



Bioinformatyczne Studenckie Koło Naukowe (BioSKN)
przy Wydziale Automatyki, Elektroniki i Informatyki
Politechniki Śląskiej w Gliwicach

RAPORT Z PROJEKTU

NA KONKURS

BIOINFORMATYCZNEGO STUDENCKIEGO KOŁA
NAUKOWEGO



Temat projektu:

*GeneBrowser - program, który dla zadanej listy genów,
pochodzącej z pliku anotacji GO, wybiera unikalne
symbole i pobiera odpowiednie atrybuty łącząc się
zdalnie z bazą Biomart.*

Wykonanie:

Katarzyna Jonak

Biotechnologia, rok III

e-mail: jonak.katarzyna@gmail.com

Spis treści

- 1. Cel projektu**
 - 2. Wstęp teoretyczny**
 - 3. Specyfikacja**
 - 4. Omówienie napotkanych problemów**
 - 5. Dodatkowe zastosowanie programu**
 - 6. Wnioski**
 - 7. Literatura**
-

1. Cel projektu

Celem niniejszego projektu jest stworzenie programu, który z pliku anotacji GO dla człowieka wybiera unikalne symbole genów, a następnie łączy się zdalnie z bazą BioMart i wyszukuje informacje na temat podanej listy genów, korzystając z metody wyszukiwania jaką oferuje narzędzie BioMart. Program w wyniku zwraca cztery pliki z zapisanymi informacjami pobranymi z bazy BioMart. W pierwszym pliku znajdują się Ensembl ID (ID to identyfikatory genów), w drugim ID symboli (dla człowieka to ID HGNC), w trzecim UniProt/SwissProt ID i w czwartym symbole Affymetrix dla danej mikromacierzy (dla człowieka jest to mikromacierz HG U-133 PLUS2). W każdym z czterech plików zapisywany jest również dany symbol genu przy odpowiadającym mu wyniku.

Program GeneBrowser, po przeprowadzeniu niewielkich modyfikacji kodu, może również znaleźć zastosowanie podczas wyszukiwania informacji dotyczących innych organizmów, których pliki anotacji GO są dostępne na stronie Gene Ontology.

2. Wstęp teoretyczny

Projekt GeneBrowser łączy ze sobą kilka przedsięwzięć rewolucjonizujących świat nauk biologicznych, w którym wyszukiwanie danych dotyczących konkretnych genów i ich produktów w nadmiarze informacji stało się bardzo utrudnione, bądź wręcz niemożliwe. Mowa tu głównie o programie Gene Ontology (Ontologie Genowe – wspólne ontologie dla różnych baz danych) oraz bazie BioMart, która pozwala na wyszukiwanie informacji o genach w różnych bazach danych.

Program wyszukujący informacje o genach został wykonany w języku programowania JAVA. Język ten jest obecnie jednym z najszybciej rozwijających się języków programistycznych o szerokim spektrum zastosowania, dużą liczbą bibliotek i wielu udogodnieniach dla programistów. Dlatego też zdecydowano się na opracowanie programu w tym języku.

2.1. Gene Ontology

Gene Ontology (GO – Ontologie Genowe, strona internetowa: www.geneontology.org) to projekt prowadzony przez Europejski Instytut Bioinformatyki (EBI). Jest on częścią większego projektu Open Biomedical Ontologies (OBO – Otwarte Ontologie Biomedyczne). Oba przedsięwzięcia są próbą stworzenia kontrolowanego słownictwa dla różnych biologicznych i medycznych domen [1].

Projekt GO został stworzony w 1998 roku przez konsorcjum badaczy analizujących genomy trzech organizmów modelowych: *Drosophila melanogaster*, *Mus musculus* i *Saccharomyces cerevisiae*. Wkrótce wiele innych baz danych połączyło się z GO przekazując nie tylko ontologie, ale i narzędzia do analizy danych. Obecnie większość największych baz danych roślin, zwierząt i mikroorganizmów wspiera ten projekt. W 2008 roku GO zawierało już ponad 24 500 terminów odpowiadających rozmaitym organizmom żywym [1]. Obecnie GO jest standardowym narzędziem bioinformatycznym.

Współcześnie istniejące biologiczne bazy danych opisują szerokie spektrum cech i danych o konkretnych organizmach. Ze względu na to biologom coraz trudniej wyszukać informacje dotyczące danego organizmu, gdyż każda baza posiada własne ontologie. Celem projektu Gene Ontology jest przede wszystkim utrzymywanie i tworzenie słownictwa genów i atrybutów produktów genowych (normalizowanie reprezentacji genów i ich atrybutów u organizmów żywych, kontrolowanie słownictwa i wspólnej ontologii do opisu procesów biochemicznych), anotacja genów i ich produktów oraz asymilacja i szerzenie danych anotacji, wprowadzenie narzędzi umożliwiających łatwiejszy dostęp do wszystkich aspektów danych dostarczanych przez projekt.

GO to próba rozwiązania problemu niespójnego nazewnictwa produktów genowych w różnych bazach danych. Projekty sekwencjonowania genomów i eksperymenty mikromacierzowe produkują generowane elektronicznie dane dostarczane do systemów komputerowych. Bio-ontologie zostały stworzone jako systemy, które umożliwiają porozumiewanie się komputerom i ludziom.

Ontologie dotyczą trzech głównych domen:

- komponentów komórkowych,
- funkcji molekularnych – elementarnie aktywnych produktów genowych na poziomie molekularnym (np. wiązanie lub kataliza),
- procesów biologicznych – operacje zdarzeń molekularnych z zdefiniowanym początkiem i końcem, odnoszących się do funkcjonowania żyjących jednostek: komórek, tkanek, organów i całych organizmów.

2.1.1. Pliki anotacji GO

W programie wykorzystywane są symbole genów znajdujące się w pliku anotacji GO dla człowieka i myszy. Struktura plików tego typu nie jest skomplikowana, jednakże plik zawiera bardzo dużo, często zbędnych informacji (np. zbiory genów, które niedostępne są w bazie Biomart – geny, o których nie ma informacji w bazach danych przeszukiwanych przez Biomart).

