NLP-Werkstatt
SS2017
Automatische Erkennung von HTML-Entities in Hexadezimalkodierung
Hausaufgabe – Abgabe bis 03.05.2017, 14:00 Uhr

Aufgabe Extrahieren Sie alle hexadezimal kodierten *HTML-Entities* (Sonderzeichen bzw. Emoticons/Emojis) mit einem selbstgeschriebenen Programm in der Programmiersprache Python oder Java aus dem Textfragment der beiliegenden Datei *emojis.html*[1]. Abzugeben ist das selbsterstellte Programm.

Der Inhalt der Datei *emojis.html*:

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
      html4/loose.dtd">
2 <html>
3 <head></head>
4 <body>
5 Juhuu! Es ist Fr&#x00FC;hling &#x1F600; und die &#x2600;&#xFE0F; scheint!
6 <br/>
7 &#x2753; Wo lebstu? Hier Schei&#x00DF; Wetter &#x1F327; &#x1F620; &#x26A1;...
8 </body>
9 </html>
```

Für die Abgabe können folgende Szenarien implementiert werden:

1. Extraktion des Texts (auf Zeilen 5 und 7 der Eingabedatei):
   a) in der Implementierung hartkodiert angeben (z.B. als Strings)
   b) aus der Datei *emojis.html* einlesen

2. Extraktion der hexadezimal kodierten HTML-Entities:
   a) hartkodiert in der Implementierung angeben (z.B. als Schlüssel-Wert-Paare von Hexadezimal-Codes und Beschreibung – für den Character ☺ könnte das folgende Schlüssel-Wert-Paar in einem Dictionary angelegt werden: { `"&#x1F600;": "grinning face"` })
   b) aus den Textdateien *Unicode_UTF-8-character_table_tab.txt* und *emoji-test_tab.txt* einlesen – Diese Dateien enthalten die HTML-Entities und ihre Beschreibung in einem TAB-separierten Format: Jede Zeile enthält den Eintrag zu einer HTML-Entity: `<Hexadezimal-Code> \t <darstellbarer Charakter> \t <Beschreibung>`

[1] Die in der Aufgabenstellung erwähnten Eingabedateien sind unter http://www.cl.uni-heidelberg.de/courses/ss17/annotierteKorpora/material/data_HA01.zip herunterzuladen.

3. Ausgabe
   a) auf die Konsole
   b) in eine Ausgabedatei namens *found_html-entities.txt* mit hartkodierter Angabe des Dateinamens in der Implementierung
   c) in eine Ausgabedatei, deren Namen man als Kommandozeilenargument angibt
4. Ausgabeformat:
   a) Für jeden gefundenen Hexadezimal-Code im Text je eine Zeile mit folgendem Inhalt: `<Code>: <Beschreibung>`.
   b) Ausgabeninhalt wie in 4a, aber hübscher formatiert, ggf. mit weiteren, eigens ausgedachten Informationen angereichert.

Mindestanforderung: Für eine erfolgreiche Abgabe sollten alle **a)**-Punkte in der Implementierung realisiert werden: Hartkodierte Angabe des Texts; Extraktion der HTML-Entities mithilfe einer hartkodiert gestalteten Datenstruktur; und Ausgabe der gefundenen HTML-Entities mit ihrer Beschreibung auf die Konsole.

Hinweise: Bei der Implementierung soll man auf folgende Sachen achten:

- ausreichende Dokumentation des Codes, inklusive Angabe des Autors
- Behandlung von eventuellen Fehlermöglichkeiten
- übersichtliche Gestaltung der Klassen und Methoden

Mögliche Zusatzaufgaben für Interessierte

- Implementieren Sie die Aufgabe in beiden Programmiersprachen.
- Implementieren Sie die Aufgabe so, dass viele Szenarien mit dem gleichen Programm möglich sind (– nur sinnvolle Kombinationen betrachten).
- Lesen Sie die hexadezimale Kodierung der HTML-Entities direkt aus den beiliegenden Dateien *Unicode_UTF-8-character_table_orig.html* und *emoji-test.txt*[2] ein.

[2]Die Dateien sind auch herunterladbar unter http://www.utf8-chartable.de und http://www.unicode.org/Public/emoji/5.0/emoji-test.txt