УДК 004

Создание программы-переводчика на языке программирования C#

Эрдман Александр Алексеевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В статье рассмотрен процесс создания программы-переводчика текста с различных языков. Программа написана на языке программирования C# с использованием облачного сервиса машинного перевода компании Яндекс. Результатом исследования будет являться оконное приложение-переводчик и описание его работы.
Ключевые слова: C#, API, приложение

Creating a translator program in the C# programming language

Erdman Alexander Alekseevich
Sholom-Aleichem Priamursky State University
Student

Abstract
The article describes the process of creating a text translator program from various languages. The program is written in the C# programming language using the Yandex cloud machine translation service. The result of the study will be a windowed translator application and a description of its work.
Keywords: C#, API, application

1 Введение

1.1 Актуальность

В современном мире возможность общаться с людьми из разных культур и языков становится все более важной. Однако языковые барьеры часто могут мешать эффективному общению. В данном случае на помощь приходят программы-переводчики. Эти программы используют компьютерные алгоритмы для преобразования текста с одного языка на другой, что позволяет людям эффективно общаться без необходимости свободно владеть несколькими языками. В данном исследование рассмотрен процесс разработки программы-переводчика с использованием языка программирования C# и Yandex Translate API. C# — это популярный язык программирования, который широко используется в разработке ПО, что делает его отличным выбором для создания программы-переводчика. В статье представлен процесс создания приложения-переводчика, а также ключевые концепции и приемы разработки.

1.2 Обзор исследований

Т.В. Ромашкина и В.И. Шагалиев рассмотрели основополагающие возможностей языка C# для создания программ [1]. Д.И. Максимов рассмотрел принципы разработки модульных приложений на основе Managed Extensibility Framework, а также основные возможности фреймворка [2]. П.И. Кучинский, Ф.Г. Зограф, С.И. Трегубов и другие разработали программу на языке C# с использованием макросов и API-технологий [3]. А.Х. Альхузайи и С.М. Куценко в своём исследование рассмотрели процесс создания приложения на C#, а также аспекты использования запросов API [4]. В.Н. Зуева, И.А. Грудноv изучили применение API при разработке программного обеспечения на примере API социальных сетей [5]. И.С. Быкова и А.А. Горбунов в своей статье рассмотрели принципы реализации целевой функции путем применения инструмента API [6].

1.3 Цель исследования

Целью исследования является создание программы-переводчика текста с помощью языка программирования C# и API Яндекс.Переводчика.

2 Материалы и методы

Для создания программы используется язык программирования C#, библиотека Windows Forms и API Яндекс.Переводчик. В качестве IDE используется Visual Studio.

3 Результаты и обсуждения

Для начала создания программы требуется создать проект на базе библиотеки WinForms – библиотеки создания оконных приложений. Кроме библиотеки WinForms требуется подгрузить через диспетчер пакетов (NuGet) дополнительный модуль YandexLinguistics.NET, который позволяет работать с достаточно большим числом различных Яндекс API для работы с текстом. После подключения всех модулей создаётся дизайн приложения. Дизайн приложения представлен с минимальным количеством графических объектов: button(кнопка) – кнопка, которая будет использоваться для вызова каких-либо событий в приложения, например, вызов функции перевода текста; comboBox(выпадающий список) - элемент управления, который позволяет пользователю выбирать элемент из раскрывающегося списка или вводить собственный текст, а именно список языков; menuStrip(меню) – простое меню, с помощью которого можно управлять приложением; richTextBox(поле для ввода текста) – область, которая позволяет вводить текст и отображать его. Для добавления списка языков в элемент comboBox требуется использовать свойство «Edit items», которое представляет из себя текстовое поле, в которое записаваются варианты выпадающего списка, причём для разделения вариантов (языков) используется написание варианта отдельно на новой строчке. Общий вид приложения выглядит прост и интуитивно понятен для конечного пользователя (рис. 1).