Projekt Gene Ontology zawiera pliki anotacji nie tylko dla człowieka (*Homo sapiens*), ale również dla innych organizmów. Każdy taki plik zawiera ogólne informacje na temat plików anotacji oraz 17 kolumn. W każdej z kolumn znajdują się dane dotyczące konkretnego genu (Tabela1):

Nr kolumny	Oznaczenie	Przykład
1	Baza danych, z której został pobrany wpis	UniProtKB
2	ID (unikalny identyfikator) obiektu w bazie danych	P12345
3	Symbol obiektu w bazie danych	PH03
4	Kwalifikator (opcjonalnie)	NOT
5	ID (identyfikator) GO	GO:0003993
6	Odsyłacz	PMID:2676709
7	Kod ewidencyjny	IMP
8	Z (opcjonalnie)	GO:0000346
9	Jeden z trzech rodzajów ontologii	F
10	Nazwa obiektu (opcjonalnie)	Toll-like receptor 4
11	Synonim obiektu (białka)	hToll
12	Typ obiektu	Protein
13	Takson	Talon:9606
14	Data ostatniej anotacji	20090118
15	Atrybut opisujący źródło anotacji	SGD
16	Rozszerzenie anotacji (opcjonalnie)	(CL:00005776)
17	ID produktu genu	UniProtKB:P12345-2

Tabela1. Oznaczenia kolumn w plikach anotacji GO [1].

2.2. BioMart

BioMart (strona internetowa: www.biomart.org) to zorientowany na zapytania system zarządzania danymi stworzony przez Instytut Badań nad Rakiem w Ontario (OICR) i Europejski Instytut Bioinformatyki. System może być użyty do każdego typu danych i jest narzędziem dostępnym bez ograniczeń (licencja pod LGPL) [2].

BioMart umożliwia przeszukiwanie kilkudziesięciu różnych biologicznych i medycznych baz danych. Jest prostym narzędziem do uzyskania informacji interesujących dla użytkownika, które znajdują się w kilku bazach. Bez przeszukiwania kilku baz z osobna, BioMart umożliwia odnalezienie informacji o danych genach i ich produktach korzystając tylko z jednego narzędzia.

2.2.1. MartView

Zapytania do bazy można tworzyć na kilka sposobów. Jednym z nich jest korzystanie z wyszukiwarki online – MartView (www.biomart.org/martview). Aby skorzystać z wyszukiwarki należy wybrać bazę danych („Choose database”). W niniejszym projekcie posłużono się bazą danych „Ensemble Genes 62 (Sanger UK)”. Następnie należy wybrać interesujący użytkownika organizm. Po tym wyborze pojawiają się opcje przeprowadzenia filtracji („Filters”), tzn. według jakiego kryterium informacje o genach mają być wyszukiwane. Po wyborze filtru „Gene” (wyszukiwanie po genie) w przedstawianym projekcie należy wybrać „ID list limit” i „HGNC symbol(s)” (wyszukiwanie po symbolu HGNC danego genu – dla człowieka). W atrybutach („Attributes”) wybierane są rodzaje danych, które użytkownik chce uzyskać. Po dokonanych wyborach i wciśnięciu przycisku „Results” otrzymuje się tabelę wyników.

2.2.2. Tworzenie zapytań poza MartView

Zapytania można tworzyć również innymi sposobami bez korzystania z wyszukiwarki na stronie BioMart. Podczas tworzenia zapytań w MartView można uzyskać alternatywne struktury zapytania używając do tego celu przycisków „URL”, „XML” oraz „Perl”. W każdym z tych przypadków podawana jest alternatywna ścieżka dostępu do rezultatów. W przypadku adresu URL, z którego skorzystano podczas tworzenia programu, struktura zapytania wygląda następująco:

```
http://www.biomart.org/biomart/martview?VIRTUALSCHEMANAME=default& ← źródło  
ATTRIBUTES= & ← wyszukiwane atrybuty  
FILTERS=& ← zastosowany filtr
```

2.3. Wyszukiwane atrybuty

W programie wyszukiwane są następujące atrybuty:

- symbole i identyfikatory genów (HGNC dla człowieka),
- identyfikatory Ensembl,
- identyfikatory UniProt/SwissProt,
- symbole Affymetrix dla mikromacierzy HG U-133 PLUS2 (dla człowieka).

HGNC to skrót od HUGO Gene Nomenclature Committee – jednego z najbardziej aktywnych komitetów działających pod organizacją The Human Genome Organization (HUGO), zaangażowanej w projekt poznania ludzkiego genomu (Human Genome Project). HGNC zawiera unikalne symbole i nazwy ponad 31000 ludzkich genów, z których około 19000 to geny kodujące białka [3][4].

Projekt Ensembl (strona internetowa: www.ensembl.org) to projekt genomowych baz danych organizmów eukariotycznych. Baza Ensembl została stworzona przez EMBL i EBI w połączeniu

z Wellcome Trust Sanger Institute. Celem powstania bazy było stworzenie systemu software, który pozwoliłby na automatyczną anotację na wyselekcjonowanych genomach organizmów eukariotycznych. Projekt Ensemble rozpoczął się w 1999 roku, kilka lat przed ukończeniem projektu poznania ludzkiego genomu [5].

Swiss-Prot (strona internetowa: www.expasy.org/sprot) to baza danych zawierająca sekwencje białkowe, która została stworzona w 1986 roku przez Amosa Bairoch'a i ciągle jest rozwijana przy udziale Szwajcarskiego Instytutu Bioinformatyki (SIB) i EBI [6]. Baza zawiera wiele informacji dotyczących funkcji białek, ich struktury przestrzennej i modyfikacji posttranslacyjnych. UniProt jest oparte o sekwencje białkowe. Baza danych wiedzy UniProt (UniProtKB – strona internetowa: www.uniprot.org) jest centrum ogromnych kolekcji informacji na temat białek [7].

Affymetrix to firma zajmująca się produkcją mikromacierzy DNA. Została założona w 1991 roku i od tamtej pory produkuje mikro-chipy służące do analizy ekspresji genów [8]. Geny analizowane za pomocą mikromacierzy pochodzących z firmy Affymetrix mają specjalne oznaczenia zależne od mikromacierzy, na której były analizowane i organizmu, z którego pochodzą.

