



# Kontext

# Modellierung in RDF

Patrick Sacher



# Kontext

# Modellierung in RDF

0. Eine kleine Einführung in  
RDF

# Was ist RDF ?

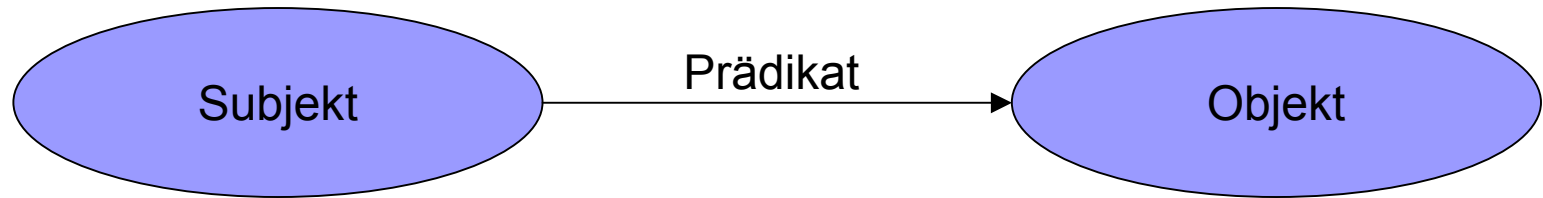
- RDF = Resource Description Framework
- Wird meistens in XML gecoded. 🤖💻
- RDF basiert auf der Idee, dass Dinge, die *beschrieben* werden sollen, *Merkmale* haben und diese Merkmale *Werte*.

# Terminologie

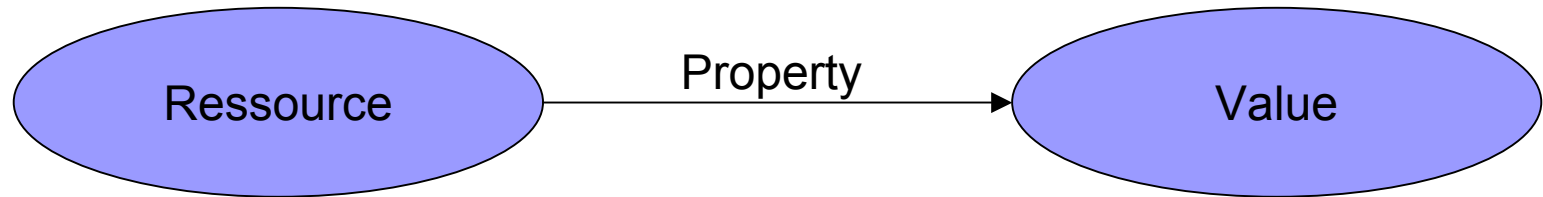
- **Ressourcen:** alles was man mit RDF-Ausdrücken beschreiben kann (z.B. eine Webseite, Teile einer Webseite, ein HTML-Element, ein Objekt, das man nicht via Web erreichen kann).  
Wird eindeutig durch eine URI (Uniforme Resource Identifier) identifiziert.
- **Property:** ein Attribut oder eine Beziehung, die für die Beschreibung einer Ressource benutzt wird.
- **Value:** der Wert des Statements.
- **Statement:**  
 $[Ressource] + [Merkmal] + [Wert]$

# Beispiele für Statements

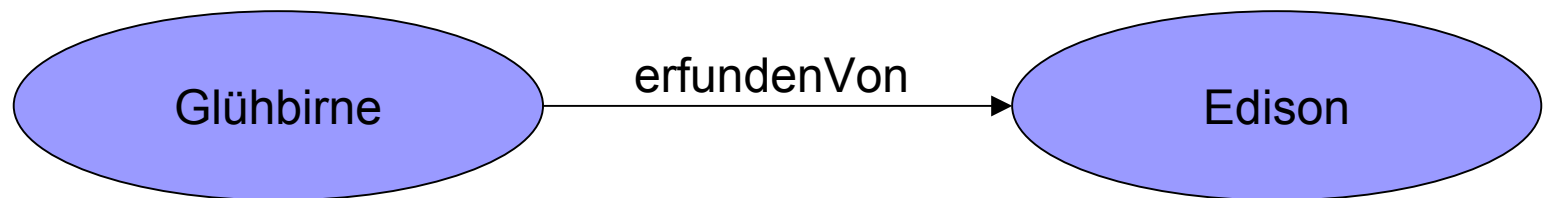
Statements kann man sich als Aussagen vorstellen:



