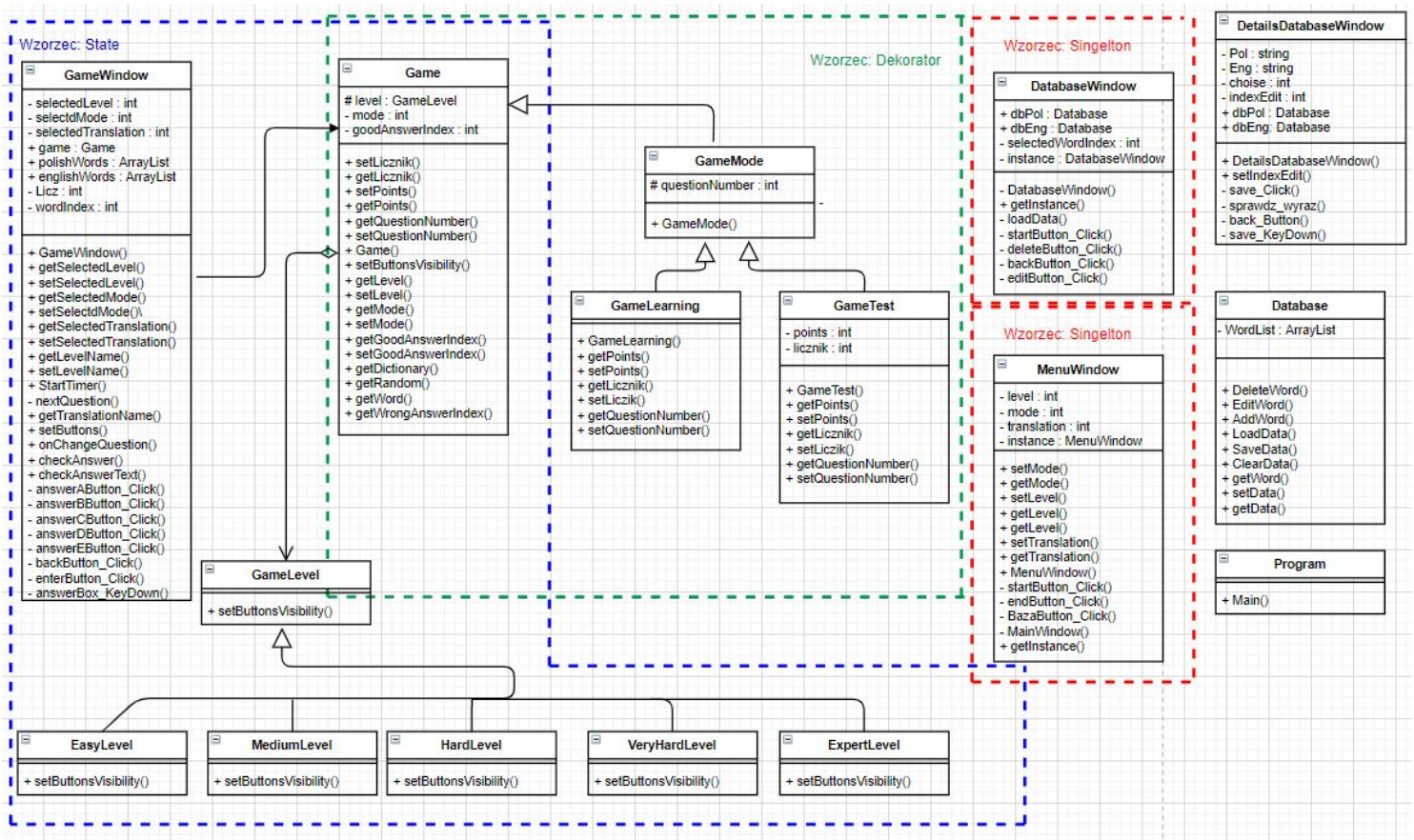
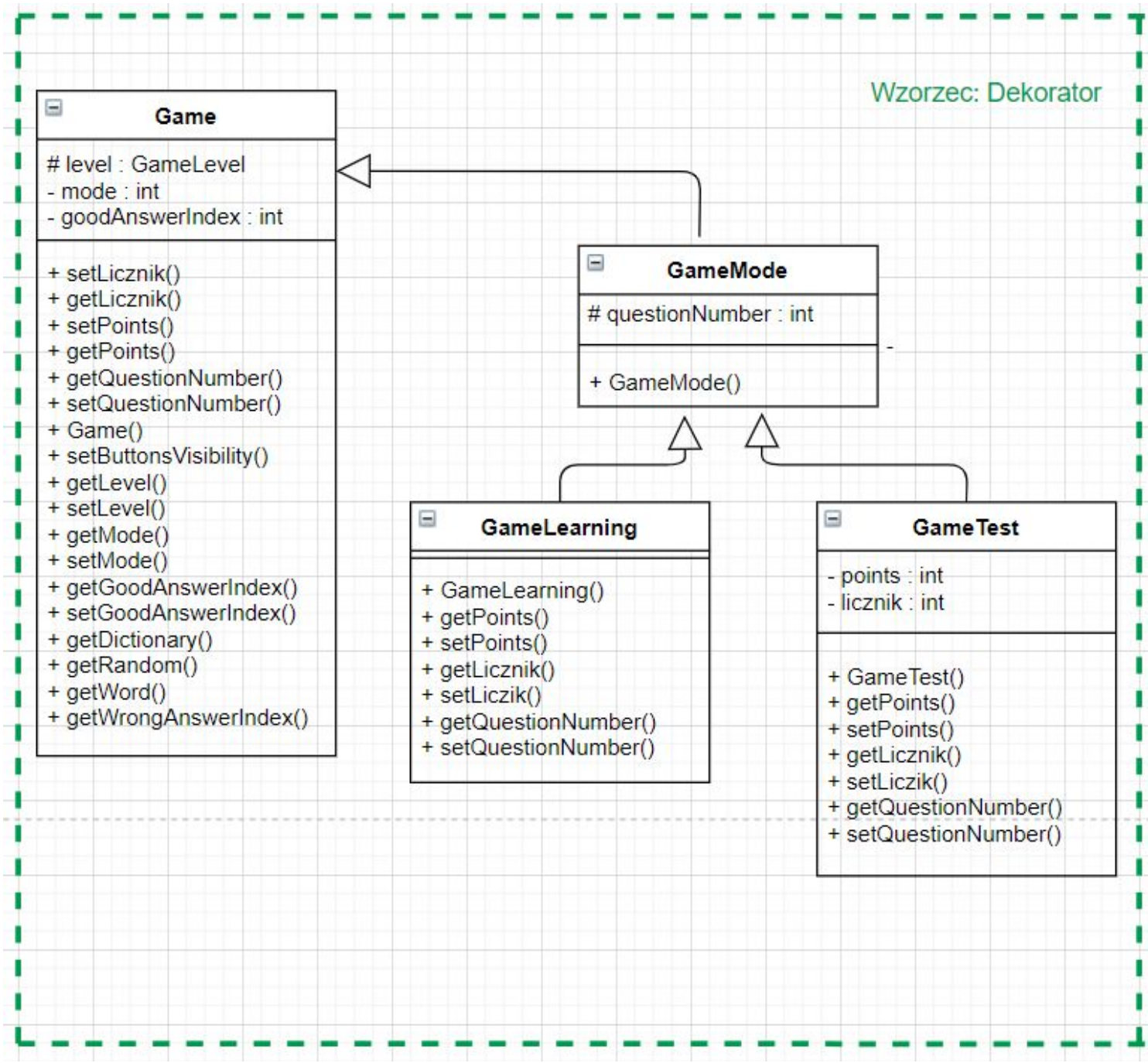
Wydział informatyki Pracownia Specjalistyczna Zaawansowanych Technik Programistycznych	Data: 21.01.2020r.
Projekt Temat: Program do nauki słówek języka obcego. Grupa: PS 6 Skład zespołu: 1.Warakomski Maciej 2.Wasiluk Maksymilian	Prowadzący: mgr inż. Daniel Reska Ocena:

