Einführung in RDF

Markus Stocker

23. April 2018

Rekapitulation

- Wozu brachen wir Schemata?
- Was sind die Document Type Definition (DTD) und XML Schema?
- Was für Vorteile hat XML Schema gegenüber der DTD?

Übersicht

- Kurzgefasst: Was ist RDF?
- Exkurs *Semantic Web*
- RDF: Die Details

RDF

- Resource Description Framework
- Ein “System zur Beschreibung von Ressourcen” (Wikipedia)
- Ursprünglich für Beschreibung von Metadaten über *Web* Ressourcen
- In der Praxis sind heutzutage Ressourcen beliebige Dinge
- Web Ressourcen, physische objekte, Konzepte, immaginäre Entitäten

RDF

- Grundlegender Baustein des *Semantic Web*
- W3C Recommendation seit 1999
- Mehr dazu gleich, ...
- Zuerst etwas über das Semantische Web

Semantische Web (*Semantic Web*)

- Ursprünglich als Erweiterung des *World Wide Web* gedacht
- Intelligentere Suche und, generell, Daten Verarbeitung ermöglichen
- Beabsichtige Bedeutung von Daten Maschinen zur Verfügung stellen
- Informationsverarbeitung, Daten und deren Bedeutung

Semantische Web (*Semantic Web*)

- XML ermöglich Spezifikation von Sprachen mit formaler Syntax
- Maschinen können XML Daten auf syntaktische Korrektheit prüfen
- Daten haben allerdings keinerlei formale Bedeutung (Semantik)
- Was bedeutet `<planet>`?
- `<planet>` und `<Planet>`: Syntaktisch ungleich. Gleiche Bedeutung?

Semantische Web (*Semantic Web*)

- `<planet>`: Himmelskörper der eine Sonne umkreist
- Bedeutung explizit spezifizieren, mittels formaler Sprache
- Dadurch ist die Bedeutung maschinenverarbeitbar
- Zudem ist das Konzept *Planet* als Ressource identifiziert
- Dazu benutzt man ein *Uniform Resource Identifier* (URI)
- Zum Beispiel `http://example.org#Planet`
- Oder `http://example.org#R34GH4`
- Schreibweisunabhängige Identifizierung: `planet`, `Planet`, `PLANET`
- Sprachunabhängige Identifizierung: `pianeta`, `planeetta`, `planète`

Semantische Web (*Semantic Web*)

- Praktische Ziel ist Maschinen Zugang zu *mehr* Information geben
- Ein Ideal gegen sich das WWW entwickeln könnte, oder sollte
- Einige Hürden
  - Wie modelliert man menschliches Wissen mittels formaler Sprache?
  - Wie können Maschinen Wissen in automatisieter Verarbeitung nutzen?
  - Wie kann Information (Daten und Bedeutung) integriert werden?
- Dafür wurde einiges an Technologie entwickelt
- Einige werden wir im Detail anschauen

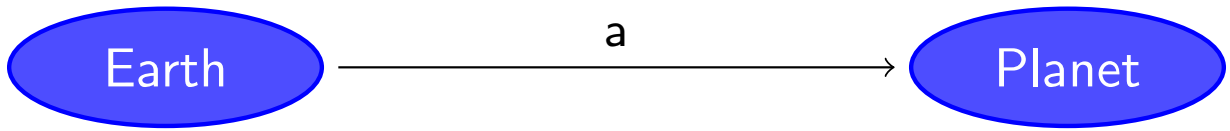
Resource Description Framework (RDF)

- Formale Sprache zur Beschreibung strukturierter Information
- Ermöglicht Datenaustausch unter Einhaltung der Bedeutung
- Ursprünglich (ca. 1999) ein Datenmodell für Metadaten
- Hauptsächlich Metadaten über Web Ressourcen (Seiten, Bilder, etc.)
- Heute generell zur Representation semantischer Information
- Ressourcenorientiert, nicht dokumentenorientiert
- Grundstein des *Semantic Web*

RDF

- Ein RDF Dokument spannt ein gerichter Graph
- Eine Menge aus Knoten die über gerichtete Kanten verknüpft sind
- Kleinste Menge besteht aus zwei Knoten
- Diese sind mittels einer Kante verknüpft
- Diese Struktur bildet elementare Informationseinheit
- Sie wird *statement* genannt

