

Politechnika Lubelska
Wydział Podstaw Techniki
Dokumentacja programu

GENERATOR DOBREJ ZABAWY

Ewa Podlowska & Sofiya Rylova
13 lutego 2022




Praca wykonana pod patronatem:
dr Paweł Właż

Spis treści

1. Główne zadania programu	2
2. Pobranie wieku i ilości osób	4
3. Losowanie filmu	6
4. Losowanie restauracji	9
5. Losowanie gry	12
6. Dodawanie gry	15
7. Losowanie muzyki	17
8. Losowanie prawdy czy wyzwania	20
9. Obsługa ikonki	22
10. Kierunek rozbudowy programu	23

1. Główne zadania programu

Każdy z nas był zapewne w sytuacji, gdy nie potrafił zdecydować się co chciałby obejrzeć, zjeść, albo po prostu miał ochotę spróbować czegoś nowego. Sytuacja znacznie komplikuje się, gdy decyzja nie należy tylko do nas, lecz kilku osób i każdy ma odmienne zdanie. Dlatego właśnie stworzyliśmy program ułatwiający te wybory i umilający wieczór. Generator dobrej zabawy, bo taka jest jego nazwa, losuje dla nas film z danego gatunku, typuje potrawę z wyselekcjonowanej kuchni i przenosi nas do strony internetowej restauracji gdzie ta potrawa znajduje się w menu. Oprócz tego możemy dokonać wyboru spośród 16 rodzajów muzyki i posłuchać playlistę na stronie internetowej You Tube. Następnie mamy do dyspozycji rodzaje i akcesoria do gry, z uwzględnieniem liczby osób otrzymujemy propozycję takiejże rozrywki. Na koniec mamy do wyboru prawdę i wyzwanie, ta gra nie potrzebuje żadnych innych akcesoriów, wystarczy tylko kliknąć aby wziąć udział w zabawie, jest więc dobrą alternatywą dla tych, którzy nie znają zasad innych gier. Dodatkowo interaktywna ikonka wprowadza przyjazną atmosferę podczas użytkowania programu.



GENERATOR DOBREJ ZABAWY

Wiek	<input type="text" value="0"/>	Ilość osób	<input type="text" value="1"/>	<input type="button" value="↑"/> <input type="button" value="↓"/>
Rodzaj	<input type="text"/>	Film	<input type="text"/>	<input type="button" value="Losuj"/>
Kuchnia	<input type="text"/>	Alkohol	<input type="text"/>	
Danie	<input type="text"/>	Restauracja	<input type="text"/>	Kliknij tutaj
	<input type="button" value="Losuj"/>			
Rodzaj muzyki	<input type="text"/>	Link	Kliknij tutaj	
Rodzaj gry	<input type="text"/>	Gra	<input type="text"/>	<input type="button" value="Losuj"/>
Akcesoria	<input type="text"/>			
Dodaj grę	<input type="text" value="nazwa"/>	Rodzaj gry	<input type="text"/>	
Ilość osób	<input type="text" value="min"/>	<input type="text" value="maks"/>	<input type="button" value="dodaj"/>	
Akcesoria	<input type="text"/>			
Prawda	<input type="button" value="Losuj"/>	Wyzwanie	<input type="button" value="Losuj"/>	
<input type="button" value="Wyjdź"/>				

Rysunek 1: główne okno programu

2. Pobranie wieku i ilości osób

Wiek jest stałą wpisaną przez użytkownika w pole wxTextCtrl. Zostaje pobrany przy pomocy funkcji getInt która przyjmuje wskaźnik do widgeta wxTextCtrl. Ilość osób pobierana jest przez funkcję GetValue z wxSpinCtrl.

```
void FunDialog::OnwiekText(wxCommandEvent& event)
{
    int wiekInt = getInt(wiek);
    if (wiekInt >= 60)
    {
        ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/old.png"))));
    }
    else if (wiekInt >= 18)
    {
        ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/medium_age.png"))));
    }
    else
    {
        ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/baby.png"))));
    }
}

void FunDialog::Onilosc_osobChange(wxSpinEvent& event)
{
    if (ilosc_osob->GetValue() < 1) {
        ilosc_osob->SetValue(1);
    }
}
```