2.4. Język programowania Java

Java to szybko rozwijający się obiektowy język programowania używany obecnie w prawie każdej dziedzinie życia i w każdej ważniejszej gałęzi przemysłu. To uniwersalny i bezpieczny język, z którego korzysta ponad 7 milionów programistów świata. Uniwersalność, przenośność technologii Java oraz ich wydajność i bezpieczeństwo stosowania pozwala programistom m.in. na:

- tworzenie programów, które można uruchomić z poziomu przeglądarki internetowej i usług sieciowych,
- uruchamianie programów na dowolnej platformie,
- pisanie efektywnych aplikacji m.in. do zdalnych procesorów, telefonów komórkowych i innych urządzeń opartych na układach cyfrowych,
- opracowywanie aplikacji obsługujących fora i sklepy internetowe, formularze HTML itp. [9].

Java rozwija się coraz prężniej dostosowując się do ówczesnych wymagań. Wraz z rozwojem bioinformatyki coraz szybciej zaczęły rozwijać się języki programowania, które w założeniu miały ułatwić pracę biologom i lekarzom. Stworzone zostały liczne biblioteki umożliwiające łatwiejszą analizę danych biologicznych znajdujących się w bazach danych, czy otrzymywanych w wyniku eksperymentów laboratoryjnych. Również Java podążyła w tym kierunku tworząc projekt BioJava – zestaw bibliotek do analizy danych biologicznych i medycznych. Biblioteki BioJavy ciągle są rozbudowywane. Nie ma obecnie bibliotek związanych z przeszukiwaniem baz typu BioMart, dlatego też w projekcie posłużono się typowym kodem Javy.

3.4.1. Środowisko Eclipse i projekt Launch4j

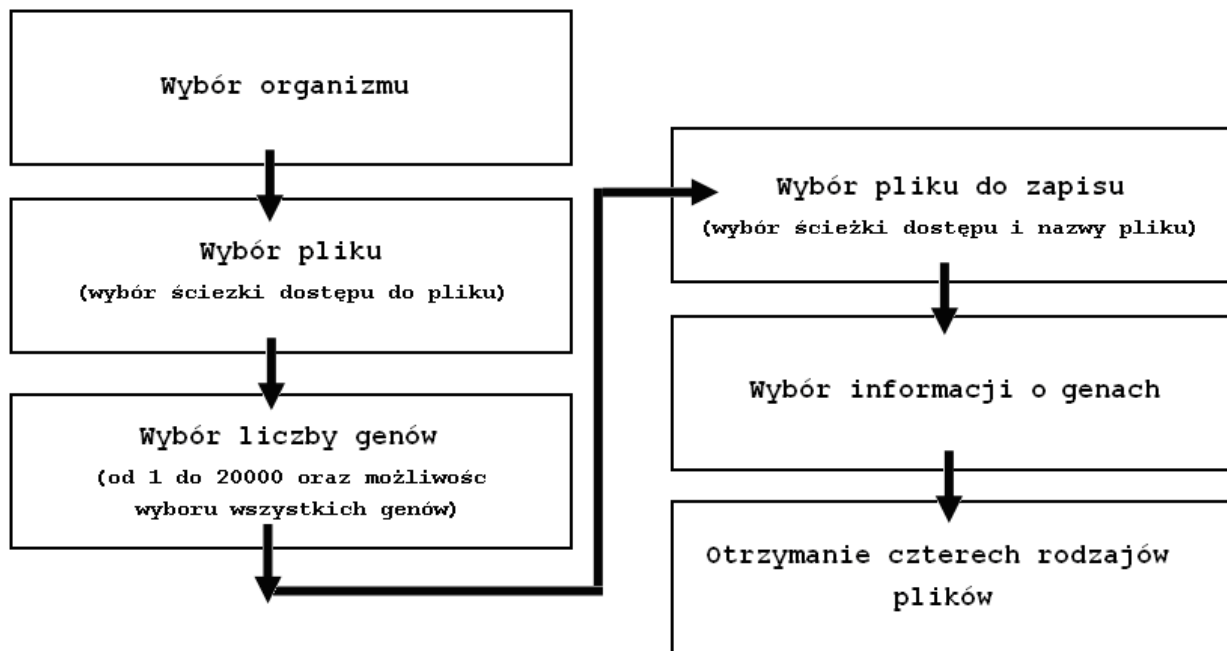
Projekt GeneBrowser został wykonany w środowisku Eclipse – jest to platforma napisana w Javie i dostępna dla wszystkich platform, które posiadają własną implementację maszyny wirtualnej Javy. Eclipse nie dostarcza żadnych narzędzi niezbędnych do tworzenia kodu i aplikacji. Jednak jej zaletą jest obsługa wtyczek (plugins), które rozszerzają jej funkcjonalność i umożliwiają np. tworzenie GUI, współpracę z serwerami aplikacji czy serwerami baz danych. Eclipse umożliwia zapis programu w rozszerzeniu .jar. Plik o takim rozszerzeniu może zostać opakowany w plik wykonywalny z rozszerzeniem .exe. Umożliwia to zastosowanie takich projektów jak Launch4j (z tego programu skorzystano podczas tworzenia programu GeneBrowser).

3. Specyfikacja

3.1. Specyfikacja zewnętrzna

Celem stworzonego programu jest pobranie informacji o unikalnych identyfikatorach HGNC, Ensemble, UniProt/SwissProt i symboli Affymetrix dla konkretnej mikromacierzy DNA. Program nie ma wbudowanego GUI, jednak można go uruchomić korzystając z pliku wykonywalnego .exe, który uruchamia konsolę i okna dialogowe.

