

# **Uniwersytet WSB Merito w Toruniu**

Wydział Finansów i Zarządzania w Toruniu

Kierunek: Informatyka w biznesie

## **Projekt inżynierski**

**Opracowanie aplikacji desktopowej oferującej funkcjonalność słownika języka  
angielskiego i niemieckiego wraz z modulem do nauki słownictwa**

Autorzy:

Tomasz Izydoreczyk, 40389

TORUŃ, 2025

|   |    |
|---|----|
| WSTĘP .....   | 3  |
| Rozdział I Opis planu stworzenia aplikacji .....                      | 4  |
| Rozdział II Kod aplikacji głównej .....                               | 10 |
| Rozdział III Kod aplikacji do nauki słownictwa .....                  | 25 |
| Rozdział IV Opis mechaniki działania aplikacji .....                  | 31 |
| Rozdział V Opis metody widzę, piszę i czytam, to czego się uczę ..... | 34 |
| ZAKOŃCZENIE .....   | 36 |
| SPIS WYKORZYSTANYCH ŹRÓDEŁ I OPRACOWAŃ .....                          | 37 |
| SPIS RYSUNKÓW .....   | 38 |
| SPIS TABEL .....  | 40 |

## WSTĘP

Na rynku oprogramowania dostępnych jest wiele różnych aplikacji do nauki. Nie są to jednak aplikacje dedykowane konkretnej dziedzinie, lecz ogólnie nauce. Przykładem jest aplikacja Anki, która opiera swoje działanie na powtarzaniu treści, które stworzy użytkownik. Istnieją aplikacje typu GoogleTranslate<sup>1</sup> lub DeepL<sup>2</sup>. Aplikacje takie jednak nie łączą się z żadnym zorganizowanym źródłem danych, takim jak np. słowniki języków obcych. Dlatego postanowiłem stworzyć taką aplikację.

Celem tego projektu jest stworzenie aplikacji oferującej funkcjonalność słownika języka angielskiego i niemieckiego oraz modułu do nauki słownictwa. Założenia tej aplikacji to możliwość zorganizowanego przechowywania danych w sposób usystematyzowany oraz możliwość stworzenia na ich podstawie osobnych zestawów danych służących nauce słownictwa. Zestawy służące nauce w założeniu będą obsługiwane przez osobny moduł aplikacji o nazwie „Nauka”. Moduł ten będzie umożliwiał rozróżnienie źródła danych, tzn. czy słowa do nauki pochodzą ze słownika języka angielskiego lub niemieckiego. Dodatkowo nauka w tym module będzie opierała się o metodę „widzę, wpisuję i czytam” to czego się uczę. W poniższym projekcie zaprezentowano proces tworzenia dwóch aplikacji oferujących wyżej wymienione funkcjonalności oraz udowodnienie założonej hipotezy.

W rozdziale pierwszym zawarto opis planu stworzenia aplikacji desktopowej oferującej funkcjonalność podwójnego słownika językowego – niemiecki i angielski wraz z modulem do nauki słownictwa. W rozdziale drugim opisano w sposób szczegółowy kod aplikacji głównej. W rozdziale trzecim opisano kod aplikacji – moduł do nauki słownictwa. Rozdział czwarty zawiera opis mechaniki aplikacji, czyli co, gdzie i dlaczego? Ostatni rozdział zawiera opis metody do nauki słownictwa – „widzę, piszę i czytam” to, czego się uczę.

---

<sup>1</sup> Tłumacz Google: [translate.google.pl](https://translate.google.pl)

<sup>2</sup> Tłumacz DeepL: [www.deepl.com/pl/translator](https://www.deepl.com/pl/translator)

## Rozdział I Opis planu stworzenia aplikacji

Zgodnie z opisem przedstawionym we wstępie celem niniejszego projektu jest stworzenie aplikacji desktopowej oferującej funkcjonalność słownika języka angielskiego i niemieckiego wraz z modułem do nauki słownictwa. Aplikacja będzie składać się z trzech modułów wbudowanych w dwie aplikacje. Moduły te to:

- słownik języka angielskiego – zawiera możliwość zapisywania słów angielskich i ich znaczenia polskiego oraz wskazanie typu słowa. Do wyboru są: czasownik, czasownik frazowy, idiom, przymiotnik, przysłówki, rzeczownik, zwrot oraz inne. Dodatkowo aplikacja umożliwia definiowanie własnej kategorii słów. Predefiniowana kategoria słowa to INNE. Jest to domyślna kategoria każdego nowego słowa. Można ją zmienić.
- słownik języka niemieckiego – zawiera możliwość zapisywania słów niemieckich i ich znaczenia polskiego oraz wskazanie typu słowa. Do wyboru są: czasownik, rekcja czasownika, przymiotnik, rekcja przymiotnika, przysłówki, rzeczownik, rekcja rzeczownika oraz zwrot. Dodatkowo możliwe jest dodanie własnej kategorii słowa jak w przypadku słownika angielskiego, działa to na tej samej zasadzie. Słownik niemiecko – polski umożliwia dodawanie bardziej złożonych niemieckich zagadnień gramatycznych. Są to: rekcja czasownika, rekcja rzeczownika, rekcja przymiotnika.
- moduł do nauki słownictwa – ten moduł umożliwia naukę słownictwa. Wbudowany jest w osobną aplikację „Nauka”. W modułach słowników istnieje opcja wyeksportowania słów do modułu nauka. Moduł nauka obsługuje dwa rodzaje plików. Są to odpowiednio pliki o rozszerzeniu „\*.xmla” dla języka angielskiego oraz „\*.xmln” dla języka niemieckiego. W oknie głównym aplikacji „Nauka” mamy możliwość wybrania odpowiedniego pliku, z którego chcemy się uczyć. Następnie przechodząc do modułu nauki otwiera się formularz, który umożliwia naukę słów za pomocą metody „widzę, wpisuję i widzę” to co czytam, co jest hipotezą tego projektu.

W oknie głównym aplikacji „Słownik” jest kilka modułów. Pierwszy z modułów to moduł z przyciskami do sterowania aplikacją. Są to kolejno przyciski: „Słowa” oraz „Kategorie”. Drugi moduł to przyciski wyboru rodzaju słownika. Są to przyciski typu radiobutton. Trzeci moduł to moduł wyszukiwania. Umożliwia on wyszukiwanie wg słowa. Czwarty moduł to moduł z listą dostępnych słów. Tu również dynamicznie pojawiają się wyniki wyszukiwania zgodnie z przyjętymi kryteriami wyszukiwania. Piąty moduł i zarazem ostatni zawiera pole, w którym pojawiają się szczegóły wybranego słowa. Na samym dole aplikacji znajduje się komponent statusbar, w którym możemy zobaczyć liczbę słów na liście oraz

znajduje się tam podpowiedź jak wpisywać niemieckie litery takie jak: ä, ß, ö, ü. Teraz po krótko zostanie omówione działanie i zakres każdego modułu.



Rysunek 1 Główne okno aplikacji Słownik

Pierwszy moduł aplikacji to jak już wspomniano moduł z przyciskami sterowania. Pierwszy przycisk czyli przycisk „Słowa” umożliwia zarządzanie słowami. W zależności od wybranego rodzaju słownika, przycisk ten otwiera odpowiedni formularz, na którym znajdują się odpowiednie pola umożliwiające sterowanie słowami. Na formularzu tym jest pole typu DataGridView, w którym uwidocznione są wszystkie dostępne słowa w słowniku. Na tym etapie można usunąć wybrane słowo. Pod spodem tego komponentu znajduje się przycisk „Dodaj słowo”, który otwiera kolejny formularz, dzięki któremu można już bezpośrednio dodać słowo. Usunięcie słowa ze słownika polega na zaznaczeniu go na liście i kliknięciu przycisku „Usuń słowo”. Domyślnie, gdy nie wybrano żadnego słowa przycisk ten jest nieaktywny. Uaktywnia się dopiero gdy użytkownik zaznaczy pozycję na liście. Na tym formularzu znajduje się oprócz opisanego wcześniej przycisku „Usuń słowo” oraz „Dodaj słowo” przycisk „Zamknij”.

Rysunek 2 Formularz do zarządzania słowami angielskimi (u góry) oraz niemieckimi (na dole)

Drugi przycisk w oknie głównym w module pierwszym to przycisk „Kategorie”. Umożliwia on zarządzanie kategoriami słów dostępnymi dla słownika angielskiego i niemieckiego. Przycisk ten przenosi użytkownika na kolejny formularz. Znajdują się na nim lista dodanych kategorii, przycisk „Dodaj kategorię” oraz „Usuń kategorię” i przycisk „Zamknij”, które umożliwiają odpowiednio dodanie, usunięcie kategorii oraz zamknięcie formularza. Ponadto po wpisaniu nazwy nowej kategorii uaktywnia się przycisk umożliwiający dodanie nowej kategorii.

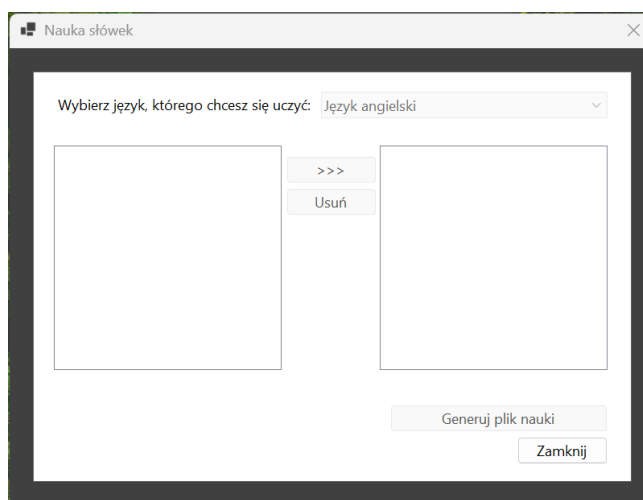
Rysunek 3 Formularz do zarządzania kategoriami

Drugi moduł umożliwia wybór rodzaju słownika. Są następujące opcje do wyboru: EN-PL, PL-EN, DE-PL oraz PL-DE. Pierwsze dwie opcje obsługują słownik języka angielskiego. Kolejne dwie opcje obsługują słownik języka niemieckiego. Nie ma opcji łączących dwa słowniki, to znaczy, nie istnieje słownik DE-EN i EN-DE. Istnieje jednak opcja

umożliwiająca dodanie takiego rozwiązania w przyszłości bazując na dwóch dostępnych słownikach.

Trzeci moduł to moduł wyszukiwania. Są w nim jedno pole tekstowe do wpisania szukanego słowa. Wyszukiwanie w polu tekstowym jest dynamiczne. Oznacza to, że lista filtrowana jest automatycznie przy wpisywaniu tekstu do pola wyszukiwania.