GENERATOR DOBREJ ZABAWY

Wiek

Ilość osób

▲▼

Rysunek 2: wybór wieku z przedziału 0-17



GENERATOR DOBREJ ZABAWY

Wiek

Ilość osób

▲▼

Rysunek 3: wybór wieku z przedziału 18-59



GENERATOR DOBREJ ZABAWY

Wiek

Ilość osób

▲▼

Rysunek 4: wybór wieku 60+

3. Losowanie filmu

Użytkownik wybiera kategorię filmu z wxChoice. W klasie odczytywanie zapisana jest funkcja odczytaj plik zwracająca odczytany plik tekstowy w formie jednej zmiennej typu string. Linie oddzielone są od siebie znakami \r co ułatwia dalsze wykorzystanie tych danych. W klasie Film zapisana jest funkcja zwracająca tytuł filmu, rok produkcji oraz wymagany wiek. Linijka zawierająca te informacje jest w schemacie "tytuł, (rok), wiek \r, z tego względu funkcja odszukuje przecinki i pobiera kolejno tytuł, rok oraz wiek. W funkcji Onlos_filmClick znajduje się sprawdzenie czy wiek użytkownika jest mniejszy niż wiek wymagany filmu, jeśli tak, to należy znaleźć inny film odpowiadający wiekowi użytkownika. Gdy prawidłowy film zostaje znaleziony, jego tytuł wyświetlamy przy pomocy SetValue w wxTextCtrl.

```
Film::Film(string linia)
{
    int pierwszyPrzecinek = linia.find(',', 0);
    int drugiPrzecinek = linia.find(',', pierwszyPrzecinek + 1);

    int obecnaPozycja = 1;

    tytul = linia.substr(obecnaPozycja, pierwszyPrzecinek - 1 - obecnaPozycja);

    obecnaPozycja = pierwszyPrzecinek + 3;

    rok = stoi(linia.substr(obecnaPozycja, drugiPrzecinek - 2 - obecnaPozycja));

    obecnaPozycja = drugiPrzecinek + 2;

    wymaganyWiek = stoi(linia.substr(obecnaPozycja));
}

const string& Film::getTytul() const
{
    return tytul;
}

int Film::getRok() const
{
    return rok;
}

int Film::getWymaganyWiek() const
{
    return wymaganyWiek;
}
```

```

void FunDialog::Onlos_filmClick(wxCommandEvent& event)
{
    ikonka->SetBitmap(wxBitmap(wxImage( T("dane/ikonki/film.png"))));

    int gatunek = rodz_filmu->GetCurrentSelection();
    const vector<Film>& wybraneFilmy = filmy[gatunek];
    int wiekInt = getInt(wiek);

    int wymaganyWiek;
    Film const* wylosowanyFilm;

    do
    {
        int indeksFilmu = losuj(wybraneFilmy.size());
        wylosowanyFilm = &wybraneFilmy[indeksFilmu];

        wymaganyWiek = wylosowanyFilm->getWymaganyWiek();

    } while (wiekInt < wymaganyWiek);
    string pom = (wylosowanyFilm->getTytul());
    film->SetValue(wxString(pom.c_str(), wxConvUTF8));
}

```

Rysunek 5: Wybór rodzaju filmu

Rodzaj	<div> <div>▼</div> <div> akcji animowane dramaty fantasy horror komedie komedie akcji komedie romantyczne kryminały mystery przygodowe romantyczne science fiction superbohaterzy thrillery </div> </div>	Film	<input type="text"/> <input type="button" value="Losuj"/>
Kuchnia		Alkohol	<input type="text"/>
Danie		Restauracja	<div> <div>▼</div> <div> Kliknij tutaj </div> </div>
Rodzaj muzyki		Link	Kliknij tutaj