In RDF würde das dann so aussehen:



Die Glühbirne wurde erfunden von Edison

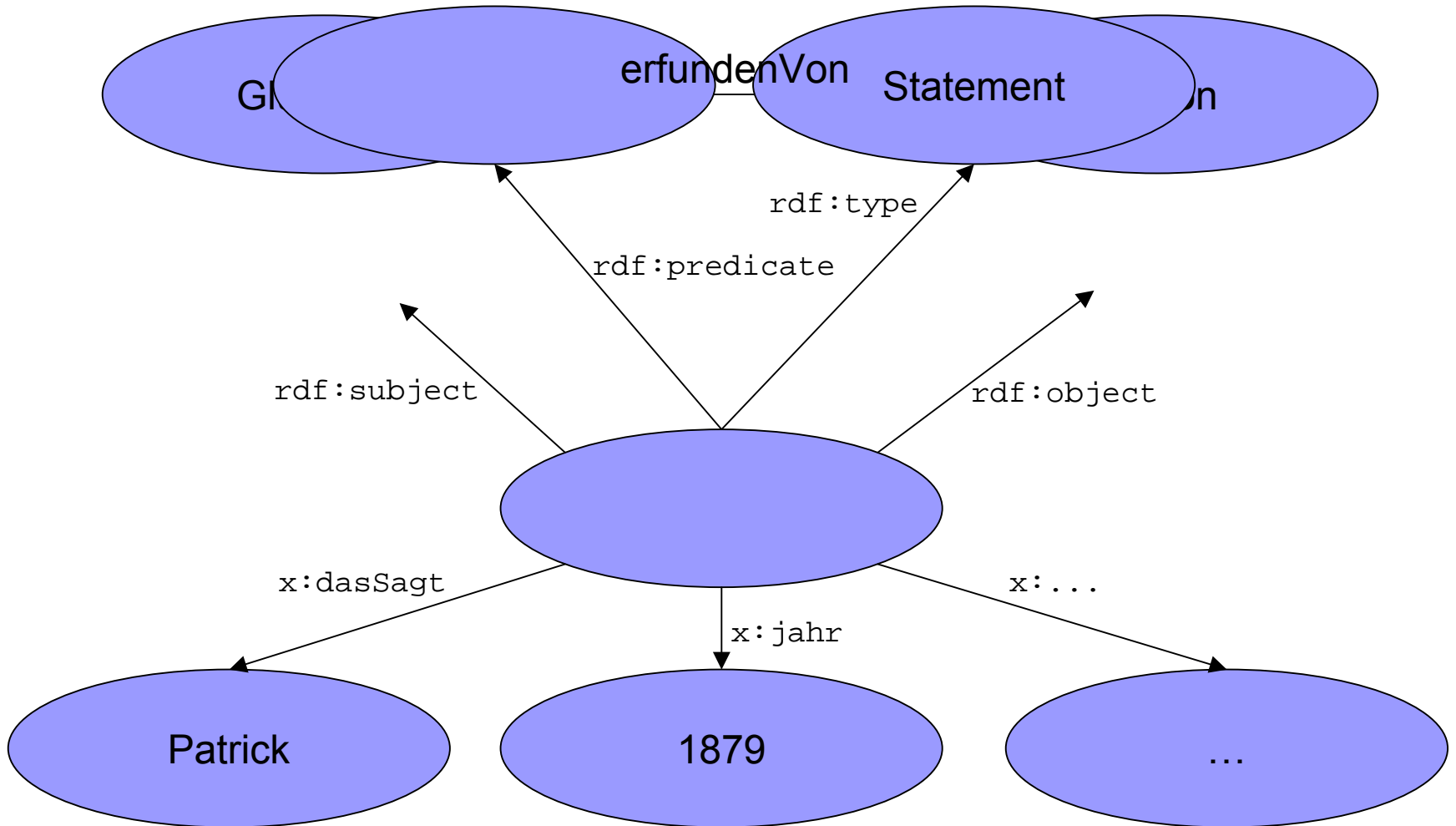


# Reified Statements

- Um Aussagen über Statements treffen zu können, müssen diese Statements zuerst zu Ressourcen umgewandelt werden. (*Reification*).
- Dadurch sind im Prinzip beliebig hohe Rekursionen möglich.

# Reified Statements (Beispiel)

Patrick sagt, dass die Glühbirne erfunden wurde.