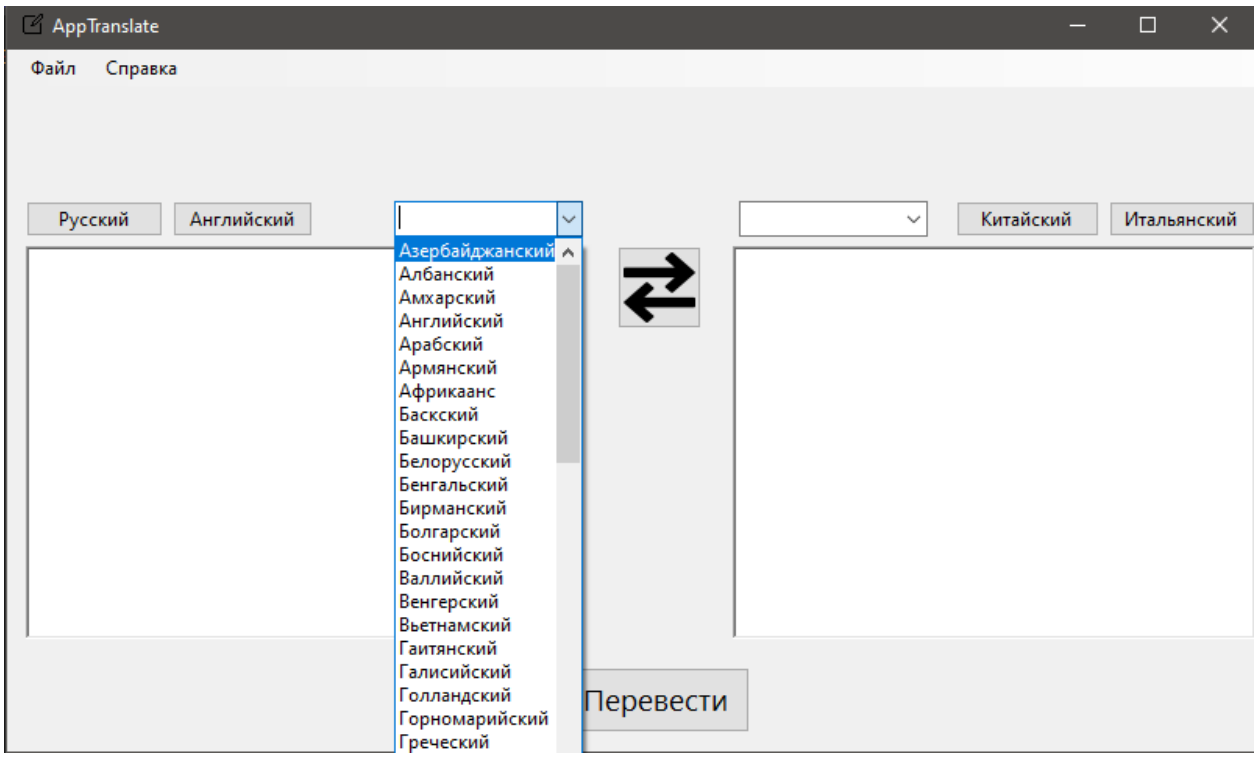

Рисунок 1. Общий вид программы

Графическая составлявшая приложения готова. Далее следует получение ключа API Яндекс.Переводчика и программирования функций. Получение ключа API осуществляется на официальном сайте Яндекс в разделе «разработчики/ключи». Для получения ключа необходимо иметь аккаунт Яндекс. После получения ключа все приготовления закончены и далее следует программирование приложения.

Основным классом приложения выступает public partial class Form1. В данном классе будет прописан весь основной функционал программы. Для работы со словарём будет использоваться отдельный класс для удобства и практичности его редактирования.

В классе Form1 прописывается функциональность кнопок, то есть обработчики нажатия на кнопки, за исключением кнопки «Перевести». В приложение имеются четыре кнопки с названием часто переводимых или популярных языков: Русский, Английский, Китайский и Итальянский. В данном случае считается, что часто переводят русские и английские тексты на китайский и итальянский язык. Элементы comboBox1 и comboBox2 позволяют выбирать пользователю с какого на какой язык осуществить перевод. Вышеупомянутые кнопки с популярными языками позволяют пользователю при нажатии на них выбирать в comboBox1 или comboBox2 соответствующие языки. На программном уровне данные возможности реализованы с помощью обработчика нажатия на кнопку (рис. 2). С помощью метода private void button_ Click реализуется непосредственно само нажатие кнопки. Внутри данного обработчика находится метод selectedItem для объекта comboBox, который позволяет выбрать соответствующий язык из списка. Для того, чтобы