Rysunek 6: przykład losowania komedii



GENERATOR DOBREJ ZABAWY

Wiek

19

Ilość osób

1

Rodzaj


komedie

Film

Jumanji: Przygoda w dżungli

Losuj

Rysunek 7: przykład losowania filmu romantycznego



GENERATOR DOBREJ ZABAWY

Wiek

19

Ilość osób

1

Rodzaj


romantyczne

Film

Emili w Paryżu

Losuj

Rysunek 8: przykład losowania dramatu



GENERATOR DOBREJ ZABAWY

Wiek

36

Ilość osób

1

Rodzaj

dramaty

Film

Dom Gucci

Losuj

4. Losowanie restauracji

W klasie `Jedzenie` odczytywana jest ze zmiennej typu `string` nazwa dania oraz restauracje które te danie mają w menu. Funkcje `odczytajJedzenie`, `odczytajAlkohol` oraz `Onlos_kuchniaClick` odpowiadają za wyświetlenie nazwy dania w `wxTextCtrl` oraz za dynamiczne dodanie przy pomocy `AppendString` nazw restauracji w `wxChoice`. Gdy użytkownik wybierze rodzaj dania, wylosuje nazwę potrawy oraz pojawią się restauracje w `wxChoice` użytkownik może wybrać jeden z tych punktów gastronomicznych. Po jego wybraniu funkcja `OnresturacjaSelect` przekazuje link do danej restauracji w `HyperlinkCtrl2`. Po kliknięciu na napis "kliknij tutaj" otwiera się przeglądarka ze stroną internetową wybranej restauracji.

```
void FunDialog::odczytajJedzenie(const char* sciezkaPliku, TypJedzenia typ)
{
    string text = odczytajPlik(sciezkaPliku);
    int koniecLinii = 0;

    while ((koniecLinii = text.find('\r')) != -1)
    {
        string linia = text.substr(0, koniecLinii);
        text = text.substr(koniecLinii + 1);
        jedzenie[typ].emplace_back(linia);
    }
    cout << "jedzenie " << typ << " " << jedzenie[typ].size() << '\n';
}

void FunDialog::odczytajAlkohol(const char* sciezkaPliku, TypJedzenia typ)
{
    string text = odczytajPlik(sciezkaPliku);
    int pozycja = 0;
    int koniecLinii = 0;

    string linia;
    while ((koniecLinii = text.find('\r', pozycja)) != -1)
    {
        linia = text.substr(pozycja, koniecLinii - pozycja);
        alkohole[typ].push_back(linia);
        pozycja = koniecLinii + 1;
    }
    linia = text.substr(pozycja);
    alkohole[typ].push_back(linia);
}
```

```

void FunDialog::odczytajRestauracje(const char* sciezkaPliku)
{
    string text = odczytajPlik(sciezkaPliku);
    int koniecLinii = 0;

    while ((koniecLinii = text.find('\r')) != -1)
    {
        string linia = text.substr(0, koniecLinii);
        text = text.substr(koniecLinii + 1);

        int przecinek = linia.find(',');
        string restauracjaStr = linia.substr(0, przecinek);
        string linkRestauracji = linia.substr(przecinek + 1);

        restauracje[restauracjaStr] = linkRestauracji;
    }

    cout << restauracje.size() << '\n';
}

void FunDialog::Onlos_kuchniaClick(wxCommandEvent& event)
{
    ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/yummy.png"))));

    int wybranaKuchnia = kuchnia->GetCurrentSelection();

    const vector<Jedzenie>& jedzenieKuchni = jedzenie[wybranaKuchnia];
    int wylosowane = losuj(jedzenieKuchni.size());

    const Jedzenie& wylosowaneJedzenie = jedzenieKuchni[wylosowane];
    const vector<string>& wylosowaneRestauracje = wylosowaneJedzenie.getRestauracje();

    danie->Clear();
    *danie << wylosowaneJedzenie.getNazwa();

    restauracja->Clear();
    for (int i = 0; i < wylosowaneRestauracje.size(); ++i) {
        restauracja->AppendString(wxString((wylosowaneRestauracje[i]).c_str(), wxConvUTF8));
    }
}


```

Rysunek 9: Wybór rodzaju kuchni

Kuchnia	indyjskie	Alkohol	<input type="text"/>
Danie	<div> <div>mejsykańskie</div> <div>indyjskie</div> <div>sushi</div> <div>amerykańskie</div> <div>polskie</div> <div>wegetariańskie</div> <div>kebaby</div> <div>desery</div> <div>fast food</div> </div>	Restauracja	<div> <div></div> <div>Kliknij tutaj</div> </div>
Rodzaj muzyki		Link	<div> <div></div> <div>Kliknij tutaj</div> </div>

Rysunek 10: Wybór rodzaju kuchni

Rysunek 11: przykład losowania dania wegetariańskiego



GENERATOR DOBREJ ZABAWY

Wiek Ilość osób

Rodzaj Film

Kuchnia Alkohol

Danie Restauracja


[Kliknij tutaj](#)