Użytkownik programu wybiera nazwę organizmu, o którym chce uzyskać informacje (w tym przypadku „Homo sapiens (Human)”). Następnie użytkownik dostaje możliwość wczytania symboli genów bezpośrednio z pliku, który wybiera, korzystając z okna wyboru pliku. Takie okno umożliwia również wybór ścieżki dostępu do danych. Użytkownik ma również możliwość wyboru liczby genów, o których informacje chce uzyskać. I tak można wybrać od 1 do 20000 genów (wówczas zostanie wybranych pierwsze n symboli genów z pliku, gdzie n oznacza liczbę całkowitą z przedziału od 1 do 20000 wpisaną przez użytkownika). Jeżeli użytkownik wpisze „0” wówczas z pliku pobierane są wszystkie symbole genów. Po wybraniu liczby genów pojawia się okno zapisu plików z otrzymanymi wynikami. Wszystkie pliki, niezależnie od tego ile informacji o genach użytkownik będzie chciał uzyskać, zapisywane są pod jedną nazwą wpisaną przez użytkownika. Każdy z tych plików ma jednak w ostateczności inną nazwę zależną od tego, jakie informacje się w nim znajdują (nazwa pliku zawierającego informacje o ENSEMBL ID ma strukturę: wpisana_nazwa_EnsemblID.txt; nazwa pliku dla HGNC ID: wpisana_nazwa_GeneId.txt; nazwa pliku dla UniProt/SwissProt ID: wpisana_nazwa_UniProtSwissProtId.txt, nazwa pliku dla symbolu Affymetrix dla mikromacierzy HG U-133 PLUS2: wpisana_nazwa_AffyHgU133Plus2.txt). Użytkownik po wyborze nazwy pliku ma możliwość wyboru liczby wyników: czy chce otrzymać tylko wyniki Ensembl ID, HGNC ID, UniProt/SwissProt ID, czy symbolu Affymetrix, czy też chce uzyskać wszystkie podane powyżej informacje o genach.



Rys1. Schemat działania programu widoczny dla użytkownika

Program podbierając z pliku symbole genów wybiera tylko unikatowe symbole. Oznacza to, że jeżeli w pliku anotacji GO pojawiają się kilkakrotnie te same symbole genów, program wybiera taki symbol tylko raz, pozostałe ignorując.

Podobnie program zachowuje się przy zapisie danych do plików: zapisuje tylko unikatowe wyniki, tzn. takie, których treść pierwszej kolumny (zawiera ona wyszukiwane w bazie BioMart informacje) nie powtarza się więcej niż raz w tym pliku. Ignorowane przy zapisie do pliku są również puste komórki.

3.2. Specyfikacja wewnętrzna i szczegółowy opis działania programu

Program został napisany w języku Java. GeneBrowser posiada jedną główną klasę o nazwie *Browser*. W pliku *Browser.java* znajduje się cały skrypt omawianego programu wraz z szerokim objaśnieniem poszczególnych części skryptu.

Program w celu poprawnego wykonania korzysta z czterech pakietów: *java.io* (klasy strumieni wejścia-wyjścia), *java.util* (klasy użytkowe organizujące obsługę struktur danych), *java.net* (zawiera klasy niezbędne do tworzenia oprogramowania wykorzystującego połączenia z Internetem) oraz *java.swing* (służy do tworzenia graficznego interfejsu użytkownika).

W klasie *Browser* znajduje się skrypt programu. Klasa zawiera kilka metod stworzonych przez wykonawcę projektu:

- *Browser()*,
- *reduceUselessShit()*,
- *displayBioMartResults()*,
- *main()*.
- *compareRowArrays()*,

3.2.1. Skrótowy opis metod

Funkcja main()

Funkcja main() rozpoczyna działanie programu. Tutaj znajdują się wszystkie okna dialogowe kierujące zapytania do użytkownika. Metoda umożliwia użytkownikowi wybór poszczególnych cech, jakimi program powinien się kierować podczas pobierania informacji na temat genów z pliku anotacji GO.

Podstawowe funkcje wykorzystywane w tej części programu to:

- add – dodaje element do istniejącej zmiennej (np. tablicy), jeżeli taki element jeszcze tam nie istnieje,
- charAt() – zwraca określony znak, na który wskazuje dany indeks,
- close() – zamyka strumień i uwalnia inne źródła związane z nim,
- equals() – dokonuje porównania dwóch obiektów typu łańcuch znaków,
- FileReader() – jedna z podstawowych klas, służąca do odczytywania informacji zawartych w pliku,
- FileWriter() – zapisuje dane do pliku,
- parseInt() – funkcja parsuje łańcuch znaków na dane typu integer (liczb całkowitych),
- readLine() – odczytuje linie tekstu,
- size() – funkcja zwracająca rozmiar danej zmiennej,
- split() – rozdziela łańcuch znaków używając określonego separatora,
- trim() – funkcja pozwalająca na usunięcie białych znaków z końca i początku pobranych łańcuchów znaków.

Pozostałe funkcje

Metoda Browser() odczytuje dane z pliku przechodząc po ich kolejnych liniach. Z łańcucha znaków wyciągany jest interesujący programistę fragment tekstu (funkcja substring()). Funkcja indexOf() zwraca konkretne indeksy, bez łańcuchów znaków.

displayBioMartResults() korzysta głównie z funkcji związanych z podłączaniem się do sieci i do bazy BioMart: URLConnection() – za jej pomocą program łączy się zdalnie z siecią, OpenConnection() – otwieranie połączenia z Internetem.

Funkcje compareRowArrays() i reduceUselessShit() są używane w programie w celu pozbycia się wyników powtarzających się. Nie używa się tutaj skomplikowanie działających funkcji, jak w poprzednich przypadkach.

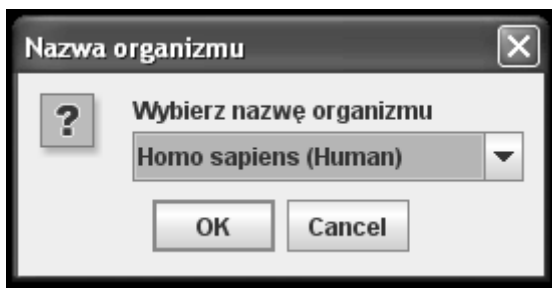
3.2.2. Podział programu na części ze względu na wykonywane czynności

Część I. Wybór nazwy organizmu

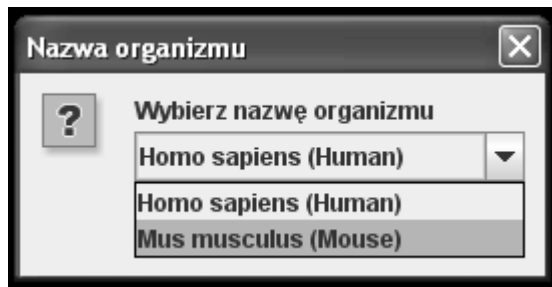
Projekt umożliwia wprowadzenie przez użytkownika nazwy organizmu, o którym informacje chce uzyskać. Pierwotnym założeniem projektu było wykonanie programu tylko dla genów ludzkich, jednak zauważono, że program w takiej formie, w jakiej został napisany, może być również wykorzystany do analizy genów myszy (*Mus musculus*). Stąd też w opcji wyboru organizmu pojawiają się dwie nazwy.