Czwarty moduł zawiera listę typu listbox, w której wyświetlane są wszystkie dostępne słowa w danym słowniku, a także wyniki wyszukiwania. Lista przywraca wyjściową zawartość, gdy pole wyszukiwania zostanie wyzerowane. Dodatkowo pod listą dostępnych słów znajduje się przycisk „Nauka słów”, który umożliwia wygenerowanie zestawu z wybranymi słowami do nauki. Po kliknięciu tego przycisku otwiera się kolejny formularz. Na formularzu tym pierwszym krokiem, który musi podjąć użytkownik jest wybranie słownika, na podstawie którego chce stworzyć listę słówek do nauki. Następnie należy wybrać słowo, którego chce się uczyć i kliknąć przycisk „>>>”. Po wybraniu wszystkich słów należy kliknąć przycisk „Generuj plik nauki”. Przycisk ten umożliwia zapisanie odpowiedniego pliku we wskazanej lokalizacji. Pliki te są obsługiwane następnie przez aplikację „Nauka”.



Rysunek 4 Formularz do generowania plików do nauki

Piąty i ostatni moduł umożliwia wyświetlenie szczegółowych informacji o wybranym słowie z listy. Składa się on z pola typu richtextbox i w sposób kolorowy pokazuje informacje o danym słowie.

Program zapisuje słowa w dwóch plikach xml<sup>3</sup> i jest napisany w języku C#<sup>4</sup> w technologii .Net<sup>5</sup>. Język ten rozpoczęto projektować i wdrażać w latach 2000.<sup>6</sup> Technologia

<sup>3</sup> eXtensible Markup Language

<sup>4</sup> Obecna wersja języka C# to 13 i jest obsługiwana przez platformę .Net 9

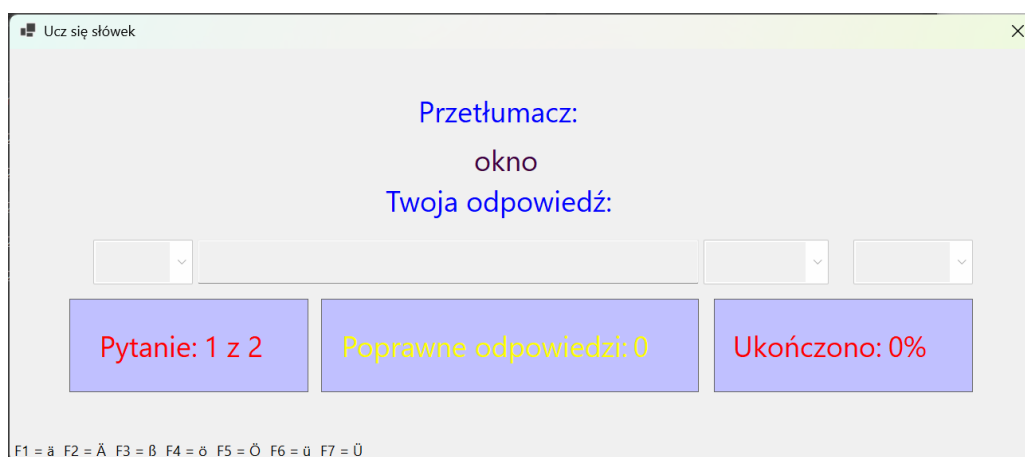
<sup>5</sup> Wymawiaj „dot Net”

<sup>6</sup> Pierwsze spotkanie w ramach projektu C# miało miejsce w 1998 roku

.Net zakłada kompilowanie kodu do kodu pośredniego (IL – Intermediate Language). Następnie kompilator generuje pliki wykonywalne \*.exe<sup>7</sup> lub \*.dll<sup>8</sup> w zależności od wybranej platformy. Następnie podczas uruchamiania działa Wspólne Środowisko Uruchomieniowe (ang. Common Language Runtime), które dokonuje właściwej kompilacji dla danej platformy systemowej.

Pliki xml, które generuje aplikacja to odpowiednio „kategorie.xml” oraz „baza.xml”. Pliki te odpowiednio przechowują dane słownika oraz kategorie. Słowa do nauki przechowywane są w osobnych plikach. Rozszerzenia tych plików to „\*.xmla” dla języka angielskiego i „\*.xmln” dla języka niemieckiego. Pliki te dokładnie zostaną omówione w rozdziale czwartym.

Moduł do nauki słownictwa to osobna aplikacja, która wymaga istnienia pliku o rozszerzeniu „\*.xmla” lub „\*.xmln” i na starcie umożliwia użytkownikowi wybranie danego pliku. Może istnieć jednocześnie wiele plików nauki. Pliki te generowane są w głównej aplikacji w module nauka i następnie eksportuje. Moduł nauka po wyborze pliku oblicza ilość słów, które w nim się znajdują. Następnie w polu textbox, które jest typu tylko do odczytu<sup>9</sup> u góry formularza głównego pokazuje się pierwsze słowo z listy z pliku w języku polskim. Zadaniem użytkownika jest odpowiedzenie w polu poniżej. Polega to na wpisaniu tłumaczenia słowa. Następnie po naciśnięciu klawisza enter pojawia się kolejne słowo do przetłumaczenia, a na dole okna aktualizowana jest statystyka odpowiedzi. Zawiera ona kolejno trzy pola: „pytanie z”, które zawiera aktualne pytanie wraz z pokazaniem ile pytań jest dostępnych, ilość prawidłowych odpowiedzi oraz postęp w procentach. Nie ma możliwości powrotu do wcześniejszego pytania.



Rysunek 5 Aplikacja do nauki słownictwa

<sup>7</sup> Plik wykonywalny w systemach z rodziny MS Windows

<sup>8</sup> Dynamic Link Library – biblioteka ładowana dynamicznie

<sup>9</sup> Ang. ReadOnly

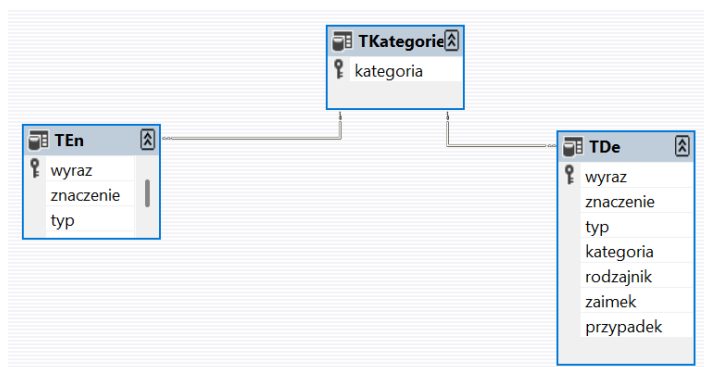


Dodatkowo cała aplikacja (słownik oraz moduł do nauki) przechowywane są w repozytorium na [github.com](https://github.com/toizy91/PROJEKT_INZ), co umożliwia każdemu pobranie aplikacji wraz z kodem źródłowym. Jest to także zabezpieczenie przed utratą danych na wypadek awarii lub kradzieży sprzętu, na którym tworzona jest praca. Repozytorium domyślnie jest prywatne. Link do repozytorium: [https://github.com/toizy91/PROJEKT\\_INZ](https://github.com/toizy91/PROJEKT_INZ)

## Rozdział II Kod aplikacji głównej

Poniżej znajduje się dokładnie opisany kod głównej aplikacji wg kolejnych formularzy i ich funkcjonalności. Dodatkowo pierwszy moduł stanowi okno główne aplikacji, które użytkownik widzi zaraz po uruchomieniu.

Opis kodu rozpoczęty zostanie od opisu zestawu danych<sup>10</sup>, na którym opiera się zapisywanie plików i danych w aplikacji. Oto schemat stworzony w tym zestawie:



Rysunek 6 Schemat zestawu DataSet (bazy danych)

Na powyższym schemacie widać trzy tabele. Są to: TEn – tabela odpowiedzialna za przechowywanie danych w słowniku języka angielskiego, TDe – tabela odpowiedzialna za przechowywanie danych w słowniku języka niemieckiego oraz TKategorie – tabela odpowiedzialna za kategorię.

Pierwsza tabela, czyli TEn zawiera cztery pola. Są to: wyraz, znaczenie, typ oraz kategoria. Pola te odpowiadają słowu angielskiemu (kolumna wyraz), jego znaczeniu polskiemu (kolumna znaczenie), typowi słowa<sup>11</sup> (kolumna typ) oraz kategorii (kolumna kategoria).

Druga tabela, czyli TDe zawiera siedem pól. Pierwsze cztery są takie same jak w tabeli TEn i oznaczają to samo. Dlatego omówione zostaną tylko pozostałe trzy. Są to odpowiednio: kolumna rodzajnik – przechowuje rodzajnik rzeczownika, kolumna zaimek odpowiada za zaimek oraz kolumna przypadek przechowuje przypadek<sup>12</sup>. Ostatnie dwie kolumny są aktywne, gdy użytkownik wybierze podczas wstawiania nowego słowa do słownika rodzaj słowa jako rekcję.

<sup>10</sup> Ang. DataSet – zestaw danych, który umożliwia wstawianie tabel i relacji między nimi. Jest to uproszczona wersja bazy danych. Nie wymaga łączenia się z żadną bazą, zestaw jest przechowywany w plikach w lokalnym katalogu aplikacji.

<sup>11</sup> Do wyboru są: czasownik, czasownik frazowy, idiom, inne, przymiotnik, rzeczownik oraz zwrot

<sup>12</sup> Do wyboru są: Nom., Gen., Dat., oraz Akk. – odpowiednio: mianownik, dopełniacz, celownik oraz biernik

Ostatnia tabela to TKategorie. Posiada ona jedynie jedną kolumnę kategoria. Tabela ta odpowiada za przechowywanie dostępnych kategorii, które zdefiniuje użytkownik.

Obie tabele TEn i TDe posiadają ustawiony klucz podstawowy<sup>13</sup> na kolumnę wyraz. Tabela TKategorie posiada ustawiony klucz podstawowy na jedną kolumnę czyli kategoria. Ponadto tabela TKategorie i TDe oraz TEn wiąże relacja jeden do wielu, co gwarantuje, że wiele słów obcych może posiadać tylko jedną kategorię. Wszystkie tabele przechowywane są w pliku xml „baza.xml”. Plik ten znajduje się w katalogu, gdzie jest główny plik wykonywalny aplikacji. Dane zapisywane są do pliku za pomocą metody „WriteXml”<sup>14</sup>.

Teraz opisany zostanie kod aplikacji. Opis zostanie rozpoczęty od opisu kodu okna głównego. Poniżej znajdują się listingi kodu, a pod nimi ich opis. Każdy listing pokazuje tylko jedną funkcję.

```
int TYP_SLOWNIKA = 0;
const string FILE_NAME_BASE = "baza.xml";
1 odwołanie
private void btnZarzSlow_Click(object sender, EventArgs e)
{
    if (TYP_SLOWNIKA == 0 || TYP_SLOWNIKA == 1) //en
    {
        DS_form_en f = new DS_form_en(this);
        f.ShowDialog();
    }
    else if (TYP_SLOWNIKA == 2 || TYP_SLOWNIKA == 3)
    {
        zarzSlowNiem f = new zarzSlowNiem(this);
        f.ShowDialog();
    }
}
```

Rysunek 7 Podstawowe stałe oraz funkcja btnZarzSlow\_Click

Na samym początku zdefiniowana została zmienna typu int<sup>15</sup> o nazwie TYP\_SLOWNIKA oraz domyślnie przypisana do niej wartość 0, która oznacza, że domyślnym słownikiem po uruchomieniu programu jest słownik angielsko – polski. Następnie zdefiniowana jest stała FILE\_NAME\_BASE, która przechowuje globalną nazwę dla pliku z danymi. Dalej mamy funkcję, która jest wywoływana po kliknięciu na przycisk „Słowa” w oknie głównym programu. W zależności od wybranego rodzaju słownika tworzona jest instancja<sup>16</sup> odpowiedniego formularza, a następnie formularz ten jest pokazywany.