Rysunek 12: przykład losowania dania indyjskiego

Kuchnia Alkohol

Danie Restauracja

Rysunek 13: przykład wyboru restauracji, w przypadku, kiedy masz poniżej 18 lat



GENERATOR DOBREJ ZABAWY

Wiek Ilość osób

Rodzaj Film

Kuchnia Alkohol

Danie Restauracja

5. Losowanie gry

Użytkownik wybiera rodzaj gry oraz akcesoria i z uwzględnieniem ilości osób zostaje losowana odpowiednia gra. Klasy TypGry oraz Gra odpowiadają za pobranie nazwy, akcesoriów oraz minimalnej I maksymalnej ilości graczy. W FunMain.cpp funkcje odczytajGry oraz Onlos_graClick odpowiadają za dalsze czynności aby przy pomocy SetValue wyświetlić nazwę gry lub komunikat, że nie znaleziono gry spełniającej takie kryteria jak aktualna liczba osób w grupie, akcesoria które wybrali użytkownicy oraz typ gry. Nie znajdziemy na przykład gry, która jest karciana I do której żadne akcesoria nie są potrzebne.

```
Gra::Gra(string linia)
{
    int obecnaPozycja = 0;
    int pierwszyPrzecinek = linia.find(',', obecnaPozycja);
    int drugiPrzecinek = linia.find(',', pierwszyPrzecinek + 1);
    int trzeciPrzecinek = linia.find(',', drugiPrzecinek + 1);
    int czwartyPrzecinek = linia.find(',', trzeciPrzecinek + 1);

    nazwa = linia.substr(obecnaPozycja, pierwszyPrzecinek);
    obecnaPozycja = pierwszyPrzecinek + 1;

    string typString = linia.substr(obecnaPozycja, drugiPrzecinek - obecnaPozycja);
    typ = toTypGry(typString);
    obecnaPozycja = drugiPrzecinek + 1;

    string minLiczbaGraczyStr = linia.substr(obecnaPozycja, trzeciPrzecinek - obecnaPozycja);
    minLiczbaGraczy = stoi(minLiczbaGraczyStr);
    obecnaPozycja = trzeciPrzecinek + 1;

    string maxLiczbaGraczyStr = linia.substr(obecnaPozycja, czwartyPrzecinek - obecnaPozycja);
    maxLiczbaGraczy = stoi(maxLiczbaGraczyStr);
    obecnaPozycja = czwartyPrzecinek + 1;

    string akcesoriumString = linia.substr(obecnaPozycja);
    akcesoria = toAkcesoria(akcesoriumString);
}
```



```

void FunDialog::Onlos_graClick(wxCommandEvent& event)
{
    ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/gry.png"))));

    TypGry wybranyTypGry = (TypGry)gra->GetCurrentSelection();
    Akcesoria wybraneAkcesorium = (Akcesoria) akcesoria->GetCurrentSelection();

    int liczbaGraczy = ilosc_osob->GetValue();

    const vector<Gra>& wybraneGry = gry[wybranyTypGry];
    vector<Gra> gryDoWyboru;
    for (const auto& gra : wybraneGry)
    {
        if (gra.getAkcesoria() == wybraneAkcesorium
            && liczbaGraczy >= gra.getMinLiczbaGraczy()
            && liczbaGraczy <= gra.getMaxLiczbaGraczy())

            gryDoWyboru.push_back(gra);
    }

    if (gryDoWyboru.size() == 0)
    {
        TextCtrl1->SetValue("Nie ma takich gier");
    }
    else
    {
        int wylosowanyIndeks = losuj(gryDoWyboru.size());
        string pom = (gryDoWyboru[wylosowanyIndeks].getNazwa());
        TextCtrl1->SetValue(wxString(pom.c_str(), wxConvUTF8));
        // message box z instrukcja gry
    }
}

```

Rysunek 14: Wybór rodzaju gry

Rodzaj gry	<div> <div>karciane</div> <div> karciane planszowe inne </div> </div>	Gra	<div> <div>poker</div> <div>Losuj</div> </div>
Akcesoria			

Rysunek 15: Wybór akcesoriów

Rodzaj gry	<div> <div>karciane</div> <div> karty kartki długopisy karty inne nic </div> </div>	Gra	<div> <div>poker</div> <div>Losuj</div> </div>
Akcesoria			
Dodaj grę	<div> <div>nazwa</div> <div></div> </div>	Ilość osób	<div> <div>min</div> <div>maks</div> </div>

Rysunek 16: przykład losowania gry

Rodzaj gry	karciane	Gra	uno	Losuj
Akcesoria	inne			

Rysunek 17: przykład losowania gry w zależności od wieku i ilości osób

Wiek	22	Ilość osób	2	
Rodzaj		Film		Losuj
Kuchnia		Alkohol		
Danie		Restauracja		
	Losuj		Kliknij tutaj	
Rodzaj muzyki		Link	Kliknij tutaj	
Rodzaj gry	karciane	Gra	test1	Losuj
Akcesoria	nic			