RDF

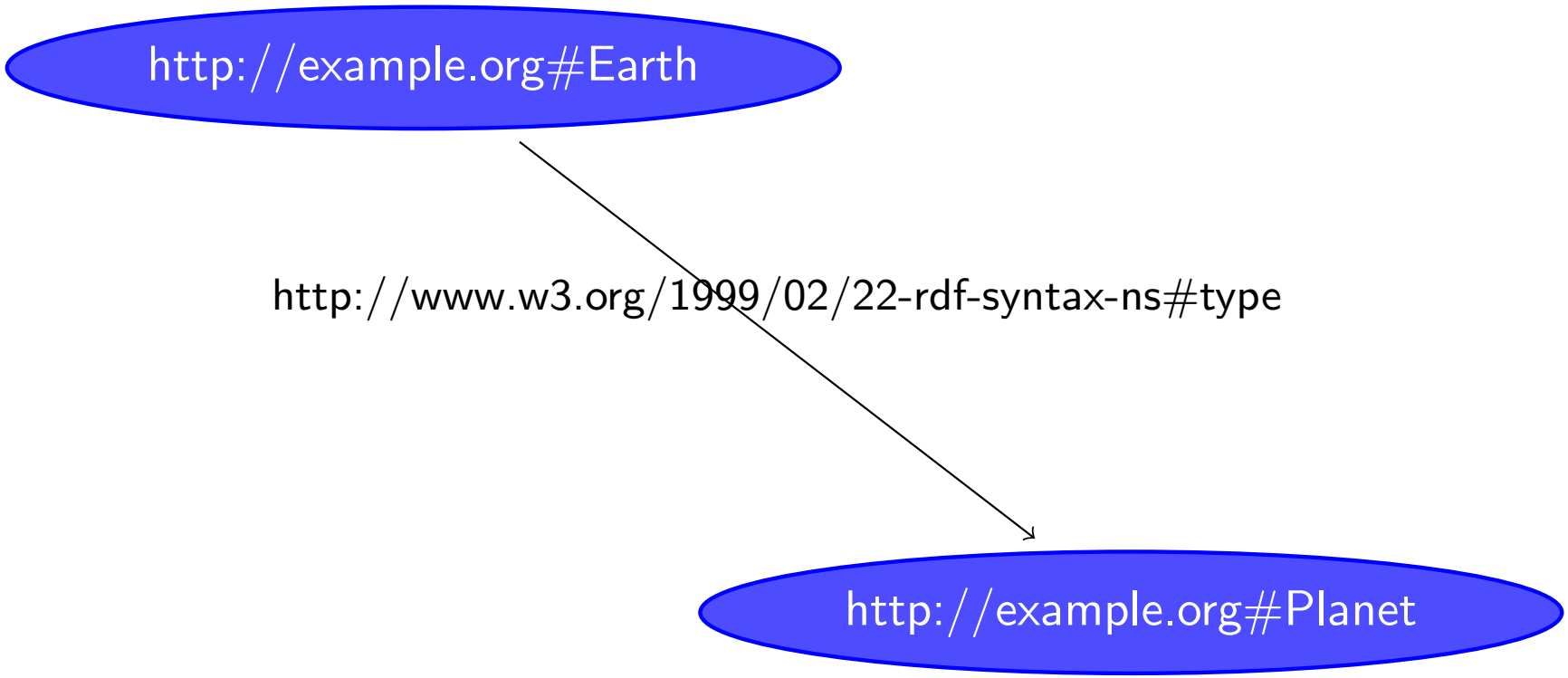


"Earth is a Planet"

RDF

- Knoten und Kanten werden mittels Identifikatoren benannt
- Man verwendet dafür *Uniform Resource Identifier* (URI)
- Dies unterstützt die (globale) Unterscheidung von Ressourcen
- Die URI ist eine Referenz zur beabsichtigen Ressource
- Beispiele: Planeten, Bücher, etc. als konzepte oder physische Dinge

RDF


http://example.org#Earth
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://example.org#Planet

URI

- URI ist eine Generalisierung des *Uniform Resource Locator* (URL)
- URL ist somit eine Unterart der URI
- Jeder URL ist auch ein URI
- URL allgemein bekannt als Webadresse, z.B. `http://google.com`
- URL identifiziert *und* lokalisiert eine Ressource
- Meist eine digitale Ressource die im Internet lokalisiert ist
- Lokalisiert auf einem IP identifizierten Rechner
- URI ist ein Identifikator, lediglich identifiziert eine Ressource
- Diese Ressource muss nicht digital sein
- Zudem muss sie nicht im Internet lokalisiert sein
- Tippt man ein URI in den Browser, erhält man u.U. einen Fehler