1. Diagram klas



2. Wzorce użyte w projekcie.

2.1 Dekorator



2.1.1 - Wzorzec Decorator (dekorator) został użyty w celu zmniejszenia liczby klas potrzebnych do odpowiedniego zachowania aplikacji zależnie od wyboru danych opcji przez użytkownika (tj. trybu, z którego użytkownik chce korzystać). Wzorzec ten pozwolił na uporządkowanie klas oraz zmniejszenie ilości klas odpowiedzialnych za wszystkie opcje w menu.

2.1.2 - Klasa Game jest Komponentem.
Klasa GameMode jest Dekoratorem i dziedziczy z klasy Game.
Klasa GameLearning oraz GameTest są konkretnymi Dekoratorami A oraz B i dziedziczą z klasy GameMode.

2.1.3 - Dekorator korzysta z 4 klas: *Game.cs, GameMode.cs, GameLearning.cs, GameTest.cs .*
Wzorzec Dekorator, jest wzorcem należącym do grupy *wzorców strukturalnych.* Umożliwia on dodawania nowej funkcjonalności do aplikacji podczas działania programu. Dekorator daje nam większe możliwości niż statyczne dziedziczenie.

2.1.4 - Możliwość rozbudowy aplikacji za pomocą wzorca Dekorator o kolejne tryby inne niż tryb nauki i tryb testu, np. nauka wpisywania poprawnych form słówka w zdania,
lub znalezienie błędu oraz poprawa zdań.

Metody, których działanie zmieniają Dekoratory:

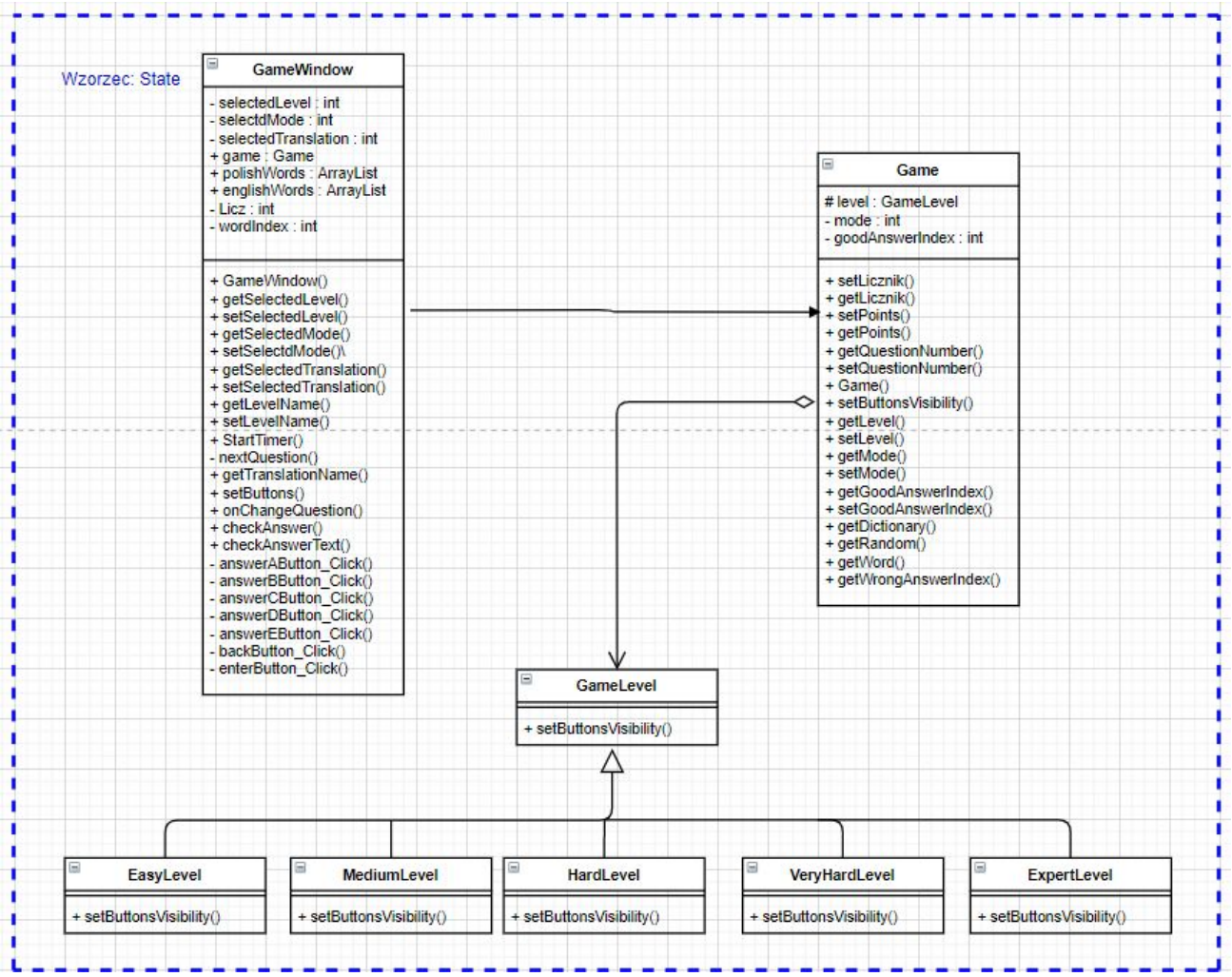
getPoints() - w klasie GameTest zwraca liczbę uzyskanych punktów, w klasie GameLearning zwraca wartość domyślną 0, gdyż metoda ta jest nieużywana w trybie nauki