6. Dodawanie gry

Dodawanie nowej gry do pliku odbywa się w klasie Gra w funkcji zapisz przy pomocy fstream. W FunMain.cpp funkcja OndodajClick pobiera potrzebne dane takie jak nazwa gry, typ gry, akcesoria potrzebne do gry, minimalną i maksymalną liczbę graczy. Funkcja sprawdza także czy gra o danej nazwie istnieje już w pliku tekstowym, jeśli tak informuje o tym użytkownika, jeśli nie, zapisuje grę i wyświetla odpowiedni komunikat w wxMessageBox.

```
void FunDialog::OndodajClick(wxCommandEvent& event)
{
    string nazwaGry = nazwa->GetValue().ToString();
    TypGry typGry = (TypGry)rodz_dod->GetCurrentSelection();
    Akcesoria akcesoriaGry = (Akcesoria)akc_dod->GetCurrentSelection();
    int minLiczbaGraczy = getInt(min);
    int maxLiczbaGraczy = getInt(maks);

    bool graIstnieje = false;
    for (int i = 0; i < ILOSC_TYPOW_GIER; ++i)
    {
        for (const Gra& gra : gry[(TypGry) i])
        {
            if (nazwaGry == gra.getNazwa())
            {
                graIstnieje = true;
            }
        }
    }
    if (graIstnieje)
    {
        nazwa->SetValue("nazwa");
        min->SetValue("min");
        maks->SetValue("maks");
        wxMessageBox("Gra już istnieje");
    }
    else
    {
        Gra gra(nazwaGry, typGry, minLiczbaGraczy, maxLiczbaGraczy, akcesoriaGry);
        gry[typGry].push_back(gra);

        gra.zapisz();

        wxMessageBox("zapisano gra");
    }
}
```



```

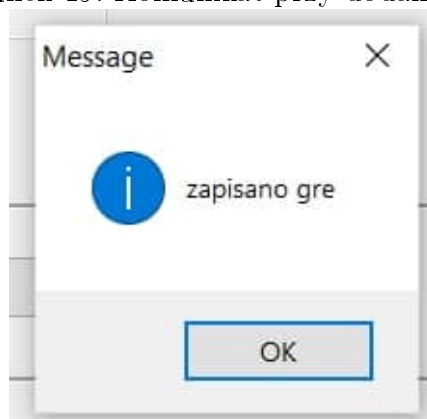
void Gra::zapisz()
{
    fstream plik("dane/reszta/grz.txt", ios_base::app);
    plik << nazwa << ',' << toString(typ)
        << ',' << minLiczbaGraczy
        << ',' << maxLiczbaGraczy << ',' << toString(akcesoria)
        << "\r\n";
}

```

Rysunek 18: Przykładowe dodawanie gry

Dodaj grę	<input type="text" value="rubik"/>	Rodzaj gry	<input type="text" value="inne"/>	Ilość osób	<input type="text" value="2"/>	<input type="text" value="8"/>
Akcesoria	<input type="text" value="nic"/>	<input type="button" value="dodaj"/>				

Rysunek 19: Komunikat przy dodaniu gry



7. Losowanie muzyki

Nie wyobrażam sobie dobrego wieczoru bez nastrojowej muzyki, dlatego właśnie użytkownik ma do wyboru aż 16 gatunków muzyki. Po wybraniu jednego z nich z wxChoice i kliknięciu na HyperlinkCtrl program przenosi nas na stronę internetową YouTube z gotową playlistą stosowną do wybranego gatunku muzyki. Niektóre playlisty liczą aż ponad 2000 utworów, więc każdy znajdzie coś dla siebie. Odczyt linku z pliku tekstowego jest prosty i wykonuje się (przy uprzednim zapisaniu zawartości pliku w zmiennej typu string w klasie odczytywanie) w funkcji odczytajMuzyke. W pliku znajduje się nazwa playlisty oddzielona przecinkiem od linka, a po wczytaniu w klasie odczytywanie linijki oddzielone są od siebie \ r. Ustawienie odpowiedniego linka w HyperlinkCtrl odbywa się w funkcji OnmuzykaSelect.