URI

- Im Gegensatz zu URL, gibt es keine Institution die URI vergibt
- Eine URL wie z.B. `markusstocker.com` is registriert
- An mehreren Stellen, inklusive Hosting Provider DNS
- Man bezahlt für die verschiedenen Services
- Ein URI kann man "einfach so" erstellen
- Wobei es schon ein paar Regeln für "cool URIs" gibt
- Es ist von Vorteil wenn URIs eindeutig sind
- Wobei das nicht gewährleistet wird
- Eine Ressource kann durchaus mehrere URIs haben
- Zudem kann ein URI für mehrere Ressourcen benutzt werden
- URI Aufbau ist wichtig um Änderungsbedarf zu minimieren
- Schlecht sind z.B. Technologieabhängigkeiten (wie `.php`) oder *?query*

URI Aufbau

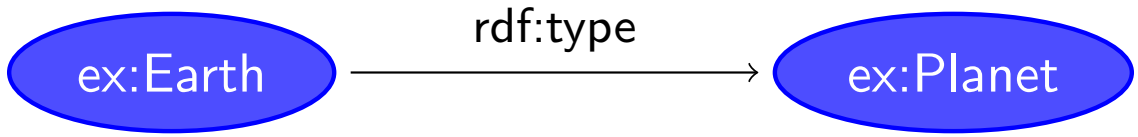
`scheme:[//authority]path[?query][#fragment]`

- `scheme`: Meist `http`, selbst wenn nicht auf dem Web lokalisiert
- Beispiel: `http://example.org#Earth` ist ein URI
- `authority`: Meist ein Domänennamen, z.B. `//example.org`
- `path`: Meist hierarchisch strukturierte Pfade, z.B. `/astro/vocab`
- `?query`: Optional, für URL Parameter; für URI abgeraten
- `#fragment`: Wird für URI oft verwendet um Dinge zu identifizieren

URI Abkürzen

- Es ist möglich URIs abzukürzen
- Dies geschieht mittels eines Präfix in Form `prefix:name`
- Der Präfix ist frei wählbar, muss aber definiert werden
- Identifikatoren in dieser Form werden auch *qualified names* genannt
- Beispiel
  - Definiere Präfix `ex:` für `http://example.org#`
  - Für den URI `http://example.org#Earth`
  - Ergibt sich der ensprechende *qualified name* `ex:Earth`

URI Kürzung in RDF



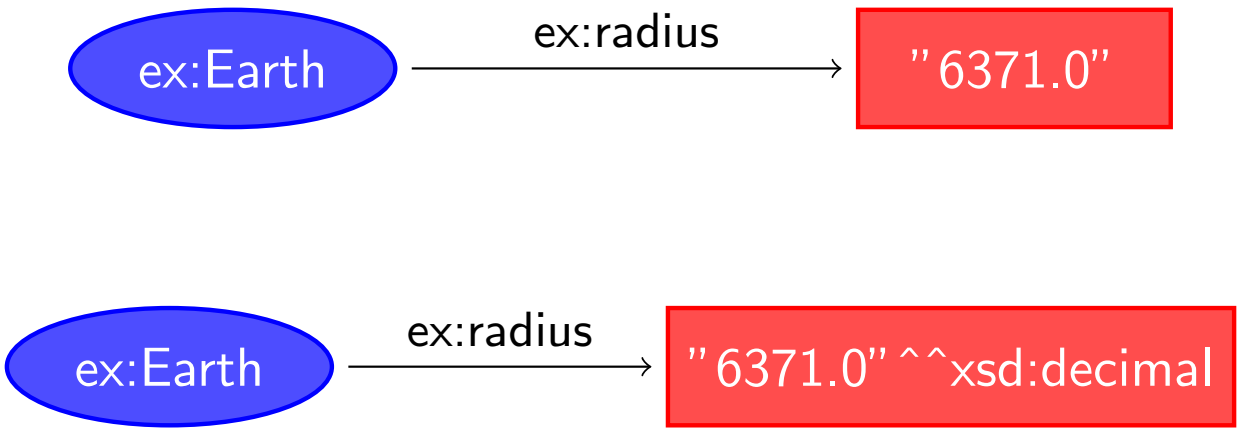
RDF Knoten: Nicht nur URI

- In unseren Beispielen waren bisher Knoten mittels URI benannt
- In RDF gibt es zwei weitere Knoten Formen
- RDF Knoten für Datenwerte, das sogenannte *literal* wie z.B. `6371.0`
- Der sogenannte *blank node*, sprich unbenannte Knoten

