

jobsta

Study - Job - Money

Project for
Introduction to Semantic Systems
(188.399-2019W)

Group 01

Cem Bicer (01425692)
Helmuth Breitenfellner (08725866)
Laszlo Kiraly (09227679)
Gerald Weber (00125536)

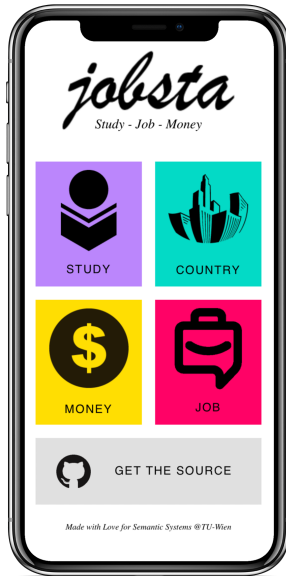
2020-01-31

The logo for 'jobsta' is written in a black, cursive, handwritten-style font. The letters are connected, with a prominent 'j' and a stylized 'a'.

Study - Job - Money

- For Software Developers and Data Scientists
- Asks for experience, age, location
- Answers to following questions:
 - *What shall I study?*
 - *Where shall I work?*
 - *What shall I practise?*
 - *How can I improve?*

The Mobile App



Data Sources

- **Kaggle User Survey**
Data Scientists, Country, Job Role, Programming Language, Income
- **StackOverflow User Survey**
Software Developer, Country, Job Role, Programming Language, Income
- **GitHub Repositories Data**
Repository URL, Popularity, Programming Language, Issues
- **TISS Lectures**
Lectures, Lecturer, Description, Programming Language

Kaggle Survey

- <https://www.kaggle.com/c/kaggle-survey-2019>
- Used Jupyter Notebook for Pre-Processing
- Created RDF XML directly from Python
- **Challenge:** high number of one-hot-encoded values, had to extract unique values

StackOverflow Survey

- <https://insights.stackoverflow.com/survey/2018>
- Used Python for Pre-Processing
- Created instances directly from Python
- **Challenge:** Excel, Numbers and TextMate could all not open the csv file (>90.000 entries) properly

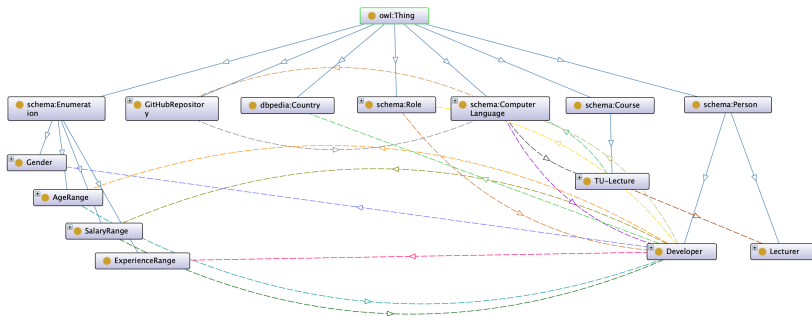
GitHub Repositories Data

- <http://ghtorrent.org/>
- Used Bash & R Script for Pre-Processing
- Created RDF XML directly from Python
- **Challenge:** *huge* data archive (>100GB) had to be filtered / preprocessed

TISS Lectures

- <https://tiss.tuwien.ac.at/course/courseList.xhtml?dswid=6403&dsrid=238>
- Used Python Script
- Created RDF XML directly from Python (using `rdflib`)
- **Challenge:** web scraping, identifying the programming language from text

Ontology #1



Ontology #2

- Created with Protégé
- Reusing existing Ontologies
 - schema.org
 - dbpedia.org
- Entites:
 - Developer
 - dbpedia:Country
 - schema:Course
 - GitHubRepository
 - schema:ComputerLanguage
 - ...
- Object properties:
 - dealsWith
 - developsIn
 - schema:homeLocation
 - ...

Harmonize Data I

- Age Ranges
 - Different Age Ranges
- Salary vs. Salary Range
 - Salary Range in Kaggle
 - Salary Value in Stackoverflow
- Roles
 - Combined from Surveys into List
 - e.g. Frontend Developer -> Software Engineer
 - ... C-Suite Executive -> Manager

Harmonize Data II

- Countries
 - dbpedia linked to external data
- Gender
 - Single Selection in Kaggle
 - Multiple Selections in Stackoverflow
- Computer Language
 - Combined from Surveys into List
 - Field in Github Repository
 - Extracted from TISS Lecture Description

SPARQL Queries #1

“As a developer I live in (Austria) and I want more than (150000 USD per year). What courses at TU Wien deal with programming languages which high-earners are using?”

SPARQL Queries #1 (cont.)

```
1 ▼ PREFIX group1: <http://www.semanticweb.org/sws/ws2019/group1#>
2 PREFIX schema: <http://schema.org/>
3 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
4 PREFIX dbpedia: <http://dbpedia.org/resource/>
5
6 SELECT DISTINCT ?lecture ?language
7 ▼ WHERE {
8     ?lecture group1:dealsWith ?language .
9     ?developer group1:developsIn ?language .
10    ?salaryRange group1:maxSalary ?salary .
11    ?developer group1:hasSalaryRange ?salaryRange .
12    ?developer schema:homeLocation dbpedia:Austria .
13    FILTER (?salary > "150000"^^xsd:integer)
14 }
15 LIMIT 25
```

SPARQL Queries #2

“As a developer I live in (Austria) and I can program in (Python) and I want more than (55000 USD per year). Should I stay or should I go?”

SPARQL Queries #2 (cont.)

```
8 ASK
9 WHERE {
10   ?developer a group1:Developer .
11   ?developer schema:homeLocation ?country .
12   ?developer group1:developsIn ?language .
13 }
14 {
15   SELECT ?country (AVG(?avgRange) as ?averageK)
16   WHERE {
17     ?developer a group1:Developer .
18     ?developer schema:homeLocation ?country .
19     ?developer group1:developsIn ?language .
20     ?developer group1:hasSalaryRange ?salaryRange .
21     ?salaryRange group1:minSalary ?minSalary .
22     ?salaryRange group1:maxSalary ?maxSalary .
23     BIND ((?minSalary + ?maxSalary)/2 AS ?avgRange)
24   }
25   GROUP BY ?country
26 }
27 {
28   SELECT ?country (AVG(?salaryValue) as ?averageS)
29   WHERE {
30     ?developer a group1:Developer .
31     ?developer schema:homeLocation ?country .
32     ?developer group1:developsIn ?language .
33     ?developer group1:salary ?salaryValue .
34   }
35   GROUP BY ?country
36 }
37 BIND ((?averageK + ?averageS)/2 as ?average)
38 FILTER (?language = group1:Python && ?country = dbpedia:Austria && ?average > "55000"^^xsd:integer)
39 }
40 GROUP BY ?country ?average
```


Lessons Learned

- Iterative process to come up with final idea
- Scraping TISS: no ID access to fields
- <http://schema.org> not equal to <http://www.schema.org>
- GraphQL Github API vs. Database Dump
- Harmonizing data can be tedious

Questions?

Thank you for your attention!