<sup>13</sup> Ang. Primary Key – zapewnia unikalność danej kolumny, tzn. że wartości w niej nie mogą się powtórzyć

<sup>14</sup> Metoda dostępna z klasy DataSet – zapisuje podany schemat do pliku xml i przyjmuje jako argument nazwę pliku

<sup>15</sup> Typ integer – inaczej typ stałoprzecinkowy – przechowuje liczbę całkowitą

<sup>16</sup> Instancja – obiekt utworzony na podstawie danej klasy

```

1 odwołanie
private void btnZarzKat_Click(object sender, EventArgs e)
{
    DK_form f = new DK_form();
    f.ShowDialog();
}

```

Rysunek 8 Funkcja wywoływana po kliknięciu na przycisk "Kategorie"

Powyższa funkcja działa analogicznie jak poprzednia funkcja. Jediną różnicą jest fakt, że funkcja ta odpowiada za utworzenie i pokazanie formularza do zarządzania kategoriami. Wywoływana jest w momencie, gdy użytkownik kliknie na przycisk „Kategorie” w oknie głównym.

```

1 odwołanie
private void rbEnPl_CheckedChanged(object sender, EventArgs e)
{
    TYP_SLOWNIKA = 0;
    dsBaza1.Clear();
    if (File.Exists(FILE_NAME_BASE))
        dsBaza1.ReadXml(FILE_NAME_BASE);

    lbSlowa.DataSource = dsBaza1.TEn;
    lbSlowa.DisplayMember = "wyraz";
    lbSlowa.SelectedIndex = -1;
}

1 odwołanie
private void rbPlEn_CheckedChanged(object sender, EventArgs e)
{
    TYP_SLOWNIKA = 1;
    dsBaza1.Clear();
    if (File.Exists(FILE_NAME_BASE))
        dsBaza1.ReadXml(FILE_NAME_BASE);

    lbSlowa.DataSource = dsBaza1.TEn;
    lbSlowa.DisplayMember = "znaczenie";
    lbSlowa.SelectedIndex = -1;
}

```

Rysunek 9 Funkcje ustawiające dane dla słownika języka angielskiego

Powyższe dwie funkcje dotyczą ustawiania źródła danych<sup>17</sup> dla słownika języka angielskiego. Wywoływane są po kliknięciu odpowiedniego pola typu radiobutton. Pierwsza z nich ustawia zmienną TYP\_SLOWNIKA na 0 i wczytuje odpowiednie dane do komponentu listbox, co powoduje, że na liście pojawiają się słowa w kierunku angielski > polski. Natomiast druga funkcja działa analogicznie, lecz ustawia zmienną TYP\_SLOWNIKA na 1 co oznacza, że lista będzie wyświetlała słowa w kierunku polski > angielski.

<sup>17</sup> Ang. DataSource – komponenty przechowujące dane posiadają taką właściwość co oznacza źródło danych, z którego mają zostać pobrane elementy

```

1 odwołanie
private void rbDePl_CheckedChanged(object sender, EventArgs e)
{
    TYP_SLOWNIKA = 2;
    dsBaza1.Clear();
    if (File.Exists(FILE_NAME_BASE))
        dsBaza1.ReadXml(FILE_NAME_BASE);

    lbSlowa.DataSource = dsBaza1.TDe;
    lbSlowa.DisplayMember = "wyraz";
    lbSlowa.SelectedIndex = -1;
}

1 odwołanie
private void rbPlDe_CheckedChanged(object sender, EventArgs e)
{
    TYP_SLOWNIKA = 3;
    dsBaza1.Clear();
    if (File.Exists(FILE_NAME_BASE))
        dsBaza1.ReadXml(FILE_NAME_BASE);

    lbSlowa.DataSource = dsBaza1.TDe;
    lbSlowa.DisplayMember = "znaczenie";
    lbSlowa.SelectedIndex = -1;
}

```

Rysunek 10 Analogiczne funkcje dla słownika języka niemieckiego

Powyższe dwie funkcje są kontynuacją dwóch poprzednich. Różnią się jedynie tym, że ustawiają zmienną TYP\_SLOWNIKA na 2 oraz 3, co oznacza odpowiednio pracę listy w oknie głównym w trybie niemiecki > polski i polski > niemiecki.

```

1 odwołanie
private void okno_glowne_Load(object sender, EventArgs e)
{
    if (File.Exists(FILE_NAME_BASE))
    {
        dsBaza1.ReadXml(FILE_NAME_BASE);
    }
}

```

Rysunek 11 Funkcja wywoływana podczas ładowania formularza

Funkcja powyżej jest pierwszą funkcją wywoływaną przy tworzeniu okna głównego programu. Odpowiada ona za załadowanie danych do zestawu danych z pliku.

```

1 odwołanie
private void txtWyszukaj_KeyDown(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.F1)
    {
        txtWyszukaj.Text += "ä";
        txtWyszukaj.Focus();
        txtWyszukaj.SelectionStart = txtWyszukaj.Text.Length;
        txtWyszukaj.SelectionLength = 0;
    }
    else
    if (e.KeyCode == Keys.F2)
    {
        txtWyszukaj.Text += "ß";
        txtWyszukaj.Focus();
        txtWyszukaj.SelectionStart = txtWyszukaj.Text.Length;
        txtWyszukaj.SelectionLength = 0;
    }
    else
    if (e.KeyCode == Keys.F3)
    {
        txtWyszukaj.Text += "ö";
        txtWyszukaj.Focus();
        txtWyszukaj.SelectionStart = txtWyszukaj.Text.Length;
        txtWyszukaj.SelectionLength = 0;
    }
    else
    if (e.KeyCode == Keys.F4)
    {
        txtWyszukaj.Text += "ü";
        txtWyszukaj.Focus();
        txtWyszukaj.SelectionStart = txtWyszukaj.Text.Length;
        txtWyszukaj.SelectionLength = 0;
    }
}

```

Rysunek 12 Funkcja odpowiadająca za możliwość wpisywania w polu wyszukiwania niemieckich liter

Funkcja obsługuje zdarzenie onKeyDown<sup>18</sup> dla kontrolki typu textbox. W tym przypadku metoda ta odpowiada za monitorowanie klawiszy funkcyjnych od F1 do F4 i reaguje po naciśnięciu któregoś z nich poprzez dopisanie do pola wyszukiwania litery niemieckiej.

```

Odwołania: 4
public void Refresh()
{
    if (File.Exists(FILE_NAME_BASE))
    {
        dsBaza1.Clear();
        dsBaza1.ReadXml(FILE_NAME_BASE);
    }
}

```

Rysunek 13 Funkcja odświeżająca zestaw danych

<sup>18</sup> Zdarzenie występuje gdy rozpoczęto naciskanie klawisza, zdarzenie jest dostępne dla kontrolki typu textbox

Funkcja Refresh odpowiada za odświeżenie zestawu danych czyli całej bazy danych. Powoduje ponowne odczytanie pliku „baza.xml”. Funkcja ta jest wykorzystywana, gdy otwarte są pozostałe formularze. Wywoływana jest np. po dodaniu lub usunięciu słowa lub kategorii.

```

1 odwołanie
private void lbSlova_SelectedIndexChanged(object sender, EventArgs e)
{
    rtxOpis.Clear();
    if (lbSlova.SelectedIndex > -1)
    {
        if (TYP_SLOWNIKA == 0) //en - pl
        {
            DataRowView drV = (DataRowView)lbSlova.SelectedItem;
            rtxOpis.Lines = new string[]
            {
                "Słowo: " + drV.Row[0].ToString()+"\n"+
                "Tłumaczenie: " + drV.Row[1].ToString()+"\n"+
                "Typ słowa: " + drV.Row[2].ToString()+"\n"+
                "Kategoria: " + drV.Row[3].ToString()
            };
        }
        else
        if (TYP_SLOWNIKA == 1) //pl - en
        {
            DataRowView drV = (DataRowView)lbSlova.SelectedItem;
            rtxOpis.Lines = new string[]
            {
                "Słowo: " + drV.Row[1].ToString()+"\n"+
                "Tłumaczenie: " + drV.Row[0].ToString()+"\n"+
                "Typ słowa: " + drV.Row[2].ToString()+"\n"+
                "Kategoria: " + drV.Row[3].ToString()
            };
        }
        else
        if (TYP_SLOWNIKA == 2) //de - pl
        {
            DataRowView drV = (DataRowView)lbSlova.SelectedItem;
            rtxOpis.Lines = new string[]
            {
                "Słowo: " + drV.Row[4].ToString() + " " + drV.Row[0].ToString()+"\n"+
                "Zaimek: " + drV.Row[5].ToString()+" " +
                drV.Row[6].ToString()+"\n"+
                "Tłumaczenie: " + drV.Row[1].ToString()+"\n"+
                "Typ słowa: " + drV.Row[2].ToString()+"\n"+
                "Kategoria: " + drV.Row[3].ToString()
            };
        }
        else
        if (TYP_SLOWNIKA == 3) //pl - de
        {
            DataRowView drV = (DataRowView)lbSlova.SelectedItem;
            rtxOpis.Lines = new string[]
            {
                "Słowo: " + drV.Row[1].ToString()+"\n"+
                "Zaimek: " + drV.Row[5].ToString()+"\n"+
                "Przypadek: " + drV.Row[6].ToString()+"\n"+
                "Tłumaczenie: " + drV.Row[4].ToString()+" " + drV.Row[0].ToString()+"\n"+
                "Typ słowa: " + drV.Row[2].ToString()+"\n"+
                "Kategoria: " + drV.Row[3].ToString()
            };
        }
    }
}

```

Rysunek 14 Funkcja obsługująca kliknięcie listy ze słowami

Powyższa funkcja obsługuje zdarzenie SelectedIndexChanged<sup>19</sup>. Jest to bardzo ważna funkcja ponieważ odpowiada za pobranie danych z listy ze słowami, wyszukaniu klikniętego elementu w bazie, jego odczytanie, a następnie wyświetlenie informacji o nim. W praktyce oznacza to