# Grundlagen zur verwendeten RDF-Notation

- Eckige Klammern stellen immer eine Ressource dar,
- Texte in Hochkommas Werte,
- und die Pfeile werden als die jeweils verknüpfende Eigenschaft gewertet.

```
[SubjectName] --propertyName--> [ObjectName]
```

```
[SubjectName] --propertyName--> "Value"
```

Reified Statements werden in Quadrupeln angegeben.

```
StatementId: [Subject] --predicate--> [Object]
```

Die StatementID ist eine Art „Index“ mit der man das ReifiedStatement direkt ansprechen kann.


rdfo: Bezeichnet Klassen, die wir für die Benutzung von Kontexten einführen



# Kontext Modellierung in RDF

## 1. Einführung in Kontexte

# Was ist ein Kontext?


- Kontext ist ein Fremdwort aus dem Lateinischen mit der Bedeutung „*Zusammenhang*“.
- Die Vorsilbe Kon heißt auf deutsch „*zusammen*“, das Wort text heißt so etwas wie Textur, auf deutsch übersetzt „*das Gewebe*“.
- *contexo* = *zusammenweben* = *zusammensetzen*
- *contextus* = *verflochten* = *fortlaufend*

# Was erhoffen wir uns von Kontexten in RDF?

 **Modellieren von komplexen Zusammenhängen**  
(„Real-World-Situations“)

 Die **Zusammenhänge** zwischen verschiedenen Statements oder RDF-Dokumenten deutlich machen

 **Zusätzliche Informationen** wie z.B. Autor, Hintergrund, Herkunft hinzufügen zu können (für z.B. Trust-Modelling)

 Einen Rahmen (Framework) zu erstellen, bei dem auch einzelne **Ausnahmen** berücksichtigt werden können



# Kontext Modellierung in RDF

2. Klassenentwürfe für den  
Gebrauch von Kontexten in  
RDF

# Klassentwürfe für den Gebrauch von Kontexten in RDF

## Klassen:

rdfs:Container



rdfs:Set

## Eigenschaften:

rdfs:member

- rdfs:Container erfüllt nicht unsere Anforderungen.
- z.B. Kein Hinzufügen von Statements ohne Kenntnisse über den aktuellen Inhalt möglich.
- Wir müssen eine neue Klasse entwerfen.

- **rdfs:Set**: neue Klasse, die in der Lage ist Ressourcen zu sammeln.
- **rdfs:member**: Eigenschaft von rdfs:Set. Wird zum Hinzufügen einer Ressource in das Set benutzt.

# Klassentwürfe für den Gebrauch von Kontexten in RDF

## Klassen:

rdfs:Container



rdfc:Set



rdfc:StatementSet

## Eigenschaften:

rdfc:member



rdfc:quotes

- rdfs:Set arbeitet allerdings „nur“ mit Ressourcen. Wir benötigen aber ReifiedStatements.  
→ Wir brauchen eine explizit auf ReifiedStatements (unsere Form der RS) angepasste Klasse.

- **rdfc:StatementSet:** neue Klasse, die in der Lage ist ReifiedStatements zu sammeln.

- **rdfc:quotes:** Eigenschaft von rdfc:StatementSet. Wird zum Hinzufügen eines ReifiedStatements benutzt. Kann auch immer wieder auf das Statement durch dessen StatementID zugreifen.

# Klassentwürfe für den Gebrauch von Kontexten in RDF

## Klassen:

rdfs:Container



rdfs:Set



rdfs:StatementSet



rdfs:Context

## Eigenschaften:

rdfs:member



rdfs:quotes



rdfs:asserts

- **rdfs:StatementSet** kann zwar Statements sammeln, doch gehört zu einem Kontext noch mehr.  
→ Neue Klasse benötigt.

- **rdfs:Context**: ein StatementSet mit erweiterten Eigenschaften, (z.B. für logische Operationen).

- **rdfs:asserts**: eine Untereigenschaft von rdfs:quotes und wird benutzt um ReifiedStatements zum Kontext hinzuzufügen. Diese sind dann auch eindeutig wahr in diesem Kontext.



# Kontext Modellierung in RDF

## 3. Kontexte in RDF benutzen

# Kontexte als Container

## ■ Bisherige Vorgehensweise

```
S1: [MyPage] --author-----> "PatrickSacher"  
S2: [MyPage] --content-----> "Meine Meinungen"  
A1: [S1] -----assuredBy--> "Patrick Sacher"  
A2: [S1] -----assuredBy--> "Informatik-Student"  
A3: [S2] -----assuredBy--> "Patrick Sacher"  
A4: [S2] -----assuredBy--> "Informatik-Student"
```

Das „Problem“ ist, dass wir für jedes Statement die Eigenschaften „per Hand“ verknüpfen müssen.