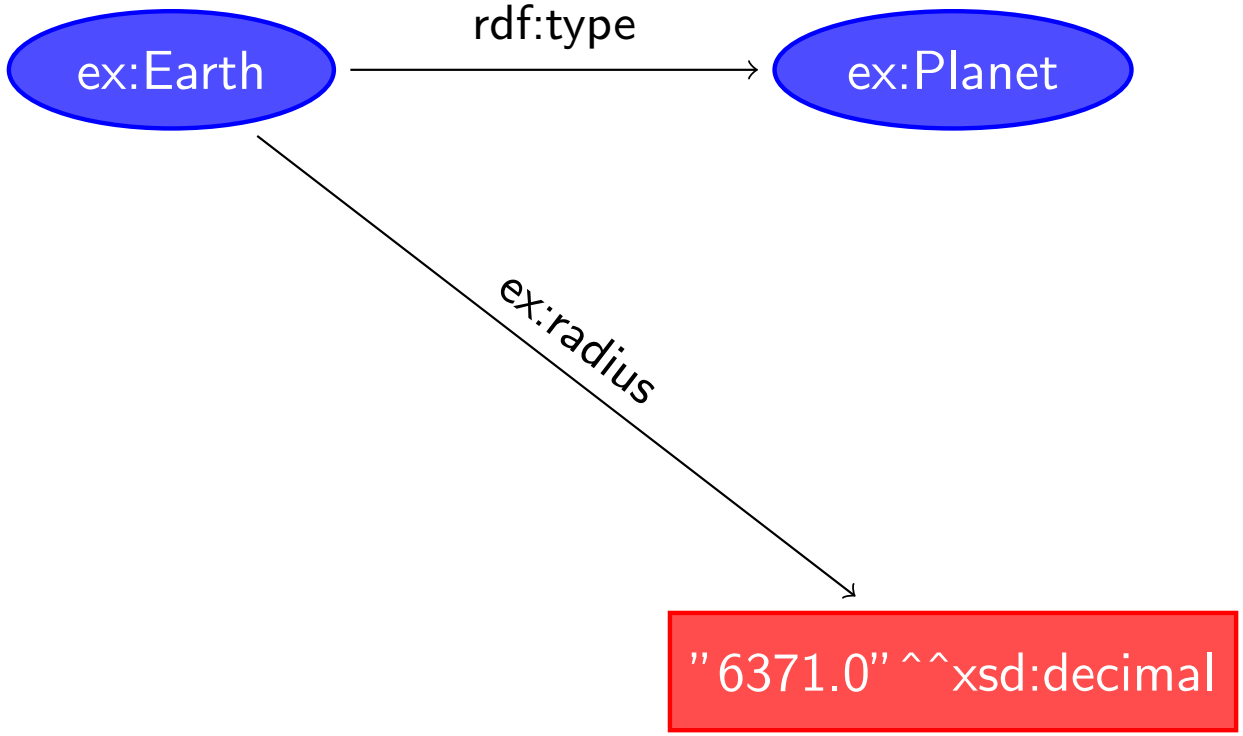
RDF Literal

- Datenwerte werden in RDF als Literale (*literals*) dargestellt
- Der Wert wird generell als Zeichenfolge dargestellt
- Zum Beispiel die Folge “6” “3” “7” “1” “.” “0”
- Die Interpretation der Zeichenfolge wird mittels Datentyp festgelegt
- Datentyp ist für Verständnis der beabsichtigten Bedeutung wichtig
- Er ist allerdings optional, untypisierte Literale sind möglich
- Diese werden immer als Zeichenfolge interpretiert
- Aus einem Literal können keine Kanten ausgehen
- Es ist somit nicht möglich Aussagen über Literale zu machen
- Literale können auch nicht als Namen für Kanten benutzt werden