<sup>19</sup> Zdarzenie wywoływane gdy kliknie się w element listy typu listbox

pobranie klikniętego słowa z listy i odczytanie odpowiednich informacji o nim. Na początku metody sprawdzamy czy użytkownik kliknął na listę i czy zaznaczony jest jakikolwiek element. Jeśli tak jest następuje przejście do sprawdzenia zmiennej TYP\_SLOWNIKA. W zależności od wybranego typu słownika, funkcja dokonuje odczytania danego elementu w bazie. W tym celu tworzona jest zmienna DataRowView<sup>20</sup>, która reprezentuje dane słowo w odpowiedniej tabeli w zestawie danych. Gdy uda się pobrać dany element tworzona jest tablica typu string<sup>21</sup>, a w niej umieszczane są informacje na temat klikniętego elementu czyli słowa. Na końcu tablica ta jest przypisywana do obiektu rxOpis<sup>22</sup>, który odpowiada za wyświetlenie opisu.

```

1 odwołanie
private void txtWyszukaj_KeyPress(object sender, KeyPressEventArgs e)
{
    int i = lbSlova.FindString(txtWyszukaj.Text);
    if (i >= 0)
    {
        lbSlova.SelectedIndex = i;
    }
    else
    {
        lbSlova.SelectedIndex = -1;
    }
}

```

Rysunek 15 Wyszukiwanie elementu na liście

Funkcja ta obsługuje zdarzenie onKeyPress<sup>23</sup> w formacie textbox<sup>24</sup>. Odpowiada za monitorowanie wpisywanych liter do pola wyszukiwania. Jeśli dany element zostanie znaleziony na liście dostępnych słów, to automatycznie zostanie zaznaczony, w przeciwnym razie żaden element nie zostanie zaznaczony.

```

1 odwołanie
private void btnNauka_Click(object sender, EventArgs e)
{
    nakaForm f = new nakaForm();
    f.ShowDialog();
}

```

Rysunek 16 Zdarzenie kliknięcia na przycisk Nauka

Jest to ostatnie zdarzenie w oknie głównym programu. Odpowiada ono za utworzenie i pokazanie odpowiedniego formularza, gdy kliknięty zostanie przycisk „Nauka”.

Teraz omówione zostaną zdarzenia poszczególnych formularzy. Pierwszy omówiony zostanie formularz służący do zarządzania kategoriami.

<sup>20</sup> Klasa odpowiadająca za pobranie informacji o wybranym elemencie

<sup>21</sup> Typ string czyli łańcuch znaków

<sup>22</sup> Zmienna typu RichTextBox – w głównym oknie formularza na dole, wyświetla informacje o wybranym elemencie

<sup>23</sup> Zdarzenie występuje gdy użytkownik naciśnie i zwolni klawisz

<sup>24</sup> Inaczej pole tekstowe



```

1 odwołanie
private void btnDodajKategorie_Click(object sender, EventArgs e)
{
    if (txtNowaKategoria.Text.Length == 0)
    {
        MessageBox.Show(this, "Wpisz kategorię!", "Kategorie", MessageBoxButtons.OK, MessageBoxIcon.Information);
        return;
    }
    if (dsBazal.TKategorie.Rows.Find(txtNowaKategoria.Text.Trim()) == null)
    {
        DataRow dr = dsBazal.TKategorie.NewRow();
        dr["kategoria"] = txtNowaKategoria.Text.Trim();
        dsBazal.TKategorie.Rows.Add(dr);
        dsBazal.WriteXml(FILE_NAME_BASE);

        MessageBox.Show(this, "Pomyślnie dodano kategorię!", "Kategorie", MessageBoxButtons.OK, MessageBoxIcon.Information);
        txtNowaKategoria.Clear();
    }
    else
    {
        MessageBox.Show(this, "Podana kategoria istnieje już w bazie!", "Kategorie", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}
}

```

### Rysunek 17 Funkcja umożliwiająca dodanie nowej kategorii

Funkcja ta odpowiada za dodanie nowej kategorii do bazy. Kategoria musi zostać wpisana przez użytkownika. Jest to warunek konieczny, inaczej przycisk „Dodaj kategorię” nie uaktywni się. Po sprawdzeniu czy użytkownik wpisał nazwę, podana nazwa jest sprawdzana w bazie danych, aby upewnić się, że podana kategoria już w niej nie istnieje. Jeśli tak jest tworzony jest obiekt typu DataRow<sup>25</sup>, który następnie zostaje wypełniony. Na końcu po dodaniu wiersza do tabeli, cała baza jest uaktualniana i zapisywana w pliku.

---

<sup>25</sup> Obiekt DataRow reprezentuje jeden wiersz w tabeli

```

1 odwołanie
private void btnUsunKategorie_Click(object sender, EventArgs e)
{
    DataRowView drv = (DataRowView)cbKategorie.SelectedItem;
    if (drv != null)
    {
        if (MessageBox.Show(this, "Czy na pewno chcesz usunąć wybraną kategorię?", "Kategorie", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
        {
            drv.Delete();
            dsBazal.WriteXml("kategorie.xml");
            MessageBox.Show(this, "Pomyślnie usunięto kategorię!", "Kategorie", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
}

```

Rysunek 18 Funkcja usuwająca wybraną kategorię

Druga i ostatnia funkcja w tym module obsługuje usunięcie wybranej kategorii z bazy. Po pobraniu wybranej kategorii i dokonaniu rzutowania<sup>26</sup> jej na typ DataRowView następuje pytanie użytkownika czy potwierdza usunięcie, i jeśli odpowiedź jest pozytywna następuje usunięcie kategorii z bazy. Na końcu cała baza jest uaktualniana i zapisywana do pliku.

Kolejnym formularzem, który zostanie omówiony jest formularz służący do zarządzania słowami. Są to w praktyce dwa podobne formularze, które różnią się jedynie rodzajem słownika, który obsługują.

```

private okno_glowne mainForm;
1 odwołanie
public zarzSlowNiem(okno_glowne form)
{
    InitializeComponent();
    mainForm = form;
    this.FormClosed += new FormClosedEventHandler(zarzSlowNiem_FormClosed);
}

```

Rysunek 19 Nietypowy konstruktor formularza

Powyższy moduł zaczyna się od nietypowego konstruktora<sup>27</sup> formularza. Do standardowego konstruktora przekazany zostaje obiekt typu okno\_glowne, który reprezentuje główne okno programu. Umożliwia to wywołanie wcześniej opisanej funkcji Refresh po zamknięciu obecnego formularza (zdarzenie FormClosed<sup>28</sup>).

<sup>26</sup> Rzutowanie służy konwersji obiektu jednego typu na inny

<sup>27</sup> Jest to funkcja wywoływana jako pierwsza, gdy dany obiekt powoływany jest do życia

<sup>28</sup> Zdarzenie występujące po zamknięciu formularza – w tym przypadku służy odświeżeniu danych w formularzu nadrzędnym

```

1 odwołanie
private void btnUsunSlovo_Click(object sender, EventArgs e)
{
    try
    {
        if (dgvSlova.SelectedRows.Count > 0)
        {
            if (MessageBox.Show(this, "Czy na pewno chcesz usunąć wybraną pozycję?", "Zarządzaj słowami niemieckimi", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
            {
                DataRowView drv = (DataRowView)dgvSlova.SelectedRows[0].DataBoundItem;
                string t = drv.Row[0].ToString();

                if (dsBazal.TDe.Rows.Find(t) != null)
                {
                    DataRow dr = dsBazal.TDe.Rows.Find(t);
                    dsBazal.TDe.Rows.Remove(dr);
                    dsBazal.WriteXml(FILE_NAME_BASE);
                    MessageBox.Show(this, "Pomyślnie usunięto wybraną pozycję!", "Zarządzaj słowami niemieckimi", MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
            }
        }
        else
        {
            return;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(this, ex.Message, "Wystąpił nieoczekiwany błąd!", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    finally
    {
        mainForm.Refresh();
    }
}

```

Rysunek 20 Funkcja usuwająca słowo ze słownika

Funkcja ta usuwa zaznaczony rekord z bazy. Działa podobnie jak usuwanie kategorii (patrz poprzedni formularz). Dodatkowo w tym przypadku funkcja opatrzona jest klauzulą try..catch..finally<sup>29</sup>, która zabezpiecza kod przed wystąpieniem nieoczekiwanego błędu.

Pozostałe funkcje w tym module dublują się z funkcjami opisanymi we wcześniejszych modułach. Są to funkcje tworzenia nowego formularza i kliknięcia na dany przycisk. Z tym modulem powiązane są dwa potomne formularze. Są to formularze bezpośrednio odpowiedzialne za dodanie słowa do słownika. Formularze te różnią się rodzajem słownika, który obsługują. Omówione zostaną więc najważniejsze metody tych formularzy.

Rysunek 21 Formularz dodawania słowa języka niemieckiego (po lewej) i angielskiego (po prawej)

<sup>29</sup> Jest to powszechnie stosowana konstrukcja służąca zabezpieczeniu kodu i obsłużeniu pojawiających się błędów (blok catch). Dodatkowo po słowie finally kod wykona się zawsze bez względu czy wystąpił błąd czy nie

```

1 odwołanie
private void cbTypSlova_SelectedIndexChanged(object sender, EventArgs e)
{
    panOgoln.Visible = true;
    switch (cbTypSlova.SelectedIndex)
    {
        case 0: //czasownik
        {
            cbPrzypadek.Enabled = false;
            cbRodzajnik.Enabled = false;
            cbZaimek.Enabled = false;
            break;
        }
        case 1: //rekcja czasownika
        {
            cbPrzypadek.Enabled = true;
            cbRodzajnik.Enabled = false;
            cbZaimek.Enabled = true;
            break;
        }
        case 2: //przymiotnik
        {
            cbPrzypadek.Enabled = false;
            cbRodzajnik.Enabled = false;
            cbZaimek.Enabled = false;
            break;
        }
        case 3: //rekcja przymiotnika
        {
            cbPrzypadek.Enabled = true;
            cbRodzajnik.Enabled = true;
            cbZaimek.Enabled = true;
            break;
        }
        case 4: //rzeczownik
        {
            cbPrzypadek.Enabled = false;
            cbRodzajnik.Enabled = true;
            cbZaimek.Enabled = false;
            break;
        }
        case 5: //rekcja rzeczownika
        {
            cbPrzypadek.Enabled = true;
            cbRodzajnik.Enabled = true;
            cbZaimek.Enabled = true;
            break;
        }
        case 6: //zwrot
        {
            cbPrzypadek.Enabled = false;
            cbRodzajnik.Enabled = false;
            cbZaimek.Enabled = false;
            break;
        }
        case 7: //inne
        {
            cbPrzypadek.Enabled = false;
            cbRodzajnik.Enabled = false;
            cbZaimek.Enabled = false;
            break;
        }
    }
}

```

Rysunek 22 Wyświetlenie odpowiednich elementów na formularzu w zależności od rodzaju słowa

