

SPARQL演習

トルヴェー アントワン （九州大学システム情報科学大学院 助教）
後藤 孝行 （九州大学共進化社会システム創成拠点 学術研究員）

2015年2月5日

SRPビル2階

連絡先：teapot-admin@isit.or.jp

本日のスケジュール

背景：セマンティックWebおよびオープンデータ（約10分）

本題1：RDFの紹介（約30分）

本題2：SPARQLの一般紹介（約15分）

休憩：10分

演習1：SPARQLを使ってみよう（約1時間）

演習2：初めてのSPARQLを使ったWebアプリ（約45分）

休憩：10分

演習3：地図アプリを作しましょう（約1時間）

始める前に . . .

ホームページ: bodik.jp

OPENDATA CONTEST 2014-2015 FUKUOKA

オープンデータコンテスト 2014-2015 福岡

オープンデータでみんなをハッピーに!

応募受付開始日	2014年 12月22日(月)
応募締切日	2015年 2月23日(月)
表彰式	2015年 3月6日(金)
募集部門	アプリケーション部門 アイデア部門

応募フォームはこちら >

ホーム | 開催概要 | 実証概要 | 応募方法 | 応募規約 | 審査・表彰 | 受賞作品 | お問い合わせ

お知らせ

2014.12.22 「オープンデータ・コンテスト2014-2015 Fukuoka」 サイトオープン

開催概要

オープンデータコンテスト2014-2015 Fukuokaの開催概要です。

実証概要

今年度の実証実験の概要をご案内いたします。

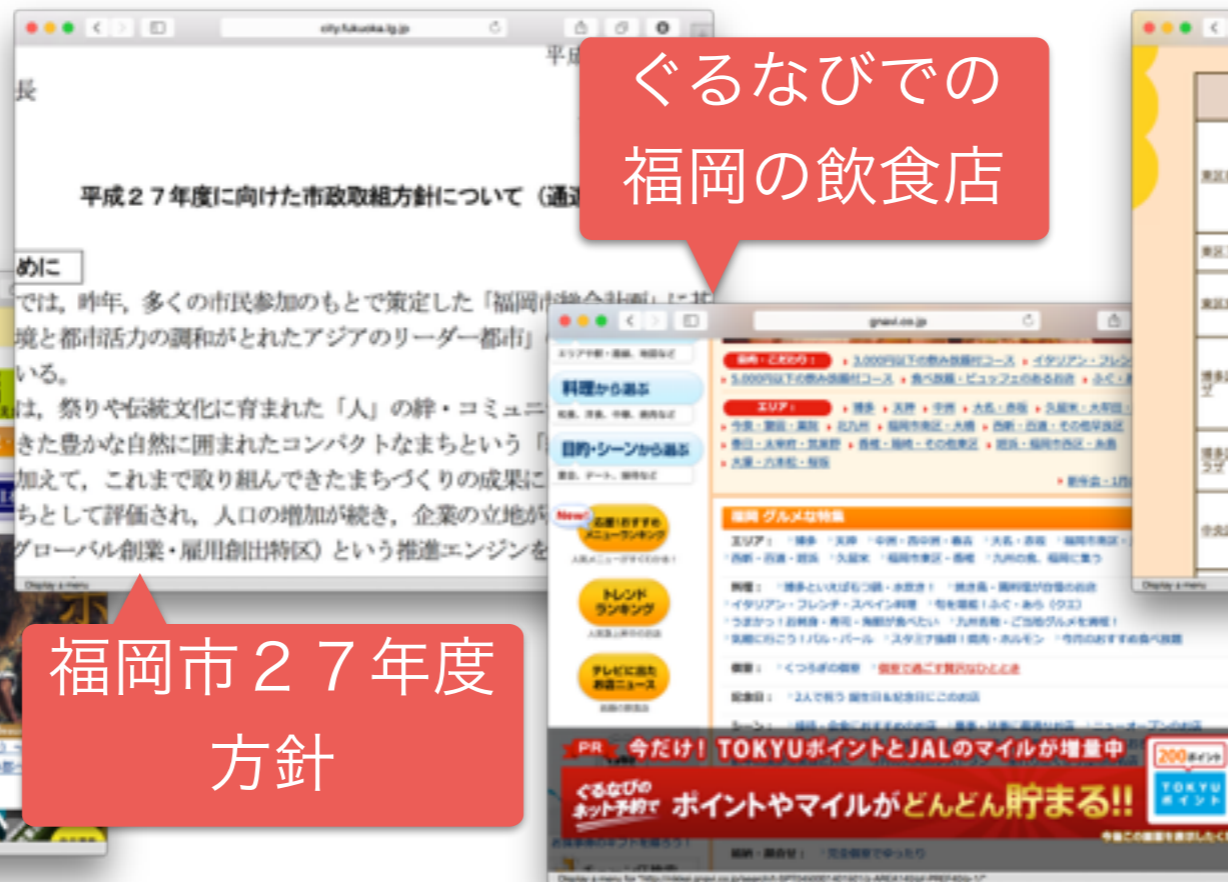
背景セマンティックWeb について

既存のWebの課題：検索

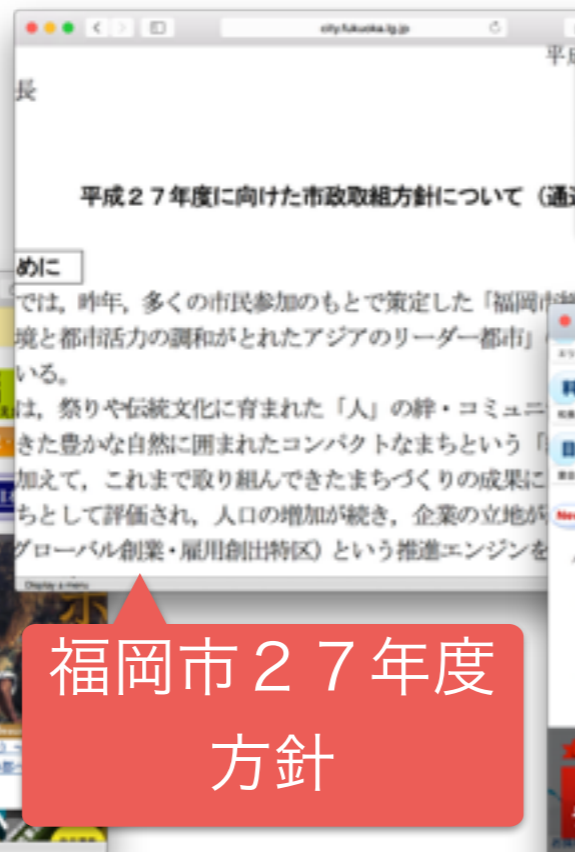
福岡市ホームページ



ぐるなびでの
福岡の飲食店



福岡市27年度
方針



福岡市のこども
プラザ一覧

施設名	住所	電話・FAX 番号	休館日	アクセス	地図
東区東区子どもプラザ	福岡市東区東区2-1-1 セビリアス西鉄東区 2階	092-660-3063	毎週月曜、毎月 第3日曜	西鉄東区線「西鉄東区」 下車徒歩5分。西鉄バス 「東区」下車徒歩 5分。JR東区線「東区」 下車徒歩5分。	
東区三宮子どもプラザ	東区三宮5-40-1階	092-660-6267	毎週水曜、毎月 第2日曜	西鉄東区線「三宮駅」下 車徒歩5分	
東区東区子どもプラザ	東区東区1-1-1 ゆめタウン博多 2階	092-260-0320	毎週水曜、毎月 第3日曜	西鉄バス「東区三丁」下 車徒歩1分	
博多区山王子どもプラザ	博多区山王 5-10-11 博多区山王 センター併設	092-472-6006	毎週日曜、毎月 第1、3、5日曜 、第1、3、5日曜 、第1、3、5日曜	西鉄バス「山王台」下 車徒歩5分。西鉄バス 「山王」下車徒歩 5分。	
博多区博多子どもプラザ	博多区博多 1-4-11-1 階	092-660-6711	毎週水曜、毎月 第4土曜	西鉄バス「博多駅前」 下車徒歩5分。西鉄バス 「博多」下車徒歩 5分。	
中央区子どもプラザ	中央区東区 2-2-4	092-741-2664	毎週月曜（祝日 の場合は第2日 曜）、毎月第1（日 曜の場合は第2日 曜）、12月26日	地下鉄東区線「東区」下 車徒歩5分。西鉄バス 「東区二丁目」下車 徒歩5分。	

- ・ インターネットは膨大な情報が閲覧可能
- ・ が、正規化・構造化されていないため、
検索は困難

セマンティックWebによる 解決策

ページやページの中にあるエレメントに
属性情報を付加する

属性情報等は検索可能とする

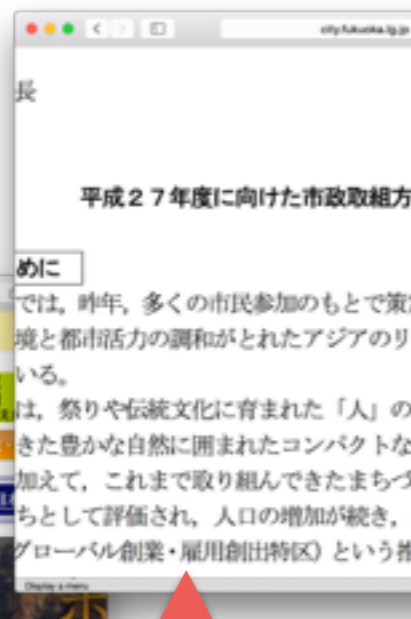
標準技術を提案する

ページ属性情報

about: 福岡市
type: ホームページ
lastupdate: 2015-2-1



about: 福岡市
type: 市政方針資料
fiscal year: 2015



about: 福岡市
type: 飲食店一覧
lastupdate: 2015-2-5



about: 福岡市
type: こどもプラザ一覧
type: 子育て支援情報

施設名	住所	電話・FAX 番号	休館日	アクセス	画像	地図
東区東区子どもプラザ	福岡市東区 東区2-1-1 セビオビル 2階	092-463-3263	毎週月曜、毎月 第3日曜	西鉄東横線「西鉄東横」 下車徒歩5分、西鉄バス 「西鉄東横」下車徒歩 3分、西鉄バス「西鉄東横」 下車徒歩3分		
東区三ツ子子どもプラザ	東区三ツ子 5-40-1	092-462-4267	毎週水曜、毎月 第2日曜	西鉄東横線「三ツ子」下 車徒歩5分		
東区東区子どもプラザ	東区東区1- 1-1 中の タウンビル 2階	092-292-5329	毎週水曜、毎月 第3日曜	西鉄バス「東区三ツ子」 下車徒歩1分		
博多区山王子どもプラザ	博多区山王 5-13-10 博多区山王 センタービル	092-472-4006	毎週日曜、毎月 第1土曜、第 3土曜、第5 土曜、第7土 曜、第9土曜 (12月28日)	西鉄バス「山王台」下 車徒歩5分、西鉄バス 「山王台」下車徒歩 5分		
博多区博多子どもプラザ	博多区博多 1-4-11 1 階	092-562-9711	毎週金曜、毎月 第4土曜	西鉄バス「博多駅前」 下車徒歩5分、西鉄バス 「博多駅前」下車徒歩 5分		
中央区子どもプラザ	中央区東区 2-2-4	092-741-3554	毎週月曜(祝日 の場合は翌日)、 毎月第1日曜 (12月28日)	地下鉄有明線「東区」下 車徒歩5分、西鉄バス 「東区二丁目」下車 徒歩5分		

ページエレメント属性情報

about: 福岡市
type: ホームページ
lastupdate: 2015-2-1



about: 福岡市
type: 市政方針資料
fiscal year: 2015



about: 福岡市
type: 飲食店一覧
lastupdate: 2015-2-5



施設名	住所	電話・FAX 番号	休館日	アクセス	画像	地図
東区香椎子どもプラザ	福岡市東区香椎駅前2-5-2-1 セピアテラス西鉄香椎2階	092-663-3263	毎週月曜、毎月第3日曜	西鉄貝塚線「西鉄香椎」下車徒歩すぐ、西鉄バス「西鉄香椎」下車徒歩すぐ、西鉄バス「香椎」下車徒歩3〜5分、JR鹿児島本線・香椎線「香椎駅」下車徒歩3分		
東区三苦子どもプラザ	東区三苦5-1-40 1階	092-692-6267	毎週水曜、毎月第2日曜	西鉄貝塚線「三苦駅」下車徒歩5分		
東区山王子どもプラザ	東区東浜1-1-1 ゆめタウン博多2階	092-292-5320	毎週木曜、毎月第3金曜	西鉄バス「東出三丁目」下車徒歩1分		
博多区山王子どもプラザ	博多区山王1-13-10 博多市民センター併設	092-472-6006	毎週日曜、毎月末日（土・日曜、祝日のときは、直後の土・日曜、祝日の場合は開館）、12月28日	西鉄バス「山王公園前」「山王一丁目」下車徒歩5分		
博多区博多南子どもプラザ	博多区竹丘1-4-11 1階	092-592-9711	毎週金曜、毎月第4土曜	西鉄バス「JR南福岡駅」下車徒歩6分、JR鹿児島本線「南福岡駅」下車徒歩6分		
中央区子どもプラザ	中央区長浜2-2-4	092-741-3564	毎週月曜（祝日の場合は翌日）、毎月末日（日曜の場合は開館）、12月28日	地下鉄空港線「赤坂」下車徒歩6分、西鉄バス「長浜二丁目」下車徒歩すぐ		

about: 福岡市
type: こどもプラザ一覧
type: 子育て支援情報

東区三苦子どもプラザ	東区三苦5-1-40 1階	092-692-6267	毎週水曜、毎月第2日曜	西鉄貝塚線「三苦駅」下車徒歩5分		
------------	---------------	--------------	-------------	------------------	--	--

about: 東区東浜子どもプラザ
type: こどもプラザ
type: 子育て支援情報
address: 東区東浜1-1-1ゆめタウン博多2階
telephone: 092-292-5320

更に！



団体（自治体等）、各個人が生成・蓄積する膨大なデータをWebで公開・構造化したら、**より便利な社会へ！**

しかし

データの利用条件（ライセンス）を明確する必要がある
オープンデータ

データ間の関係性を表さないと探索が困難
リンクトオープンデータ（LOD）

オープンデータ級



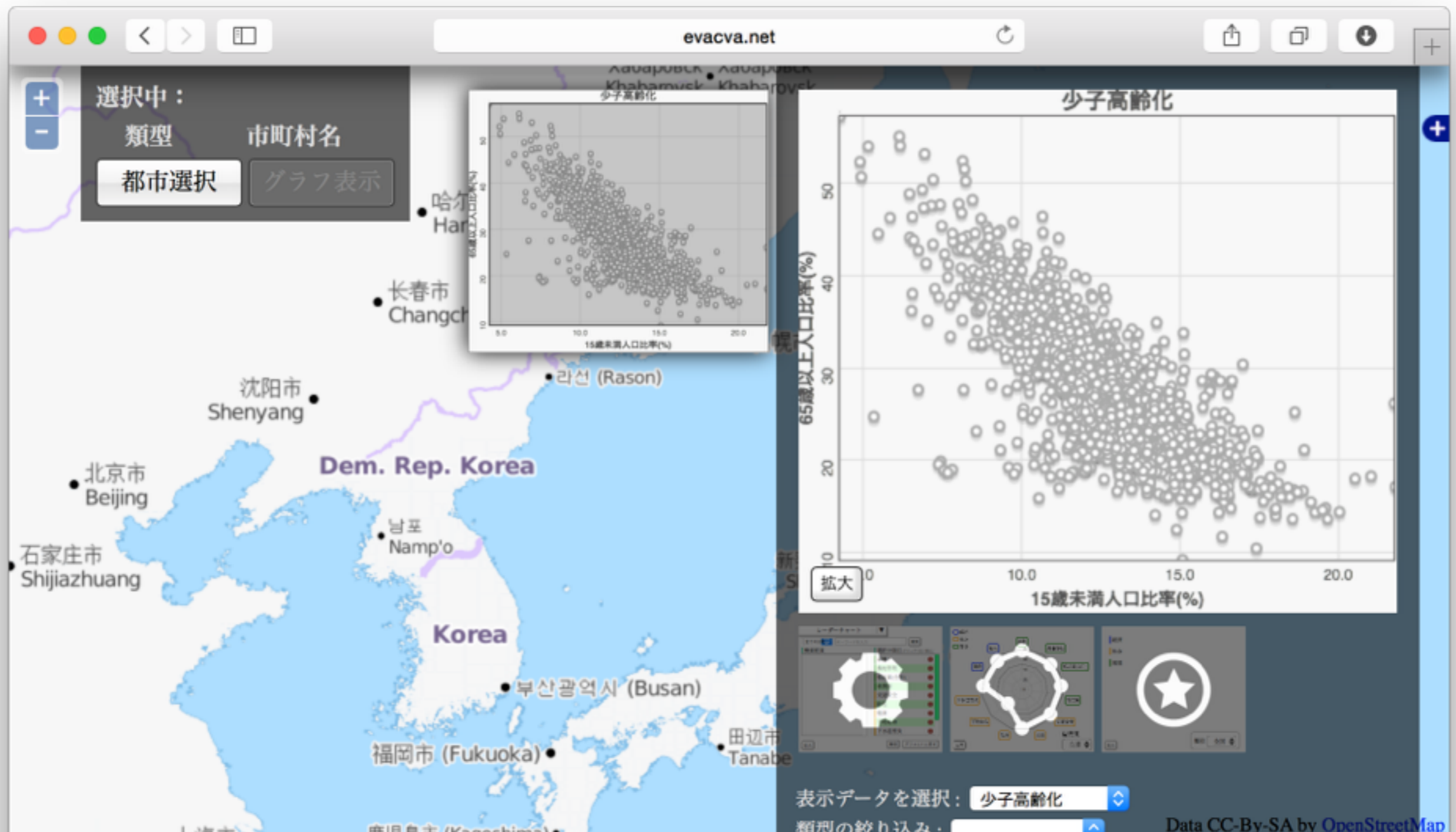
二次利用可能
ライセンス

- ★オープンライセンス
- ★★構造化データ
- ★★★オープン形式 エクセルよりも、CSV等
- ★★★★RDFを利用
- ★★★★★リンクRDFを利用

事例 1 : かなざわ育なび



事例 2 : Evacva



本題 1 RDFについて

セマンティックWebの データモデル

Resource Description Framework

リソース (=物)