RDF Literal: Beispiel



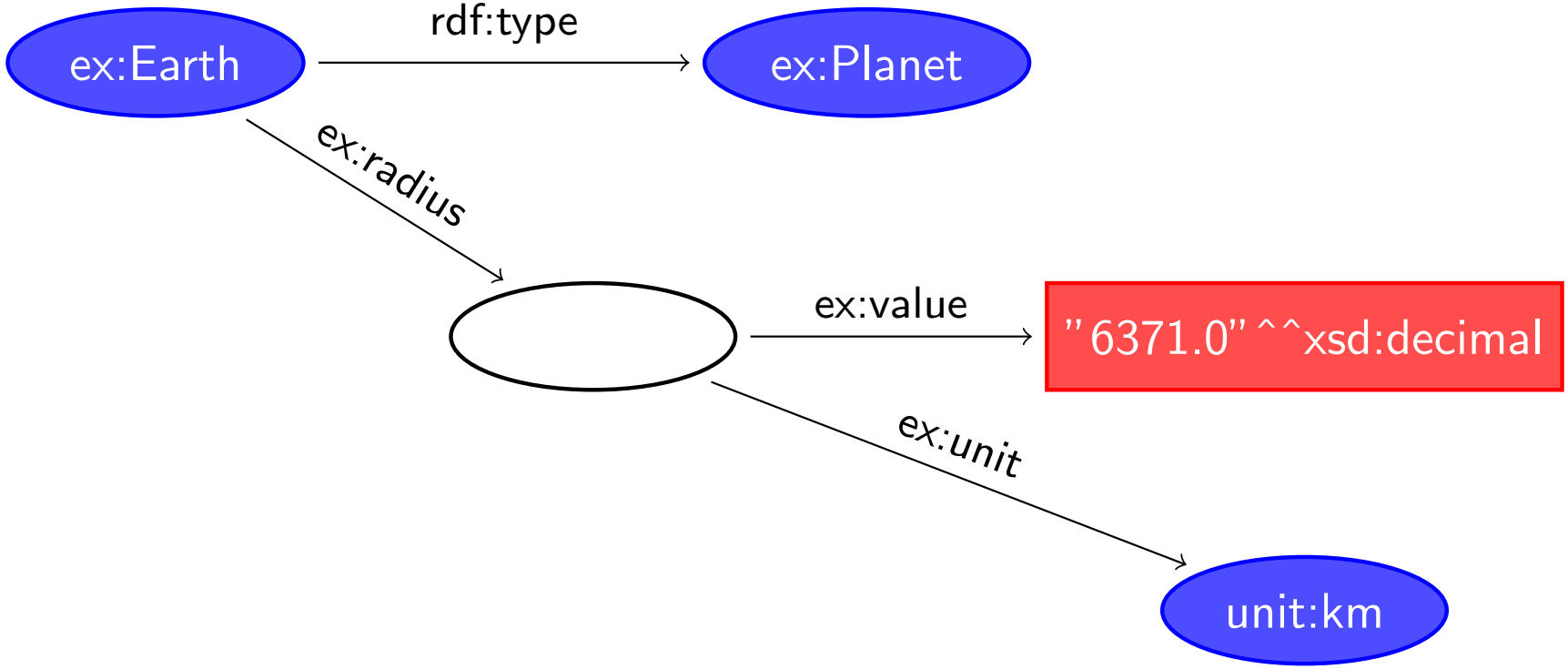
RDF Graph



RDF *Blank Node*

- Unbenannte Knoten die man nicht (global) referenzieren kann
- Die Knoten sind nicht mittels URI identifiziert
- Keine Ressourcen die man beschreiben möchte
- Nur strukturelle Funktion um mehrwertige Relationen darzustellen
- Kanten selbst können nicht unbenannt sein

RDF *Blank Node*: Beispiel



Graphen und Bäume

- XML erzeugt Bäume: Warum RDF Graphen?
- Graph Datenstruktur ist wesentlich flexibler
- Eine einfache Vereinigung zweier Graphen ist unproblematisch
- Diese dürfen getrennt sein
- Die einfache Vereinigung zweier Bäume ist kein Baum
- Man muss weitere Vorkehrungen treffen
- Zudem kann man beliebige Knoten einfach relationieren
- Bäume haben eine starre Struktur
- In Beziehung stehende Information kann im Baum entfernt sein

Zusammenfassung

- RDF ist eine Sprache zur Beschreibung strukturierter Information
- Ist ein grundlegener Baustein des *Semantic Web*
- Maschinelle Informationsverarbeitung, Daten und deren Bedeutung
- Das *statement* als zentrales RDF Konzept
- Eine *statement* Menge (RDF Dokument) spannt einen Graphen
- URI als wichtige Technologie zur Benennung von Knoten und Kanten
- Graphen sind flexiblere Datenstrukturen als Bäume