## ■ Vorgehensweise mit Kontexten

```
S1: [MyPage] ----author-----> "Patrick Sacher"  
S2: [MyPage] ----content-----> "Meine Meinungen"  
S3: [PSContext] -rdfc:asserts----> [S1] }  
S4: [PSContext] -rdfc:asserts----> [S2] }  
S5: [PSContext] -rdfc:applyToAll-> [_ab] }  
S5: [_ab] -----assuredBy-----> "Patrick Sacher"  
S6: [_ab] -----assuredBy-----> "Informatik-Student"
```

- Statements zum Kontext hinzufügen.
- [Kontext] –applyToAll-> [Variable]
- [Variable] mit Allem verknüpfen.

→ Eigenschaften der Variable werden durch applyToAll auf alle Statements im Kontext übertragen.

Wir können leichter zusätzliche Informationen hinzufügen.  
Nützlich für z.B. Trust-Modelling,...

# Kontexte mit logischen Operationen

## Klassen:

rdfs:Container



rdfc:Set



rdfc:StatementSet



rdfc:Context

## Eigenschaften:

rdfc:member



rdfc:quotes



rdfc:asserts

rdfc:assumes

- Um Kontexte mit logischen Funktionen ausstatten zu können, führen wir eine neue Eigenschaft ein.

- **rdfc:assumes**: eine Eigenschaft von rdfc:Context
- Vergleichbar mit einem „wenn“.

# Kontexte mit logischen Operationen (Beispiel)

```
S1: [Holmes(1)] --rdf:type--> [Person]
S2: [Holmes(1)] --surname--> "Holmes"
S3: [Holmes(1)] --isa-----> "detective"
S4: [Holmes(1)] --isa-----> "justice"
S5: [Holmes(2)] --isa-----> "detective"
S6: [Holmes(2)] --does-----> "solve crimes"
S7: [Holmes(2)] --pay-----> "low"
S8: [Holmes(3)] --isa-----> "justice"
S9: [Holmes(3)] --does-----> "sit in court"
S10: [Holmes(3)] --pay-----> "high"

C11: [Holmes(fiktiv)] --rdf:type-----> [rdcf:Context]
C12: [Holmes(fiktiv)] --rdcf:asserts--> [S1]
C13: [Holmes(fiktiv)] --rdcf:asserts--> [S2]
C14: [Holmes(fiktiv)] --rdcf:asserts--> [S3]
C15: [Holmes(fiktiv)] --rdcf:assumes--> [Entscheidung]

C21: [Holmes(real)] ----rdf:type-----> [rdcf:Context]
C22: [Holmes(real)] ----rdcf:asserts--> [S1]
C23: [Holmes(real)] ----rdcf:asserts--> [S2]
C24: [Holmes(real)] ----rdcf:asserts--> [S4]
C25: [Holmes(real)] ----rdcf:assumes--> [Entscheidung]

C31: [Entscheidung] --rdf:type-----> [rdcf:Context]
C32: [Entscheidung] --rdcf:assumes--> [S5]
C33: [Entscheidung] --rdcf:assumes--> [S6]
C34: [Entscheidung] --rdcf:asserts--> [S7]
C35: [Entscheidung] --rdcf:assumes--> [S8]
C36: [Entscheidung] --rdcf:assumes--> [S9]
C37: [Entscheidung] --rdcf:asserts--> [S10]
```



[S1-S10] - Holmes war Detektiv und Richter. Der Detektiv wird schlecht bezahlt, wogegen der Richter gut bezahlt wird. Diese beiden Holmes könnten also nicht ohne Widerspruch verbunden werden.

[C11-C15] - Der fiktive Sherlock Holmes wird mit der zu ihm passenden Eigenschaften verknüpft. Weitere Eigenschaften (die der Bezahlung) werden durch assumes mit einem eigenen Kontext verknüpft, der Regeln für diesen Widerspruch beinhaltet (dort wird dann entschieden welche Bezahlung diesem Holmes zusteht).

[C21-C25] - Das gleiche wie C11-C15 nur für den realen Holmes

[C31-C37] - Der „Entscheidungs-Kontext“, der die nötigen Regeln für die Bezahlung enthält. Wenn [S5] und [S6] vorhanden ist, dann gilt [S7]. Wenn [S8] und [S9] vorhanden ist, dann gilt [S10].