W celu wyboru organizmu skorzystano z klasy `JOptionPane` należącej do biblioteki `swing` obsługującej GUI w Javie. Wynik wywołania programu przedstawia się następująco:

1. Dwukrotne kliknięcie na ikonie programu GeneBrowser powoduje otwarcie się konsoli (wiersz poleceń), a wraz z nią pierwszego okna dialogowego:



Rys.2. Okno dialogowe służące do wyboru nazwy organizmu



Rys.3. Dwie opcje wyboru

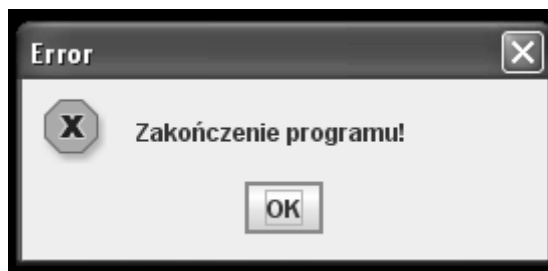
Otwarcie okna dialogowego powyższego typu wykonuje się za pomocą kodu:

```
Object selection = JOptionPane.showInputDialog(null, "Wybierz nazwę organizmu",  
"Nazwa organizmu", JOptionPane.QUESTION_MESSAGE, null, null, null);
```

(Cały kod, razem z powyższym, został dokładnie omówiony w skrypcie, dlatego nie będzie przedstawiany w raporcie.)

2. Użytkownik wybiera nazwę organizmu i zatwierdza ją przyciskiem „OK.”. Jeżeli użytkownik wybierze przycisk „Cancel” pojawia się błąd (Rys.4.) i następuje zamknięcie programu. Taki efekt uzyskuje się stosując skrypt:

```
if(selection == null) {  
    JOptionPane.showMessageDialog(null, "Zakończenie programu!", "Error",  
JOptionPane.ERROR_MESSAGE);  
    System.exit(0);  
}
```



Rys.4. Błąd podczas wywołania programu skutkuje pojawieniem się odpowiedniej wiadomości i zamknięciem systemu.

Wynik zapisywany jest do zmiennej `latinName` (łacińska nazwa organizmu).

Część II. Wybór pliku anotacji GO, tworzenie plików z wynikami

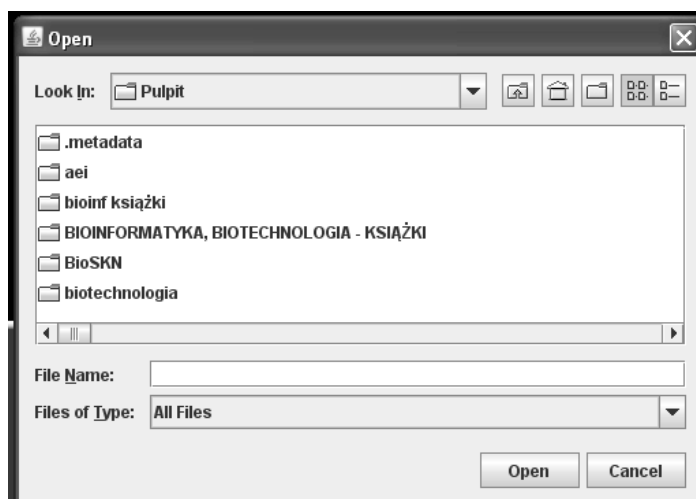
Użytkownik ma możliwość wyboru pliku, w którym znajdują się informacje o genach interesującego go organizmu. Za pomocą skryptu:

```
JFileChooser wybor = new JFileChooser();
wybor.setCurrentDirectory(new File("."));
int wynik = wybor.showOpenDialog(null);
```

otwierane jest okno pozwalające na wybór ścieżki dostępu do pliku (Rys.5). Struktura plików wczytywanych przez program powinna wyglądać następująco:

- 1 kolumna – ciąg znaków rozpoczynających się od znaku 'U',
- 2 kolumna – może być pusta,
- 3 kolumna – symbole genów.

Użytkownik ma więc możliwość stworzenia własnego pliku zawierającego symbole genów danego organizmu (obecnie program obsługuje bazę BioMart tylko dla człowieka i myszy). Użytkownik może również skorzystać bezpośrednio z pliku pobranego ze strony Gene Ontology (plik anotacji GO), który spełnia powyższe kryteria dotyczące struktury pliku.

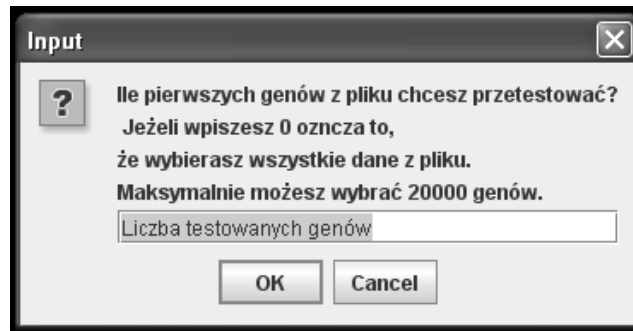


Rys.5. Okno wyboru pliku z symbolami genów.

Użytkownik wybiera również liczbę genów, o których chce uzyskać informacje z bazy.

Wywoływane jest więc okno dialogowe (Rys.6.):

```
how = JOptionPane.showInputDialog(null, "Ile pierwszych genów z pliku chcesz  
przetestować?\n Jeżeli wpiszesz 0 oznacza to, \n" + "że wybierasz wszystkie dane z  
pliku.\nMaksymalnie możesz wybrać 20000 genów.", "Liczba testowanych genów");
```



Rys.6. Wybór liczby genów.

Użytkownik ma możliwość wyboru wszystkich genów, które znajdują się w danym pliku, wybranym przez niego we wcześniejszym etapie. Po wyborze liczby genów i zatwierdzeniu przyciskiem „OK.” pojawia się kolejne okno dialogowe, które pozwala na wybór nazwy pliku i ścieżki dostępu do pliku, w którym mają zostać zapisane informacje pobrane z bazy BioMart.