記述

枠組み

W3Cはインターネットで使われている技術（HTML、Javascript、CSS等）を標準化する団体。
1994年に結成されました。



- **W3C**の標準技術
- 複数の標準技術の集合（抽象的なセマンティックス、テキスト表現等）
- 現在のバージョンは**1.1**です

RDF1.1データモデル(1)

トリプル=物3つ

ちなみに、カープル=物2つ/シングルトン=物1つ

全てデータは**トリプル**で表現する

主語

例：福岡

述語

=プロパティ名

例：人口

目的語

=プロパティ値

例：1521881人

RDF1.1データモデル(2)

主語

型：IRI

述語

型：IRI

目的語

型：IRI又は
リテラル

- IRIとは？
 - International（国際） Resource（リソース） Identifier（識別子）
 - 人物を指すユニークな識別子
 - URLはIRIの例です（Webサイトを指す）
- リテラルとは？
 - 単なる文字列
 - RDFではタイプや言語を付けることもできる

RDFモデルのサンプル

主語

述語

目的語

型・言語

city.fukuoka.lg.jp

rdf:type

schema:website

city.fukuoka.lg.jp

schema:about

Fukuoka City

en

city.fukuoka.lg.jp

schema:dateVisited

2015-2-5

xsd:date



さっきの福岡市ホームページ

RDFのグラフ形式

主語

述語

目的語

型・言語

city.fukuoka.lg.jp

rdf:type

schema:website

city.fukuoka.lg.jp

schema:about

Fukuoka City

ja

city.fukuoka.lg.jp

schema:dateVisited

2015-2-5

xsd:date



city.fukuoka.lg.jp

rdf:type

schema:website

schema:about

"Fukuoka City"@en

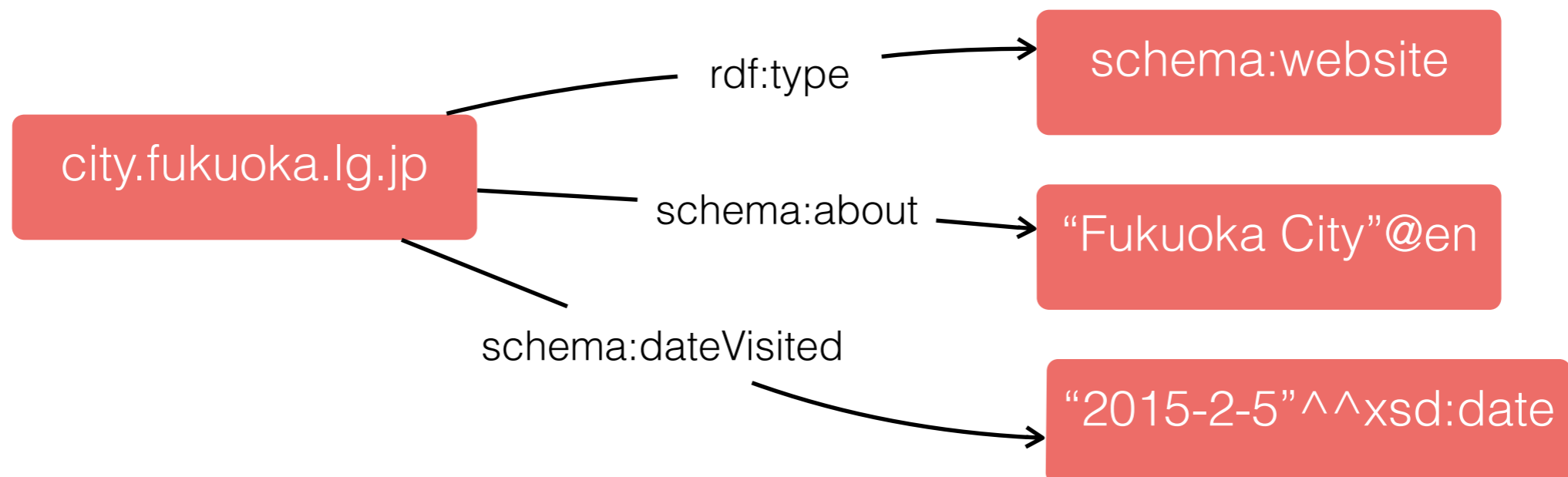
schema:dateVisited

"2015-2-5"^^xsd:date

RDFのグラフ形式

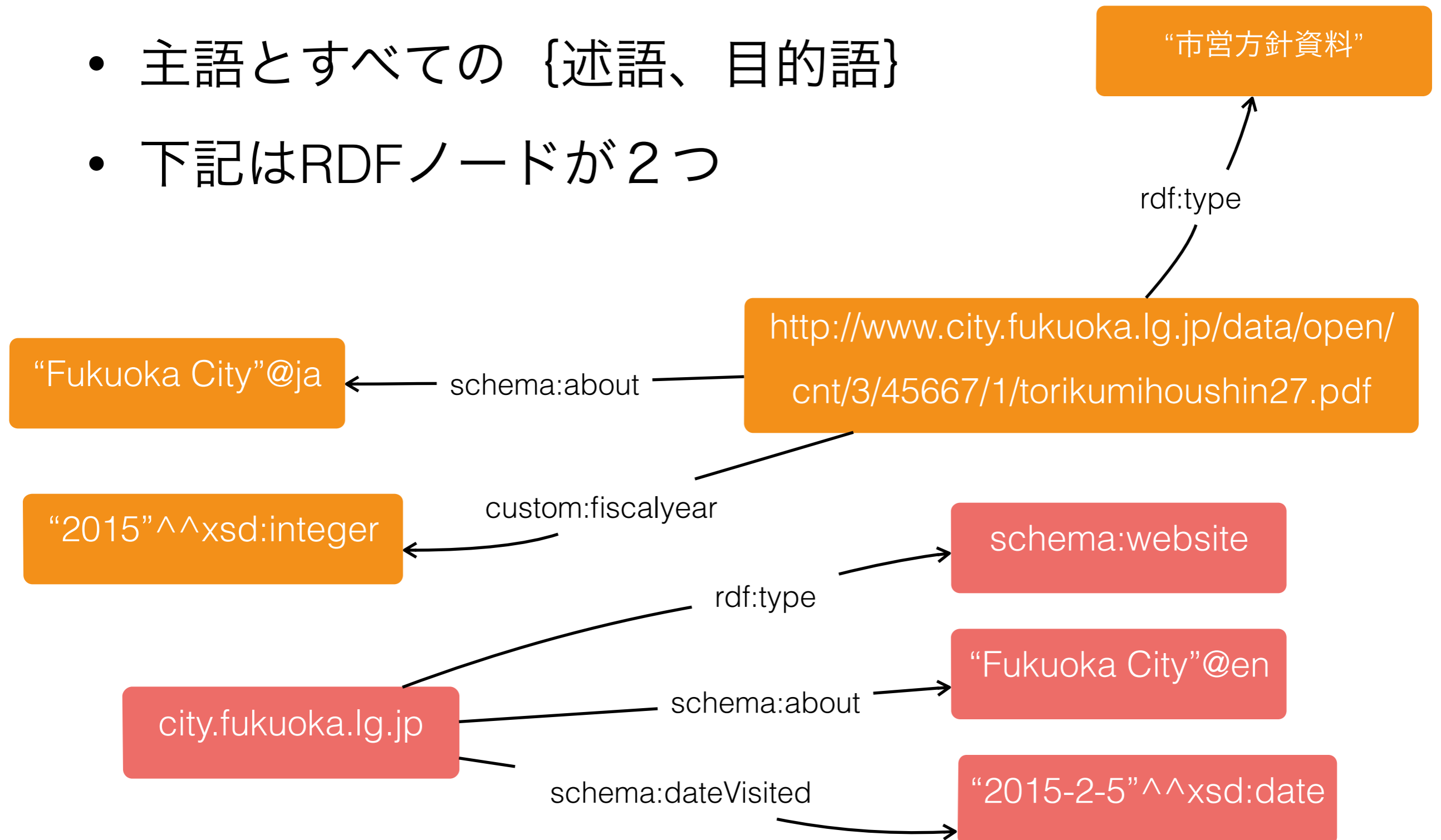
- 主語と目的語を矢印で繋ぐ
- 矢印は述語を表す（ラベル付き）
- 目的語の型や言語は下記のシンタックスで表す
 - 言語：“XXX”@lang
 - 型：“XXX”^^type

RDF/Turtle
シンタックス
(後で説明する)



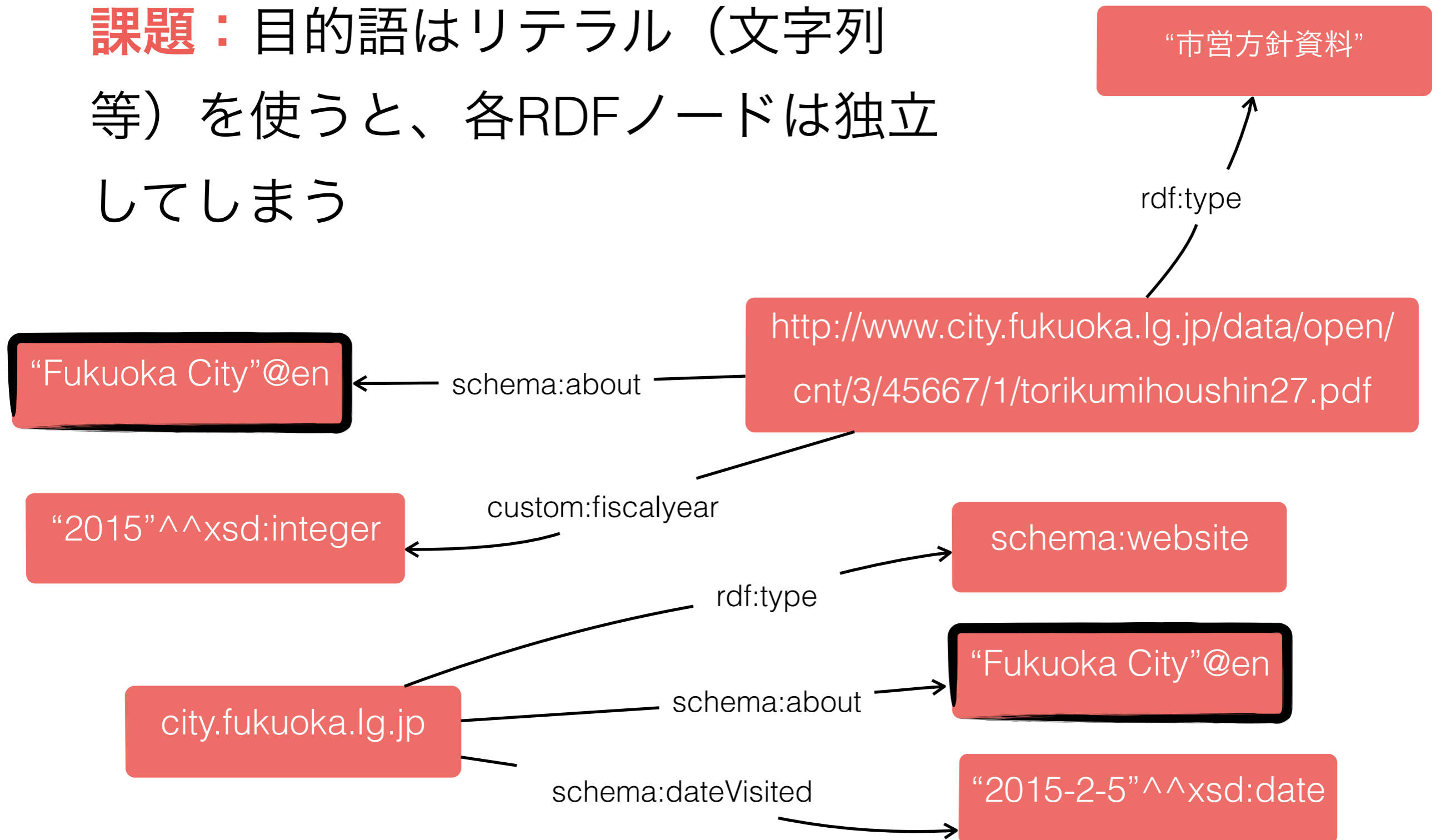
RDFノードとは

- 主語とすべての {述語、目的語}
- 下記はRDFノードが2つ



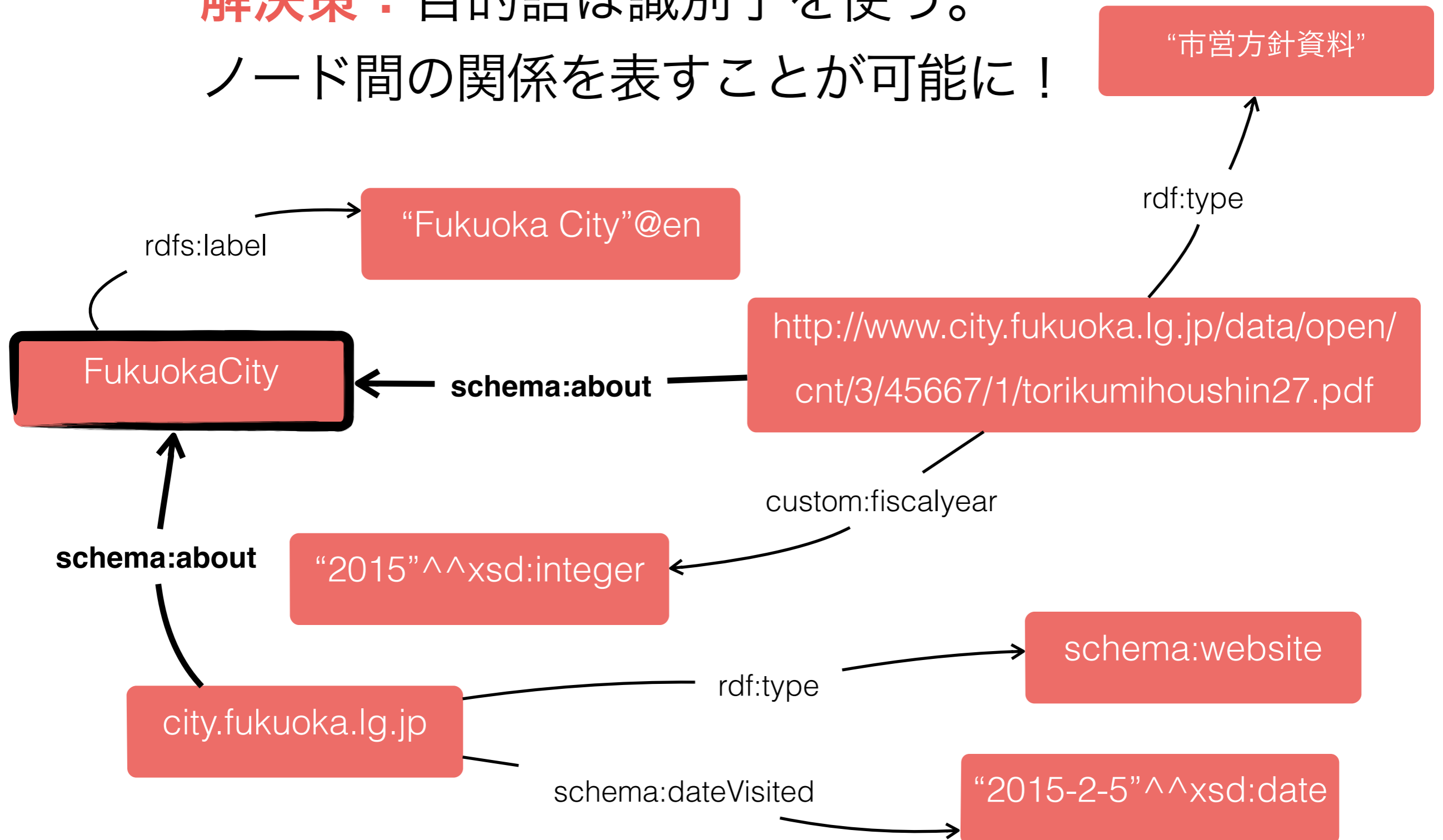
リンクRDF(1)

課題： 目的語はリテラル（文字列等）を使うと、各RDFノードは独立してしまう



リンクRDF(2)

解決策：目的語は識別子を使う。
ノード間の関係を表すことが可能に！



語彙について

- RDFデータの使っている述語等は出来るだけ統一する必要がある
 - データベース間にリンクを貼るために
 - データベースを探索ために
- よく使われている語彙セットがあります
 - W3Cの標準語彙（少ない！）：RDF語彙集合、RDFスキーマ（RDFS）語彙集合
 - その他：Friend of a friend、Schema.org、Geo
- 人気語彙セットのランキング：<http://www.prefix.cc>
 - 名前空間（接頭辞）で整理する