Rysunek 20: Wybór rodzaju muzyki

```
void FunDialog::odczytajMuzyke(const char* sciezkaPliku)
{
    string text = odczytajPlik(sciezkaPliku);
    int koniecLinii = 0;

    gatunkiMuzyki.resize(ILOSC_GATUNKOW_MUZYKI);

    while ((koniecLinii = text.find('\r')) != -1)
    {
        string linia = text.substr(0, koniecLinii);
        text = text.substr(koniecLinii + 1);

        int przecinek = linia.find(',');
        string gatunekStr = linia.substr(0, przecinek);
        string adresStrony = linia.substr(przecinek + 1);

        GatunekMuzyki gatunekMuzyki = toGatunekMuzyki(gatunekStr);

        gatunkiMuzyki[gatunekMuzyki] = adresStrony;
    }
}
```

```
void FunDialog::OnmuzykaSelect(wxCommandEvent& event)
{
    ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/music.png"))));
    int wybranaMuzyka = muzyka->GetCurrentSelection();
    HyperlinkCtrl1->SetURL(gatunkiMuzyki[wybranaMuzyka]);
}
```

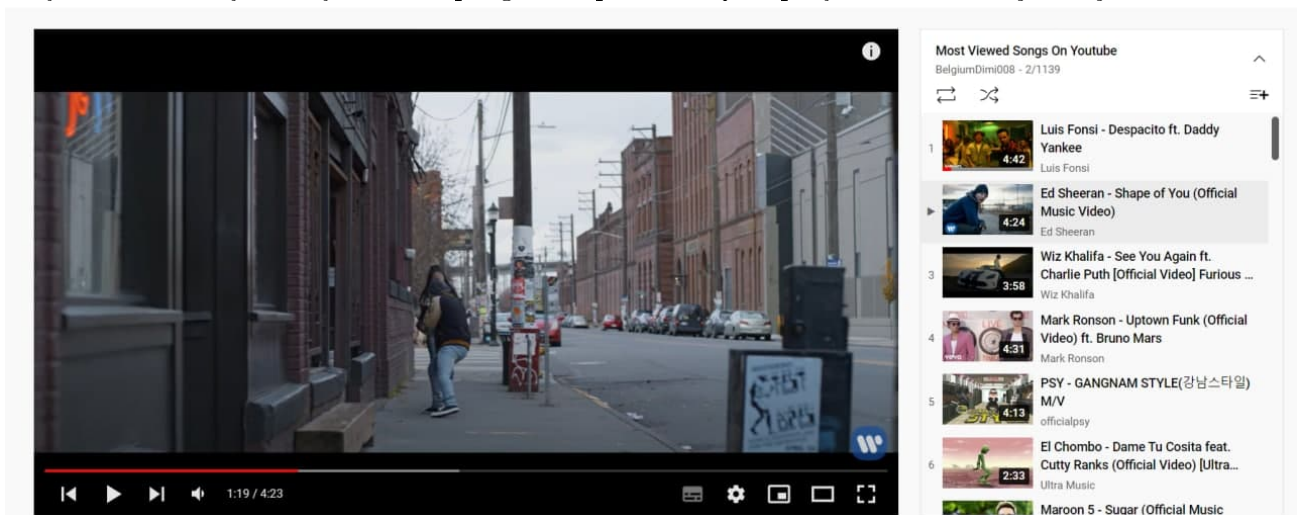
Rysunek 21: Wybór rodzaju muzyki

Rodzaj muzyki	<div><div>▼</div><div>pop stare pop nowe stare stare polskie rap rap polski klasyka filmowa disco polo rock rock polski hip-hop elektroniczna klubowa reggae jazz</div></div>	Link	Kliknij tutaj
Rodzaj gry		Gra	<input type="text"/> Losuj
Akcesoria			
Dodaj grę	<input type="text" value="nazwa"/>	Ilość osób	<input type="text" value="min"/> <input type="text" value="maks"/> <input type="button" value="dodaj"/>
Akcesoria	<input type="text"/>		
Prawda	<input type="button" value="Losuj"/>	Wyzwanie	<input type="button" value="Losuj"/>

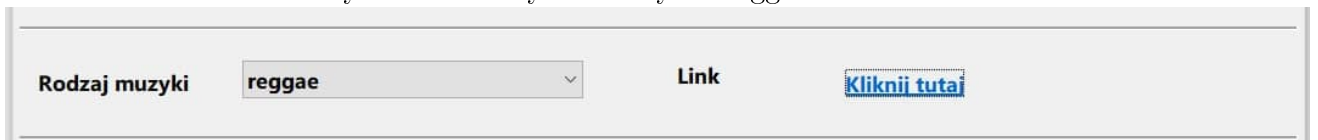
Rysunek 22: Wybór muzyki "pop nowe"