Jest to funkcja, która odpowiada za włączenie i wyłączenie odpowiednich elementów, w zależności od wyboru rodzaju słowa, które użytkownik chce dodać. Różnice widać w przypadku dodawania słowa typu: rzeczownik, rekcja rzeczownika, rekcja przymiotnika oraz rekcja czasownika.

```

1 odwołanie
private void btnDodajSlovo_Click(object sender, EventArgs e)
{
    try
    {
        if (dsBazal.TDe.Rows.Find(txtWyrasz.Text.Trim()) == null)
        {
            if (cbRodzajnik.Enabled == true)
            {
                string t = txtWyrasz.Text.Trim();
                t = char.ToUpper(t[0]) + t.Substring(1);
                txtWyrasz.Text = t;

                DataRow dr = dsBazal.TDe.NewRow();
                dr["wyrasz"] = txtWyrasz.Text.Trim();
                dr["znaczenie"] = txtZnaczenie.Text.Trim();
                dr["typ"] = cbTypSlova.Text;
                dr["kategoria"] = cbKategoria.Text;
                dr["rodzajnik"] = cbRodzajnik.Text;
                dr["zaimek"] = cbZaimek.Text;
                dr["przypadek"] = cbPrzypadek.Text;

                dsBazal.TDe.Rows.Add(dr);
                dsBazal.WriteXml(FILE_NAME_BASE);

                MessageBox.Show(this, "Pomyślnie dodano słowo do bazy!", "Dodaj słowo", MessageBoxButtons.OK, MessageBoxIcon.Information);
                txtWyrasz.Clear();
                txtZnaczenie.Clear();
            }
            else
            {
                MessageBox.Show(this, "Podane słowo już istnieje w bazie!", "Dodaj słowo", MessageBoxButtons.OK, MessageBoxIcon.Information);
                txtWyrasz.Clear();
                txtZnaczenie.Clear();
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(this, ex.Message, "Wystąpił nieoczekiwany błąd", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
}

```

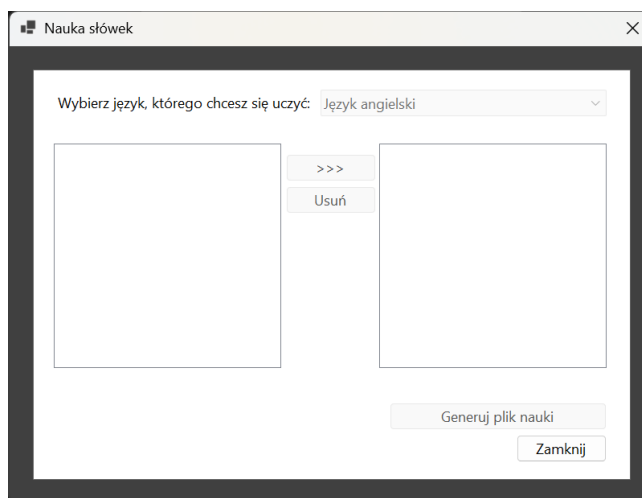
### Rysunek 23 Funkcja dodająca słowo niemieckie do słownika

Powyższa funkcja działa analogicznie do funkcji dodającej kategorię. Jest ona tylko bardziej rozbudowana i zawiera pewne elementy wyróżniające. Na samym początku następuje sprawdzenie czy podane słowo istnieje już w słowniku. Jeśli nie, to wykonywany jest blok kodu mający dodać nowe słowo. Na wstępie następuje sprawdzenie czy jest aktywne pole „rodzajnik”. Jeśli tak jest, oznacza to, że użytkownik dodaje rzeczownik lub rekcję rzeczownika<sup>30</sup>. Dla zachowania poprawności gramatycznej<sup>31</sup>, program dokonuje konwersji pierwszej litery wpisywanego wyrazu. Następnie powoływany jest element typu DataRow i uzupełniane są odpowiednie jego pola. Po tym następuje zapis do bazy, jej uaktualnienie, a następnie zapisanie do pliku, co kończy się komunikatem o powodzeniu operacji.

<sup>30</sup> W języku niemieckim przed rzeczownikiem zawsze stoi rodzajnik

<sup>31</sup> Rzeczowniki w języku niemieckim piszemy wielką literą

Został do omówienia ostatni moduł głównej aplikacji. Jest to mianowicie moduł odpowiedzialny za generowanie plików służących później w aplikacji „Nauka” do uczenia się słówek.



Rysunek 24 Moduł do generowania plików nauki

```

1 odwołanie
private void cbJęzyk_SelectedIndexChanged(object sender, EventArgs e)
{
    switch (cbJęzyk.SelectedIndex)
    {
        case 0:
        {
            lbSłowaWY.DataSource = dsBaza1.TEn;
            lbSłowaWY.DisplayMember = "wyraz";
            lbSłowaWY.SelectedIndex = -1;
            typ_zrodla = 0;
            break;
        }
        case 1:
        {
            lbSłowaWY.DataSource = dsBaza1.TDe;
            lbSłowaWY.DisplayMember = "wyraz";
            lbSłowaWY.SelectedIndex = -1;
            typ_zrodla = 1;
            break;
        }
    }
    cbJęzyk.Enabled = false;
}

```

Rysunek 25 Funkcja wybierająca, z którego słownika będą brane słowa do nauki

Za pomocą konstrukcji switch..case<sup>32</sup> wybierany jest język, z którego będą czerpane słowa do nauki. W zależności od wyboru (język angielski lub niemiecki) ustawiane są odpowiednie właściwości<sup>33</sup> komponentu zawierającego listę słów. Należy zauważyć, że po wyborze rodzaju źródłowego słownika nie ma już możliwości jego zmiany – pole wyboru języka zostanie automatycznie wyłączone po wyborze. Zapewnia to spójność danych w plikach nauki.

<sup>32</sup> Jest to instrukcja wielokrotnego wyboru

<sup>33</sup> W języku C# każdy obiekt posiada pewne cechy, są to właściwości

```

1 odwołanie
private void btnDodajDoNauka_Click(object sender, EventArgs e)
{
    try
    {
        if (lbSlowaWY.SelectedIndex >= -1)
        {
            DataRowView drV = (DataRowView)lbSlowaWY.SelectedItem;

            if (typ_zrodla == 0) //słownik ang
            {
                if (dsBazaEn1.TEn.Rows.Find(drV.Row[0]) == null)
                {
                    DataRow dr = dsBazaEn1.TEn.NewRow();
                    dr["wyraz"] = drV.Row[0].ToString();
                    dr["znaczenie"] = drV.Row[1].ToString();
                    dsBazaEn1.TEn.Rows.Add(dr);
                    lbSlowaDoNauk.DataSource = dsBazaEn1.TEn;
                    lbSlowaDoNauk.DisplayMember = "wyraz";
                }
                else
                {
                    MessageBox.Show(this, "Podany element istnieje już na liście!", "Nauka", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    return;
                }
            }
            else
            {
                if (typ_zrodla == 1) //słownik niemiecki
                {
                    if (dsBazaDe1.TDe.Rows.Find(drV.Row[0]) == null)
                    {
                        DataRow dr = dsBazaDe1.TDe.NewRow();
                        dr["wyraz"] = drV.Row[0].ToString();
                        dr["znaczenie"] = drV.Row[1].ToString();
                        dr["rodzajnik"] = drV.Row[4].ToString();
                        dr["zaimek"] = drV.Row[5].ToString();
                        dr["przypadek"] = drV.Row[6].ToString();
                        dsBazaDe1.TDe.Rows.Add(dr);
                        lbSlowaDoNauk.DataSource = dsBazaDe1.TDe;
                        lbSlowaDoNauk.DisplayMember = "wyraz";
                    }
                    else
                    {
                        MessageBox.Show(this, "Podany element istnieje już na liście!", "Nauka", MessageBoxButtons.OK, MessageBoxIcon.Information);
                        return;
                    }
                }
            }
        }
    }
    catch (Exception Ex)
    {
        MessageBox.Show(this, Ex.Message, "Nieoczekiwany błąd", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

Rysunek 26 Zdarzenie dodania słowa do listy do nauki

W zależności od rodzaju słownika (zmienna `typ_zrodla`) wybierana jest odpowiednia baza<sup>34</sup>, do której będą zapisywane dane do nauki. Najpierw następuje sprawdzenie, czy wybrane słowo nie zostało już wcześniej dodane do nauki. Jeśli nie to tworzony jest nowy wiersz (`DataRow`) i przepisane do niego są odpowiednie pola z bazy pierwotnej. W przypadku języka angielskiego są to pola „wyraz” oraz „znaczenie”, a w przypadku języka niemieckiego są to pola „wyraz”, „znaczenie”, „rodzajnik”, „zaimek” oraz przypadek. Pola są tak wybrane, aby umożliwiały jednoznaczne określenie danego słowa.

<sup>34</sup> W rzeczywistości są to dwa osobne zestawy typu `DataSet`, każdy odpowiada za inny język

```

1 odwołanie
private void btnGenerujPlik_Click(object sender, EventArgs e)
{
    try
    {
        if (lbSlovaDoNauk.Items.Count > 0)
        {
            if (typ_zrodla == 0) //plik angielski
            {
                zapiszPlikDial.FilterIndex = 1;
                if (zapiszPlikDial.ShowDialog() == DialogResult.OK)
                {
                    dsBazaEn1.WriteXml(zapiszPlikDial.FileName);
                    this.Close();
                }
            }
            else if (typ_zrodla == 1) //plik niemiecki
            {
                zapiszPlikDial.FilterIndex = 2;
                if (zapiszPlikDial.ShowDialog() == DialogResult.OK)
                {
                    dsBazaDel1.WriteXml(zapiszPlikDial.FileName);
                    this.Close();
                }
            }
        }
    }
    catch (Exception Ex)
    {
        MessageBox.Show(this, Ex.Message, "Nieoczekiwany błąd", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

Rysunek 27 Funkcja generująca plik do nauki

Jest to kluczowa funkcja dla tego modułu. Używa komponentu typu `FileSaveDialog`<sup>35</sup>, który w zależności od wybranego rodzaju słownika zapisuje w wybranej przez użytkownika lokalizacji plik z odpowiednim rozszerzeniem. W przypadku nauki słów z języka angielskiego pliki zapisywane są z rozszerzeniem „\*.xmla”, a w przypadku słów z języka niemieckiego jest to rozszerzenie „\*.xmln”. Pliki te służą jako wejście w programie „Nauka”, który omówiony zostanie w kolejnych rozdziałach.

Jak widać omówiony kod nie jest zbyt długi. Starałem się omówić poszczególne moduły w sposób logiczny i przejrzysty. W kolejnych rozdziałach przedstawione zostaną schematy działania aplikacji oraz jej mechanika oraz omówiona zostanie dokładniej aplikacja „Nauka”, która służy do uczenia się słów, wygenerowanych w programie „Słownik”.

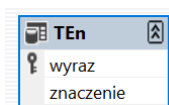
<sup>35</sup> Jest to okno dialogowe, służące do zapisania pliku – standardowe okno zapisu pliku w systemie Windows



### Rozdział III Kod aplikacji do nauki słownictwa

Poniżej znajduje się dokładnie opisany kod aplikacji służącej nauce słownictwa. Aplikacja ta zawiera jeden moduł, który jest jednocześnie głównym oknem programu.