自分のパソコンでサイトを見
ても大丈夫ですよ！

語彙集合サンプル：RDFS

「RDFS」語彙セット

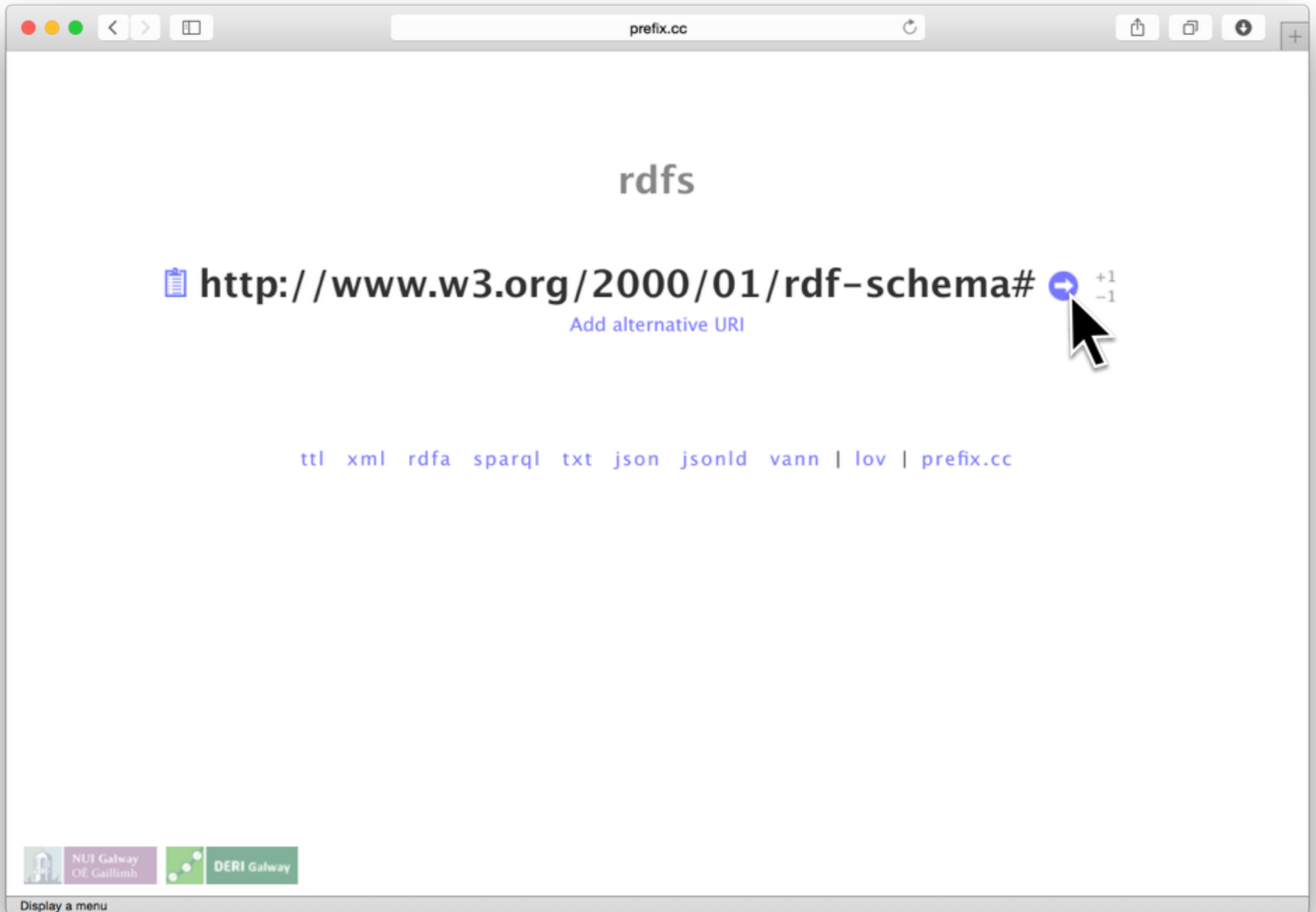
- RDFスキーマ
- W3C標準語彙
- RDFデータベースの構造を記述するための語彙

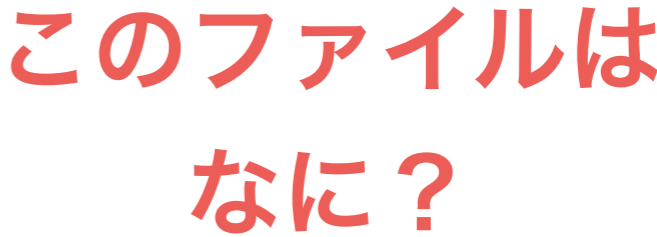
RDFデータベースのスキーマはRDFで記述する

popular

1. yago
2. rdf
3. foaf
4. dbp
5. dc
6. owl
7. rdfs
8. ont
9. dbo
10. onto
11. skos
12. geo
13. rss
14. gldp
15. sioc
16. sc
17. fb
18. geonames
19. xsd
20. gr
21. dcterms
22. dct
23. dbpedia
24. akt
25. org
26. commerce

prefix.ccによると人気
ランキングは7位
(2015/2/2の時点)





RDFテキスト形式

- ここまでの発表ではRDFを抽象的にしか扱ってない
- が、RDFデータはテキスト形式で格納する必要がある
- 標準フォーマットはRDF1.1規格に記載されている：
 - RDF/Json
 - RDF/XML
 - RDF/Turtle（拡張子：ttl）

さっきのファイル
の形式

Turtleシンタックス(1)

SPARQLで使います
ので、覚えましょう！

- トリプルを効率よく表現できる
 - 特にRDF/XMLよりも文字数がすくなく、読みやすい
- トリプルをそのまま書きます（順序：主語、述語、目的語）
 - ピリオドで区切る「。」
 - リテラルはダブルクォートで区切る「"”」 / IRIは「<>」で区切る
 - 言語やリテラル型は「@」と「^^」を利用する（RDFグラフと同様）

```
<http://bodic.org/subject1> <http://www.w3.org/2000/01/rdf-schema#label> "名前"@ja.  
<http://bodic.org/subject1> <http://www.w3.org/2000/01/rdf-schema#label> "name"@en.  
<http://bodic.org/subject1>  
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://bodic.org/people>.
```

- 「PREFIX」キーワードで名前空間を定義し、IRIを短く書ける

```
PREFIX bodic: <http://bodic.org/>  
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>  
bodic:subject1 rdfs:label "名前"@ja.  
bodic:subject1 rdfs:label "name"@en.  
bodic:subject1 rdf:type bodic:people.
```

名前空間を使うと「<...>」
が不要になる

Turtleシンタックス(2)

- 主語又は述語を結合し、更に短く書ける！
 - 主語の合同：セミコロンで {述語、目的語} を区切る
 - 述語の合同：コンマで目的語を区切る

```
PREFIX bodic: <http://bodic.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
bodic:subject1 rdfs:label "名前"@ja.
bodic:subject1 rdfs:label "name"@en.
bodic:subject1 rdf:type bodic:people.
```



```
PREFIX bodic: <http://bodic.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
bodic:subject1
  rdfs:label "名前"@ja,"name"@en;
  rdf:type bodic:people.
```

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

<http://www.w3.org/2000/01/rdf-schema#> a owl:Ontology ;
    dc:title "The RDF Schema vocabulary (RDFS)" .

rdfs:Resource a rdfs:Class ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "Resource" ;
    rdfs:comment "The class resource, everything." .

rdfs:Class a rdfs:Class ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "Class" ;
    rdfs:comment "The class of classes." ;
    rdfs:subClassOf rdfs:Resource .

rdfs:subClassOf a rdf:Property ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "subClassOf" ;
    rdfs:comment "The subject is a subclass of a class." ;
    rdfs:range rdfs:Class ;
    rdfs:domain rdfs:Class .

rdfs:subPropertyOf a rdf:Property ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "subPropertyOf" ;
    rdfs:comment "The subject is a subproperty of a property." ;
    rdfs:range rdf:Property ;
    rdfs:domain rdf:Property .

rdfs:comment a rdf:Property ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "comment" ;
    rdfs:comment "A description of the subject resource." ;
    rdfs:domain rdfs:Resource ;
    rdfs:range rdfs:Literal .

rdfs:label a rdf:Property ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "label" ;
    rdfs:comment "A human-readable name for the subject." ;
    rdfs:domain rdfs:Resource ;
    rdfs:range rdfs:Literal .
```

例：rdfs:label の定義

rdfs:labelという述語を
定義するRDFノード


```
rdfs:label a rdf:Property ;
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
    rdfs:label "label" ;
    rdfs:comment "A human-readable name for the subject." ;
    rdfs:domain rdfs:Resource ;
    rdfs:range rdfs:Literal .
```

rdfs:label (<http://www.w3.org/2000/01/rdf-schema#label>)
名前を指定するような標準的な述語。

RDFノード型について

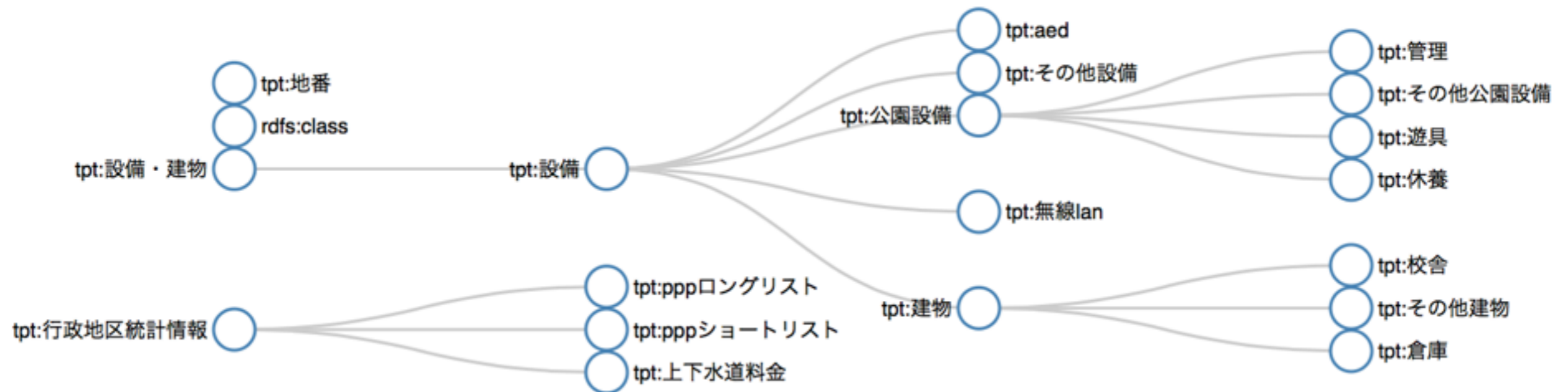
「a」キーワード又は「rdf:type」でノード型を指定

- Turtle 「a」キーワードは「rdf:type」述語と同義
- RDFスキーマ語彙を使うと、ノード型の親子関係および利用可能述語を定義可能
- RDFで、データと同じデータベースの中にスキーマを定義するとのことです
- スキーマを守るのはデータ登録者の責任である（トリプルストアは確認してくれません）