setPoints() - w klasie GameTest ustawia liczbę uzyskanych punktów, w klasie GameLearning metoda ta jest pusta, ponieważ nie jest używana w trybie nauki

getLicznik() - w klasie GameTest zwraca ilość sekund jaka pozostała do udzielenia poprawnej odpowiedzi na pytanie, w klasie GameLearning zwraca wartość domyślną 0, gdyż metoda ta jest nieużywana w trybie nauki

setLicznik() - w klasie GameTest ustawia ilość sekund jaka pozostała do udzielenia poprawnej odpowiedzi na pytanie, w klasie GameLearning metoda ta jest pusta, ponieważ nie jest używana w trybie nauki

2.2 State



2.2.1 - Wzorzec State został użyty, aby pozbyć się wielu instrukcji warunkowych **if** i uzyskiwać różne działanie metody *setButtonsVisibility()* zależnie od wybranego przez użytkownika poziomu trudności. Metoda ta ustawia widoczność odpowiednich dla danego poziomu trudności przycisków i TextBoxa (wykorzystywanego na poziomie Ekspert).

Poszczególnym poziomom trudności, tj. Łatwy, Średni, Trudny, Bardzo Trudny i Ekspert odpowiadają analogicznie klasy EasyLevel, MediumLevel, HardLevel, VeryHardLevel oraz ExpertLevel.

2.2.2 - Klasa GameWindow jest Klientem.
Klasa Game jest Kontekstem.
Klasa GameLevel jest Stanem.
Klasa EasyLevel jest jednym ze stanów i dziedziczy z klasy GameLevel.
Klasa MediumLevel jest jednym ze stanów i dziedziczy z klasy GameLevel.
Klasa HardLevel jest jednym ze stanów i dziedziczy z klasy GameLevel.
Klasa VeryHardLevel jest jednym ze stanów i dziedziczy z klasy GameLevel.