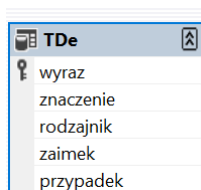
Opis kodu należy rozpocząć od opisu zestawów danych, na których opiera się odczytywanie plików nauki w aplikacji. Program ten korzysta bowiem z dwóch zestawów danych. Są to osobne zestawy dla plików nauki zawierających słowa ze słownika języka angielskiego i niemieckiego. Pliki te różnią się rozszerzeniami. Opis zestawu rozpocząć należy od zestawu dla języka angielskiego. Oto schemat stworzony w tym zestawie:



| TEn       |  |
|-----------|--|
| wyraz     |  |
| znaczenie |  |

Rysunek 28 Tabela w zestawie dla plików języka angielskiego

Jak widzimy zestaw dsBazaEn, bo taka jest jego nazwa zawiera jedną prostą tabelę. Tabela ta zawiera dwie kolumny. Są to kolumna wyraz i kolumna znaczenie. Przechowują one odpowiednio wyraz angielski oraz jego polskie znaczenie. Kolejny zestaw danych to dsBazaDe, który przechowuje tabelę dla plików języka niemieckiego. Zawiera on następujące kolumny: wyraz, znaczenie, rodzajnik, zaimek oraz przypadek. Nazwy tych kolumn odpowiadają poszczególnym zagadnieniom gramatycznym języka niemieckiego. Obie tabele mają ustawiony klucz podstawowy dla kolumny wyraz. Zapewnia to niepowtarzalność każdego słowa. Oto tabela TDe:



| TDe       |  |  |  |  |
|-----------|--|--|--|--|
| wyraz     |  |  |  |  |
| znaczenie |  |  |  |  |
| rodzajnik |  |  |  |  |
| zaimek    |  |  |  |  |
| przypadek |  |  |  |  |

Rysunek 29 Tabela w zestawie dla plików języka niemieckiego

Następnie omówiony zostanie kod aplikacji Nauka. Kod ten nie jest zbyt długi. Składa się z 243 linii. Po uruchomieniu aplikacji użytkownik widzi okno dialogowe typu OpenFileDialog<sup>36</sup>, które służy do wyboru pliku nauki. W oknie tym użytkownik musi wybrać typ pliku, który chce otworzyć. Do wyboru są pliki dla języka angielskiego – opcja Język angielski oraz dla języka niemieckiego – opcja Język niemiecki. Jeśli użytkownik nie wybierze żadnego pliku aplikacja

<sup>36</sup> Jest to standardowe okno dialogowe w systemach Windows służące otwarciu plików

zakończy swoje działanie. Jest to rodzaj zabezpieczenia, aby nie uruchomić tzw. pustego pliku. Jeżeli użytkownik wskaże plik, to po zatwierdzeniu okna dialogowego wykona się poniższy kod:

```
1 odwołanie
private void oKnoGlowne_Load(object sender, EventArgs e)
{
    if (otworzPlikDial.ShowDialog() == DialogResult.OK)
    {
        string filename = otworzPlikDial.FileName;
        if (filename.Contains(".xmla"))
        {
            dsBazaEn1.ReadXml(filename);
            lbSlowa.DataSource = dsBazaEn1.TEn;
            lbSlowa.DisplayMember = "znaczenie";
            typ_pliku = 0;
            cbRodzajnik.Enabled = false;
            cbPrzypadek.Enabled = false;
            cbZaimiek.Enabled = false;
            btnStart_Click(this, e);
        }
        else if (filename.Contains(".xmln"))
        {
            dsBazaDe1.ReadXml(filename);
            lbSlowa.DataSource = dsBazaDe1.TDe;
            lbSlowa.DisplayMember = "wyraz";
            typ_pliku = 1;
            cbRodzajnik.Enabled = true;
            cbPrzypadek.Enabled = true;
            cbZaimiek.Enabled = true;
            btnStartDe_Click(this, e);
        }
    }
    else
    {
        Application.Exit();
    }
}
```

Rysunek 30 Kod wykonywany podczas otwarcia pliku do nauki

Jak widać po kliknięciu przycisku otwórz w oknie dialogowym otwarcia pliku pobierana jest nazwa otwieranego pliku. Następnie sprawdzeniu podlega jego rozszerzenie i w zależności od rozszerzenia następuje odczytanie danych do odpowiedniego zestawu danych oraz uzupełnienie zmiennych globalnych<sup>37</sup>, które posłużą do generowania statystyki nauki. Oto zmienne globalne, których używa program:

```
int pytanieZ = 0;
double wszystkiePytania = 0;
string poprOdp = "";
string wyraz = "";
string rodzajnik = "";
string zaimiek = "";
string przypadek = "";
double procUkon = 0.0f;
double krok = 0.0f;
int poprOdpLicz = 0;

int typ_pliku = 0; //xmla
```

Rysunek 31 Zmienne globalne użyte w programie

W programie użyte zostały następujące zmienne:

<sup>37</sup> Zmienne globalne to takie zmienne, które dostępne są w całym kodzie programu z każdego jego miejsca

Tabela 1 Zmienne globalne użyte w programie Nauka

| Lp. | Zmienna          | Typ                  | Wartość domyślna | Opis                                  |
|-----|------------------|----------------------|------------------|---------------------------------------|
| 1   | pytanieZ         | int                  | 0                | Przechowuje indeks aktualnego pytania |
| 2   | wszystkiePytania | double <sup>38</sup> | 0                | Zawiera liczbę wszystkich pytań       |
| 3   | poprOdp          | string               | ""               | Przechowuje poprawną odpowiedź        |
| 4   | wyraz            | string               | ""               | Przechowuje wyraz                     |
| 5   | rodzajnik        | string               | ""               | Przechowuje rodzajnik                 |
| 6   | zaimek           | string               | ""               | Przechowuje zaimek                    |
| 7   | przypadek        | string               | ""               | Przechowuje przypadek                 |
| 8   | procUkon         | double               | 0.0f             | Zawiera procent ukończenia nauki      |
| 9   | krok             | double               | 0.0f             | Krok procentowy                       |
| 10  | poprOdpLicz      | int                  | 0                | Liczba poprawnych odpowiedzi          |

Po wstępnym wygenerowaniu i wypełnieniu zmiennych globalnych, program przechodzi do wywołania funkcji startowej. Odpowiednio dla plików nauki słownika angielskiego wywoływana jest procedura `btnStart_Click`, a dla plików nauki słownika niemieckiego wywoływana jest procedura `btnStartDe_Click`. Obie procedury odpowiedzialne są za wystartowanie nauki i wypełnienie pól dla pierwszego pytania z listy. Oto ich kody:

Odwołania: 2

```
private void btnStart_Click(object sender, EventArgs e)
{
    if (lbSlowa.Items.Count > 0)
    {
        DataRowView drV = (DataRowView)lbSlowa.Items[pytanieZ];
        txtPytTres.Text = (drV.Row[1].ToString());
        wyraz = drV.Row[1].ToString();
        poprOdp = drV.Row[0].ToString();
        poprOdpLicz = 0;
        wszystkiePytania = System.Convert.ToInt32(lbSlowa.Items.Count);
        krok = Math.Round(100 / wszystkiePytania, 2);
        nastPyt();
    }
}
```

Rysunek 32 Funkcja startowa dla plików zawierających słowa angielskie

<sup>38</sup> Zmienna typu double jest to typ zmiennoprzecinkowy podwójnej precyzji – odpowiada za przechowywanie liczb „po przecinku”

Odwołania: 2

```
private void btnStartDe_Click(object sender, EventArgs e)
{
    if (lbSlova.Items.Count > 0)
    {
        DataRowView drV = (DataRowView)lbSlova.Items[pytanieZ];
        txtPytTres.Text = (drV.Row[1].ToString());
        wyraz = drV.Row[1].ToString();
        rodzajnik = drV.Row[2].ToString();
        zaimek = drV.Row[3].ToString();
        przypadek = drV.Row[4].ToString();
        poprOdp = drV.Row[0].ToString();
        poprOdp.Trim();
        poprOdpLicz = 0;
        wszystkiePytania = System.Convert.ToInt32(lbSlova.Items.Count);
        krok = Math.Round(100 / wszystkiePytania, 2);
        nastPyt();
    }
}
```

Rysunek 33 Funkcja startowa dla plików zawierających słowa niemieckie

Obie funkcje działają podobnie. Swoje działanie zaczynają od sprawdzenia, czy na liście zostały poprawnie wczytane słowa z danego pliku. Jeśli tak, to z odpowiedniej tabeli pobierany jest wiersz zawierający szczegóły na temat wybranego słowa – służy temu rzutowanie na obiekt typu DataRowView. Następnie uzupełniane są zmienne przechowujące informacje na temat danego słowa, które nie zostały uzupełnione w poprzednim etapie. Każda z funkcji startowych kończy się wywołaniem funkcji odpowiedzialnej za wczytywanie kolejnych pytań. Są to odpowiednio NastPyt i NastPytDe. Oto ich kody:

Odwołania: 4

```
private void nastPyt()
{
    if (pytanieZ < lbSlova.Items.Count)
    {
        DataRowView drV = (DataRowView)lbSlova.Items[pytanieZ];
        wyraz = drV.Row[1].ToString();
        poprOdp = drV.Row[0].ToString();
        txtPytTres.Text = wyraz;
        txtOdp.Clear();
        pytanieZ++;
        txtOdp.Focus();
    }
    else
    {
        procUkon = 100;
        MessageBox.Show(this, "Ukończono naukę!", "Ucz się słówek", MessageBoxButtons.OK, MessageBoxIcon.Information);
        wyraz = "";
        poprOdp = "";
        txtOdp.Clear();
        txtPytTres.Clear();
        txtPytTres.Enabled = false;
        txtOdp.Enabled = false;
        timer1.Enabled = false;
        Application.Exit();
    }
}
```

Rysunek 34 Funkcja wczytująca następne pytanie ze słownika angielskiego

Odwołania: 2

```
private void nastPytDe()
{
    if (pytanieZ < lbSlowa.Items.Count)
    {
        DataRowView drV = (DataRowView)lbSlowa.Items[pytanieZ];
        wyraz = drV.Row[1].ToString();
        rodzajnik = drV.Row[2].ToString();
        zaimek = drV.Row[3].ToString();
        przypadek = drV.Row[4].ToString();
        poprOdp = drV.Row[0].ToString();
        poprOdp.Trim();
        txtPytTres.Text = wyraz;
        txtOdp.Clear();
        pytanieZ++;
        txtOdp.Focus();
    }
    else
    {
        procUkon = 100;
        MessageBox.Show(this, "Ukończono naukę!", "Ucz się słówek", MessageBoxButtons.OK, MessageBoxIcon.Information);
        wyraz = "";
        poprOdp = "";
        rodzajnik = "";
        zaimek = "";
        przypadek = "";
        txtOdp.Clear();
        txtPytTres.Clear();
        txtPytTres.Enabled = false;
        txtOdp.Enabled = false;
        timer1.Enabled = false;
        Application.Exit();
    }
}
```

