loadTextFiles();

Console.WriteLine(this.commands["startingChatBot"]);

Console.WriteLine(this.commands["startedChatBot"]);

Console.WriteLine("Chatty - " + this.commands["changeLanguage"]);

Console.Write(" - ");

this.input = readInput();

while (input != "bye")

{

if (this.input.Length > 0)

{

answerToInput();

}

Console.Write(" - ");

this.input = readInput();

}

}

string readInput()

{

return Console.ReadLine().ToLower().Replace('.', ' ').Replace(',', ' ').Replace('-', ' ');

}

void answerToInput()

{

string[] parts = input.Split(' ');

foreach (string part in parts)

{

if (answers.ContainsKey(part))

{

if(part == this.lastAnswer)

{

break;

}

Console.WriteLine("Chatty - " +answers[part]);

this.lastAnswer = part;

return;

}

else if(part == this.commands["#language"])

{

changeLanguage();

return;

}

}

int randdomElement = rnd.Next(smallTalk.Count);

if(randdomElement.ToString() == this.lastAnswer)

{

int counter = 0;

do

{

randdomElement = rnd.Next(smallTalk.Count);

counter++;

} while (randdomElement.ToString() == this.lastAnswer && counter <3);

}

this.lastAnswer = randdomElement.ToString();

Console.WriteLine("Chatty - " + smallTalk[randdomElement]);

}

string createPathToFile(string file)

{

return Directory.GetCurrentDirectory().Split("bin")[0] + @"text\_files\" + this.currentLanguage + @"\" + file;

}

void loadCommandsFromFile(ref Dictionary<string,string> output,string path)

{

output = new Dictionary<string, string>();

string[] lines = System.IO.File.ReadAllLines(path);

foreach (string line in lines)

{

try

{

string[] parts = line.Split(':');

output.Add(parts[0], parts[1]);

}

catch (IndexOutOfRangeException)

{

Console.WriteLine("The is a error in the language file \'" + path + "\'. The line starting with \'" + line.Split(':')[0] + "\' cann't be loaded!");

}

}

}

void loadLinesFromFile(ref List<string> output, string path)

{

output = new List<string>();

string[] lines = System.IO.File.ReadAllLines(path);

foreach (string line in lines)

{

output.Add(line);

}

}

void loadTextFiles()

{

loadCommandsFromFile(ref this.answers, createPathToFile("answers.txt"));

loadCommandsFromFile(ref this.commands, createPathToFile("commands.txt"));

loadLinesFromFile(ref this.smallTalk, createPathToFile("small\_talk.txt"));

}

void changeLanguage()

{

Console.WriteLine("Chatty - " + this.commands["changeLanguageText1"]);

foreach (KeyValuePair<string, string> language in languages)

{

Console.WriteLine("\t" + language.Key + " - " + language.Value);

}

Console.WriteLine("Chatty - " + this.commands["changeLanguageText2"]);

Console.Write(" - ");

string input = readInput();

if (languages.ContainsKey(input))

{

this.currentLanguage = input;

}

else

{

this.currentLanguage = "en";

}

loadTextFiles();

Console.WriteLine("Chatty - " + this.commands["languageChanged"]);

}

}

}

Programm:

using Hue02;

ChatBot chatty = new ChatBot();

chatty.StartChat();

# Example

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung