Chromstahl-CMS

Frederick Lahde, Lars Grahmann

9. Mai 2019

Inhaltsverzeichnis

Abkürzungsverzeichnis

DOM	Document Object Model
Virtual DOM	Virtual Document Object Model
REST	Representational State Transfer
CMS	Content-Management-System
CaaS	Content as a Service
RBAC	Rolebased Access Control
CMA	Content Management Application
CDA	Content Delivery Application
DBMS	Database Management System
ERM	Entity Relationship Model

Einleitung

Thema

Theoretische Betrachtung

Content-Management-System

Content Management Application

Content Delivery Application

Bestehende Lösungen

Wordpress

Joomla

Drupal

Document Object Model

Geschichte

Standards

XML! (**XML!**) bildet eine Dokumentenhierarchie ab, welche einen maschinell lesbaren Aufbau eines Dokumentes, mithilfe von einer generischen Syntax, definiert. **XML!** wird in verschiedenen Formen in vielen Geräten und Programmen genutzt um Datenstrukturen abzubilden. Diese Datenstrukturen können in verschiedenen, Domän-spezifischen varianten beschrieben werden, wie zum Beispiel

- Scaleable Vector Graphic (SVG)

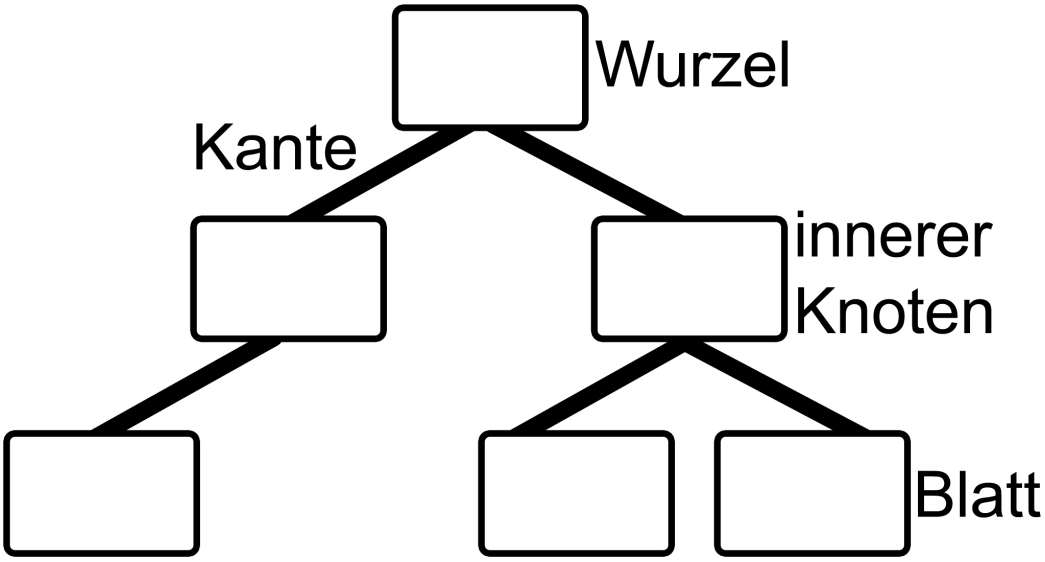

Abbildung 3.1: Baumstruktur
Von Mhombach - Eigenes Werk, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=29981537

- Rich Site Summary (RSS)
- Extensible Hypertext Markup Language (XHTML)

Das Document Object Model (DOM) definiert ein **API!** (**API!**) zum manipulieren von **XML!** als Baumstruktur. [1]

Aufbau

Eine Baumstruktur (3.1) besteht aus einem Wurzel-knoten, welcher mithilfe von Kanten “Kindknoten” oder auch “innere Knoten” besitzen kann. Hat ein Knoten keine “Kinder”, wird er als “Blatt” bezeichnet. Baumstrukturen sind für die Darstellung von Daten aufgrund ihrer einfachen Gestaltung vorteilhaft. Eine Baumstruktur kann sehr leicht mithilfe von rekursiven Funktionen durchlaufen werden, da ein Knoten immer genau ein “Elternteil” und eine Liste von “Kindern” hat.

Virtual Document Object Model

Definition

Ein Virtual Document Object Model (Virtual DOM) ist eine Repräsentation eines DOM welche durch arbiträre Datenstrukturen abgebildet werden kann. Eine häufige Implementation ist durch Javascript-Objekte, welche die Daten des DOM abbilden.

Motivation

Oft wird ein Virtual DOM verwendet, da aus manipulation der Daten automatisch eine Änderung im DOM abgebildet werden kann, und somit der Entwickler lediglich die Änderung der Daten bedenken muss.

Nachteile

Eine Repräsentation des DOM als Virtual DOM führt zwangsläufig dazu, dass die Struktur des DOM doppelt vorhanden ist. Desweiteren ist das finden der Änderungen zwischen DOM und Virtual DOM nicht trivial und langsamer als eine "direkte" Änderung der Daten im DOM. Weiterhin muss der Web-Browser zusätzlich zum finden der Änderungen zwischen DOM und Virtual DOM immer die DOM-**API!** Aufrufe ausführen.

Statemanagement

Repräsentation

Änderungen

SPA!

Datenstrukturen

Database Management System

Entity Relationship Model

Representational State Transfer

Definition

REST Verben

Rolebased Access Control

Plugins

Definition

Implementationsansätze

Dynamisch

Statisch

Methodik

Vergleich von Web-Frontend Sprachen

Javascript

Typescript

Web-ASM

Vergleich von Frontend-Frameworks

Vue

JQuery

Eigene Implementation

Vergleich von Web-Backend Sprachen

Java

Go

Javascript

Vergleich von Backend-Frameworks

Spring

Play

JavaEE

Vergleich von Datenbank-Lösungen

Relational Database

Umsetzung

Virtual DOM Implementation

Datenstrukturen

Rendering

Komponenten

Routing

Content as a Service Implementation

Datenstrukturen

Sicherheit

Datenbank Kommunikation

REST-Spezifikation

Frontend

Designentscheidungen

Implementationsdetails

Plugins

Software Development Kit

Backend

Frontend

Pluginstruktur

Diskussion

Plugins: Dynamischer Ansatz

Ausblick

Fazit

Literaturverzeichnis

[1] Elliotte Rusty Harold & W. Scott Means, XML in a Nutshell, Oreilly, 2004.