```
rdfs:label a rdf:Property ;  
    rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;  
    rdfs:label "label" ;  
    rdfs:comment "A human-readable name for the subject." ;  
    rdfs:domain rdfs:Resource ;  
    rdfs:range rdfs:Literal .
```

RDFノード型ツリーのサンプル



http://teapot.bodic.org/voc_doc.htmlより

その他の名前空間

popular

1. yago
2. rdf
3. foaf
4. dbp
5. dc
6. owl
7. rdfs
8. ont
9. dbo
10. onto
11. skos
12. geo
13. rss
14. gldp
15. sioc
16. sc
17. fb
18. geonames
19. xsd
20. gr
21. dcterms
22. dct
23. dbpedia
24. akt
25. org
26. commerce

yago. Wikipedia等から作られた
RDF知識データベース

rdf. W3DのRDF1.1規格の語彙集
合

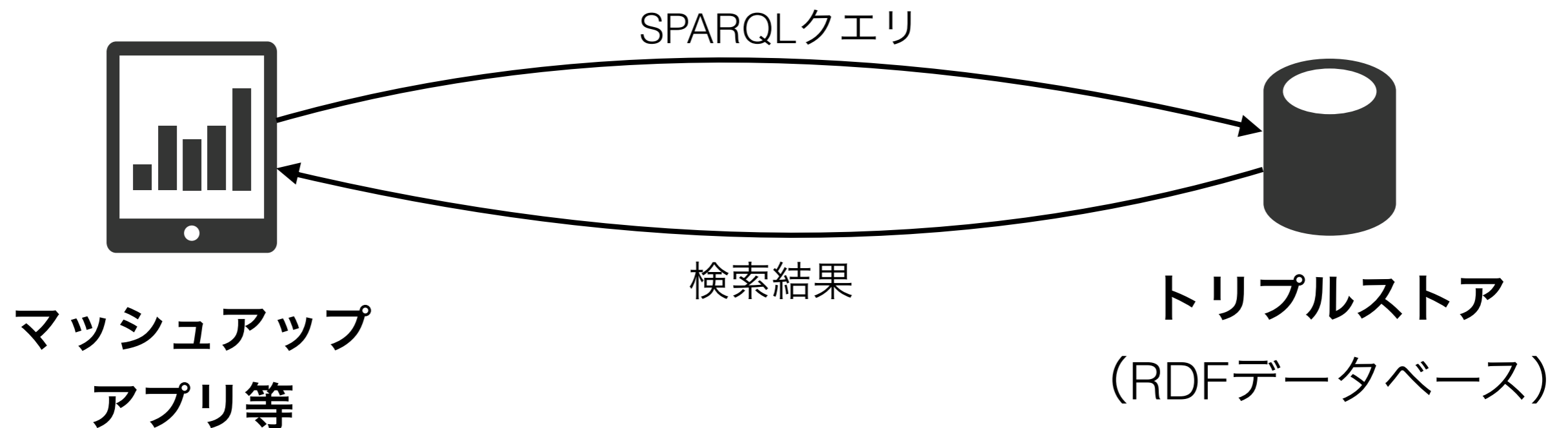
foaf. 人間関係を表す述語
(foaf=**F**riend **O**f **A** **F**riend)

dbpd. Wikipediaから作られた知
識データベース (DBペディア)

本題 2 SPARQLクエリの 基本

SPARQLとは

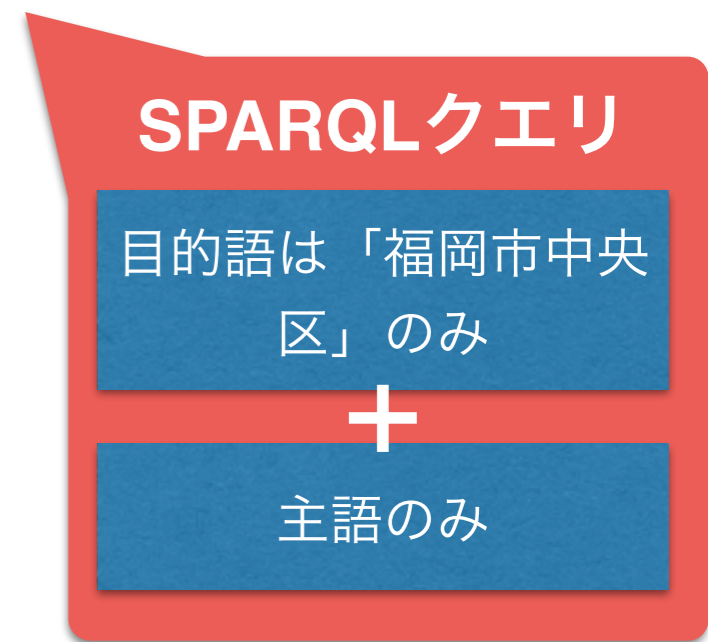
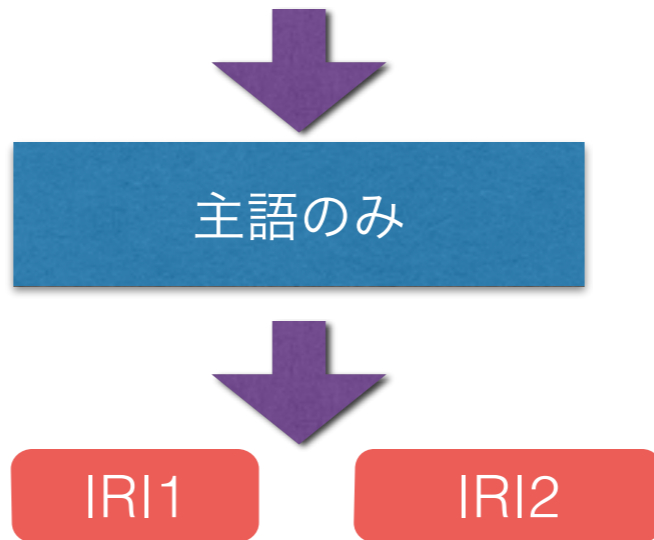
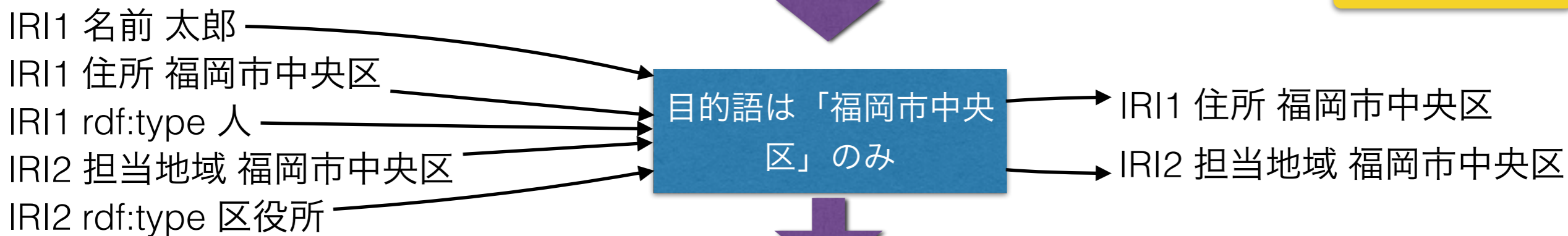
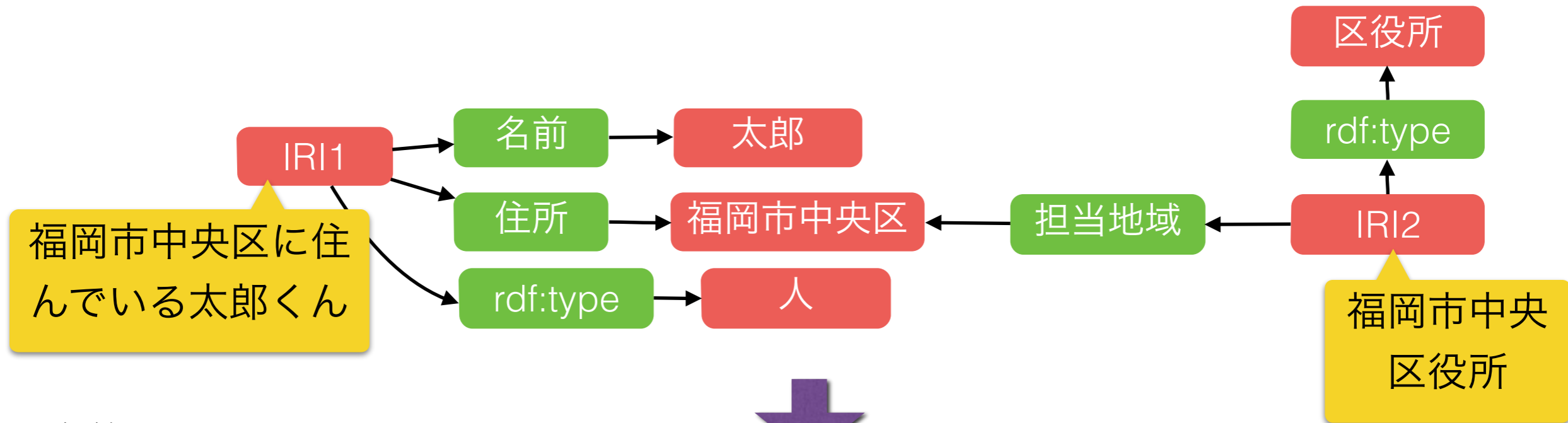
- RDF1.1のクエリ言語。W3Cの標準技術である
- クエリ言語とは
 - データの検索や操作を行うための言語
 - データモデルやデータベースシステムに合わせて設計



SPARQLの概念

- RDFトリプルを検索することで、**RDFグラフをマッチする**
- RDFトリプルを検索して、RDFトリプルを返す
- 検索条件はRDFトリプルで表現する
- RDF/Turtle Syntaxを利用

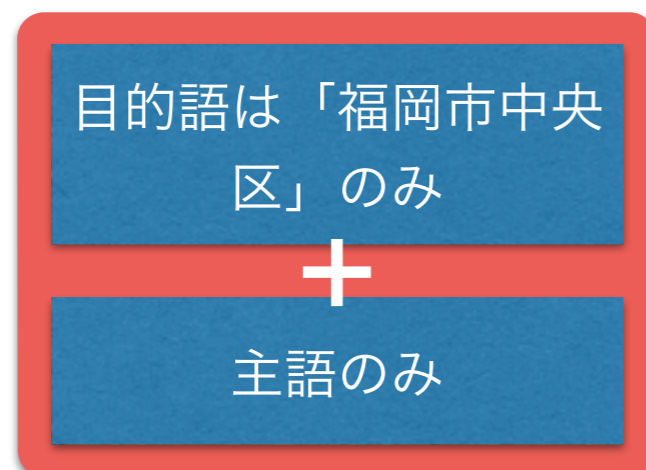
サンプル：次のスライド・・・



SPARQLシンタックス

- SPARQLクエリは主に2つに分けれる：
 - **SELECT部分**：返して欲しい変数を指定する（例：主語のみ）
 - **条件部分**：検索条件を指定する（例：目的語は「福岡市中央区のみ」）
- RDF/Turtleシンタックスを利用
 - 特にRDF/Turtleで書かれたRDFトリプルで検索条件を表現する
- RDFトリプルの主語・述語・目的語は変数または固定値である
 - **固定値**を使うと、条件とする
 - **変数**を使うと、選択する
- 変数名は必ず「?」で始まる

リテラル（文字列等）：クォートを利用（"..."）
URI：ブラケットを利用（<...>）



SELECT部分

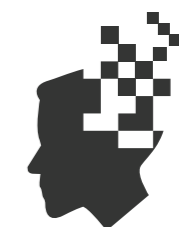
SELECT ?s

条件部分

WHERE { ?s ?p “福岡市中央区” }

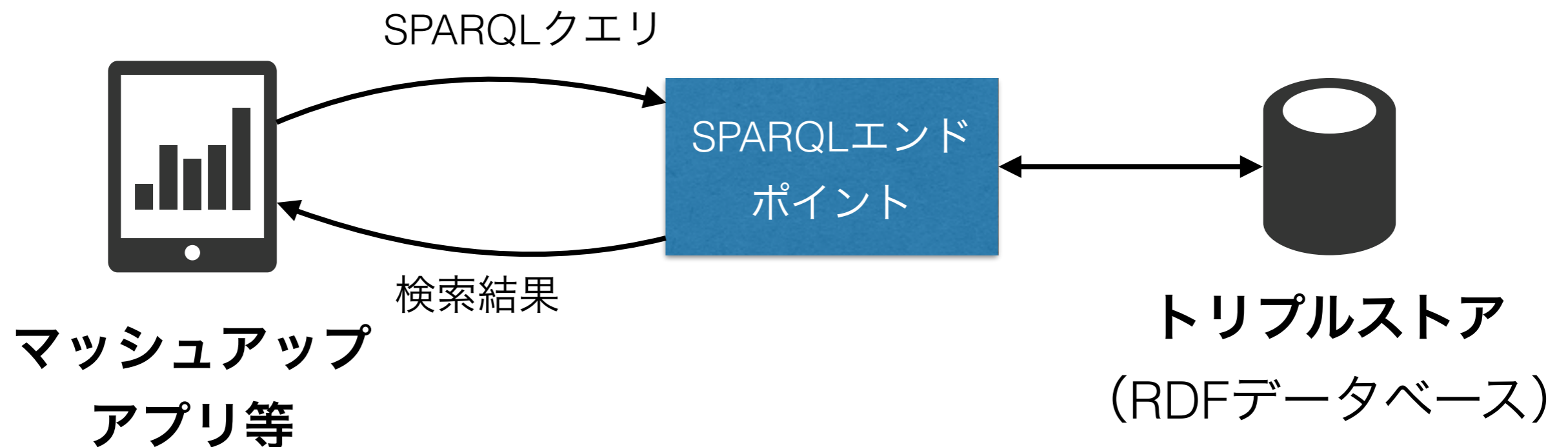
関係データベースとの比較

	関係データベース (MySQL等)	RDF
クエリ言語	SQL	SPARQL
データベース	関係データベース	トリプルストア (グラフストア)
データモデル	関係モデル	RDF1.1モデル



SPARQLエンドポイント

- SPARQLクエリを受ける場所
 - 普段はHTTP API経由
 - Webページ上のフォームを提供するエンドポイントも多い（裏はHTTP APIを利用）
- マッシュアップアプリ等がクライアント
 - サーバ側（PHP、Ruby等）
 - クライアント側（AJAX）



今から演習

演習 1 SPARQLを使って みよう



エンドポイント 1

東日本大震災アーカイブ

<http://fukushima.archive-disasters.jp/>



The screenshot shows a web browser window with the address bar displaying `fukushima.archive-disasters.jp`. The browser tabs include "SPARQLの活用 - 第4回AIツール入門講座", "Virtuoso SPARQL Query Editor", "sparql エンドポイント - Google Search", and "SPARQL Endpoint | 東日本大震災アーカ...". The page header features the logo of the Great East Japan Earthquake Archives Fukushima. The main heading is "東日本大震災アーカイブ Fukushima - SPARQLエンドポイント(テスト版)". Below this, there are three columns of information:

- SPARQLでさがす**: This section explains that the SPARQL endpoint stores metadata converted from RDF. It allows users to search using SPARQL queries.
- キーワードでさがす**: This section describes a keyword search function for the metadata. It states that search results are displayed one item at a time.
- データの構造**: This section provides information about the structure of the metadata, including SPARQL examples and API usage instructions.

Below the search options, there is a section for "PREFIX(名前空間接頭辞)追加:" listing various vocabularies such as Dublin Core, DCMI Metadata Terms, NDL Metadata Terms, Data Catalog Vocabulary, Geo vocabulary, GeoNames Ontology, JM P2.0, The RDF Vocabulary (RDF), The RDF Schema vocabulary (RDFS), Creative Commons, Friend of a Friend (FOAF), The Ontology for Media Resources, NDL東日本大震災アーカイブメタデータ, PREMIS, vCard, and "東日本大震災アーカイブ" 基盤構築事業 福島プロジェクトメタデータ. At the bottom, there is a section for "サンプルSPARQLクエリ:" with the example query "全検索 全データのタイトルを取得 URIと緯度経度情報を取得".

 **SPARQLでさがす**

このSPARQLエンドポイントには、東日本大震災アーカイブ Fukushimaメタデータを RDFに変換したメタデータを蓄積しており、SPARQLを利用して検索することがで

PREFIX(名前空間接頭辞)追加:

[Dublin Core](#) [DCMI Metadata Terms](#) [NDL Metadata Terms](#) [Data Catalog Vocabulary](#) [Geo vocabulary](#) [GeoNames Ontology](#) [JM P2.0](#) [The RDF Vocabulary \(RDF\)](#) [The RDF Schema vocabulary \(RDFS\)](#) [Creative Commons](#) [Friend of a Friend\(FOAF\)](#) [The Ontology for Media Resources](#) [NDL東日本大震災アーカイブメタデータ](#) [PREMIS](#) [vCard](#) 「東日本大震災アーカイブ」基盤構築事業 福島プロジェクトメタデータ

サンプルSPARQLクエリ:

[全検索](#) [全データのタイトルを取得](#) [URIと緯度経度情報を取得](#)

SPARQL クエリ

```
1 select ?p ?o { <http://fukushima.archive-disasters.jp/id/resource/M2013011819254782265> ?p ?o }
```

SPARQL検索

①SPARQL検索を選択

②SPARQLクエリを入力

③SPARQLクエリを実行する

④検索結果を確認する

SPARQLの活用 - 第4回AIツール入門講座 Virtuoso SPARQL Query Editor sparql エンドポイント - Google Search SPARQL Endpoint | 東日本大震災アーカ...

東日本大震災アーカイブ Fukushima
The Great East Japan Earthquake Archives Fukushima

SPARQLでさがす キーワードでさがす データの構造

35 / 35 (表示件数/ヒット件数) ダウンロード: RDF/XML形式 N3/Turtle形式 CSV形式

p	o
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://kn.ndl.go.jp/terms/Resource
http://www.w3.org/2000/01/rdf-schema#seeAlso	http://fukushima.archive-disasters.jp/infolib/id/M2013011819254782265
http://purl.org/dc/elements/1.1/title	福島県内各地方 環境放射能測定値 (暫定値) 第9496.pdf報
http://fukushima.archive-disasters.jp/metadata/terms/keyword	県北保健福祉事務所東側駐車場 (福島市)
http://fukushima.archive-disasters.jp/metadata/terms/keyword	郡山合同庁舎東側入口付近 (郡山市)
http://fukushima.archive-disasters.jp/metadata/terms/keyword	白河合同庁舎駐車場 (白河市)
http://fukushima.archive-disasters.jp/metadata/terms/keyword	会津若松合同庁舎駐車場 (会津若松市)
http://fukushima.archive-disasters.jp/metadata/terms/keyword	南会津合同庁舎駐車場 (南会津町)
http://fukushima.archive-disasters.jp/metadata/terms/keyword	南相馬合同庁舎駐車場 (南相馬市)
http://fukushima.archive-disasters.jp/metadata/terms/keyword	いわき合同庁舎駐車場 (いわき市)
http://fukushima.archive-disasters.jp/metadata/terms/keyword	

サンプルクエリを利用

① 「すべてのタイトルを取得」を選択

サンプルSPARQLクエリ:

全検索 全データのタイトルを取得 URIと緯度経度情報を取得

SPARQLクエリ

```
1 PREFIX dc: <http://purl.org/dc/elements/1.1/>
2 select ?s ?o from <http://fukushima.archive-disasters.jp/rdf/resource>
3 where {?s dc:title ?o}
4
```

②クエリが自動的に導入される ↓

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
select ?s ?o from <http://fukushima.archive-disasters.jp/rdf/resource>
where {?s dc:title ?o}
```

名前空間「dc」を定義

省略可能ですが、述語名などに「<http://purl.org/dc/elements/1.1/>」の代わりに「dc」を使えるようになるため、便利。

クエリ対象ノード識別子の名前空間

指定する名前空間の中しかノードを検索しない（省略可能）

PREFIX dc: <<http://purl.org/dc/elements/1.1/>>

select ?s ?o from <<http://fukushima.archive-disasters.jp/rdf/resource>>

where {?s dc:title ?o}

クエリの条件

述語が「<http://purl.org/dc/elements/1.1/title>」のすべてのトリプルを探す

返して欲しい変数

クエリは他の変数を使ってもいいですが、そちらに書いてある変数のみ返される。

サンプルクエリの結果

select ?s ?o where { ?s dc:title ?o }

s	o
http://fukushima.archive-disasters.jp/id/resource/M2013011818013170719	お知らせ版No2
http://fukushima.archive-disasters.jp/id/resource/M2013011818013170718	お知らせ版No1
http://fukushima.archive-disasters.jp/id/resource/M2013011818013270720	お知らせ版No3
http://fukushima.archive-disasters.jp/id/resource/M2013011818013270721	お知らせ版No4
http://fukushima.archive-disasters.jp/id/resource/M2013011818013370722	お知らせ版No5
http://fukushima.archive-disasters.jp/id/resource/M2013011818013370723	お知らせ版No6
http://fukushima.archive-disasters.jp/id/resource/M2013011818013470724	お知らせ版No7
http://fukushima.archive-disasters.jp/id/resource/M2013011818013470725	お知らせ版No8
http://fukushima.archive-disasters.jp/id/resource/M2013011818013570727	お知らせ版No10
http://fukushima.archive-disasters.jp/id/resource/M2013011818013570726	お知らせ版No9
http://fukushima.archive-disasters.jp/id/resource/M2013011818013670729	お知らせ版No12



問題 1

タイトルが「お知らせ版No2」になっている
ノードについてすべての情報（述語、目的語）
を検索してください

答 1

案 1 : 前の検索結果を活かせる

前の結果では「dc:title」が「お知らせ版No2」となっているノードの識別子がありましたので、そのまま使う

案 1 :
select ?p ?o
where {<http://fukushima.archive-disasters.jp/id/resource/M2013011818013170719> ?p ?o}

案 2 :
PREFIX dc: <http://purl.org/dc/elements/1.1/>
select ?p ?o
where {?s dc:title "お知らせ版No2" . ?s ?p ?o }

案 2 : タイトルの文字列から検索

「?s dc:title "お知らせ版No2"」を検索条件として、すべての述語と目的語を検索

各クエーリを説明しましょう・・・

案1 クエリの説明

変数「p」と「o」を返す



```
select ?p ?o
```

```
where {<http://fukushima.archive-disasters.jp/id/resource/  
M2013011818013170719> ?p ?o}
```



主語を指定する

検索条件のトリプルは主語を指定するので、主語が指定したもののトリプルのみ検索する。述語と目的語は変数を指定するので、絞り込み条件に使わない。

案2 クエリの説明

変数「p」と「o」を返す

検索トリプルで使う「s」は返さない。

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>  
select ?p ?o  
where {?s dc:title "お知らせ版No2" . ?s ?p ?o}
```

条件トリプル2：情報取得

条件トリプル1で選択した「s」に対して、すべての述語と目的語を選択し、それぞれ「p」と「o」に格納。


条件トリプル1：絞り込み条件

述語＝「dc:title」と目的語＝「お知らせ版No2」のトリプルのすべて選択して、それらの主語は変数「s」に格納する。

で？



検索結果を見ると・・・

第4回 AIツール入門講座		Virtuoso SPARQL Query Editor	sparql エンドポイント - Google...	SPARQL Endpoint 東日本大...	rdf turtle - Google Search	+
 東日本大震災アーカイブ Fukushima The Great East Japan Earthquake Archives Fukushima		SPARQLでさがす		キーワードでさがす		データの構造
http://fukushima.archive-disasters.jp/metadata/terms/keyword		農畜産物損害賠償手続き説明会				
http://fukushima.archive-disasters.jp/metadata/terms/keyword		国民健康保険への加入				
http://purl.org/dc/terms/language		"jpn"^^<http://purl.org/dc/terms/ISO639-2>				
http://purl.org/dc/terms/spatial		nodeID://b764270				
http://purl.org/dc/terms/audience		一般				
http://fukushima.archive-disasters.jp/metadata/terms/accessFree		なし				
http://purl.org/dc/elements/1.1/creator		nodeID://b764267				
http://purl.org/dc/elements/1.1/contributor		nodeID://b764268				
http://purl.org/dc/terms/abstract		おしらせ版				
http://xmlns.com/foaf/0.1/page		http://fukushima.archive-disasters.jp/infolib/id-c/M2013011818013170719				
http://purl.org/dc/elements/1.1/source		http://fukushima.archive-disasters.jp/				
http://fukushima.archive-disasters.jp/metadata/terms/accessPeriod		なし				
http://ndl.go.jp/dcndl/terms/alternative		oshirase_no.02.pdf				

URLをアクセスすると . . .



RDFストアの探索について

悩み：初めてSPARQLエンドポイントを見る時に、使っている語彙（述語、ノード型等）が分からない



解決策：とりあえず、語彙の閲覧を出来る様なクエリを投げる！

データベースの中に使われているノード型をリストアップしましょう

下記のクエリーを使うと全てのノードのノード型を検索できる

