Как да търсим в библиотеката от народни песни?

Кирил Киров

Магратеа ЕООД

23.11.2010 г.

Съдържание

1 Търсене и резултати

2 Език за търсене

3 Реализация

4 Интеграция на външни системи

Търсене

- Търсенето в библиотеката от народни песни става чрез използване на web browser и прилича на търсене в Google
- Заявката за търсене се пише на прост език, в който се използват български и английски думи, както и специални символи
- Търсенето се осъществява на базата на предварително създаден индекс

Резултати

- Резултатите от търсенето се визуализират в табличен вид
- Всеки ред от таблицата отговаря на отделна песен и съдържа връзки към нейните файлове
- Резултатите са определен брой песни, подмножество на индексираните

Термин

Търсенето по термин става с една единствена дума, например:

Търсене по термин

```
стоян
```

Фраза

Търсенето по фраза става с две или повече думи заградени с двойни кавички, например:

Търсене по фраза

```
"ожадня стоян за вода"
```

Поле

Търсенето по поле става като се специфицира полето преди термина или фразата, например:

Търсене по поле

```
code:ba_002_2_04
content:"ожадня стоян за вода"
```

Условие

Търсенето по условие става, като се опишат логическите връзки между търсените думи или фрази, например:

Търсене по условие

```
стоян AND радка
стоян AND радка AND NOT "пяла: радка"
```

Маска

Търсенето по маска става, като част от търсената дума или фраза се замести с * или ?, например:

Търсене по маска

`ст*ян`

Метаданни

Търсенето по метаданни става, като търсената дума или част от търсената фраза се заместят с LaTeXкоманда, например:

Търсене по метаданни

```
ст*ян AND area\{ямболско\}
```

Демонстрация

Демонстрация на търсенето

Използван софтуер

- Ruby - език за програмиране
- Ferret - търсеща машина
- Sinatra - платформа за създаване на web сайтове
- Thin - web сървър
- HAML - език за описание на web шаблони

Организация на SVN хранилището

- `bin` - изпълними файлове
- `doc` - документация на проекта
- `ferret` - изходни файлове, индекс на Ferret
- `Gemfile` - списък с използвани Ruby библиотеки
- `haml` - HAML сорс код
- `latex` - текстове на песни
- `lib` - Ruby сорс код
- `lilypond` - ноти
- `mp3` - изпълнения на песните дигитализирани от магнитен носител
- `out` - изходни файлове от компилация на LilyPond и LaTeX
- `public` - публични, статични файлове за web сървъра
- `Rakefile` - списък със задачи (генериране на документация и тестване)
- `README.rdoc` - описание на проекта
- `test` - Ruby сорс код за тестване на системата
- `vendor` - Ruby библиотеки

Инсталация

За да работи системата трябва да имаме инсталирани Ruby 1.8 и Rubygems. Изпълняваме следните команди:

Команди

```
$ svn co https://svn.magrathea.bg/folk
$ cd folk
$ gem bundle
$ bin/rake
```

Компилация и индексиране

Преди да използваме системата трябва да компилираме сорс файловете от LilyPond и LaTeX, а след това да индексираме съдържанието във Ferret:

Команди

```
$ ruby bin/lilypond
$ ruby bin/latex
$ ruby bin/index
```

Стартиране

За да използваме системата през Web е необходимо да стартираме web сървъра и да отворим с browser посочения адрес (например http://localhost:8080).

Команди

```
$ ruby bin/folk
>> Thin web server (v1.2.7 codename No Hup)
>> Maximum connections set to 1024
>> Listening on 0.0.0.0:8080, CTRL+C to stop
```

folk.rb

```
module Folk
 # Типове директории, според съдържащите се в тях файлове
 DirTypes=["latex", "lilypond", "mp3", "midi", "pdf", "eps"]
 # Съответствие на типовете файлове с MIME типовете
 FileTypes={"latex"=>"text/plain", "lilypond"=>"text/lilypond",
            "mp3"=>"audio/mpeg", "midi"=>"audio/midi",
            "pdf"=>"application/pdf",
            "eps"=>"application/postscript"}
 # Списък с директории
 Dirs = [
    { :type => "latex", :path => "txt1", :files => Regexp.new(".txt$") },
    { :type => "lilypond", :path => "td1", :files => Regexp.new(".ly$")},
    { :type => "mp3", :path => "mp3", :files => Regexp.new(".mp3$") },
    { :type => "midi", :path => "out", :files => Regexp.new(".midi$") },
    { :type => "pdf", :path => "out", :files => Regexp.new(".pdf$") },
    { :type => "eps", :path => "out", :files => Regexp.new(".eps$") }
 ]
end
```

index.rb

```
require 'ferret'
require 'folk'
class Folk::Index
   def initialize(options={})
      @default_field = options[:default_field] || :content
      @init = options[:init] || false
      @path = options[:path] || 'ferret'
      @to_index = Array.new
      @index = Index::Index.new(:default_field => @default_field, :path => @path)
      @index.field_infos.add_field(:code, :index => :untokenized, :store => :yes)
      @index.field_infos.add_field(:content, :index => :yes, :store => :yes)
      # За всеки тип директория добавяме поле в индекса
      Folk::DirTypes.each do |type|
         @index.field_infos.add_field(type.to_sym, :index => :untokenized, :store => :yes)
      end
   end
   # Създава index с две обхождания
   def run
      # Pass 1, code & content
      Folk::Dirs.each do |dir|
         if dir[:type] == "latex"
                                 # code ...
         end
      end
      # Pass 2
      Folk::Dirs.each do |dir|
         Find.find(dir[:path]) do |path|
                                 # code ...
         end
      end
      # Запис на резултатите в индекса
      @to_index.each do |doc|
         @index << doc
      end
   end
end
```

search.rb

```
require 'ferret'
require 'folk'

class Folk::Search
        attr_accessor :results

   def initialize(options={})
      @index = options[:index] || 'ferret'
      @field = options[:field] || 'content'
      @searcher = Ferret::Search::Searcher.new(@index)
      @parser = Ferret::QueryParser.new(:default_field => @field,
           :analyzer => Ferret::Analysis::StandardAnalyzer.new)
      @query = @parser.parse(options[:query])
      @results = Array.new
   end

   # Старт на търсенето
   def run
      @searcher.search_each(@query, :limit => @limit) do |doc, score|
      @results << { :document => @searcher[doc],
                    :highlight => @searcher.highlight(@query, doc, 'content').to_s,
                    :score => score }
      end
   end
end
```

Локално използване на файловете

Всеки участник в проекта има възможност да изтегли локално на компютъра си сорс кода на песните чрез Subversion - да прави поправки в тях и да изпраща промените обратно в хранилището.

Ruby код

Системата може да бъде разширявана с различна функционалност чрез код написан на Ruby.

HTTP интерфейс

Външни системи могат да използват HTTP REST интерфейса за комуникация с:

- Ferret индекса
- Ferret търсене
- Достъп до файловете с песните