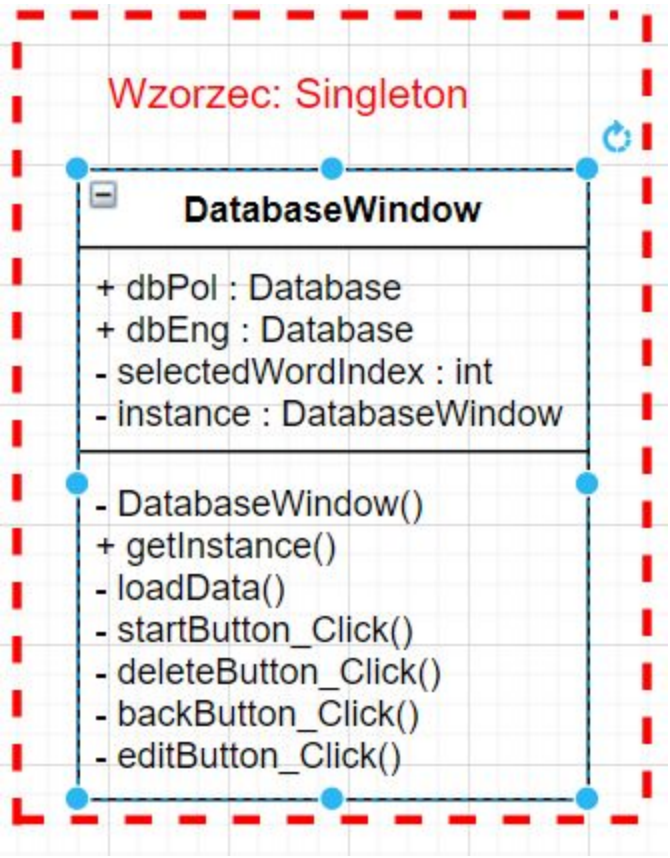
Klasa ExpertLevel jest jednym ze stanów i dziedziczy z klasy GameLevel.

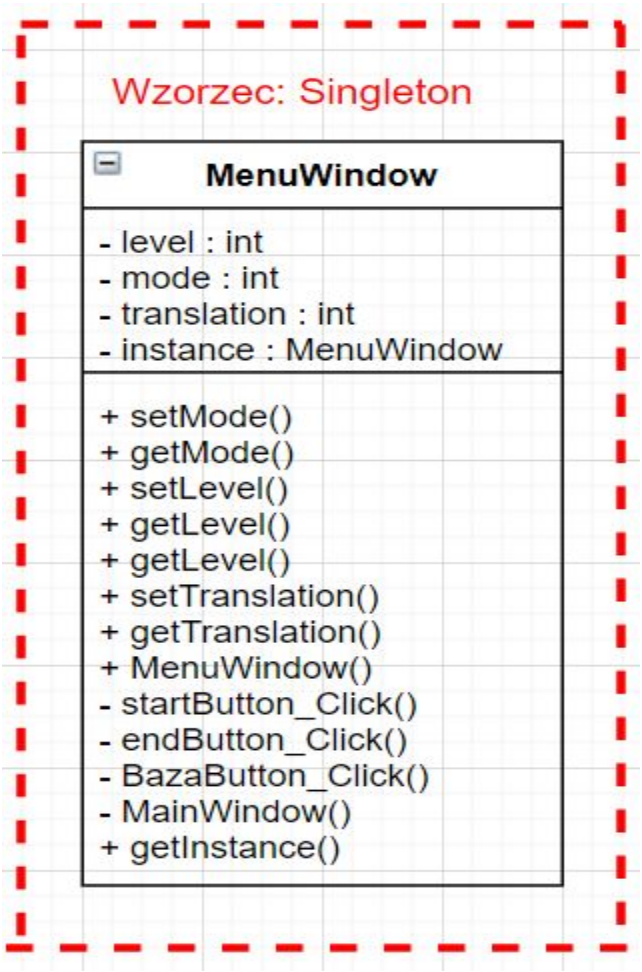
2.2.3 - State korzysta z 8 klas: *GameWindow.cs, Game.cs, GameLevel.cs, EasyLevel.cs, MediumLevel.cs, HardLevel.cs, VeryHardLevel.cs, ExpertLevel.cs.*
Wzorca State używamy, gdy stan obiektu może się zmieniać, gdy mamy dużo skomplikowanych instrukcji warunkowych lub tam gdzie logika warunkowa jest bardzo skomplikowana.

2.2.4 - Wektorem zmian wzorca State jest zachowanie się obiektu (działanie metod) w zależności od stanu. Jeśli chodzi o możliwość rozbudowy aplikacji za pomocą wzorca State, to można dodawać kolejne poziomy trudności. Dla każdego nowego poziomu trudności wystarczy dodać kolejną klasę dziedziczącą z klasy GameLevel oraz zaimplementować odpowiednio znajdującą się w niej abstrakcyjną metodę *setButtonsVisibility()*.

