

Programmierung 2

Programmieraufgabe „Wordcloud“ (*vorläufiger Stand*)

Prof. Dr. Oliver Hummel



**angezogene Handbremse oder so,
bin kein Biologe**





Wir wollen ein Tool entwickeln, mit dem klickbare HTML-Wordclouds gebaut werden können

- Word- oder Tag-Clouds analysieren eine Menge von Text und stellen die darin enthaltenen Wörter je nach Frequenz in verschiedenen Größen dar

add adding ai animation center cloud code color
community content cool create CSS data-attribute data-show-value
data-weight demo dev development different
display focus font-size functionality hrefhtmlimage instead
item jan javascript li lines links list looks popular profile read rem
remove size style tag terms ul value web weight work

Quelle: <https://tagcrowd.com>




User Story: Als Professor möchte ich Wordclouds aus meinen Dateien (Folien) erstellen können, um Studierenden einen Überblick über die enthaltenen Themen geben zu können.

Main Features

- Beliebige Ordner mit folgenden Dateitypen sollen analysierbar sein
 - .txt
 - .pdf (z.B. PDFBox)
 - .docx, .pptx (z.B. Apache POI)
- Link-Vorgabe soll gesetzt werden können, bspw.
 - bspw.: <https://www.google.com/search?q=XXXXXX>
- Stopwords sollen über eine Datei gesetzt und über die UI ergänzt werden können
- Generierte Datei im Browser anzeigen: `Runtime.getRuntime().exec(command);`



In Anlehnung an Tagcrowd.com

Language of text: 

Ignore common words in this language

Maximum number of words to show?
25 – 100 is a good range

Minimum frequency?
Don't show infrequent words

Show frequencies? ☒ no ☐ yes
Show word count next to each word

Group similar words? (English only)
eg: learn, learned, learning -> learn
☐ no ☒ yes

Convert to lowercase? ☒ lowercase ☐ original
eg: PhD -> phd, FBI -> fbi, Rio -> rio

Don't show these words:
Exclude unwanted words. sog. Stopwords

und dafür

sog. Stemming mit Apache Lucene

- Worte alphabetisch sortiert oder nicht
- Statistische Angaben loggen, wie Gesamtzahl der Worte, versch. Worte o.ä.
- (neu): Ausgabe Wordcloud als CSV-Datei, optional als Excel



Abgabe: Maven-Projekt über Gitty (koehler und hummel mit Schreibrechten zum Repo hinzufügen)

- individuell zu erstellen und zu testieren
- Verwendung von Branches ist obligatorisch

Nutzen Sie eine Layered Architecture

- inkl. voll qualifizierten Package-Namen

Erstellen Sie sinnvolle JUnit-Tests für Ihre Domänenlogik

Code-Qualität (PMD) beachten

Optional: Verwendung einer GUI



Bitte geben Sie bei Übernahme von mehr als 3 Zeilen Code (Internet/KI) die Quelle oder den Prompt an

- z.B. in einer Datei quellen.txt im Ihrem Projektverzeichnis

Verwenden Sie für die Textextraktion bitte Bibliotheken

- Apache POI für Office-Formate
- Apache PDFBox für PDFs
- Apache Lucene für das sogenannte Tokenizing und Stemming

Für Log-Ausgaben einen Logger verwenden

- s. Log4J im MVN-Projekt

Sie erhalten eine vorbereitete HTML-Datei

- in diese müssen „nur noch“ die Worte mit ihrer Frequenz geschrieben werden



Dependencies finden

Go to: <https://central.sonatype.com>

Find OSS Components

As stewards of Central for nearly 20 years and inventors of both software supply chain management and Nexus Repository, Sonatype knows that the integrity of your build is critical.

pdfbox 3.0.4 published 3 months ago in **org.apache.pdfbox**

The Apache PDFBox library is an open source Java tool for working with PDF documents.

pdfbox 3.0.4
Used in: 316 components

[Overview](#) [Versions](#) [Dependents](#) [Dependencies](#)

Overview

Description

The Apache PDFBox library is an open source Java tool for working with PDF documents.

Snippets

Apache Maven [Copy to clipboard](#)

```
<dependency>
  <groupId>org.apache.pdfbox</groupId>
  <artifactId>pdfbox</artifactId>
  <version>3.0.4</version>
</dependency>
```

*Achtung, OSS-Libs haben
üblicherweise eine Lizenz,
die u.U. die Offenlegung
des eigenen Quellcodes
notwendig machen kann!*


```
String pdfFilePath = "src/main/resources/some.pdf";  
File pdfFile = new File(pdfFilePath);  
  
try (PDDocument document = Loader.loadPDF(pdfFile)) {  
    PDTextStripper pdfStripper = new PDTextStripper();  
    String text = pdfStripper.getText(document);  
  
    System.out.println(text);  
  
} catch (Exception e) {  
    e.printStackTrace();  
}
```



```
String text = "Der schnelle braune Fuchs springt über den lahmen Hund.";
```

```
try (Analyzer analyzer = new GermanAnalyzer()) {  
    TokenStream tokenStream = analyzer.tokenStream(null, text);  
    CharTermAttribute termAttribute =  
        tokenStream.addAttribute(CharTermAttribute.class);  
  
    tokenStream.reset();  
  
    while (tokenStream.incrementToken()) {  
        System.out.println(termAttribute.toString());  
    }  
  
} catch (IOException e) {  
    e.printStackTrace();  
}
```