```
select ?type  
where {?s a ?type . }
```

- しかし、重複している。重複をなくすため、「**DISTINCT**」キーワードを利用できる

```
select distinct ?type  
where {?s a ?type . }
```



問題 2

東日本大震災アーカイブから利用されている
すべての述語をリストアップして下さい

エンドポイント 2

公共施設等情報のオープンデータ実証

<http://teapot.bodic.org>

ニックネームは「Teapot」
(ティーポット)

総務省「情報流通連携基盤の公共施設等情報における実証」

公共施設等情報のオープンデータ実証 開発者サイト

総務省「情報流通連携基盤の公共施設等情報における実証」



本サイトは、平成26年度の総務省による公共施設等情報を取り扱うオープンデータ実証「情報流通連携基盤の公共施設等情報における実証」での、オープンデータ開発者サイトです。本実証は、地方自治体が保有する公共施設等情報や統計情報などを情報流通連携基盤共通APIを通し公開することで、公共施設に関する様々なアプリケーション開発が促進されることを目的としています。福岡市、福岡県、糸島市の協力を得て実施しています。データの利用に関しましては[利用規約](#)をよくお読みいただき、利用規約に同意の上、ご利用ください。



問題 3

まず、データベースの構成を探索しましょう！

URL: teapot.bodic.org/test_api.html

利用されているノード型および述語を閲覧しましょう

「a」キーワードは使えない場合は「rdf:type」述語を直接つかいましょう！

．．．ノード型が沢山あります！

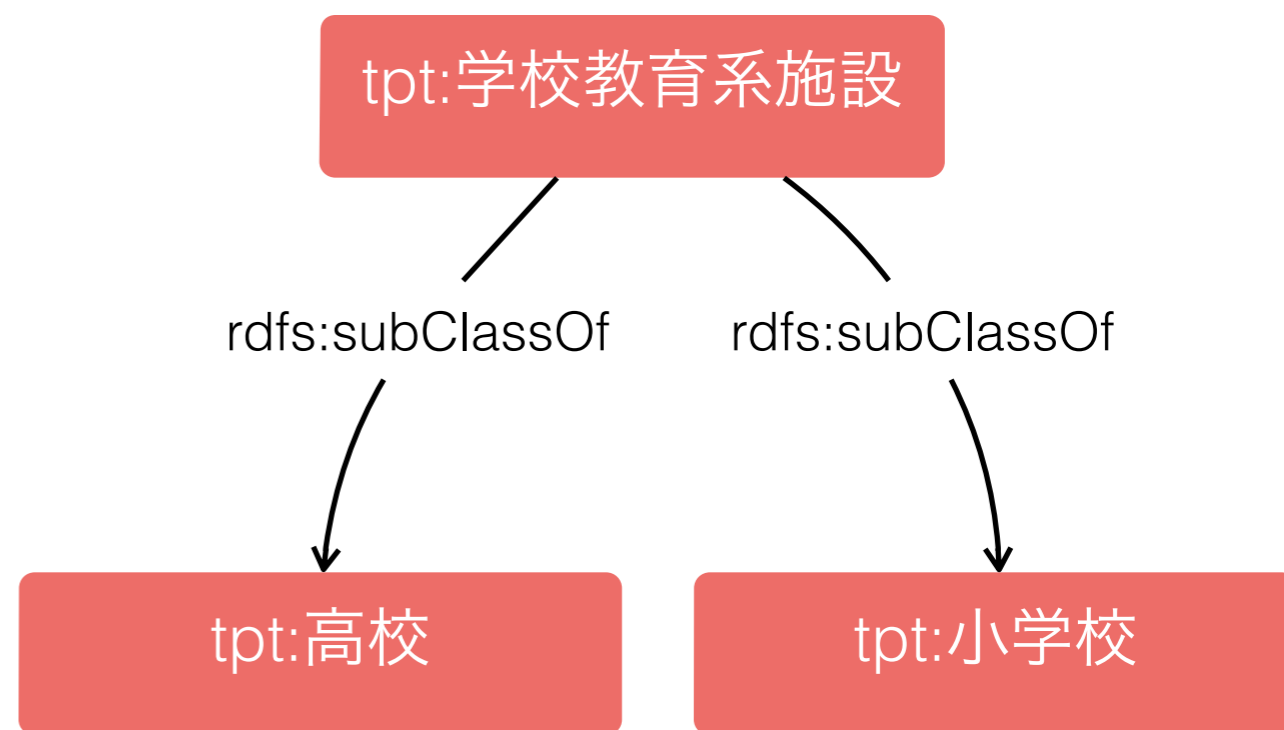


階層的ノード型の探索

TeapotはRDFS語彙を利用して、ノード型の親子関係を表す（述語：「rdfs:subClassOf」）

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?t1 ?t2  { ?t1 rdfs:subClassOf ?t2. }
```

```
{
  "t1": {
    "type": "uri",
    "value": "http://teapot.bodic.org/type/小学校"
  },
  "t2": {
    "type": "uri",
    "value": "http://teapot.bodic.org/type/学校教育施設"
  }
},
{
  "t1": {
    "type": "uri",
    "value": "http://teapot.bodic.org/type/高校"
  },
  "t2": {
    "type": "uri",
    "value": "http://teapot.bodic.org/type/学校教育施設"
  }
}
```



なんでこんなわかり難い
ことをするのか？

演習で実感しましょう





問題 4

全ての小学校の名前を検索しましょう

施設名は「rdfs:label」
述語で格納している



問題 5

構造化ノード型を利用して、より正確なクエリを書きましょう！

全ての学校教育系施設の名前を検索しましょう

SPARQLでRDFグラフを辿る

SPARQLの「*」や「+」オペレータを使う
と、グラフを辿る

```
prefix rdfs: <...>
select ?s{ ノード 2 rdfs:subClassOf ?s. }
```

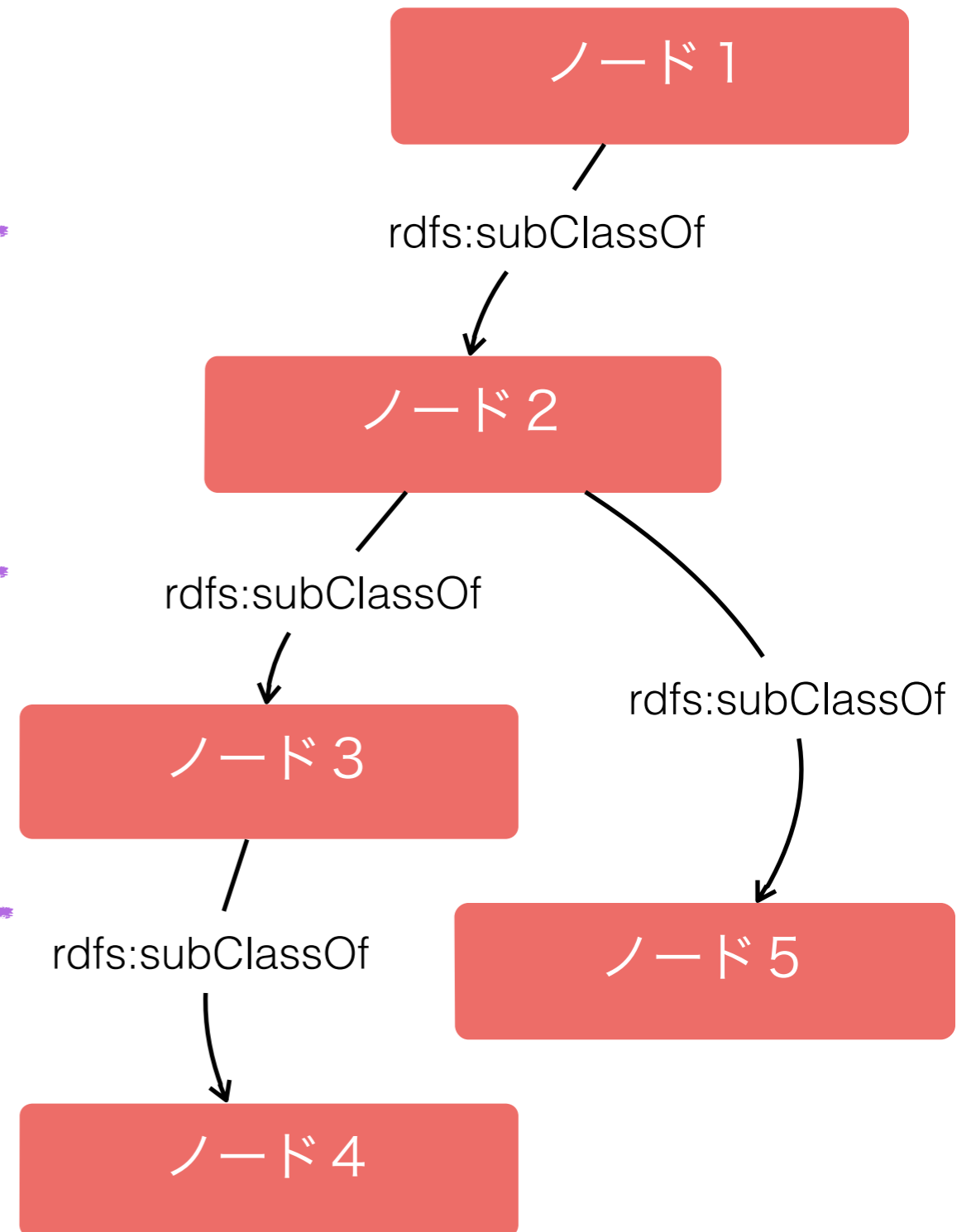
結果：ノード 3、 5

```
prefix rdfs: <...>
select ?s{ ノード 2 rdfs:subClassOf+ ?s. }
```

結果：ノード 3、 4、 5

```
prefix rdfs: <...>
select ?s{ ノード 2 rdfs:subClassOf* ?s. }
```

結果：ノード 2、 3、 4、 5





問題 6

施設の全てのサブタイプを検索しましょう！

SPARQLで文字列をフィルタをかける

SPARQLで「**FILTER**」キーワードを使うと諸々はフィルターをかけられる

例 1 (比較演算) : select ?s { ?s foaf:age ?age .

FILTER (age < 10 && age > 5) }

年齢 ∈]5,10[

例 2 (正規表現) : select ?s { ?s foaf:name ?name .

FILTER (regexp(?name, "taro")) }

"kentaro", "kantaro", "taro" ...

例 3 (正規表現) : select ?s { ?s foaf:name ?name .

FILTER (regexp(?name, "^山")) }

○山田、山口
×中山



問題 7

中央区にある全ての施設を検索してください。

住所の文字列は「[http://teapot.bodic.org/
predicate/所在地](http://teapot.bodic.org/predicate/所在地)（文）」述語で格納している

その他の便利なSPARQL(1)

OPTIONAL キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix tpp: <http://teapot.bodic.org/predicate/>
select ?name ?address
{ ?s rdfs:label ?name ; OPTIONAL { tpp:所在地 (文) ?address } }
```

「tpp:所在地 (文)」述語がなくても、
「rdfs:name」さえあれば合致する

その他の便利なSPARQL(2)

UNION キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix tpt: <http://teapot.bodic.org/type/>
select ?name
{
  ?s rdfs:label ?name.
  { ?s rdf:type tpt:小学校 } UNION { ?s rdf:type tpt:高校 }
}
```

高校と小学校の名前を検索する

その他の便利なSPARQL(3)

COUNT キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
select COUNT(?name)
{
  ?s rdfs:label ?name.
}
```

rdf:labelのあるノード
の数を返す

その他の便利なSPARQL(4)

GROUP BY キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
select COUNT(?name)
{
  ?s rdfs:label ?name ; rdf:type ?type
} group by ?type
```

各タイプ毎にrdfs:labelのある
ノードの数を返す

その他の便利なSPARQL(5)

LIMIT / OFFSET キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
select COUNT(?name)
{
  ?s rdfs:label ?name ; rdf:type ?type
} group by ?type offset 50 limit 10
```

合致するデータの中から50個目
から10個を返す（59個目まで）

演習 2 初めてのWebアプリ リケーション

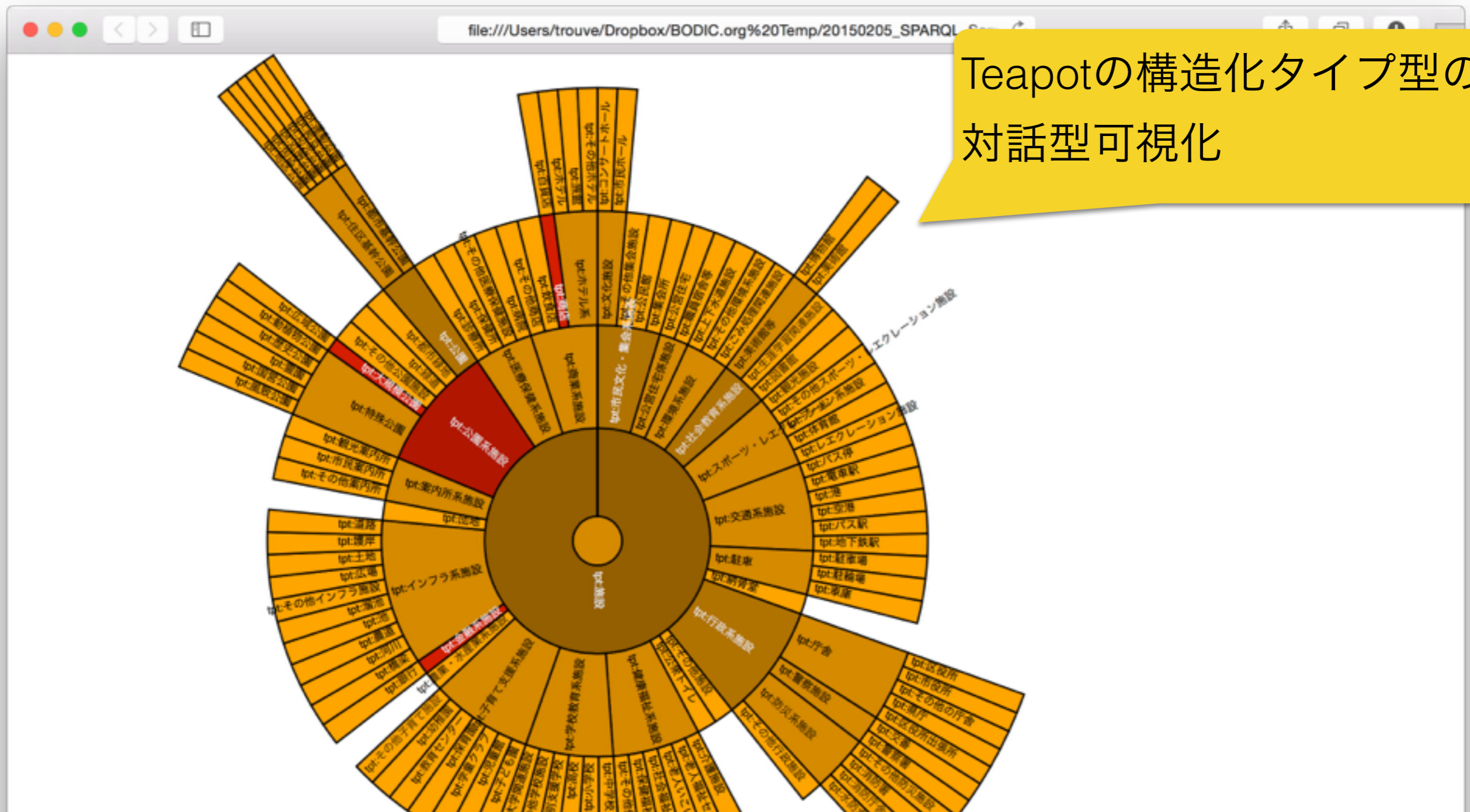


演習について

- 演習の教科書下記のリンクにあります：
 - <http://www.bodic.org/?p=119>
- 各演習はファイル3つあります：
 - stepX.html：ページの構成
 - stepX.js：ジャヴァスクリプトソースコード
 - stepX.css：スタイルシート
- 基本的にStepXはStep(X-1)のファイルを書き加えると想定する
 - できるだけ自分で考えてコードを書いてください
 - 分からない時のみ教科書からコピーペーストできる

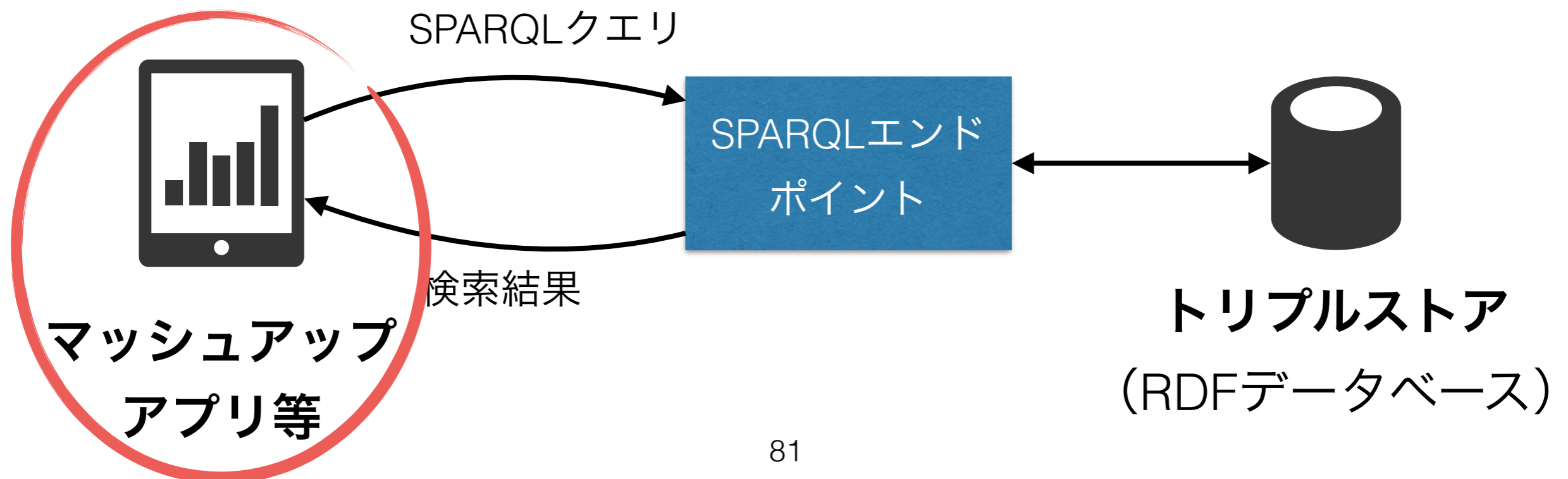
一番編集することが多い

この演習で出来上がるもの



Webアプリケーションとは

- Web Browserで利用できるアプリケーション
- クライアント側で利用する技術
 - 言語はHTML、CSS、Javascript
 - その他：AJAX（次のスライドで説明する・・・）
- サーバ側は今日しません



AJAXについて

Aynchronous **J**avascript **A**nd **X**ML

- クライアント側から非同期的に遠隔内容をアクセスするような技術
- 本演習はAJAX経由でクエリを投げて、動的にWebページ内容を更新する

JQueryについて

- WebブラウザがJavascript経由で提供している機能を簡単に利用できるようなライブラリ
- 本演習では下記の機能を活躍する：
 - DOMエレメントへの選択（セレクト）
 - AJAX機能（POST関数）
 - DOMエレメントの内容を変更
- JQuery機能は「\$」オブジェクトから利用できる
 - DOMエレメント選択：\$(セレクト)
 - AJAX POST：\$.post(...)
 - DOMエレメント内容変更：\$(...).append()

Document Object Model (**DOM**)

HTML等で使われているエレメントの規格および操作APIの仕様



演習 2 Step 1

SPARQLの検索結果を取得しよう

<http://www.bodic.org/?p=242>

教科書の指示に基づいて、Teapotサーバへクエリを投げて、得られた結果はページに表示しましょう

例えば、全てのノード型親子関係を表すトリプルを取得する

クエリー結果を一部ずつロード

- **課題**：データが大量に合致した場合は、結果は数メガを超えます
 - サーバ側はGzip圧縮使うことも多いですが、ブラウザによってサポートしていない
- **解決策**：データを一気にダウンロードではなく、一部ずつ行う
- **実装方法**：SPARQLの「limit」と「offset」キーワードを利用する
- **例**：