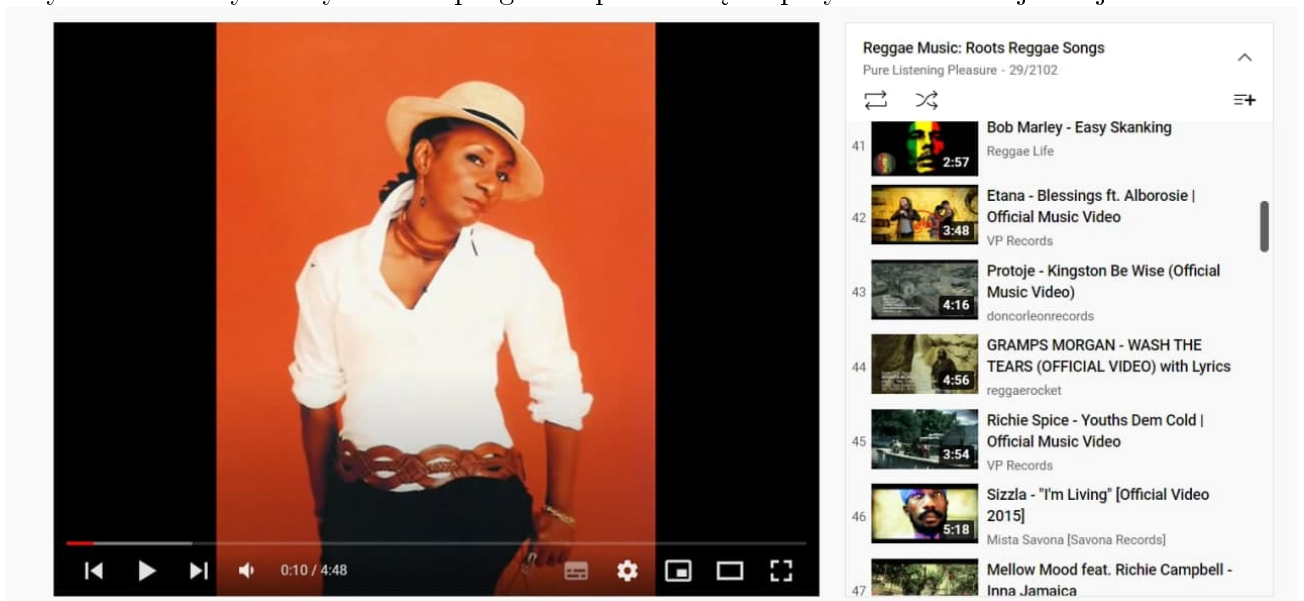
Rysunek 23: Wynik wywołania programu po kliknięciu przycisku "Kliknij tutaj"



Rysunek 24: Wybór muzyki "reggae"



Rysunek 25: Wynik wywołania programu po kliknięciu przycisku "Kliknij tutaj"



8. Losowanie prawdy czy wyzwania

Odczytanie plików z prawdą i wyzwaniem jest proste, gdyż każda prawda i każde wyzwanie jest zapisane w osobnej linii. Po wczytaniu plików przez klasę odczytywanie pozostaje tylko funkcjom odczytajPrawdy i odczytajWyzwanie znaleźć w zmiennej typu string \r oddzielające każdą linię tekstu. Wylosowana prawda lub wyzwanie jest wyświetlana przy pomocy funkcji onprawdaClick i onwyzwanieClick w wxMessageBox.

```
void FunDialog::odczytajPrawdy(const char* sciezkaPliku)
{
    string text = odczytajPlik(sciezkaPliku);
    int koniecLinii = 0;

    while ((koniecLinii = text.find('\r')) != -1)
    {
        string linia = text.substr(0, koniecLinii);
        text = text.substr(koniecLinii + 1);
        prawdy.push_back(linia);
    }
}

void FunDialog::odczytajWyzwania(const char* sciezkaPliku)
{
    string text = odczytajPlik(sciezkaPliku);
    int koniecLinii = 0;

    while ((koniecLinii = text.find('\r')) != -1)
    {
        string linia = text.substr(0, koniecLinii);
        text = text.substr(koniecLinii + 1);
        wyzwania.push_back(linia);
    }
}
```

```

void FunDialog::OnprawdaClick(wxCommandEvent& event)
{
    ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/prawda.png"))));

    int indeks = losuj(prawdy.size());
    const string& text = prawdy[indeks];
    wxString s2w(string text);
    cout << text << '\n';
    wxMessageBox(wxString(text.c_str()), wxConvUTF8);

    ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/prawda2.png"))));
}

void FunDialog::OnwyzwanieClick(wxCommandEvent& event)
{
    ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/wyzwanie.png"))));

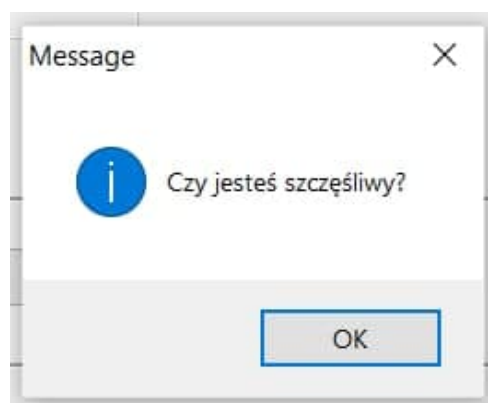
    int indeks = losuj(wyzwania.size());
    const string& text = wyzwania[indeks];
    wxString s2w(string text);
    cout << text << '\n';
    wxMessageBox(wxString(text.c_str()), wxConvUTF8);

    ikonka->SetBitmap(wxBitmap(wxImage(_T("dane/ikonki/wyzwanie2.png"))));
}

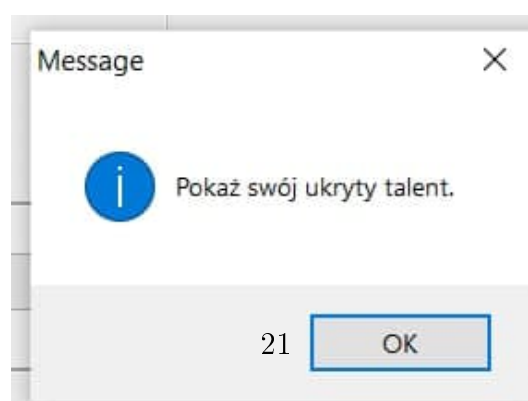
```