в начале в comboBox по умолчанию уже был выбран какой-либо язык, также используется метод selectedItem.

```
public Form1()
{
    InitializeComponent();

    comboBox1.SelectedItem = "Русский";
    comboBox2.SelectedItem = "Английский";
}

1 reference
private void button1_Click(object sender, EventArgs e)
{
    comboBox1.SelectedItem = "Русский";
}

1 reference
private void button5_Click(object sender, EventArgs e)
{
    comboBox1.SelectedItem = "Английский";
}

1 reference
private void button2_Click(object sender, EventArgs e)
{
    comboBox2.SelectedItem = "Китайский";
}

1 reference
private void button3_Click(object sender, EventArgs e)
{
    comboBox2.SelectedItem = "Итальянский";
}
```

Рисунок 2. Функционал кнопок с выбором языков и выпадающего списка

В интерфейсе программы присутствует кнопка быстрой смены языков. Для реализации её функциональности необходимо создать резервный экземпляр класса элемента comboBox, который позволит осуществить быструю смену значений comboBox. Для этого объявляется глобальное поле класса ComboBox cb. В классе Form1 для созданного выше поле класса cb присваивается новое значение comboBox. Присвоить значение через метод selectItem не получится, так как созданный comboBox не имеет никаких значений, а значит ему не на что ссылаться. Чтобы иметь возможность использовать метод выбора элемента в выпадающем списке, необходимо данный список (коллекцию значений comboBox) добавить. Чтобы создать

коллекцию значений используется метод AddRange (добавление массива). Значения коллекции копируются из уже существующих comboBox элементов. После вышеописанных действий резервный comboBox можно использовать. Код кнопки быстрой смены языка включает в себя обработчик событий нажатия на кнопку, а также методы selectItem у элементов comboBox (рис. 3).

```
private void button6_Click(object sender, EventArgs e)
{
    cb.SelectedItem = comboBox2.SelectedItem;
    comboBox2.SelectedItem= comboBox1.SelectedItem;
    comboBox1.SelectedItem = cb.SelectedItem;
}
```

Рисунок 3. Программа кнопки быстрой смены языка

Графический интерфейс запрограммирован. Для реализации функционала перевода создаётся новый класс (Translater.cs), в котором реализуются методы модуля YandexLinguistics.NET. В данном классе инициализируется ранее полученный ключ API с помощью строковой константы (const string). Для инициализации модуля YandexLinguistics.NET используется конструктор класса class Translater, в котором также указывается (передаётся) ключ API. Для множественно выбора и определения языков создаётся конструкция switch. Эта конструкция является частью метода GetLangPair, который имеет свойства входных и выходных параметров. Данный метод возвращает экземпляр класса LangPair, который получает метод tr.Translater. Конструкция switch представлена в двух вариантах выбор входных параметров и выходных (рис. 4).

```
switch (inputLang)
{
    case "Азербайджанский":

        lp.InputLang = Lang.Az;

        break;
    case "Албанский":

        lp.InputLang = Lang.Sq;

        break;
```

```
switch (outputLang)
{
    case "Азербайджанский":

        lp.OutputLang = Lang.Az;

        break;
    case "Албанский":

        lp.OutputLang = Lang.Sq;

        break;
```

Рисунок 4. Реализация конструкции множественного выбора switch (фрагменты конструкции)

Дале реализуется метод перевода текста (рис. 5).

```
public string Translator(string wordToTranslate, LanguagePair)
{
    return tr.Translate(wordToTranslate, langPair).Text;
}
```

Рисунок 5. Метод перевода текста

Этот метод принимает два параметра: слово, которое нужно перевести ("wordToTranslate") и пару языков ("langPair"), которая указывает исходный и целевой языки. Метод возвращает перевод слова на указанный целевой язык в виде строки. На данном методе класс Translater закончен. Для последней кнопки «Перевсти» создаётся обработчик события в классе Form1, в которой подключается класс Translate для возможности использования функций класса (рис. 6).

```
private void button4_Click(object sender, EventArgs e)
{
    richTextBox2.Clear();
    richTextBox2.Text = t.Translator(richTextBox1.Text, t.GetLangPair(comboBox1.SelectedItem.ToString(), comboBox2.SelectedItem.ToString()));
}
```

Рисунок 6. Функционал кнопки перевода

В данном обработчике задаётся параметр очистки второго richTextBox элемента для того, чтобы удалять предыдущий текст перевода из данного элемента. Параметр richTextBox2.Text принимает значение метода t.Translator, который в свою очередь возвращает переведённый текст. После окончания программирования кнопки «Перевести» приложение готово к использованию (рис. 7,8).