```
select . . . where { . . . }  
offset 1000 limit 500
```

1000個目から 1 5 0 0 個目
まで返す



演習 2 Step 2

検索結果を分割して取得しよう

教科書：<http://www.bodic.org/?p=249>

演習 2 Step 1 を書き加えて、検索結果を分割して
10個ずつ取得するように書き換えてください。



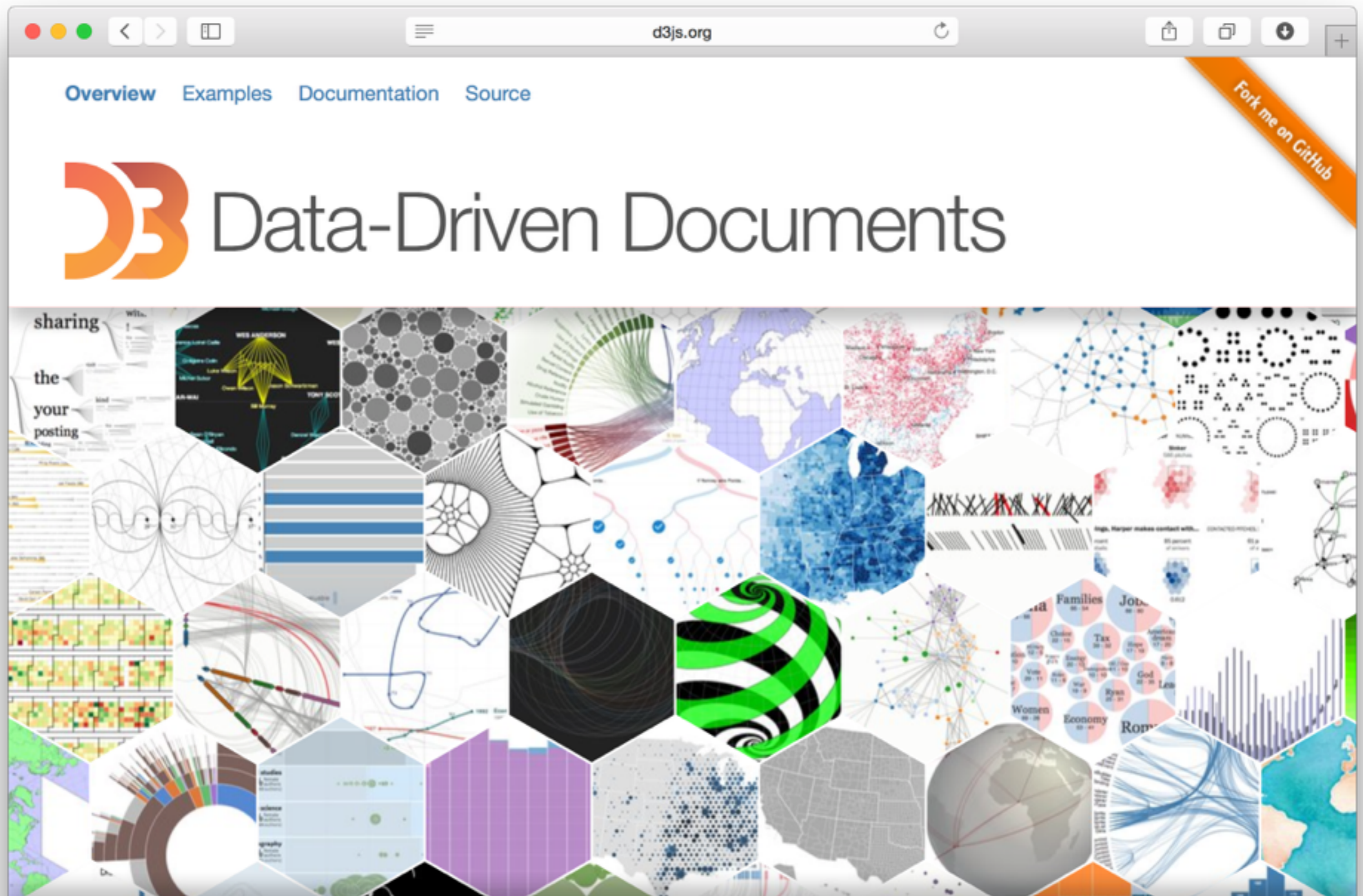
演習 2 Step 3

特定のデータを表示させよう

教科書 : <http://www.bodic.org/?p=276>

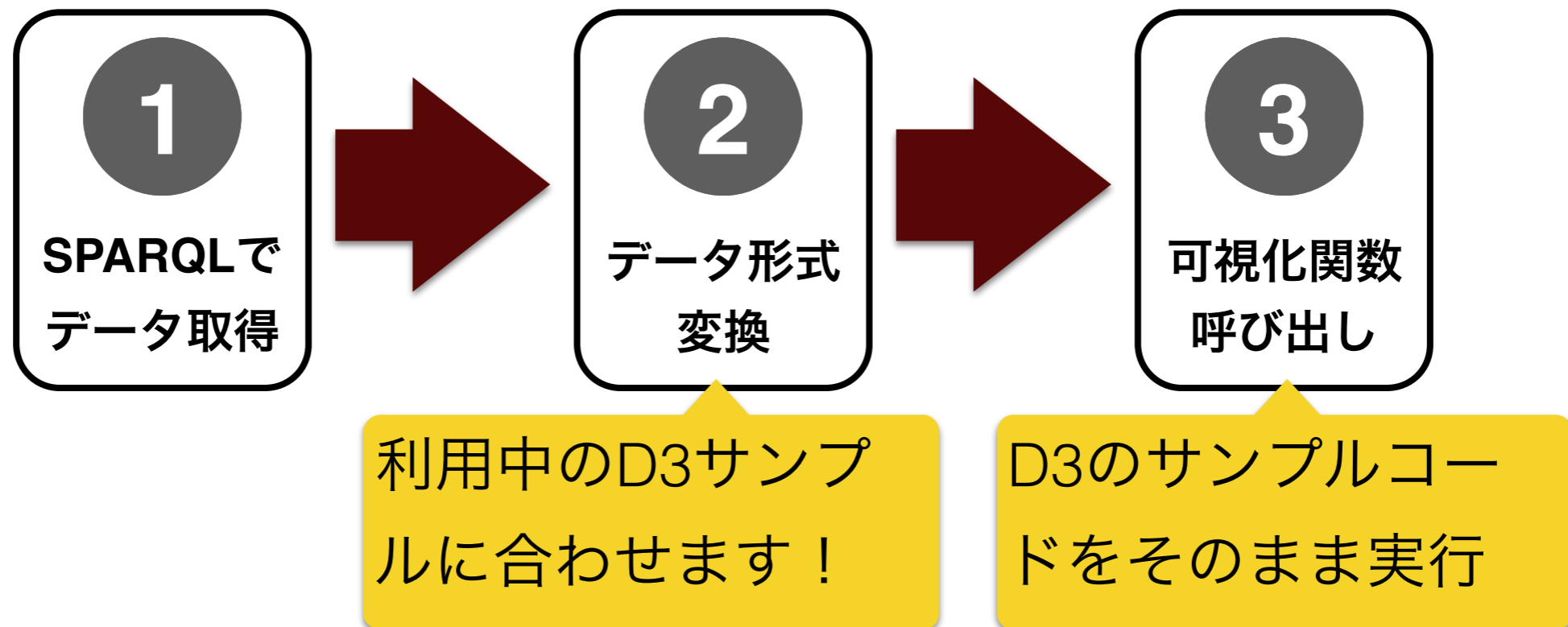
演習 2 Step 2 を書き加えてに、Teapotデータベースで利用されているノード型識別子を表示してください（生JSONではなく）。10個ずつ取得してください。

D3について

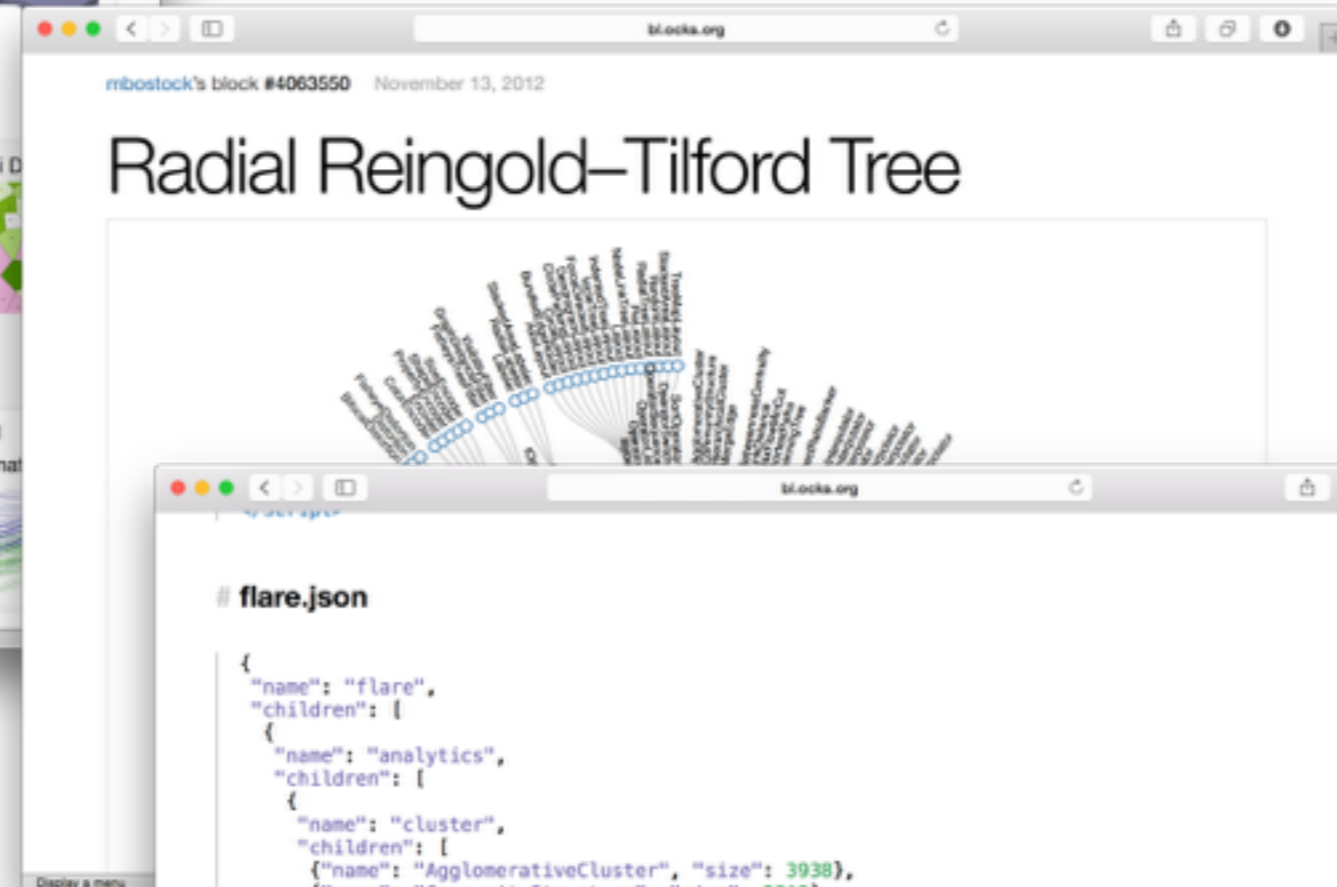
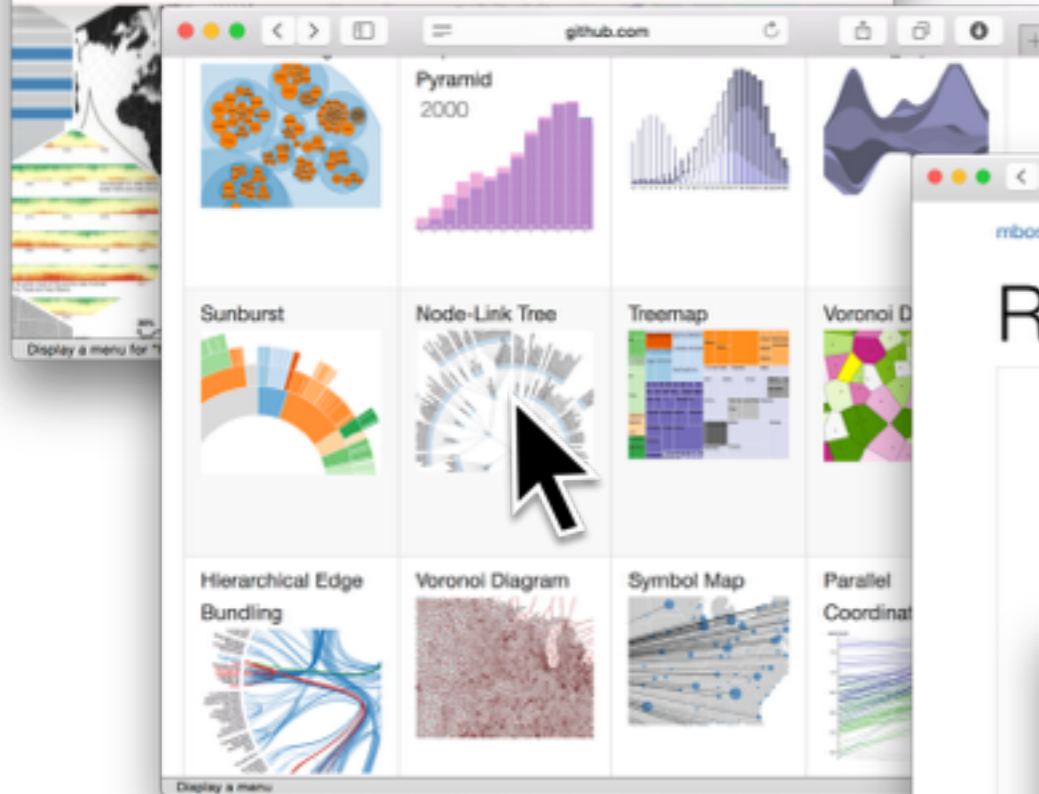


D3の使い方について

- D3のサイトに諸々なサンプルが置いてあります。そのままコピペしてもいいです！



サンプル



データ形式のサンプル

すこしわかり難い場合
もあります・・・