Ostatnim z etapów jest wybór przez użytkownika informacji, które chce uzyskać z bazy. Użytkownik ma do wyboru uzyskanie wyników dotyczących:

- Ensemble ID,
- ID genu (w przypadku człowieka HGNC, w przypadku myszy MGI),
- UniProt/SwissProt ID,
- symbol Affymetrix,
- wszystkich wymienionych.

Program zapis do pliku wykonuje korzystając z utworzonego wektora końcówek nazw plików:

```
String[] oNames = new String[] { "_EnsemblID", "_GeneID", "_UniProtSwissProtID",  
"_SymbolAffymetrix" };
```

Użytkownik może wybrać, które dane chce zapisać, dzięki zastosowaniu dwóch fragmentów skryptu:

- tworzącego okno dialogowe umożliwiające wybór

```
how = JOptionPane.showInputDialog(null, "Które z otrzymanych danych w wyniku  
wyszukiwania\n"
```

```
+ "zapisać do pliku?\n"  
+ "0 - Wszystko\n"  
+ "1 - ENSEMBL ID\n"  
+ "2 - ID GENU\n"  
+ "3 - UniProt/SwissProt ID\n"  
+ "4 - Symbol Affymetrix", "0");
```

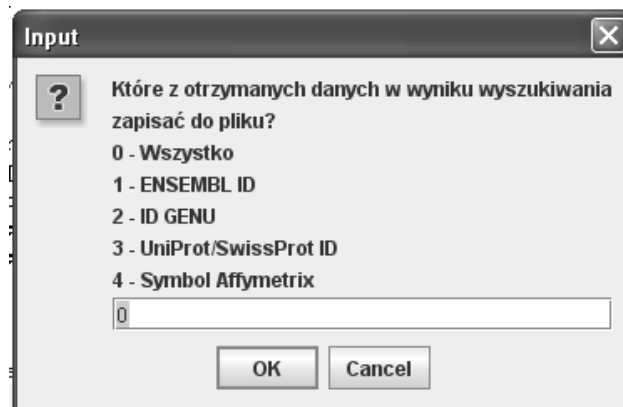
- pozwalające na odpowiedni zapis do pliku (fragment)

```
switch(hgncEnsemblSwissAffyChoice)
{
    case 0:
    {
        hgncEnsemblSwissAffyChoices[0] = true;
        hgncEnsemblSwissAffyChoices[1] = true;
        hgncEnsemblSwissAffyChoices[2] = true;
        hgncEnsemblSwissAffyChoices[3] = true;

        if (fileName != null)
        {
            for (int j = 0; j < 4; j++)
            {
                files[j] = new File(fileName + oNames[j] +
".txt");
                outputs[j] = new BufferedWriter(new
FileWriter(files[j]));
            }
            w = 0;
            break;
        }

        case 1: ...
    }
}
```

Poniżej przedstawione jest okno dialogowe umożliwiające powyższe wybory:



Rys.7. Wybór informacji, które użytkownik chce uzyskać.

Część III. Odczyt danych z pliku anotacji GO

Program odczytujący plik anotacji GO musi wziąć pod uwagę specyficzną strukturę pliku. Pierwszą rzeczą, o jakiej należało pamiętać pisząc skrypt, było wyciągnięcie informacji z pliku tylko z tych wierszy, które zawierały symbole genów pochodzące z bazy UniProtKB. Wobec powyższego stworzono skrypt, który pobiera tylko te symbole, które znajdują się w liniach rozpoczynających się od litery 'U'. Następnie należało pamiętać, iż w pliku anotacji GO symbole genów niejednokrotnie powtarzają się. Takie powtórzenia nie zawsze występują po sobie.

Stworzono kod pozwalający sprawdzić, czy gen będący w danej linijce jest taki sam jak któryś z poprzednich genów:

```
while ((one = buff.readLine()) != null) {
    if (one.charAt(0) != 'U') continue;
    String[] parts = one.trim().split("\t");
    if (!parts[2].equals(before))
    {
        inFile.add(parts[2]);
        before = parts[2];
    }
}
```

Fragment skryptu, przedstawionego powyżej, rozpoczyna się od sprawdzenia, czy linijka pliku, która jest aktualnie analizowana przez program, jest pusta. Jeżeli nie, wówczas sprawdzana jest obecność znaku 'U' na początku każdego z wierszy. Kolejne linijki dotyczą sprawdzenia, czy gdzieś w pliku znajduje się już gen o takim samym symbolu. Unikalne wyniki zapisywane są do tabeli.

Część IV. Przeszukiwanie bazy i tworzenie zapytań

Program łączy się z siecią za pomocą kilku linijek kodu:

```
URL bioMart = new URL(cd);
URLConnection connection = bioMart.openConnection();
BufferedReader in = new BufferedReader(new InputStreamReader
(connection.getInputStream()));
```

Tworzony jest obiekt klasy URL i następnie nawiązywane połączenie za pomocą funkcji `openConnection()`.

Zapytanie do bazy tworzone jest na podstawie danych uzyskanych podczas analizy wyników otrzymanych z bazy BioMart za pomocą narzędzie MartView. Zapytanie składa się z czterech części:

- adresu internetowego, z którego należy pobrać dane

```
String cd = http://www.biomart.org/biomart/martview?VIRTUALSCHEMANAME=default&
```

- wybranych atrybutów (zapytań)

```
+ "ATTRIBUTES=" + latinName + "_gene_ensembl.default.feature_page.ensembl_gene_id|"
//Ensembl ID
+ latinName +
"_gene_ensembl.default.feature_page."+sym+"_id|" //HGNC ID
+ latinName +
"_gene_ensembl.default.feature_page."+sym+"_symbol|" // symbol genu
+ latinName +
"_gene_ensembl.default.feature_page.uniprot_swissprot|" // UniProt/SwissProt ID
```

```
        + latinName + "_gene_ensembl.default.feature_page."+affy+"&"  
// symbol Affymetrix (tylko dla hsapiens)
```

gdzie `latinName` oznacza nazwę organizmu zapisaną w postaci np. `sapiens`, `sym` to symbol wykorzystywany dla danej bazy do wyszukiwania informacji o danym organizmie, `affy` to nazwa mikromacierzy DNA.