Kontekstem stanu jest klasa Game. Przy zmianie stanu zmienia się działanie metody *setButtonsVisibility()*, gdyż ustawia ona widoczność poszczególnych przycisków i TextBoxa (z poziomu trudności Ekspert) zależnie od wybranego przez użytkownika poziomu trudności.

2.3 Singleton





2.3.1 - Wzorzec Singleton został użyty w celu stworzenia ograniczenia do jednej instancji klas *DatabaseWindow* oraz *MenuWindow.* Dzięki temu unikniemy znanego problemu tworzenia się wielu okien tej samej klasy, gdy użytkownik przechodzi między oknami. Jeżeli obiekt okna już istnieje to niemożliwe będzie utworzenie nowego okna tego samego typu, a w przypadku takiej próby zostanie zwrócona instancja istniejącego już okna.

2.3.2 - Klasy *DatabaseWindow* oraz *MenuWindow* są Singletonami. Każda z nich zawiera prywatny konstruktor oraz publiczną metodę dostępu zwracającą jej instację.

2.3.3 - Wzorzec Singleton został wykorzystany w dwóch klasach *DatabaseWindow.cs* oraz *MenuWindow.cs .* Również w klasie *GameWindow.cs* pobierana jest instancja klasy MenuWindow. Oto miejsca w kodzie gdzie zaimplementowany został Singleton:

MenuWindow.cs:19-21

```
private static MenuWindow instance;

 private MenuWindow(){
…
}
```

MenuWindow.cs:34

```
public static MenuWindow getInstance()
        {
            if (instance == null || instance.IsDisposed)
                instance = new MenuWindow();
            return instance;
        }
```

DatabaseWindow.cs:20

```
private static DatabaseWindow instance;

        private DatabaseWindow()
        {
            InitializeComponent();
            loadData();
        }

        public static DatabaseWindow getInstance()
        {
            if (instance == null || instance.IsDisposed)
                instance = new DatabaseWindow();
            return instance;
        }
```

Użycia metody getInstance zostały opisane niżej.

2.3.4 - Wzorzec Singleton można wykorzystać przy rozbudowie programu w przypadku dodawania klas kolejnych okien, których instancje chcemy ograniczyć do jednej dla każdego okna (aby nie otwierało się wiele takich samych okien podczas przechodzenia pomiędzy nimi przez użytkownika).

Wszystkie użycia metody *getInstance* w programie:

MenuWindow.cs:115
DatabaseWindow db = DatabaseWindow.**getInstance();**

GameWindow.cs:315
MenuWindow menu = MenuWindow.**getInstance();**

DatabaseWindow.cs:89
MenuWindow menu = MenuWindow.**getInstance();**

Program.cs:19
MenuWindow menu = MenuWindow.**getInstance();**

3. Rozwiązanie specyficzne dla użytej technologii

Specyficznym rozwiązaniem dla języka C#, z którego skorzystaliśmy było przekazywanie referencji obiektu do funkcji poprzez wartość. Nie jest to możliwe w wielu językach, m.in. w popularnym C++, który jest jezykiem niższego poziomu niż C#. W klasie *GameWindow.cs:174* funkcja *onChangeQuestion* przyjmuje jako jeden z parametrów obiekt klasy Game. Metoda operuje na tym obiekcie i jest w stanie zmieniać jego stan. Mimo, że do funkcji przekazalismy tak naprawdę kopię referencji, jej wartość wskazuje na oryginalny obszar pamięci (na stercie), który zawiera instancję klasy Game. Dzięki temu możemy swobodnie operować przekazanym obiektem bez obaw iż po opuszczeniu metody utracimy wyniki naszych operacji.