```
# flare.json

{
  "name": "flare",
  "children": [
    {
      "name": "analytics",
      "children": [
        {
          "name": "cluster",
          "children": [
            { "name": "AgglomerativeCluster", "size": 3938 },
            { "name": "CommunityStructure", "size": 3812 },
            { "name": "HierarchicalCluster", "size": 6714 },
            { "name": "MergeEdge", "size": 743 }
          ]
        },
        { "name": "graph",
          "children": [
            { "name": "BetweennessCentrality", "size": 3534 },
            { "name": "LinkDistance", "size": 5731 },
            { "name": "MaxFlowMinCut", "size": 7840 },
            { "name": "ShortestPaths", "size": 5914 },
            { "name": "SpanningTree", "size": 3416 }
          ]
        },
        { "name": "optimization",
          "children": [
            { "name": "AspectRatioBanker", "size": 7074 }
          ]
        }
      ]
    }
  ]
}
```



演習 2 Step 4

SPARQLの検索結果を可視化しよう

教科書：<http://www.bodic.org/?p=278>

演習 2 Step 3 を書き加えて、Teapotのノード型を可視化しましょう！タイプは10

「type_sunburst.lib.js」では、定義している下記の関数をご利用ください：

- データ変換：`makeTypeTree(sparql_res)`
- グラフ生成：`sunburst(selector, json)`

クエリ変数名：

- 親クラス：`?superclass`
- 子クラス：`?class`

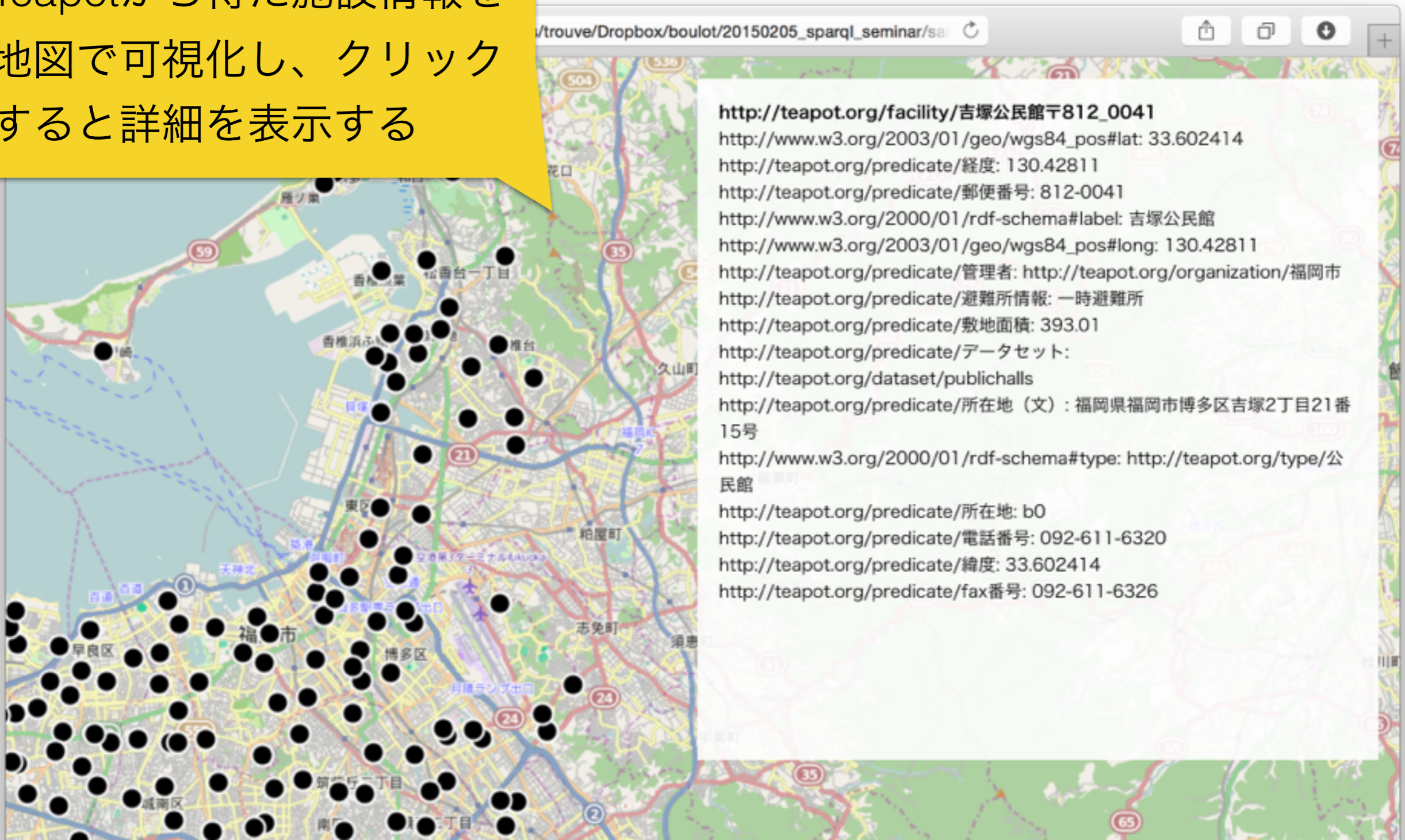
「json」は
「makeTypeTree」の出力

演習 3 地図アプリを作 りましよう

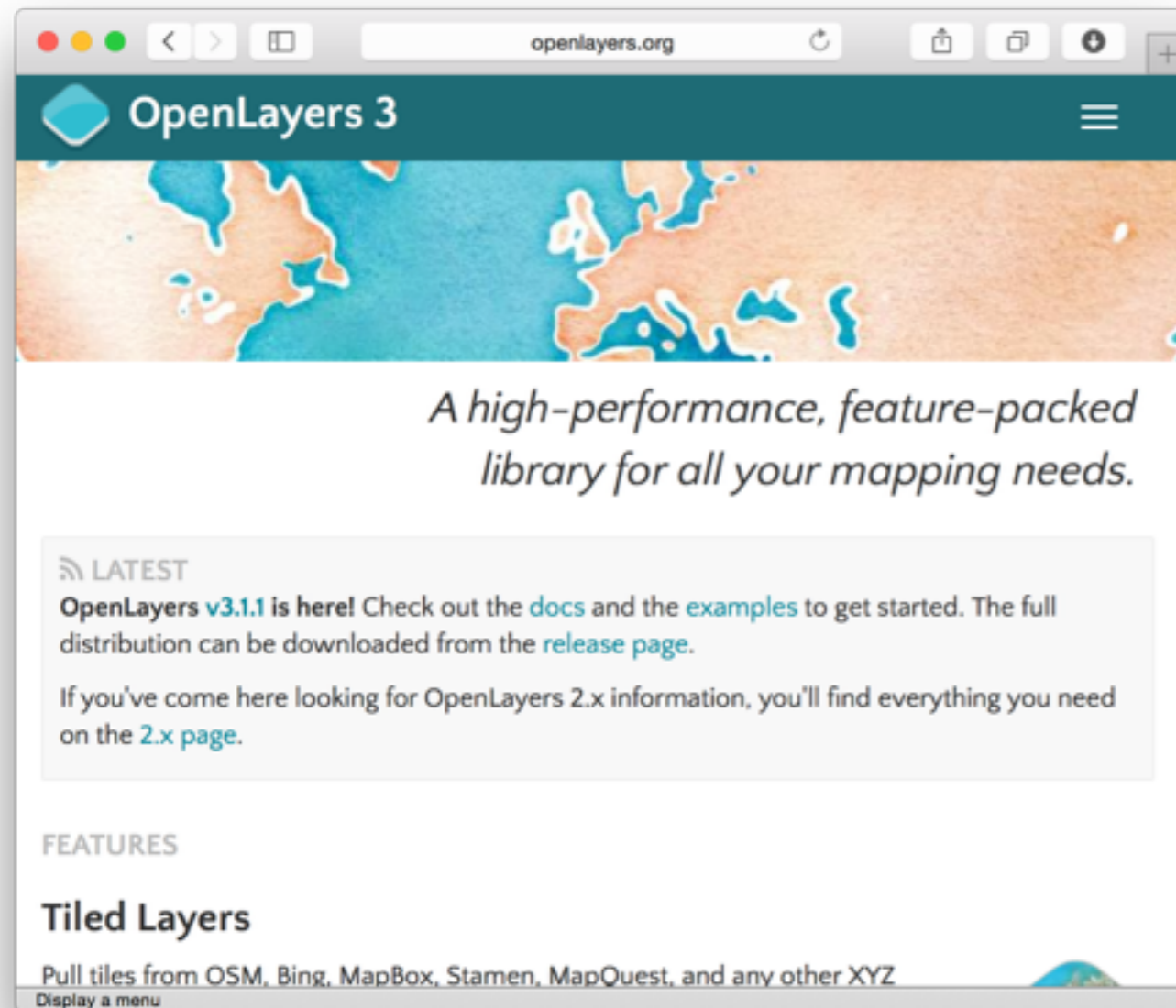


この演習で出来上がるもの

Teapotから得た施設情報を
地図で可視化し、クリック
すると詳細を表示する

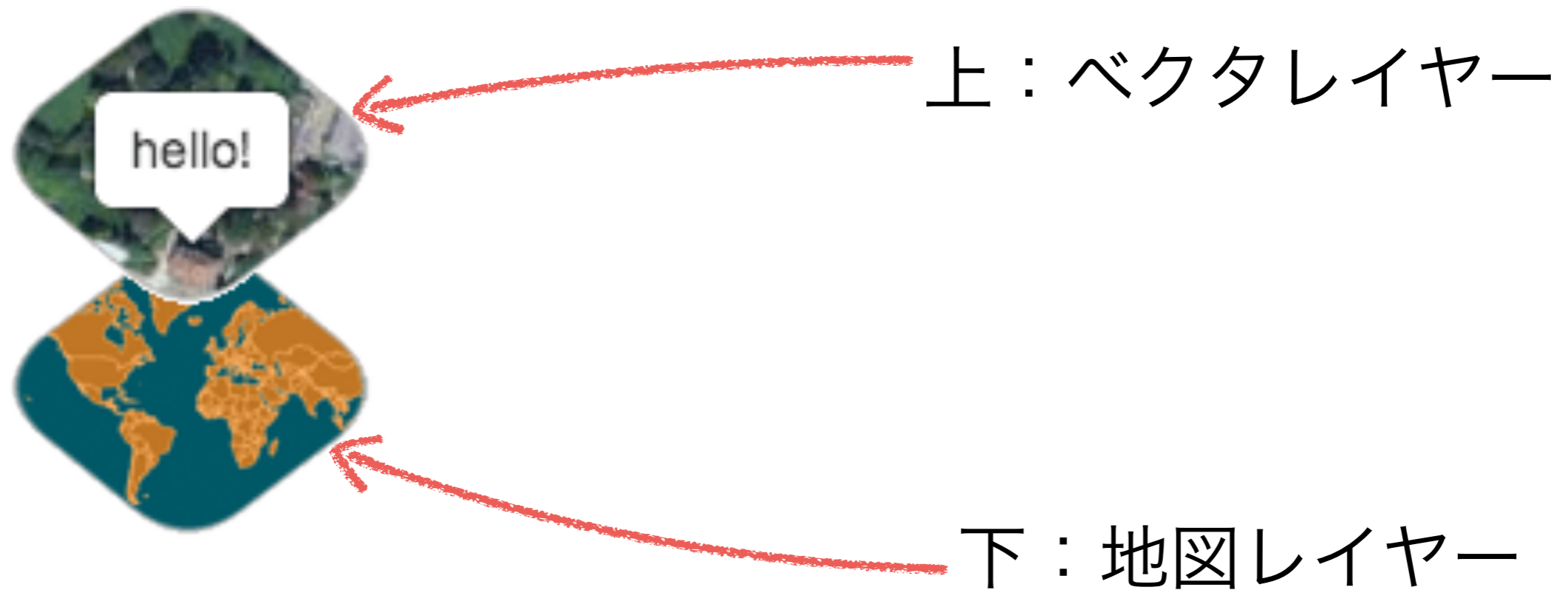


地図表示：OpenLayers3

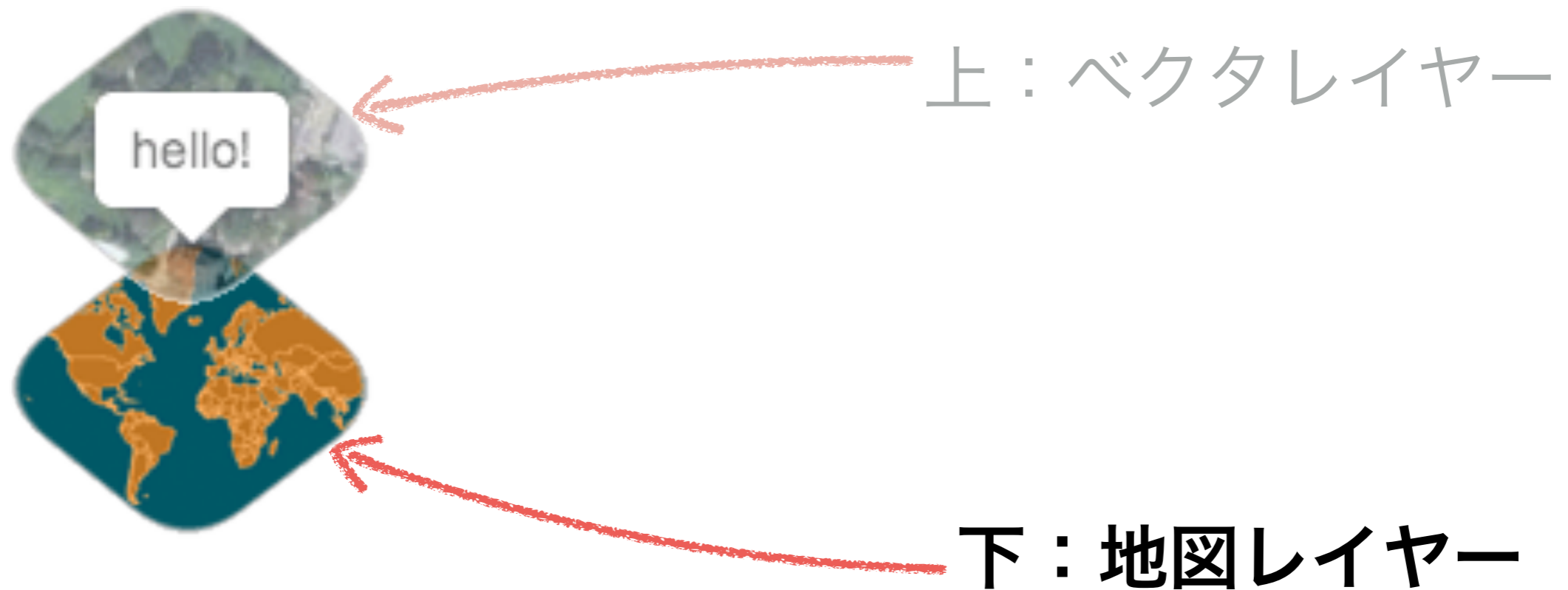


- レイヤー管理・表示ライブラリ
- 今回利用するレイヤー種
 - 地図レイヤー：地図を表示する
 - ベクターレイヤー：アイコンやベクター形式地図データを表示する

今回のユースケース



今回のユースケース



オープンレイヤーの初期化(1)

OL3オブジェクトを生成する

```
var map = new ol.Map({  
  target: "地図",  
  layers: [  
    new ol.layer.Tile({  
      source: new ol.source.OSM()  
    }),  
  ],  
});
```

対象DOM要素のID

各レイヤーの初期化

今回は地図タイルレイヤーを使う。利用可能なライヤー種はAPI説明へ↓

<http://openlayers.org/en/v3.1.1/apidoc/>

地図タイルについて

- GoogleMap等が表示する地図画像は資格単位で整理している：タイルという
- 表示する時にタイルが一個ずつ取得する
- ネットワークが遅い時に見える

今回はOpenStreetMapが提供している地図タイルを利用する



オープンレイヤーの初期化(2)