Rysunek 35 Funkcja wczytująca następne pytanie ze słownika niemieckiego

Następna i kluczowa funkcja programu Nauka to funkcja sprawdzająca czy wpisano poprawną odpowiedź. Po naciśnięciu klawisza enter program porównuje wpisaną przez użytkownika wartość z wartościami zapisanymi w odpowiednich zmiennych globalnych. Jeśli odpowiedź się zgadza, tzn. jest poprawna to zwiększają się odpowiednie liczniki oraz panel „Poprawne odpowiedzi” zapala się na kolor jasnozielony. Jeśli odpowiedź jest niepoprawna to panel ten zapala się na kolor czerwony. Funkcja sprawdzająca wpisaną wartość kończy się wywołaniem metod wczytujących następne pytanie opisanych w paragrafie powyżej. Oto kod funkcji sprawdzającej:

```

1 odwołanie
private void txt0dp_KeyDown(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Enter)
    {
        if (txt0dp.Text.Trim() == popr0dp && typ_pliku == 0)
        {
            panel2.BackColor = Color.LightGreen;
            popr0dpLicz++;
            procUkon += krok;
            txt0dp.Clear();
            nastPyt();
        }
        else if (typ_pliku == 0 && txt0dp.Text.Trim() != popr0dp)
        {
            panel2.BackColor = Color.Red;
            procUkon += krok;
            txt0dp.Clear();
            nastPyt();
        }
        else if (typ_pliku == 1 && txt0dp.Text.Trim() == popr0dp && cbRodzajnik.Text == rodzajnik && cbPrzypadek.Text == przypadek && cbZaimek.Text == zaimek)
        {
            panel2.BackColor = Color.LightGreen;
            popr0dpLicz++;
            procUkon += krok;
            txt0dp.Clear();
            nastPytDe();
        }
        else if (typ_pliku == 1 && txt0dp.Text.Trim() != popr0dp || cbRodzajnik.Text != rodzajnik || cbZaimek.Text != zaimek || cbPrzypadek.Text != przypadek)
        {
            panel2.BackColor = Color.Red;
            procUkon += krok;
            txt0dp.Clear();
            nastPytDe();
        }
    }
}

```

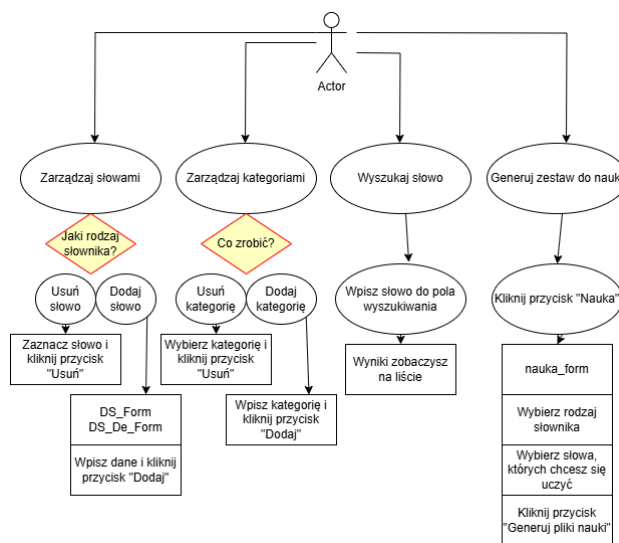
### Rysunek 36 Funkcja sprawdzająca wpisaną wartość

Ponadto należy nadmienić iż program zawiera poboczne funkcje, które nie zostały tutaj omówione. Jest to min. funkcja odpowiedzialna za możliwość wpisywania niemieckich liter. Funkcja ta została szczegółowo omówiona w rozdziale 2. Kolejną funkcją, która nie została tu omówiona to metoda, która na bieżąco monitoruje wartość odpowiednich zmiennych globalnych i odpowiada za ich aktualizację i wyświetlanie. Metoda ta jest realizowana za pomocą komponentu Timer<sup>39</sup>, który wykonuje swój kod co określony, ustawiony przedział czasu.

<sup>39</sup> Komponent Timer zawiera kluczową funkcję Tick, która wykonuje się co określony przedział czasu

## Rozdział IV Opis mechaniki działania aplikacji

Oto schemat UML<sup>40</sup> przedstawiający diagram przypadków użycia<sup>41</sup> dla aplikacji Słownik:



Rysunek 37 Diagram przypadków użycia dla aplikacji Słownik

Podstawą powyższego diagramu jest użytkownik (ang. actor). Użytkownik ma następujące akcje do wykonania po uruchomieniu głównego okna aplikacji. Są to kolejno:

1. Możliwość zarządzania słowami w słowniku
2. Możliwość zarządzania kategoriami
3. Możliwość wyszukania podanego słowa w słowniku
4. Możliwość wygenerowania pliku do nauki słownictwa

Akcje nr 1 i 3 zależne są od wybranego rodzaju słownika. Do wyboru są cztery kierunki pracy słownika. Są to:

1. EN – PL – odpowiada za słownik angielsko – polski
2. PL – EN – odpowiada za słownik polsko – angielski
3. DE – PL – odpowiada za słownik niemiecko – polski
4. PL – DE – odpowiada za słownik polsko – niemiecki

Pierwsze dwa kierunki pracy słownika przy kliknięciu przycisku „Słowa” pokazują formularz do zarządzania słowami dla słownika języka angielskiego. Natomiast pozostałe dwa kierunki pokazują formatkę do zarządzania słowami dla słownika języka niemieckiego. Te dwa formularze są bardzo podobne. Różnią się jedynie wyświetlanymi danymi. Na obu

<sup>40</sup> Język UML to Ujednolicony język modelowania systemów informatycznych

<sup>41</sup> Jest to graficzne przedstawienie przypadków użycia, aktorów oraz związków między nimi

formularzach użytkownik ma do wyboru trzy akcje: Dodaj słowo, Usuń słowo oraz Zamknij. Po kliknięciu na przycisk Dodaj słowo pokazuje się formularz dodawania nowego słowa. Przy kliknięciu na przycisk Usuń słowo, użytkownik zostanie zapytany czy potwierdza usunięcie wybranego słowa ze słownika. Jeśli potwierdzi, to dane słowo zostanie usunięte, a cały słownik uaktualniony. Kliknięcie przycisku Zamknij powoduje zamknięcie formularza do zarządzania słowami.

Kolejna akcja, która zależy od wybranego rodzaju słownika, to akcja wyszukiwania słów w zasobach danych. Wybranie konkretnego kierunku słownika powoduje przestawienie opcji DataSource i DisplayMember<sup>42</sup> dla komponentu typu Listbox, który odpowiedzialny jest za wyświetlanie dostępnych słów w danym słowniku. Pole wyszukiwania jest niezależne od wybranego rodzaju słownika. Jego działanie opiera się na wyszukiwaniu słów, które są na liście bez zaglądania w szczegóły źródła danych. Dodatkowo pole wyszukiwania posiada wbudowaną obsługę niemieckich liter. Odpowiadają za to klawisze funkcyjne od F1 do F4.

Akcja nr 2 występuje po kliknięciu przycisku „Kategorie”. Po wykonaniu tej akcji pojawi się formularz do zarządzania kategoriami. Zawiera on dwie opcje: „Dodaj kategorię” i „Usuń kategorię”. Po kliknięciu pierwszego przycisku „Dodaj kategorię”, po sprawdzeniu czy pole, gdzie użytkownik wpisuje nazwę nowej kategorii nie jest puste następuje dodanie kategorii i odświeżenie słownika. Po kliknięciu przycisku „Usuń kategorię”, jeśli została wybrana kategoria dokonuje się usunięcie wybranej kategorii. Po obu akcjach słownik zostaje uaktualniony.

Czwarta i ostanía akcja to możliwość wygenerowania plików służących nauce słownictwa. Po kliknięciu przycisku „Nauka” zostanie pokazany formularz do zarządzania nauką. Zawiera on pole typu ComboBox<sup>43</sup> służące wybraniu słownika, na podstawie którego będą generowane słowa. Następnie z listy po lewej stronie użytkownik wybiera słowo którego chce się uczyć. Aby dodać słowo do nauki, po wybraniu, należy kliknąć przycisk „>>>”. Jeśli słowo zostanie poprawnie dodane do listy nauki, pojawi się na liście po prawej stronie. Na koniec jeśli wszystkie słowa zostały już dodane należy kliknąć przycisk „Generuj pliki nauki”, co otworzy okno dialogowe, gdzie należy wskazać nazwę pliku oraz jego lokalizację i kliknąć „Zapisz”. Spowoduje to stworzenie pliku z odpowiednim rozszerzeniem. Pliki te należy otworzyć za pomocą osobnej aplikacji „Nauka”. Teraz omówiona zostanie mechanika aplikacji „Nauka”.

---

<sup>42</sup> Właściwość DataSource i DisplayMember oznaczają odpowiednio Źródło danych i wyświetlana kolumna

<sup>43</sup> Pole typu ComboBox to w praktyce lista rozwijana



Po uruchomieniu aplikacji „Nauka” użytkownik zostanie poproszony o wskazanie odpowiedniego pliku nauki. Odpowiedniego to znaczy takiego, który posiada odpowiednie rozszerzenie i strukturę. Dla plików ze słowami z języka angielskiego jest to rozszerzenie „\*.xmla”, a dla plików ze słowami z języka niemieckiego jest to „\*.xmln”. Jeśli użytkownik nie wskaże żadnego pliku aplikacja zakończy swoje działanie. Po wybraniu odpowiedniego pliku, plik zostanie wczytany do odpowiedniego zestawu danych. Pliki języka angielskiego wczytywane są do zestawu dsBazaEn, a języka niemieckiego do dsBazaDe. Po wczytaniu plików i wstępnym uzupełnieniu zmiennych globalnych takich jak liczba pytań<sup>44</sup>, krok procentowy oraz inne służące wygenerowaniu pierwszego pytania. Następnie wywoływana jest procedura Start, która wczytuje pierwsze pytanie i uzupełnia resztę zmiennych globalnych. Funkcja ta przekazuje sterowanie funkcji NastPyt, która wczytuje kolejne pytania. U góry programu użytkownik zobaczy słowo w języku polskim. Jego zadaniem jest wpisać w polu poniżej jego odpowiednie tłumaczenie. W przypadku słów z języka niemieckiego należy wybrać odpowiedni rodzajnik, zaimek oraz przypadek. Pola te nie są obowiązkowe, to znaczy, że w niektórych przypadkach ich wypełnianie nie ma sensu<sup>45</sup>. Po wpisaniu i uzupełnieniu odpowiednich pól należy kliknąć „Enter” na klawiaturze. Nastąpi sprawdzenie poprawności wpisanej odpowiedzi. Jeśli odpowiedź jest prawidłowa, to pole statystyki „Prawidłowe odpowiedzi” zostanie zaktualizowane i podświetli się na zielono. W przeciwnym razie pole to podświetli się na czerwono. Kolejno zostanie wygenerowane następne pytanie, jeśli takowe jeszcze są na liście. Jeśli użytkownik odpowiedział już na wszystkie pytania, to aplikacji wyświetli komunikat o zakończeniu nauki, po czym zakończy swoje działanie.

