Licencjacki projekt programistyczny 2011

Program do wykonywania obliczeń statystycznych związanych z grą go

Dokumentacja programisty

Wojciech Jedynak

Wrocław, 7 lipca 2011

Spis treści

1 Wprowadzenie **3**
1.1 Cel dokumentacji . . . 3

2 Organizacja projektu **3**
2.1 Ogólny opis . . . 3
2.2 Konfiguracja programu . . . 3
2.2.1 Opis formatu pliku CONFIG . . . 3
2.3 Baza danych . . . 3
2.3.1 Tabela go_stat_data . . . 4
2.4 Struktura modułów . . . 4
2.4.1 Data.SGF.Types i Data.SGF.Parsing . . . 4
2.4.2 Transformations . . . 4
2.4.3 SgfBatching . . . 5
2.4.4 Lang . . . 5
2.4.5 Configuration . . . 5
2.4.6 Pages . . . 5
2.4.7 DB . . . 5
2.4.8 Server . . . 5
2.4.9 Main . . . 5
2.5 Pozostałe pliki . . . 5

3 Kompilacja i testowanie **6**

4 Wykorzystane biblioteki i narzędzia pomocnicze **6**
4.1 Biblioteki, pakiety, moduły . . . 6
4.1.1 Śledzenie zależności (Cabal) . . . 6
4.1.2 Serwer HTTP (Happstack) . . . 6
4.1.3 Testowanie (HUnit, QuickCheck, test-framework) . . . 6
4.1.4 Zarządzanie bazą danych (HDBC, HDBC-postresql, HDBC-sqlite3) . . . 6
4.1.5 Analiza leksykalna (Parsec) . . . 6
4.1.6 Generowanie html (xhtml 3000) . . . 6
4.1.7 Pozostałe (filemanip, strict, mtl) . . . 6
4.2 Narzędzia . . . 6
4.2.1 Git i portal http://github.com . . . 6
4.2.2 Latex . . . 6

1 Wprowadzenie

1.1 Cel dokumentacji

Celem niniejszego dokumentu jest takie przedstawienie struktury projektu GoStat, aby umożliwić jego modyfikacje oraz utrzymywanie programistom, którzy znają Haskella, ale nie należeli do początkowego zespołu. Wykaz użytych bibliotek powinien był pomocny, jeśli instalacja oprogrogramowania nie powiedzie się i konieczna będzie kompilacja programu ze źródeł. Dodatkowo życzeniem autora jest nakreślenie wykonanej pracy tak, aby zainteresowane osoby były w stanie (w razie potrzeby) na wykorzystanie opisanych tu rozwiązań w swoich projektach.

2 Organizacja projektu

2.1 Ogólny opis

Program został napisany niemal w całości w Haskellu [8]. Po uruchomieniu pliku `GoStat` wewnętrzny serwer HTTP nasłuchuje na porcie 8000, a komunikacja z użytkownikiem odbywa się za pomocą interfejsu WWW. Lista katalogów, w których znajduje się kolekcja plików SGF, zapisywana jest do pliku konfiguracyjnego, wstępnie przetworzone (*znormalizowane*) gry są przechowywane w bazie danych. Dialog z użytkownikiem może odbywać się w języku polskim bądź angielskim.

2.2 Konfiguracja programu

W programie potrzebujemy przechowywać dwie informacje: jakiej bazy danych używa (chce używać) użytkownik i gdzie znajduje się jego kolekcja zapisów partii go, które chciałby analizować naszym programem. Do przechowywania ww. danych używamy bardzo prostego, autorskiego formatu. Funkcje związane z wczytywanien, analizą leksykalną oraz zapisywaniem znajdują się w module *Configuration* (w pliku `src/Configuration.hs`).

2.2.1 Opis formatu pliku CONFIG

Format pliku jest opisywany przez poniższą gramatykę w postaci EBNF:

```
config ::= declaration*
declaration ::= db | dirs | '–' *anything*
db = dbserver ':' dbversion
dbversion = sqlite3 ';' path ';' | postgresql ';'
dirs = gamedirs ':' path ';'
```

Przykładowy plik konfiguracyjny:

```
-- new config
--dbserver:postgresql;
dbserver:sqlite3;/home/wojtek/db/games.db;
gamedirs:/home/wojtek/data/;
```

2.3 Baza danych

Program używa bazy danych Sqlite3 [4]. Programista nie musi zajmować się ręczną administracją bazy danych: służy do tego moduł *DB* w pliku `src/DB.hs` .

Używana jest jedna tabela o nazwie `go_stat_data` .

2.3.1 Tabela go_stat_data

Opis pól tabeli go_stat_data

Pole	Typ	NULL dozwolone?	Opis
id	PRIMARY KEY	nie	unikatowy identyfikator gry
winner	CHAR	nie	zwycięzca gry ('b' lub 'w')
moves	VARCHAR(700)	nie	znormalizowany przebieg rozgrywki
path	VARCHAR(255)	nie	bezwzględna scieżka do gry
b_name	VARCHAR(30)	nie	pseudonim (nazwisko) czarnego
w_name	VARCHAR(30)	nie	pseudonim (nazwisko) białego
b_rank	VARCHAR(10)	tak	ranking czarnego
w_rank	VARCHAR(10)	tak	ranking białego

Jedyne pola, która nie są wymagane to pola b_rank i w_rank. Wynika to z tego, że na niektórych serwerach do gry w go nie jest wymagane podanie swojego orientacyjnego poziomu ani rankingu.