Źródło: https://www.p-programowanie.pl/c-sharp/typy-wartosciowe-referencyjne/

4. Podział Pracy

Wasiluk Maksymilian	**Warakomski Maciej**
● **Utworzenie klas i widoków:** - MenuWindow (90%) - DatabaseWindow - DetailsDatabaseWindow - GameWindow (10%) - Database (90%) - Game (50%) - GameLearning (50%) - GameTest (50%) ● **Zaimplementowanie wzorca:** - Dekorator ● **Testowanie aplikacji oraz poprawa błędów (30%)** ● **Utworzenie Diagramów** ● **Utworzenie dokumentacji projektu (50%)**	● **Utworzenie klas i widoków:** - GameWindow (90%) - Database (10%) - MenuWindow (10%) - Game (50%) - GameLevel, EasyLevel, MediumLevel, HardLevel, VeryHardLevel, ExpertLevel - GameMode - GameLearning (50%) - GameTest (50%) ● **Zaimplementowanie wzorców:** - State - Singleton ● **Testowanie aplikacji oraz poprawa błędów (70%)** ● **Utworzenie dokumentacji projektu (50%)**

5. Instrukcja użytkownika

Po uruchomieniu programu mamy przed sobą widok Menu Głównego. Dostępne są różne konfiguracje programu. Po kliknięciu przycisku START w prawym dolnym rogu aplikacji przechodzimy do okna Gry. Niezależnie od wybranej konfiguracji znajdują się tam następujące elementy:

- w lewym dolnym rogu aplikacji wyświetla się informacja o wybranej aktualnie konfiguracji
- na górze pokazuje się numer aktualnie wyświetlanego słówka
- w prawym dolnym rogu znajduje się przycisk POWRÓT, którego kliknięcie spowoduje powrót do Menu Głównego

Dostępne w Menu Głównym możliwości konfiguracji to:

- **TRYB PROGRAMU**
    - **Tryb nauki** - w trybie tym pojawiają się słówka w danym języku, a pod spodem znajdują się przyciski z odpowiedziami (od 2 do 5 zależnie od trybu lub na poziomie Ekspert brak podpowiedzi - więc informacji w opisie poziomu trudności Ekspert) - tylko jedna z nich, która jest tłumaczeniem słówka na drugi język jest poprawna. Kliknięcie poprawnej odpowiedzi skutkuje wylosowaniem kolejnego słówka i kolejnego zestawu odpowiedzi, a także zwiększeniem się licznika wyświetlonych słówek. Kliknięcie niepoprawnej odpowiedzi spowoduje zaznaczenie jej na czerwono. W tym trybie nowe słówko nie zostanie wylosowane do momentu udzielenia poprawnej odpowiedzi.
    - **Tryb testu** - w trybie tym jest do odgadnięcia 20 słówek i na udzielenie poprawnej odpowiedzi przy każdym z nich są 4 sekundy. W przypadku braku poprawnej odpowiedzi lub gdy czas się skończy, program losuje kolejne słówko. Po wyświetleniu 20 słówek wyświetlany jest wynik (liczba zdobytych punktów, gdzie za każdą poprawną odpowiedź użytkownik otrzymuje 1 punkt)

- **POZIOM TRUDNOŚCI**
    - **Łatwy** - na tym poziomie użytkownik musi wybrać jedną poprawną odpowiedź spośród dwóch odpowiedzi
    - **Średni** - na tym poziomie musi wybrać jedną poprawną odpowiedź spośród trzech odpowiedzi
    - **Trudny** - na tym poziomie musi wybrać jedną poprawną odpowiedź spośród czterech odpowiedzi
    - **Bardzo trudny** - na tym poziomie musi wybrać jedną poprawną odpowiedź spośród pięciu odpowiedzi
    - **Ekspert** - na tym poziomie musi wpisać samodzielnie w okienko poprawną odpowiedź i zatwierdzić przyciskiem DALEJ lub klawiszem ENTER

- **TŁUMACZENIE**
    - **Polski-Angielski** - w trybie tym wyświetlają się polskie słówka i trzeba odgadnąć ich angielski odpowiednik
    - **Angielski-Polski** - w trybie tym wyświetlają się angielskie słówka i trzeba odgadnąć ich polski odpowiednik