---

<sup>44</sup> W praktyce oznacza to liczbę słów w zestawie

<sup>45</sup> Tak jest, gdy wpisujemy przymiotnik oraz czasownik niemiecki

## Rozdział V Opis metody widzę, piszę i czytam, to czego się uczę

Zgodnie z celem tego projektu aplikacja, którą stworzyłem oferuje funkcjonalność podwójnego słownika języka angielskiego i niemieckiego oraz moduł do nauki słownictwa. Aplikacja ta opiera się na założonej hipotezie: „widzę, czytam i wpisuje, to czego się uczę”. Metoda ta oddziałuje jednocześnie na pięć kanałów percepcji<sup>46</sup>. Są to następująco:

1. Widzenie
2. Mówienie
3. Słuchanie
4. Pisanie
5. Powtarzanie

Widzenie. Metoda zakłada, że gdy użytkownik wpisuje jakieś słowo, widzi je na ekranie komputera. Jest to szczególnie dobra metoda nauki dla osób, które mają pamięć wzrokową. Osoby takie z łatwością zapamiętują obrazy, które widzą i uczą się poprzez widzenie.

Oto kilka charakterystycznych cech pamięci wzrokowej:

1. **Zapamiętywanie obrazów:** Łatwość w zapamiętywaniu detali widzianych obrazów, takich jak kolory, kształty i układ elementów.
2. **Przestrzenna organizacja:** Umiejętność przypominania sobie, gdzie konkretne elementy znajdują się w przestrzeni, na przykład rozmieszczenie przedmiotów w pokoju.
3. **Rozpoznawanie wzorców:** Szybkie rozpoznawanie i zapamiętywanie wzorców, takich jak twarze ludzi, schematy czy diagramy.

W opisaney metodzie najważniejsze z tych cech to zapamiętywanie obrazów oraz rozpoznawanie wzorców. Widząc wielokrotnie dane słowo użytkownik z silną pamięcią wzrokową łatwiej zapamięta to słowo.

Mówienie i słuchanie. Metoda zakłada, że użytkownik podczas wpisywania danego słowa, może je czytać zarówno w myślach jak i na głos, przez co jednocześnie słyszy to co czyta. Jest

---

<sup>46</sup> Mam tu na myśli zaangażowanie różnych zmysłów i dróg uczenia się

to szczególnie przydatne dla osób, które mają rozwiniętą pamięć słuchową. W przypadku języków takich jak język niemiecki jest to bardzo ważna umiejętność, ponieważ nierzadko w języku niemieckim wymowa różni się wyraźnie od pisowni. Zatem osoba, która czyta dane słowo w języku niemieckim może z łatwością zapamiętać jego wymowę.

Oto kilka cech pamięci słuchowej:

1. **Zapamiętywanie dźwięków:** Łatwość w zapamiętywaniu melodii, dialogów, dźwięków środowiska i innych informacji dźwiękowych.
2. **Rozpoznawanie tonów:** Umiejętność rozróżniania i zapamiętywania tonów głosu, akcentów, rytmów i melodii.
3. **Nauka przez słuchanie:** Skuteczność w nauce informacji słuchowych, na przykład poprzez słuchanie wykładów, audiobooków czy rozmów.

W prezentowanej przeze mnie metodzie szczególnie przydatne cechy to zapamiętywanie dźwięków oraz nauka przez słuchanie. Jest to szczególnie przydatne w nauce języka niemieckiego.

Pisanie i powtarzanie. Pisząc wielokrotnie dane słowo można w łatwy sposób zapamiętać słowo, którego się uczy. Dobra nauka powinna opierać się na częstym powtarzaniu. Pisanie rzeczy, której chcemy się nauczyć jest bardzo dobrą metodą nauki. W szczególności w nauce takich języków jak język niemiecki, który ma skomplikowaną pisownię niektórych słów.

Podsumowanie. Łącząc pięć wymienionych wyżej kanałów uczenia się można uzyskać dobry efektu uczenia się, w szczególności języków obcych. Jednoczesne pisanie, mówienie, słuchanie, czytanie i powtarzanie słowa, którego chcemy się nauczyć intensyfikuje naukę, przez co może ją przyspieszyć, co potwierdza skuteczność opisywanej metody, co udowadnia przyjętą w tym projekcie hipotezę.

## ZAKOŃCZENIE

Celem niniejszego projektu było stworzenie podwójnego słownika języka angielskiego i niemieckiego wraz z modułem do nauki słownictwa. Projekt ten miał na celu zbadanie, zgodnie z problemem badawczym, czy funkcjonalność podwójnego słownika języka angielskiego i niemieckiego umożliwi osiągnięcie lepszych efektów w nauce. Stworzone przeze mnie dwie aplikacje oferują taką funkcjonalność.

Pierwsza aplikacja, która jest rdzeniem tego projektu to słownik języka angielskiego lub niemieckiego w zależności od wyboru trybu pracy. Jest to aplikacja z prostym interfejsem, co sprawia, że jej obsługa jest prosta i intuicyjna.

Druga aplikacja to aplikacja będąca modułem do nauki słownictwa. Słowa, których użytkownik chce się uczyć generuje w aplikacji głównej. Dzięki takiemu rozwiązaniu zapewnia to spójność danych oraz działania. Użytkownik może uczyć się tylko słów, które wcześniej sam dodał do słownika, co daje mu możliwość swobodnego wyboru. Aplikacja do nauki wykorzystuje metodą „widzę, czytam i wpisuję” to czego się uczę, co jest hipotezą niniejszego projektu.

Podsumowując, można przypuszczać z dużym prawdopodobieństwem, że potrójny mechanizm tej metody daje dużo lepsze osiągnięcia w nauce oraz znacznie przyspiesza jej postęp. Zgodnie z opisanym w rozdziale V tego projektu schematem, w jaki dokładnie działa metoda „widzę, czytam i wpisuję” to czego się uczę możemy z całą stanowczością stwierdzić, że metoda ta umożliwia szybsze uczenie się i osiągnięcie lepszych rezultatów nauki, co potwierdza przyjętą hipotezę niniejszego projektu i pozytywnie odpowiada na przyjęty wcześniej problem badawczy.

## **SPIS WYKORZYSTANYCH ŹRÓDEŁ I OPRACOWAŃ**

1. draw.io – do tworzenia schematów - <https://app.diagrams.net>
2. Albahari Joseph, Johansen Eric – C# 8.0 w pigułce
3. Wrycza Stanisław, Marcinkowski Bartosz, Wyrzykowski Krzysztof – Język UML 2.0
4. Ulman Jeffrey D., Widom Jenniffer – Podstawowy kurs systemów baz danych
5. Michaelis Mark – C# 8.0 Kompletny przewodnik dla praktyków
6. Wielki słownik języka polskiego PAN - pamięć wzrokowa
7. Wielki słownik języka polskiego PAN – pamięć słuchowa

## SPIS RYSUNKÓW

|  |    |
|--|----|
| Rysunek 1 Główne okno aplikacji Słownik .....  | 5  |
| Rysunek 2 Formularz do zarządzania słowami angielskimi (u góry) oraz niemieckimi (na dole).....      | 6  |
| Rysunek 3 Formularz do zarządzania kategoriami .....   | 6  |
| Rysunek 4 Formularz do generowania plików do nauki .....   | 7  |
| Rysunek 5 Aplikacja do nauki słownictwa .....  | 8  |
| Rysunek 6 Schemat zestawu DataSet (bazy danych) .....  | 10 |
| Rysunek 7 Podstawowe stałe oraz funkcja btnZarzSłow_Click .....                                      | 11 |
| Rysunek 8 Funkcja wywoływana po kliknięciu na przycisk "Kategorie" .....                             | 12 |
| Rysunek 9 Funkcje ustawiające dane dla słownika języka angielskiego .....                            | 12 |
| Rysunek 10 Analogiczne funkcje dla słownika języka niemieckiego .....                                | 13 |
| Rysunek 11 Funkcja wywoływana podczas ładowania formularza .....                                     | 13 |
| Rysunek 12 Funkcja odpowiadająca za możliwość wpisywania w polu wyszukiwania niemieckich liter ..... | 14 |
| Rysunek 13 Funkcja odświeżająca zestaw danych .....  | 14 |
| Rysunek 14 Funkcja obsługująca kliknięcie listy ze słowami .....                                     | 15 |
| Rysunek 15 Wyszukiwanie elementu na liście .....   | 16 |
| Rysunek 16 Zdarzenie kliknięcia na przycisk Nauka .....  | 16 |
| Rysunek 17 Funkcja umożliwiająca dodanie nowej kategorii .....                                       | 17 |
| Rysunek 18 Funkcja usuwająca wybraną kategorię .....   | 18 |
| Rysunek 19 Nietypowy konstruktor formularza .....  | 18 |
| Rysunek 20 Funkcja usuwająca słowo ze słownika .....   | 19 |
| Rysunek 21 Formularz dodawania słowa języka niemieckiego (po lewej) i angielskiego (po prawej) ..... | 19 |
| Rysunek 22 Wyświetlenie odpowiednich elementów na formularzu w zależności od rodzaju słowa .....     | 20 |
| Rysunek 23 Funkcja dodająca słowo niemieckie do słownika .....                                       | 21 |
| Rysunek 24 Moduł do generowania plików nauki .....   | 22 |
| Rysunek 25 Funkcja wybierająca, z którego słownika będą brane słowa do nauki .....                   | 22 |
| Rysunek 26 Zdarzenie dodania słowa do listy do nauki .....   | 23 |
| Rysunek 27 Funkcja generująca plik do nauki .....  | 24 |
| Rysunek 28 Tabela w zestawie dla plików języka angielskiego .....                                    | 25 |

|   |    |
|---|----|
| Rysunek 29 Tabela w zestawie dla plików języka niemieckiego .....             | 25 |
| Rysunek 30 Kod wykonywany podczas otwarcia pliku do nauki.....                | 26 |
| Rysunek 31 Zmienne globalne użyte w programie .....                           | 26 |
| Rysunek 32 Funkcja startowa dla plików zawierających słowa angielskie .....   | 27 |
| Rysunek 33 Funkcja startowa dla plików zawierających słowa niemieckie .....   | 28 |
| Rysunek 34 Funkcja wczytująca następne pytanie ze słownika angielskiego ..... | 28 |
| Rysunek 35 Funkcja wczytująca następne pytanie ze słownika niemieckiego ..... | 29 |
| Rysunek 36 Funkcja sprawdzająca wpisaną wartość .....                         | 30 |
| Rysunek 37 Diagram przypadków użycia dla aplikacji Słownik.....               | 31 |

## **SPIS TABEL**

|   |    |
|---|----|
| Tabela 1 Zmienne globalne użyte w programie Nauka ..... | 27 |
|---|----|