Maschinenlesbare Formate

Um Metadaten darzustellen gab es im Laufe der Zeit unterschiedliche Ansätze, die populärsten sollen hier vorgestellt werden. Die Wahl der Formate richtet sich nach den Kriterien

- Flexibilität: Müssen die Daten auf eine bestimmte Art und Weise strukturiert sein?
- Verständlichkeit: Ist aus einem Datensatz direkt ersichtlich, was gemeint ist?
- Aufwand: Wie aufwendig ist die Umsetzung eines Datenformats, gibt es Tools die eingesetzt werden können?

und ist letztendlich Ihrer Institution überlassen.

CSV

CSV ist die Abkürzung für "comma seperated values". Es handelt sich also um eine kommaseparierte Liste. Dieses Format wird oft benutzt, um Informationen aus bestehenden Datenbanken zu extrahieren. Dazu werden alle Daten einer Zeile durch ein bestimmtes Zeichen (Separator) getrennt. Klassischerweise ist dies das Komma. Bei der Wahl des Separators ist zu beachten, dass dieses Zeichen im gesamten Datensatz nicht vorkommen darf, da sonst die Einteilung in Spalten nicht mehr möglich ist.
Aus Entwicklerperspektive kann man zur weiteren Verarbeitung auf die verschiedenen Werte spaltenweise zuzugreifen. Zum Ansehen dieses Datenformats kann beispielsweise Excel eingesetzt werden. In einer beispielhafte CSV-Datei kann ein Stundenplan wie folgt aussehen:

```
Stunde;Montag;Dienstag;Mittwoch;Donnerstag;Freitag
1;Mathe;Deutsch;Englisch;Mathe;Kunst
2;Sport;Französisch;Geschichte;Sport;Geschichte
3;Sport;"Religion ev;kath";Kunst;;Kunst
```

In der ersten Zeile stehen die Überschriften der jeweiligen Spalten. Danach sind die einzelnen Datensätze jeweils zeilenweise gelistet. Als Separator wird in diesem Fall das Semikolon verwendet. Dies bietet sich insbesondere bei Daten an, die Fließkommazahlen enthalten.

Vorteile

- einfache Erstellung und Export aus z.B. Excel
- einfache Verarbeitung

- einfache Aufteilung in mehrere einzelne Dateien problemlos möglich (Daten können dann nacheinander - statt alle auf einmal - geladen werden, also werden die Ladezeiten verkürzt)

Nachteile

- kein Abbildung von komplexen Strukturen möglich

Nutzungsbeispiele:

- Extrahierung von Daten aus relationalen Datenbanken
- Abbildung von strukturierten kleinen Mengen von Daten (< 10000 Datensätze)
- Numerische Datensätze (mind. 90% der Daten sind nur Zahlen)

XML

Die Abkürzung XML steht für eXtensible Markup Language. Es handelt sich hierbei um eine Auszeichnungssprache, mit der auch die Darstellung von komplexen Datenstrukturen problemlos möglich ist. Zur Strukturierung der einzelnen Informationen werden Tags eingesetzt. Ein Tag beginnt mit einer öffnenden und endet mit einer schließenden, spitzen Klammer. Eine einzelne Information wird in ein Start- und ein Endtag gekapselt.

```
<vorname>Klaus</vorname>
```

Ein Endtag zeichnet sich dadurch aus, dass es stets mit einem "/" beginnt. Die einzelnen Tags können ineinander verschachtelt werden.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<verzeichnis>
 <titel>Wikipedia Städteverzeichnis</titel>
<eintrag>
     <stichwort>Genf</stichwort>
     <eintragstext>Genf ist der Sitz von ...</eintragstext>
</eintrag>
<eintrag>
          <stichwort>Köln</stichwort>
     <eintragstext>Köln ist eine Stadt, die ...</eintragstext>
</eintrag>
</verzeichnis>
```

Die Informationen stehen also immer zwischen den Tags. Die Tags selber geben an, um was für eine Art von Information es sich handelt oder wie diese dargestellt werden soll (z.B. Paragraphen oder Hervorhebungen).
Die Strukturierung der Daten wird mit einer DTD (Document Type Definition, Schemadefinition) beschrieben. Hier wird aufgeführt, welche Tags in welche geschachtelt werden können. Dem Entwickler wird so das Benutzen der XML-Daten dadurch extrem erleichtert. Darüber hinaus ist es möglich, eine Dokumentation aus der Schemadatei zu extrahieren.

Vorteile

- Bildung von beliebig komplexen Datenstrukturen
- verhältnismäßig einfach von Menschen zu lesen
- Möglichkeit zur Auszeichnung von bestimmten Textpassagen (z.B. Text, der fett oder kursiv dargestellt werden soll)

Nachteile

- bei komplexen Datenstrukturen geht schnell die Übersicht verloren
- viel Overhead beim Transport der Daten (Daten, die nur zur Strukturierung benutzt werden)

Nutzungsbeispiele:

- Bibliotheksdaten
- Daten, für die schon ein entsprechendes Schema (DTD) existiert

JSON

JSON bezeichnet die Notation, die in der Programmiersprache JavaScript verwendet wird. JavaScript ist eine Sprache, die in erster Linie für Webanwendungen eingesetzt wird, d.h. um Webseiten zu bauen. JSON steht hier für Javascript Object Notation. In JSON gibt es lediglich Objekte und Listen als mögliche Datentypen. Listen werden mit eckigen Klammern ("[","]"), Objekte mit geschweiften Klammern ("{", "}") ausgezeichnet. Listen sind einfach Sequenzen von Inhalten, die direkt hintereinander stehen. Bei Objekten kann jeder Wert mit einem Schlüssel beschrieben werden. Dadurch ist es möglich direkt auf den Wert zuzugreifen.

```
{
      "titel": "Wikipedia Städteverzeichnis",
      "einträge": [
            {
                  "stichwort": "Genf",
                  "eintragstext": "Genf ist der Sitz von ..."
            },
            {
                  "stichwort": "Köln",
                  "eintragstext": "Köln ist eine Stadt, die ..."
            }
      ]
}
```

Das obige Beispiel ist die JSON Variante vom oben beschrieben XML-Auszug. Die Darstellung ist viel kompakter, allerdings auch nicht so aussagekräftig wie XML. Der größte Vorteil dieses Formats ist aber, dass es direkt in JavaScript ohne umständliche Konvertierung verwendet werden kann. Auch für andere Plattformen gibt es entsprechende Konvertierungstools, die das Verarbeiten von JSON bis zu einem hohen Grad automatisieren.

Vorteile

- wenig Overhead beim Übertragen der Daten
- Bildung von beliebig komplexen Datenstrukturen möglich

Nachteile

- schwer verständlich
- Auszeichnung von Textpassagen verhältnismäßig aufwendig

Nutzungsbeispiele:

- Webservices (Programmierschnittstelle, die über das Internet abrufbar ist)
- Webseiten