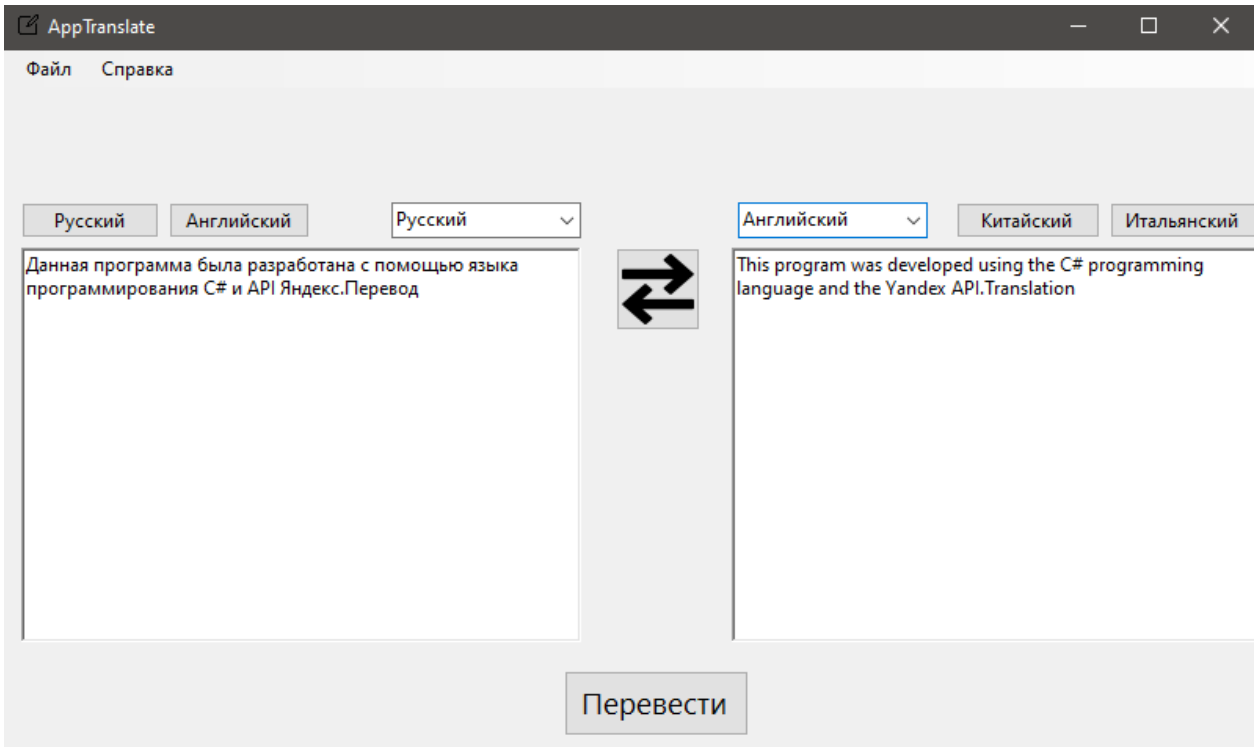

Рисунок 7. Результат работы приложения

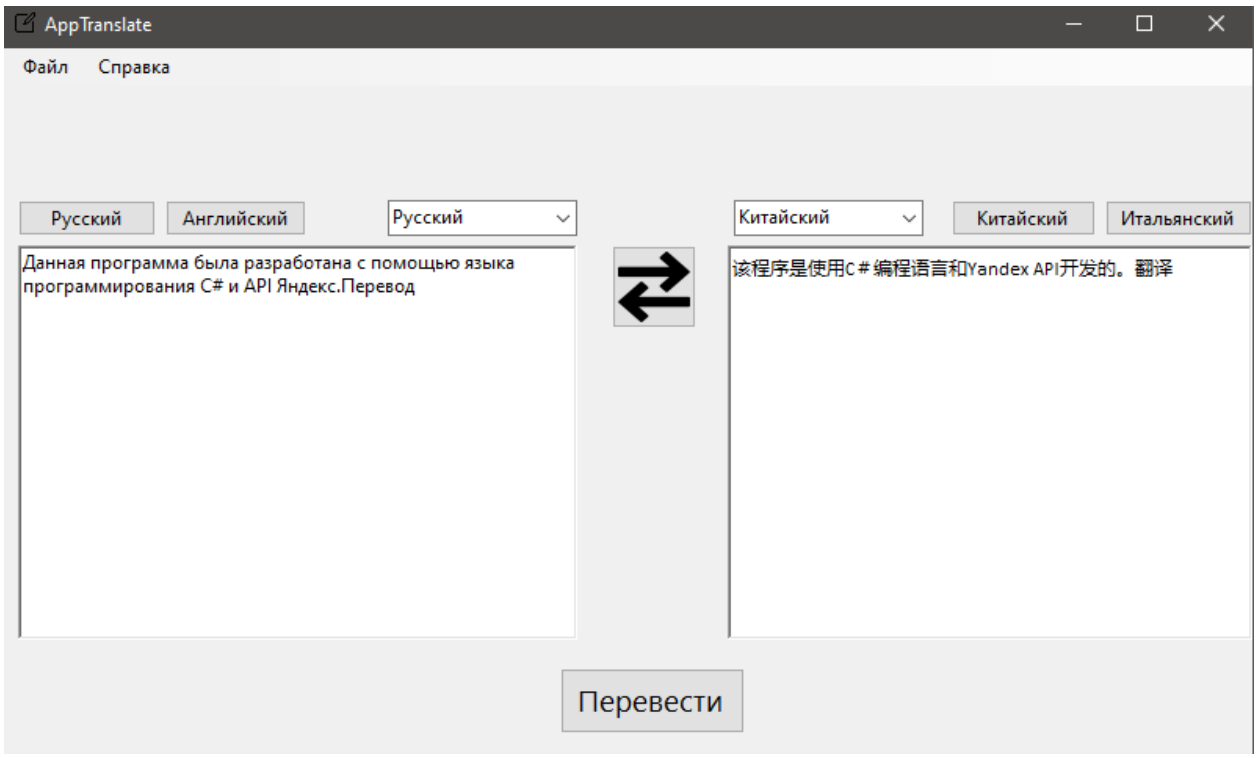

Рисунок 8. Результат работы приложения

Для полного завершения приложения программируется меню управления, а именно нажатие на пункты «Выход» и «О программе». Для этого в главном классе Form1 создаются два обработчика событий. Первый обработчик закрывает приложение, используя команду Exit; второй обработчик выводит сообщение о программе с помощью элемента MessageBox и метода команды Show (рис. 9).

```
private void выходToolStripMenuItem_Click(object sender, EventArgs y)
{
    Application.Exit();
}
0 references
private void оПрограммеToolStripMenuItem_Click(object sender, EventArgs y)
{
    MessageBox.Show("Приложение переводчик\n Перевод осуществлён с помощью сервиса 'Яндекс.Переводчик'");
}
```

Рисунок 9. Функции выхода из приложение и показа справки

Выводы

Таким образом на базе библиотеки WinForms с помощью языка программирования C# и API Яндекс.Перевод была создана программа с графическим интерфейсом и полным функционалом для перевода текста.

Библиографический список

1. Ромашкина Т.В., Шагалиев В.И. Создание приложений для Windows средствами языка C# // Хроники объединенного фонда электронных ресурсов Наука и образование. 2015. № 12 (79). С. 115.
