INSY Schummelzettel semistrukturierte Datenformate

Andreas Sünder

03.10.2022

Datenaustauschformate

CSV

CSV ist ein tabellenbasiertes Austauschformat und steht für *Comma-separated values* und wird verwendet, um Daten zwischen Applikationen zu übertragen, bspw. über das Web.

Beispiel:

```
f . name ; name ; address ; city ; state
James ; Smith ;6649 N Blue Gum St ; New Orleans ; LA
Jenna ; Darakjy ;4 B Blue Ridge Blvd ; Brighton ; MI
Art ; Venere ;8 W Cerritos Ave 54; Bridgeport ; NJ
Lenna ; Paprocki ;639 Main St ; Anchorage ; AK
```

f...name	name	address	city	state
James	Smith	6649 N Blue Gum St	New Orleans	LA
Jenna	Darakjy	4 B Blue Ridge Blvd	Brighton	MI
Art	Venere	8 W Cerritos Ave 54	Bridgeport	NJ
Lenna	Paprocki	639 Main St	Anchorage	AK

Jede Zeile steht für eine Zeile in der Tabelle, die einzelnen Zellen werden durch ein Trennzeichen getrennt.

CSV selbst ist aber nicht standadisiert, weshalb es viele verschiedene Dialekte geben kann, da beliebige Trennzeichen verwendet werden können. Etwa interpretieren die Amerikaner das Komma (,) und den Punkt (.) anders, weshalb das Trennzeichen manchmal ein Semikolon (;) ist.

JSON

Die JSON (*Javascript Object Notation*) sind Dokumente, die valide JavaScript-Objekte sind und in 1:1 in solche Objekte umgewandelt werden können. JSON ist aber so beliebt, dass diese Struktur fast überall (nicht nur in JS), hauptsächlich zur Übertragung von Daten (etwa übers Web), eingesetzt wird. Weitere Einsatzgebiete sind Ajax, Webapplikationen und Webservices.

Beispiel:

```
[{
" first_name " : " James " ,
" last_name " : " Smith " ,
" address " : " 6649 N Blue Gum St " ,
" city " : " New Orleans " ,
" state " : " LA " ,
" zip " :70116 ,
" female " : false ,
" phones " :[ " 504 -621 -8927 " ," 504 -845 -1427 " ]
} ,{
" first_name " : " Jenna " ,
" last_name " : " Darakjy " ,
" address " : " 4 B Blue Ridge Blvd " ,
" city " : " Brighton " ,
" state " : " MI " ,
" zip " :48116 ,
" female " : true ,
" phones " :[ " 810 -292 -9388 " ," 810 -374 -9840 " ]}]
```

JSON definiert verschiedene (von JS abstammende) Datentypen, wie:

- Strings
- Boolean
- Zahlen
- Arrays []
- Objekte {}

Der Ursprung bildet ein großes, übergeordnetes Objekt, welches weitere Unterobjekte speichern kann. Jedes Objekt enthält ein Set von Eigenschaften, welche jeweils aus einem Schlüssel (*Key*) und Wert (*Value*) bestehen. Dadurch können ganze Hierarchien gespeichert werden, welche dann mit einer Programmiersprache eingelesen und gespeichert werden können (Serialisierung).

XML

XML steht für (*Extensible Markup Language*) und ist eine Definitionsprache, auf welche etwa HTML basiert. Damit können Textdokumente erstellt werden, welche das Ziel haben, möglichst menschenlesbar zu sein. Wie JSON verwendet XML eine strukturierte Darstellung von Informationen, wobei ebenfalls Hierarchien möglich sind.

Beispiel:

```
<? xml version = " 1.0 " ? >
< menu >
  < food calories = " 650 " >
```

```
    < name > Belgian Waffles </ name >
    < price > $5 .95 </ price >
    < description >
      Two of our famous Belgian Waffles with plenty of real
      maple syrup
    </ description >
  </ food >
  < food calories = " 900 " >
    < name > Strawberry Belgian Waffles </ name >
    < price > $7 .95 </ price >
    < description >
      Light Belgian waffles covered with strawberries
    </ description >
  </ food >
</ menu >
```

XML wird ebenfalls bei Webapplikationen und Webservices eingesetzt.

Wie bei HTML gibt es ein Wurzelelement; einzelne Elemente müssen geöffnet und wieder geschlossen werden (Tags). Elemente können weitere Unterelemente sowie Attribute besitzen.

DOM

Das DOM (*Document Object Model*) ist ein W3C-Standard für den programmgesteuerten Zugriff auf XML-Dateien. Dabei wird ein Dokument als Baumstruktur dargestellt. Da ein gesamtes DOM-Dokument (wie HTML-Seiten) in den Speicher geladen wird, ist ein wahlfreier Zugriff möglich.

Die Spezifikation von DOM unterscheidet zwischen:

- **DOM Core**: Wichtige gemeinsame Interfaces (Node, Element, ...) und Konzepte (Namespaces, ...)
- **DOM HTML**: DOM HTML: HTML-spezifische Definitionen (HTML-Elemente, Attribute, Methoden und Events)
- **DOM XML**: Modell für Zugriff und Manipulation von XMI-Dateien

Ein wichtiges Interface, das von der Spezifikation definiert ist, stellt `Node` dar:

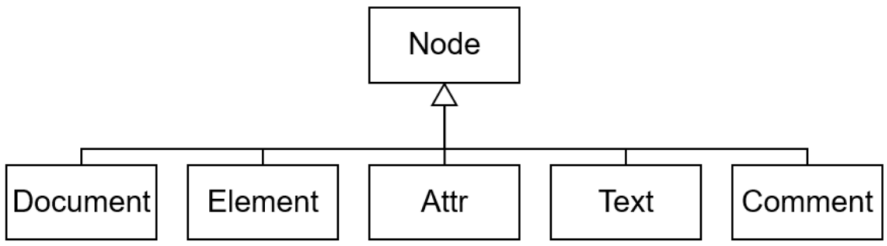

Figure 1: Das `Node` Interface

Für jeden Typ gibt es dabei unterschiedliche erlaubte Kindelemente und Rückgabewert (via `nodeValue` und `nodeName`).