W menu głównym na dole znajdują się jeszcze oprócz przycisku START (który rozpoczyna grę na wybranej wcześniej konfiguracji), przycisk WYJŚCIE (wyłącza on

program) oraz przycisk BAZA DANYCH. Po jego naciśnięciu, użytkownik przechodzi do okna Bazy Danych, gdzie wyświetlają się listy polskich i angielskich słówek, na których bazuje program. Słówka w jednym wierszu mają takie same znaczenie. Opcje dostępne w oknie Bazy Danych to:

- **DODAJ SŁOWO** - otwiera małe okno dodawania nowego słowa - należy wpisać polskie słowo oraz jego angielski odpowiednik. Po kliknięciu Zapisz, słowo zostaje dodane do obu słowników. Można też zrezygnować z dodania słowa klikając Odrzuć.
- **USUŃ SŁOWO** - aby przycisk zadziałał należy najpierw kliknąć na słowo, które chcemy usunąć ze słownika, a następnie na przycisk USUŃ SŁOWO. Wybrane słowo oraz jego odpowiednik w drugim języku zostaną usunięte na stałe z bazy danych.
- **EDYTUJ SŁOWO** - podobnie jak w przypadku przycisku USUŃ SŁOWO, aby przycisk zadziałał należy najpierw kliknąć na słowo, które chcemy edytować, następnie otwiera nam się małe okno edycji. Po zedytowaniu słowa, aby zmiany weszły w życie należy je zatwierdzić przyciskiem Zapisz. Można też zrezygnować z edycji klikając przycisk Odrzuć.
- **POWRÓT** - kliknięcie tego przycisku przeniesie nas z powrotem do Głównego Menu.

6. Instrukcja Instalacji

Aby móc skorzystać z aplikacji do nauki słówek języka obcego, należy pobrać aplikację z CEZA lub z linku: https://github.com/BladeStudios/Englearn. Następnie należy rozpakować archiwum i otworzyć folder który nam się utworzył. Należy przejść do folderu *Debug* znajdującego się w ścieżce: *ZTP/bin/Debug*. W folderze tym mamy dwa pliki *PolishDictionary.txt* oraz *EnglishDictionary.txt* są to pliki w których przetrzymujemy słówka polskie oraz angielskie i na początku są one wypełnione przykładowymi słówkami. Można do nich ręcznie wpisywać słówka oraz tłumaczenia, usuwać je i edytować, aczkolwiek interfejs aplikacji również oferuje taką możliwość. **Istnieje tych plików w folderze, w którym znajduje się plik *ZTP.exe* jest konieczne do poprawnego działania aplikacji!** Ostatnim krokiem w celu uruchomienia aplikacji Englearn jest otworzenie pliku *ZTP.exe.*

7. Analiza możliwości rozbudowy z uwzględnieniem użytych wzorców

Jedną z możliwości rozbudowy programu *Englearn* jest dodanie do aplikacji nowych poziomów trudności, co znacznie ułatwia nam wzorzec **State**, ponieważ dla każdego nowego poziomu wystarczy dodać jedną klasę dziedziczącą z klasy *GameLeve*l i zaimplementować w niej odpowiednią wersję metody *setButtonsVisibility*, która ustawia odpowiednią widoczność poszczególnych przycisków i Textboxa zależnie od poziomu.

Kolejną możliwoścIa rozbudowy aplikacji (dzięki wykorzystaniu wzorca **Decorator***),* jest dodanie nowych trybów gry. W tym programie mamy do wyboru dwa tryby: tryb nauki oraz tryb testu. Aby rozbudować aplikację można dodać np. tryb nauki poprzez zabawę, czyli stworzenie gry dzięki której użytkownik będzie mógł nauczyć się nowych słówek grając w ulubioną przez siebie grę. Wystarczy dodać jedną klasę, która będzie dziedziczyła z klasy *GameMode* i zaimplementować potrzebne pola i metody.

Również dodanie kolejnych okien w większości przypadków będzie wiązało się z użyciem wzorca **Singleton**, który idealnie nadaje się do zapewnienia istnienia tylko jednego obiektu danego okna.