2.4 Struktura modułów

Lista modułów wchodzących w skład projektu:

2.4.1 Data.SGF.Types i Data.SGF.Parsing

```
type Move = (Int, Int)
type Moves = [Move]
type PlayerName = String
data Winner = Black | White
data Result = Unfinished | Draw | Win Winner PlayerName

parseSGF :: String -> Either String SGF
getResult :: SGF -> Result
isWithHandicap :: SGF -> Bool
getBlack, getWhite, getBlackRank, getWhiteRank, date :: SGF -> String
```

Moduły *Data.Sgf.Types* i *Data.SGF.Parsing* zawierają deklaracje typów służących do przechowywania informacji o danej partii (m.in. wynik, dane graczy, lista ruchów) oraz funkcje, które pozwalają dane te wyświetlać i analizować.

2.4.2 Transformations

Przekształcenia matematyczne i normalizacja ruchów.

```
normalizeMoves :: [Move] -> [Move]

isOnMainDiagonal :: Move -> Bool
isAboveMainDiagonal :: Move -> Bool
isBelowMainDiagonal :: Move -> Bool
isOnHorizontal :: Move -> Bool
isAboveHorizontal :: Move -> Bool
isBelowHorizontal :: Move -> Bool
```

```
horizontal :: Move -> Move
rotate90 :: Move -> Move
mainDiagonalMirror :: Move -> Move

transformIntoFirst :: Triangle -> (Move -> Move)
getTransformation :: Move -> (Move -> Move)

triangles :: [Triangle]
findTriangles :: Move -> [Triangle]
findTriangle :: Move -> Triangle
```

2.4.3 SgfBatching

Konwersja plików SGF do formatu, który będzie łatwo zapisać w bazie danych.

2.4.4 Lang

Komunikaty w języku polskim i angielskim.

2.4.5 Configuration

Obsługa pliku CONFIG.

2.4.6 Pages

Tworzenie szablonów stron WWW.
[5] [6]

2.4.7 DB

Zarządzanie bazą danych.

```
createDB :: GoStatM ()
deleteDB :: GoStatM ()
addFilesToDB :: GoStatM ()
queryCountDB :: GoStatM Int
queryStatsDB :: String -> GoStatM [(String, Int, Int, Int)]
queryCurrStatsDB :: String -> GoStatM (Int, Int, Int)
queryGamesListDB :: String -> Int -> GoStatM [(Int, FilePath, String, String, String,
String, String)]
queryFindGameById :: Int -> GoStatM (Maybe (String, FilePath))
```

2.4.8 Server

Serwer HTTP.

2.4.9 Main

Punkt startowy aplikacji.

2.5 Pozostałe pliki

Pliki css, javascript, obrazki

3 Kompilacja i testowanie

4 Wykorzystane biblioteki i narzędzia pomocnicze

4.1 Biblioteki, pakiety, moduły

4.1.1 Śledzenie zależności (Cabal)

[11]

4.1.2 Serwer HTTP (Happstack)

[12]

4.1.3 Testowanie (HUnit, QuickCheck, test-framework)

4.1.4 Zarządzanie bazą danych (HDBC, HDBC-postresql, HDBC-sqlite3)

4.1.5 Analiza leksykalna (Parsec)

4.1.6 Generowanie html (xhtml 3000)

[13]

4.1.7 Pozostałe (filemanip, strict, mtl)

4.2 Narzędzia

W niniejszej sekcji wymieniono i pokrótce opisano najważniejsze narzędzia, które pozwoliły ukończyć projekt.

4.2.1 Git i portal github

Git [1] to system do kontroli wersji autorstwa Linuxa Torwardsa (TODO: sprawdzić pisownię).

Github [2] to portal, który pozwala na przechowywanie kodu źródłowego (ogólnie: repozytoriów kodu zarządzanych przez git) i udostępnienie go innym programistom. Poprzez użycie tych zasobów rozwiązałem kwestię składowania projektu i mogłem swobodnie eksperymentować: nietrafione zmiany można było wycofać jednym poleceniem.

4.2.2 Latex

System LaTeXpozwolił na utworzenie dokumentacji w formacie pdf, który jest standardem w informatyce.

Literatura

[1] *Git – narzędzie do kontroli wersji*
`http://git-scm.com/`

[2] *Portal github.com*
`https://github.com/`

[3] *Repozytorium projektu GoStat w portalu github*
`https://github.com/wjzz`

[4] *SQLite3 – lekka baza danych*
`http://www.sqlite.org/`

[5] *jQuery – biblioteka dla JavaScriptu*
`http://jquery.com/`

[6] *jQuery IU – biblioteka komponentów dla JavaScriptu*
http://jqueryui.com/

[7] *Eidogo – interaktywna plansza*
http://eidogo.com/source

[8] *Haskell – strona główna*
http://www.haskell.org/

[9] *Glasgow Haskell Compiler*
http://www.haskell.org/ghc/

[10] *Hackage – kolekcja pakietów Haskellowych*
http://hackage.haskell.org/

[11] *Cabal – narzędzie do tworzenia i instalowania pakietów Haskellowych*
http://www.haskell.org/cabal/

[12] *Happstack – serwer HTTP dla Haskella*
http://happstack.com/index.html

[13] *Biblioteka xhtml dla Haskella*
http://hackage.haskell.org/package/xhtml-3000.2.0.1