- odpowiedniego filtra

```
+ "FILTERS=" + latinName + "_gene_ensembl.default.filters."+sym+"_symbol.\"\" //  
filtracja po symbolu HGNC
```

- oraz nazwy genu

```
+ geneName;
```

Otrzymane w wyniku takich działań wyniki są przedstawione w postaci kodu HTML. Aby pobrać z kodu tylko interesujące użytkownika informacje program korzysta z kodu, który analizuje każdą linię przedstawionego w formacie HTML wyniku. Ostatecznie wyniki te zapisywane są do tabeli, która ulega dalszym modyfikacjom w kolejnej części programu.

Część V. Wybieranie unikatowych rezultatów i zapis wyników do plików

Baza BioMart daje w wyniku informacje, które powtarzają się dla tych samych genów. Aby nie dublować wyników stworzono skrypt pozwalający na odrzucenie powtarzających się wyników i zapis do pliku tylko unikatowych oraz takich, które nie posiadają tzw. pustych wyników (np. baza BioMart dla zapytania o UniProt/SwissProtID zwraca dla genu A1BG dwa wyniki:

A1BG_HUMAN	A1BG
	A1BG

Jak widać na powyższym przykładzie jeden z wyników jest wynikiem „pustym”, czyli tak na prawdę nie zawiera żadnego wyniku.

W raporcie nie będę przedstawiać skryptu realizującego powyższe zagadnienia ze względu na ich obszerność i dokładne opisanie w skrypcie programu. Najczęściej wykorzystywaną funkcją jest tutaj omówiona już wcześniej funkcja `equal()`.

3.3. Wyniki programu

3.3.1. Podstawowe GUI (okna dialogowe) i tworzenie pliku wykonywalnego

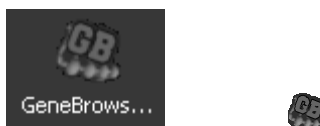
W celu lepszej pracy z programem stworzono podstawowe GUI zawierające okna dialogowe, które umożliwiają łatwą i szybką komunikację użytkownika z programem. Większość stworzonych okien w programie została opisana w poprzednich częściach rozdziału.

Stworzony projekt zapisano w formacie .jar, który z kolei przekonwertowano na plik wykonywalny .exe za pomocą programu Launch4j.

Poniżej przedstawiony jest fragment pliku .xml zawierającego informacje o stworzonym programie:

<code><minVersion>1.4.0</minVersion></code>	<code><fileDescription>Java application for finding information</code>
<code><jdkPreference>preferJre</jdkPreference></code>	<code>about genes</fileDescription></code>
<code><maxHeapPercent>50</maxHeapPercent></code>	<code><copyright>Copyright(C) 2011 Katarzyna Jonak</copyright></code>
<code><fileVersion>1.0.0.0</fileVersion></code>	<code><originalFilename>GeneBrowser.exe</originalFilename></code>
<code><txtFileVersion>1.0.0</txtFileVersion></code>	

Poniżej przedstawiono ikonę programu wykonywalnego GeneBrowse.exe:

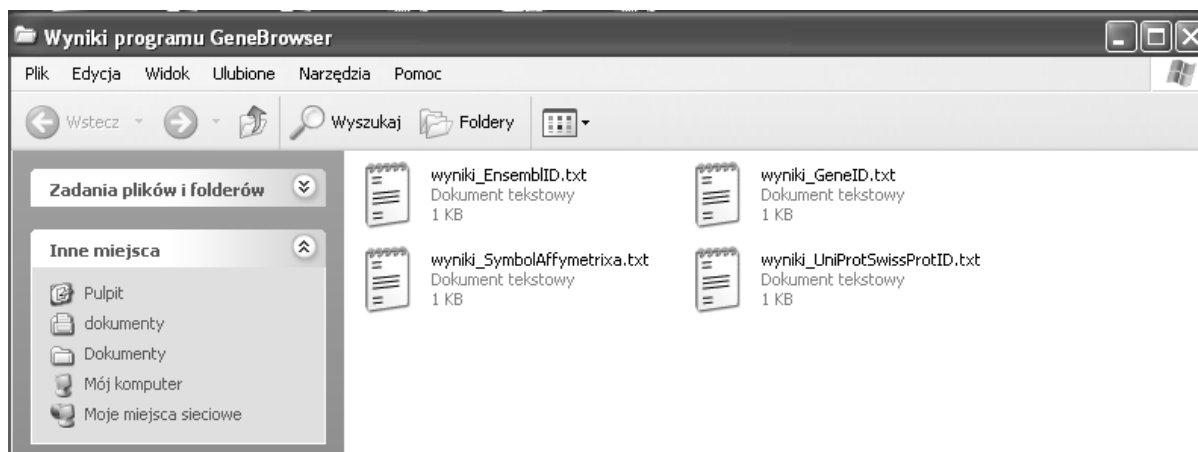


Rys.8. Ikona programu GeneBrowser.exe.

Pierwszy z obrazów przedstawia ikonę programu znajdującą się na pulpicie komputera. Drugi rysunek przedstawia obraz ikony w oryginalnych rozmiarach.

3.3.2. Otrzymane wyniki

Wynikiem działania programu są pliki (ich liczba zależy od tego, jaką opcję użytkownik wybrał podczas wyboru informacji o genach). Program daje możliwość otrzymania czterech rodzajów plików w zależności od informacji, jakie te pliki zawierają (Rys.9).



Rys.9. Wynik działania programu: cztery rodzaje plików.