Wir können logische „Abfragen“ erstellen und untersuchen,

ob Informationen zusammenschließen



# Kontext Modellierung in RDF

## 4. Weiterführende Verwendungen von Kontexten

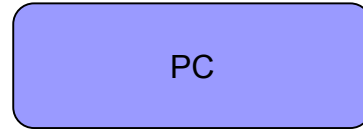
# Kontexte als Ressourcen

 Das Modellieren von komplexen Zusammenhängen wird im „normalen“ RDF zu unübersichtlich

 Durch die Idee der Kontexte als Container (mit Eigenschaften und Beschränkungen) wäre es möglich Ordnung zu schaffen

 Kontexte können auch Kontexte enthalten.

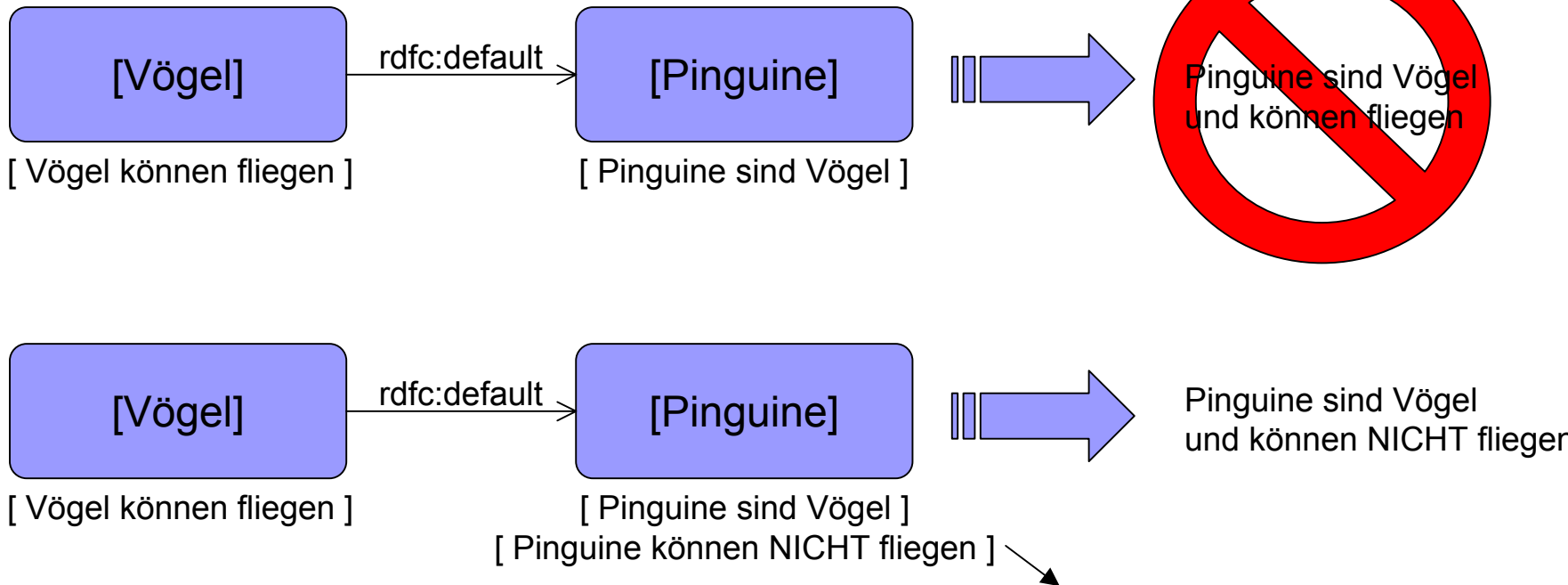
# Kontexte als Ressourcen (Beispiel)



→ Kontexte können RDF übersichtlicher machen, man kann Informationen „Stückchenweise“ betrachten.

# Default Kontexte

Default Kontexts zu betrachten, geschieht mit der Motivation, dass man eine Basis für das *Überschreiben oder Verändern von Statements*, die man vorher als wahr angesehen hat, erhalten kann (wie z.B. bei Ausnahmen).



➔ Kontexte können Ausnahmen schaffen.

Diese Aussage überschreibt die Aussage aus dem Kontext Vögel, da sie näher an dem Objekt (Pinguin) dran ist

ENDE!