```
var map = new ol.Map({  
  target: "地図",  
  layers: [...],  
  view: new ol.View({  
    center: ol.proj.transform([経度, 緯度], 'EPSG:4326', 'EPSG:3857'),  
    zoom: 10  
  }),  
});
```

表示する場所と拡大ファクターの設定

世界座標での経度・緯度を指す標準的な番号



演習 3 Step 1

地図データを表示しよう

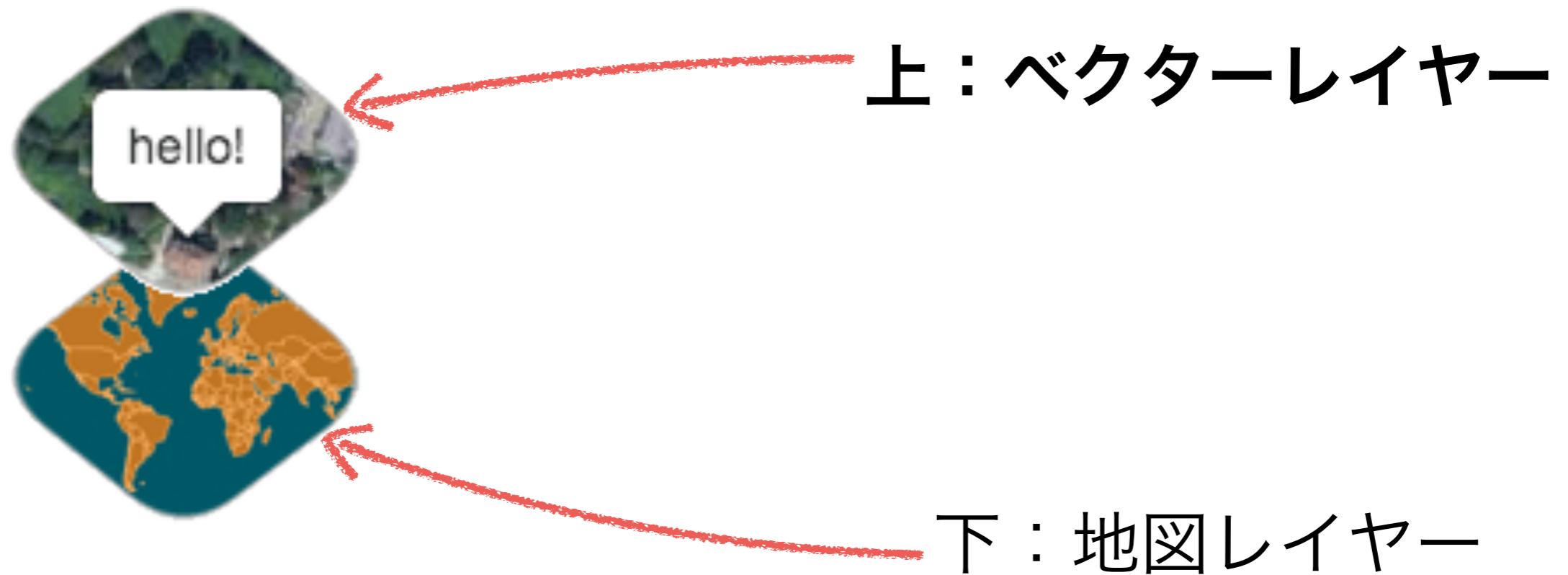
教科書：<http://www.bodic.org/?p=121>

教科書の指示に基づいて、地図を作成し、福岡周辺を表示してください。

福岡市の緯度・経度は
Wikipedia等にありますが！

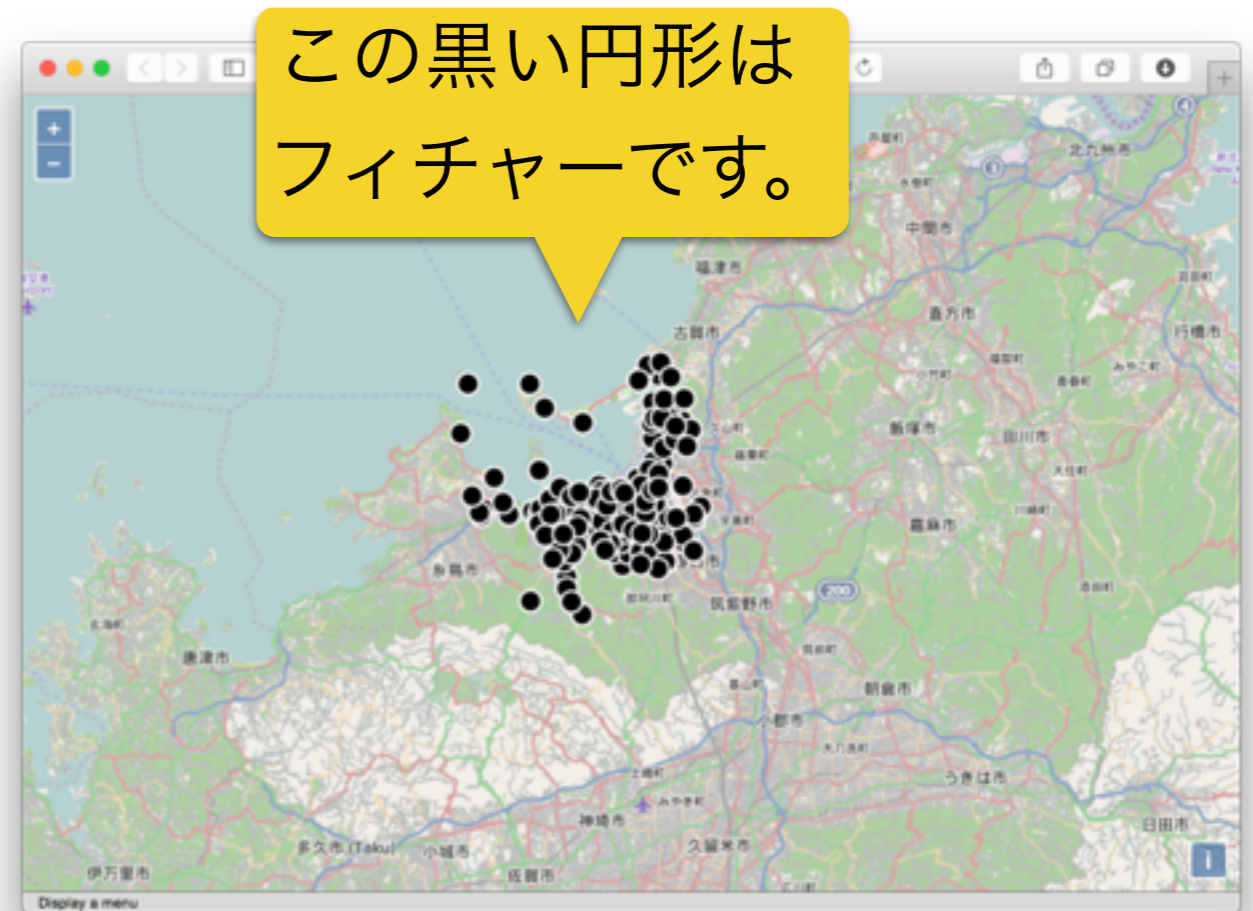


地図上にピンを表示しましょう！



フィチャーについて

- ピンはOpenLayersの言葉で「フィチャー」というオブジェクトの1種類です。
- フィチャはベクターレイヤーで表示する
- フィチャーの主な属性は
 - スタイル（見た目）
 - 位置（緯度・経度）
- 今回は黒い円形を表示する



ベクターレイヤーの初期化

```
var vectorSource = new ol.source.Vector({ features: [] });  
var map = new ol.Map({  
  target: "地図",  
  layers: [  
    new ol.layer.Tile({  
      source: new ol.source.OSM()  
    }),  
    new ol.layer.Vector({  
      source: vectorSource  
    })  
  ],  
  view: new ol.View({ ... }),  
});
```

フィチャー格納
オブジェクト

ベクターレイヤー

フィチャーの導入方法

```
var pinStyle = new ol.style.Style({  
  image: new ol.style.Circle({  
    radius: 8,  
    fill: new ol.style.Fill({color: 'black'}),  
    stroke: new ol.style.Stroke({color: 'white', width: 2})  
  })  
});
```

①スタイルオブジェクトを作成

再利用できる！

```
var iconFeature = new ol.Feature({  
  geometry: new ol.geom.Point(  
    ol.proj.transform([経度, 緯度], 'EPSG:4326', 'EPSG:3857')  
  )  
});  
iconFeature.setStyle(pinStyle);
```

②フィチャーオブジェクトを作成

```
vectorSource.addFeature(iconFeature)|
```

③フィチャー格納オブジェクトに追加

自動的にベクタレイヤが更新される。



演習 3 Step 2

地図にピンを表示させよう

教科書：<http://www.bodic.org/?p=123>

演習 3 Step 1 を書き加えて、地図上にTeapotにある公民館を地図上で黒い円形で表しましょう！

- ・ フィチャスタイルオブジェクトを作成する
- ・ SPARQLで公民館の緯度・経度を取得する（述語は「tpp:緯度」と「tpp:経度」）
- ・ 検索結果毎にフィチャーオブジェクトを作成し、フィチャー格納オブジェクトに追加する

対話機能：クリック

- OpenLayersはイベントベースでクリック等処理する
- 例えば、クリックを処理するための「map.on("click", ...)」を利用する：

```
map.on('click', function(evt) {  
    // クリックした位置にFeatureがあるかどうかを確認  
  
    var feature = map.forEachFeatureAtPixel(evt.pixel,  
        function(feature, layer) {  
            return feature;  
        })  
});
```

今回1個目のフィチャー
しか処理しない

クリックしたところでフィ
チャーがあるかどうかを確認

フィチャー情報格納

```
var feature = new ol.Feature({  
  geometry: new ol.geom.Point(  
    ol.proj.transform([lon, lat], 'EPSG:4326', 'EPSG:3857')  
  ),  
  longitude: ...,  
  latitude: ...,  
  iri: ...  
});
```

緯度・経度・
識別子を格納

```
feature.get("iri")  
feature.get("longitude")
```

フィチャー作成時に
情報を格納できる！

後は「feature.get()」を使う
と、格納した情報を読み出せる



演習 3 Step 3

クリックしてピンを表示しよう

教科書：<http://www.bodic.org/?p=125>

演習 3 Step 2 を書き加えて、公民館ピンをクリックすると識別子・緯度・経度を表示してください。

- ①Webページのどこかに表示するためのエレメント（div等）を追加する
- ②フィチャーオブジェクトに緯度・経度情報を格納する
- ③クリックイベント処理コードを追加する
- ④1で追加したエレメントの中に緯度・経度を表示する（Jquery等を使う）



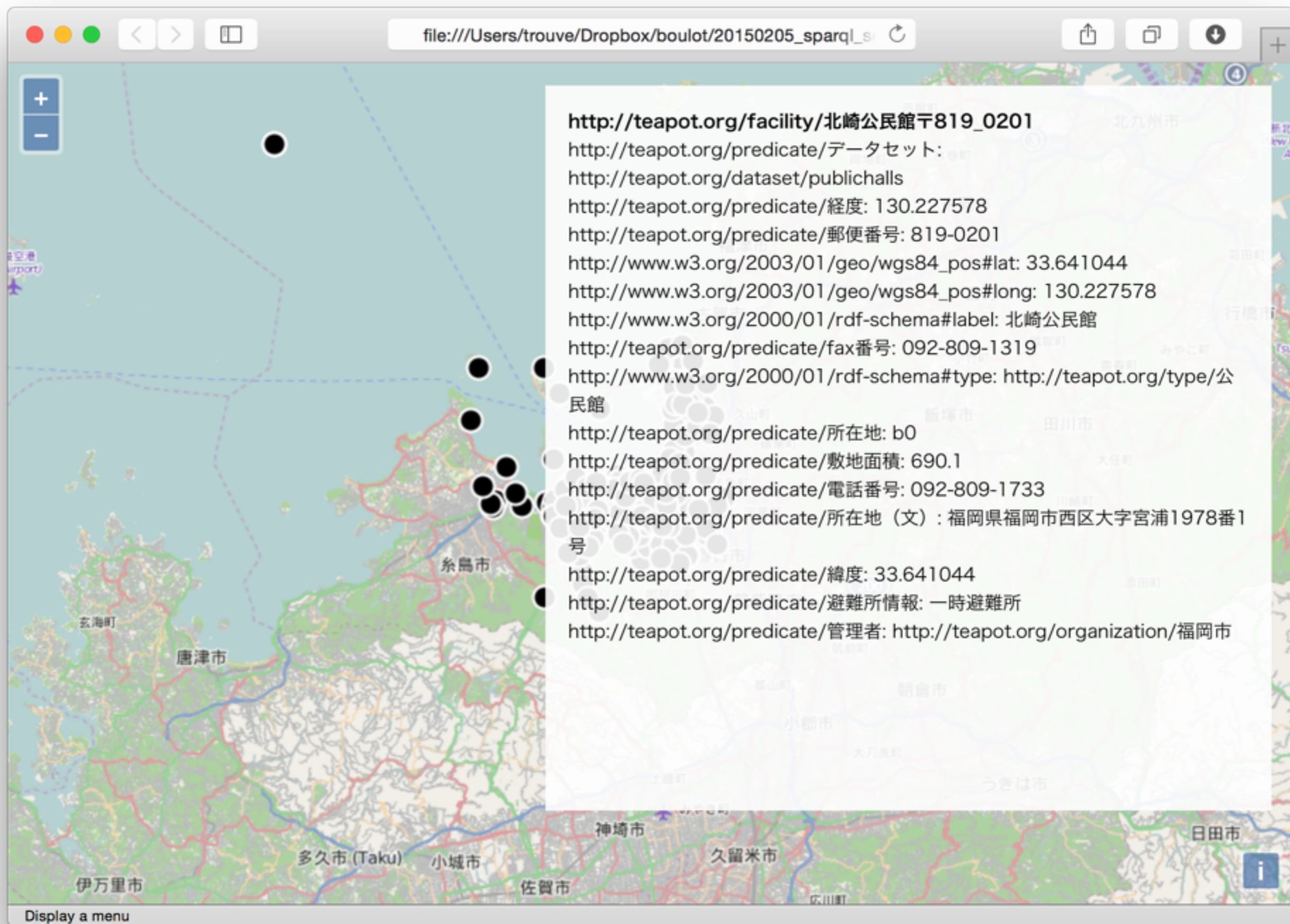
演習 3 Step 4

施設の詳細情報を表示しよう

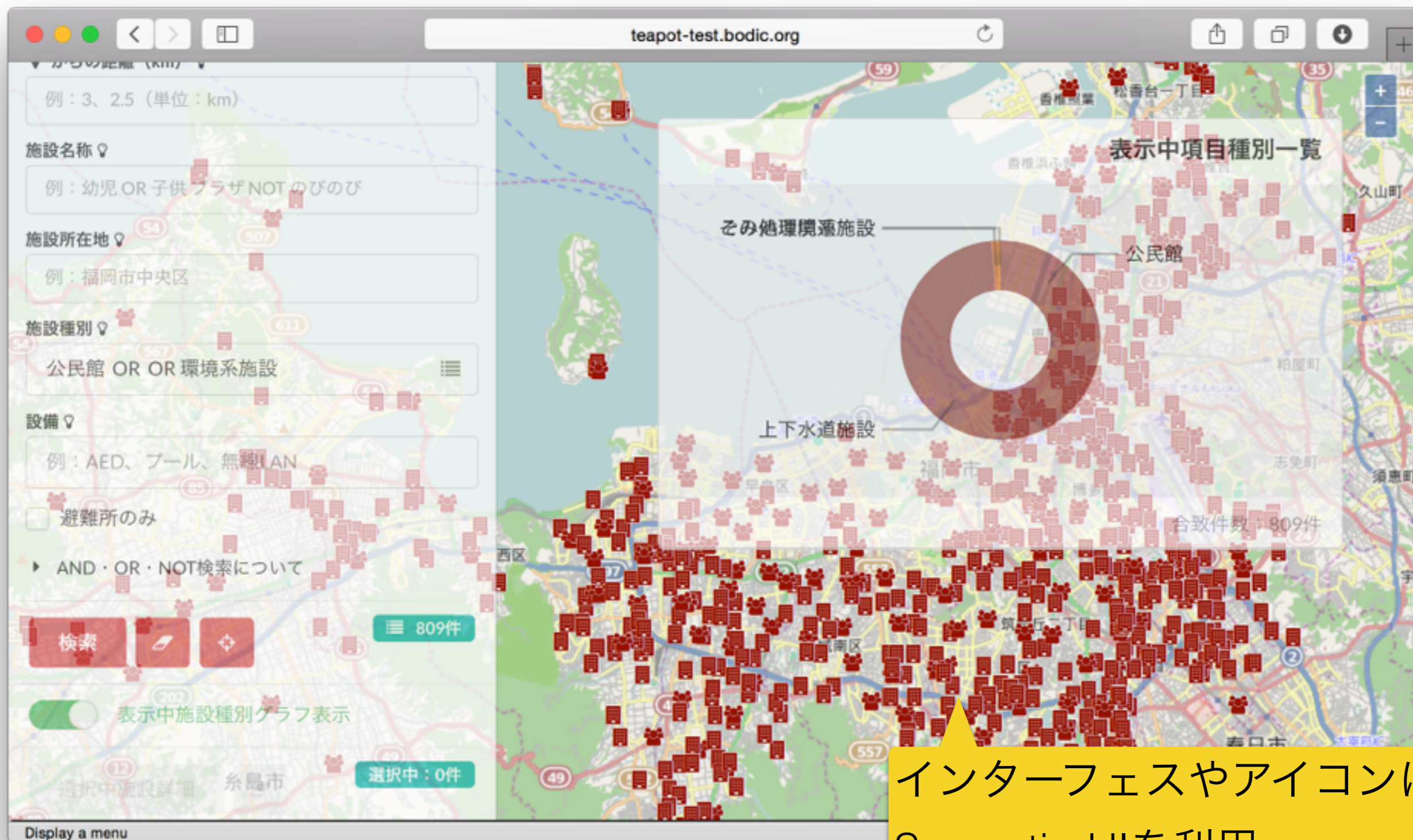
教科書：<http://www.bodic.org/?p=127>

演習 3 Step 3 を書き加えて、公民館ピンをクリックするとTeapotにある全ての情報を表示してください。

- ・ フィチャーをクリックするとTeapotにSPARQLクエリを投げる
- ・ 得られた情報は全て演習Step 3 で追加した情報表示エレメントに表示する。



今回の例を少してを加えれば



インターフェースやアイコンは
Semantic-UIを利用

終わる前に・・・



bodik.jp

OPENDATA CONTEST 2014-2015 FUKUOKA

オープンデータコンテスト 2014-2015 福岡

応募受付開始日	2014年 12月22日(月)
応募締切日	2015年 2月23日(月)
表彰式	2015年 3月6日(金)
募集部門	アプリケーション部門 アイデア部門

[応募フォームはこちら >](#)

ホーム 開催概要 実証概要 応募方法 応募規約 審査・表彰 受賞作品 お問い合わせ

お知らせ

2014.12.22 [「オープンデータ・コンテスト2014-2015 Fukuoka」サイトオープン](#)



開催概要

オープンデータコンテスト2014-2015 Fukuokaの開催概要です。



実証概要

今年度の実証実験の概要をご案内いたします。



ありがとうございました