2. Максимов Д.И. Разработка модульных приложений на C# // В сборнике:

Системы управления, технические системы: устойчивость, стабилизация, пути и методы исследования. Материалы молодежной секции в рамках IV Международной научно-практической конференции. 2018. С. 205-211.
3. Кучинский П.И., Зограф Ф.Г., Трегубов С.И., Маринушкин П.С., Левицкий А.А. Разработка системы автоматизированного проектирования радиаторов охлаждения электронных компонентов на основе API-технологий // Инженерный вестник Дона. 2015. № 3 (37). С. 62.
4. Альхузайи А.Х., Куценко С.М. Тонкости написания API-интерфейса на языке C# // Студенческий. 2019. № 41-1 (85). С. 31-33.
5. Зуева В.Н., Груднов И.А. Применение API социальных сетей при разработке программного обеспечения // В сборнике: Современные электротехнические и информационные комплексы и системы. Материалы III Международной научно-практической конференции студентов, аспирантов и преподавателей. 2021. С. 150-154.
6. Быкова И.С., Горбунов А.А. Принципы реализации целевой функции путем применения инструмента API // В сборнике: Университетский комплекс как региональный центр образования, науки и культуры. материалы Всероссийской научно-методической конференции. Министерство образования и науки РФ, ФГБОУ ВО "Оренбургский государственный университет". 2018. С. 633-635.