Rysunek 26: Przykład wylosowania prawdy



Rysunek 27: Przykład wylosowania wyzwania



9. Obsługa ikonki

W lewym górnym rogu jest wxBitmap która reaguje (przy użyciu funkcji SetBitmap) na poszczególne działania użytkownika. Domyślnie ikonka ma ustawioną emoji w okularach przeciwsłonecznych. Przy wpisywaniu wieku gdy jest on poniżej 18 lat pojawia się emoji dziecka, poniżej 60 emoji ze znacznikiem peace, a powyżej 60 emoji staruszka. Po wylosowaniu filmu emoji jest w okularach 3D, po wylosowaniu kuchni emoji trzyma widelec I nóż. Analogicznie gdy użytkownik wybiera muzykę emoji ma słuchawki, gdy wybiera gry emoji wygląda na podekscytowane. Podobnie zmienia się przy wylosowaniu prawdy I wyzwania. Po kliknięciu wyjdź emotka zmienia się na machającą na pożegnanie.



10. Kierunek rozbudowy programu

Planujemy dodać więcej filmów, obecnie jest ich po około 50 z każdego gatunku. Przy użyciu wtyczki selector gadget nie jest to czasochłonne zadanie, więc planujemy wprowadzić po 1000 tytułów filmów każdego rodzaju. Rozszerzenie wyboru kuchni i restauracji jest dość trudnym zadaniem, poświęciliśmy dużo czasu na wyszukanie restauracji, jej dań i innych restauracji w których to samo danie pojawia się w menu. Ten obszar wymaga ulepszeń, wierzę, że da się usprawnić wyszukiwanie tych obiektów bez ręcznego wpisywania tekstu. Ciekawym rozwiązaniem byłoby dodanie restauracji z kilku miast (obecnie wszystkie restauracje znajdują się w lublinie). Wówczas należałoby zapytać się użytkownika w jakim mieście przebywa. Losowanie alkoholi ogranicza się do podania rodzaju trunku, dobrze byłoby uściślić to do nazwy firmy np. piwo Namysłów lub whiskey Ballantine's. W pliku z muzyką dodałybyśmy kilka playlist do każdego gatunku aby zwiększyć wybór. Pod wylosowaną nazwą gry można wstawić link do gry np. na stronie kurnik.pl lub link do wyjaśnienia zasad gry. Dodanie nowej gry estetyczniej wyglądałoby w nowym oknie. Gra prawdy czy wyzwanie mogłaby mieć więcej opcji, aby przy dłuższej grze pytania się nie powtarzały. Po naciśnięciu przycisku Wyjdź ciekawym pomysłem byłoby stworzenie animacji, gdy ikonka zmienia się na machającą na pożegnanie emotkę można zrobić przejścia które będą kolejnymi etapami machania. Jak można zauważyć istnieje wiele możliwości poszerzenia działania programu, należy jednak zachować zdrowy umiar.