Struktura każdego z plików wygląda następująco (dla organizmu *Homo Sapiens* i testowania 20 genów):

wyniki_EnsemblID.txt - WordPad		wyniki_GeneID.txt - WordPad	
Plik Edycja Widok Wstaw Format Pomoc		Plik Edycja Widok Wstaw Format Pom	
ENSG00000121410 A1BG		5 A1BG	
ENSG00000148584 A1CF		24086 A1CF	
ENSG00000134864 A2LD1		25100 A2LD1	
ENSG00000175899 A2M		7 A2M	
ENSG00000166535 A2ML1		23336 A2ML1	
ENSG00000128274 A4GALT		18149 A4GALT	
ENSG00000118017 A4GNT		17968 A4GNT	

Rys.10 i 11. Wyniki dla 20 genów: Ensembl ID (Rys.10) i Gene ID (Rys. 11).

wyniki_UniProtSwissProtID.txt - WordPad		wyniki_SymbolAffymetrix.txt - WordPad	
Plik Edycja Widok Wstaw Format Pomoc		Plik Edycja Widok Wstaw Format Pomoc	
A1BG_HUMAN A1BG		229819_at A1BG	
A1CF_HUMAN A1CF		220951_s_at A1CF	
A2LD1_HUMAN A2LD1		232422_at A2LD1	
A2MG_HUMAN A2M		217757_at A2M	
A2ML1_HUMAN A2ML1		1564307_a_at A2ML1	
A4GAT_HUMAN A4GALT		1553505_at A2ML1	
A4GCT_HUMAN A4GNT		219488_at A4GALT	
		221131_at A4GNT	

Rys.12 i 13. Wyniki dla 20 genów: UniProt/SwissProt ID(Rys.12) i symbole Affymetrix (Rys. 13).

Pierwsza kolumna każdego pliku tekstowego zawiera wyniki pobrane z bazy BioMart. Druga linijka to symbole genów użyte do przeszukiwania bazy.

4. Omówienie napotkanych problemów

Podczas tworzenia programu pojawiły się problemy głównie wynikające z działania bazy BioMart. Program, choć z założenia stworzony tylko do obsługi plików anotacji GO dla człowieka, miał w ostateczności funkcjonować poprawnie również dla plików anotacji innych organizmów. Jednakże GeneBrowser w obecnym stanie pełni swoją funkcję tylko dla dwóch organizmów: *Homo sapiens* i *Mus musculus*. Jest to wynikiem tego, iż projekt BioMart nie uwzględnia baz danych symboli genów dla innych organizmów, aniżeli przykładowo dla człowieka (baza HGNC), czy myszy (baza MGI). Wobec powyższego fakt program nie działa dla innych organizmów niż w/w.

Innym problemem w przypadku zastosowania programu dla większej liczby organizmów było niejednolite zwracanie wyników wyszukiwania. Struktura wyników prezentowana w HTML, choć podobna dla różnych plików anotacji GO, nie była identyczna, co powodowało liczne problemy z dotarciem do szukanych wyników.

Problemem pod względem programistycznym było rozdzielenie pobranych informacji o genach do osobnych plików. Poradzono sobie z tym korzystając ze struktury *switch* i *case*. Trudności wystąpiły również podczas wyboru unikatowych symboli genów z pliku anotacji GO oraz podczas zapisu niepowtarzalnych wyników i ignorowaniu wyników zawierających puste pola.

5. Dodatkowe zastosowanie programu

Program można rozbudować m.in. o wyszukiwanie większej liczby informacji, czy też korzystanie z innych plików i baz internetowych (nie tylko bazy BioMart). Głównie należy jednak skupić się na możliwości wyszukiwania informacji w bazie BioMart bez użycia symboli genów dla innych organizmów (ze względu na brak możliwości wyszukiwania po symbolach w bazie BioMart) oraz na ujednoliceniu otrzymywanych wyników w postaci kodu HTML.

6. Wnioski

Baza BioMart jest bardzo dobrym narzędziem do przeszukiwania internetowych biologicznych baz danych, jednakże zawiera również wiele mankamentów, które nie pozwalają m.in. na wyszukiwanie informacji dotyczących innych organizmów, niż przedstawione w raporcie (brak możliwości wyszukiwania po symbolu organizmu stosowanym w pliku anotacji GO).

Przeszkodą w szerszym wykorzystaniu programu jest również mała liczba plików anotacji GO dla organizmów. Baza Gene Ontology jednak ciągle się rozwija, wobec czego program również ma szansę na rozwój.

Język programowania Java bardzo dobrze nadaje się do realizacji zagadnienia poruszanego w raporcie. Jest to język coraz częściej stosowany na świecie również w celach bioinformatycznych, a biblioteki BioJavy ciągle są poszerzane. Być może w przyszłości zostaną również poszerzone o narzędzia ułatwiające pracę z takimi bazami biologicznymi jak BioMart, bądź zastosowane zostaną rozwiązania pozwalające na szybkie przeszukiwanie kilkudziesięciu baz danych w celu znalezienia informacji o genach (na podobnej zasadzie działania jak BioMart).

Stworzony program w prosty sposób umożliwia użytkownikowi pobieranie symboli genów z pliku anotacji GO i plików o podobnej strukturze, a także innych atrybutów korzystając z bazy internetowej BioMart. Jest to wygodne narzędzie pozwalające na uzyskanie wielu informacji na temat genów danych organizmów z różnych internetowych baz danych. GeneBrowser jest ułatwieniem w pracy z bazami danych. Podczas jego zastosowania nie wybiera się ręcznie filtrów i atrybutów, jak ma to miejsce w przypadku korzystania bezpośrednio z narzędzia MartView. Program umożliwia zapis do osobnych plików każdego z otrzymanych typów wyników, co powoduje, że użytkownik w określonym momencie swojej pracy może wykorzystać dane z tego pliku, które są mu aktualnie niezbędne.

Ze względu na wygodne i bardzo proste GUI w postaci okien dialogowych użytkownik „nie gubi się” w gąszczu informacji i dokładnie wie co po kolei ma wykonać, aby otrzymać interesujące go informacje.

7. Literatura

- [1] The Gene Ontology Consortium. Źródło [http: www.geneontology.org](http://www.geneontology.org)
- [2] BioMart Project. Źródło [http: www.biomart.org](http://www.biomart.org)
- [3] HUGO Gene Nomenclature Committee. Źródło [http: www.genenames.org](http://www.genenames.org)
- [4] HUGO Human Genome Organization. Źródło [http: www.hugo-international.org](http://www.hugo-international.org)
- [5] Ensemble Genome Browser. Źródło [http: www.ensembl.org](http://www.ensembl.org)
- [6] Swiss-Prot. Źródło [http: www.expasy.org/sprot](http://www.expasy.org/sprot)
- [7] UniProt. Źródło [http: www.uniprot.org](http://www.uniprot.org)
- [8] Affymetrix. Źródło [http: www.affymetrix.com](http://www.affymetrix.com)
- [9] Horstmann C., Cornell G. *Core Java 2. Podstawy*. Wydawnictwo Helion, Warszawa 2003.