

OXPath (Oxford XPath)

Andrew Sellers
Georg Gottlob
Giovanni Grasso
Tim Furche
Christian Schallhart

Outline

- Introduction
- Actions
- Additional Axes
- Web Form Field Association
- Result Extraction
- Architecture

Goals and Motivations

- Provide a declarative formalism that extends XPath
 - Simulate user actions in order to query and manipulate a Dynamic DOM
 - Provide a means for automated systems to specify and maintain Deep Web queries
- Develop a lightweight, scalable tool for building domain-specific knowledge bases from existing, disparate web data

Goals and Motivations

DIADEM

- will generate (by learning) suitable "access paths"
 for retrieving and aggregating data from websites
- On the fly evaluation on thousand of websites and interlinked pages containing forms and menus

Main issues

- Web forms population
- Result pages navigation and data extraction
- Scalability

OXPath

- Extension of XPath
- Facilitates querying web form and retrieving returned data
- Simulates user actions (mouse events/keyboard strokes) for filling out web forms and collecting data from multiple pages
- Highly parallelizable
- Navigation across multiple pages

Actions

- Standard XPath expressions are used for node identification and navigation
- Actions to be taken on the context node are expressed by {..} brackets
- Three type of "actions"
 - Explicit input reference
 - e.g., entering "London" in a text field
 - entering values read from external sources (DB, XML, separate OXPath expression, file)
 - Events
 - submit, click, unclick, mouse events
 - Options selection by index
 - e.g., suggested options in a drop-down list, multiselection

User actions simulation



www.google.com// input[@name='q']/{"Oxford"}/
following::input[@name='btnG']/{click}

A single OXPath expression can also represent several queries (non-ground expression)

www.google.com//input[@name='q']/{"Oxford", "Cambridge", "London"}/ following::input[@name='btnG']/{click}

Its instantiation produces three different ground expressions, asking "Oxford", "Cambridge", and "London", respectively.

In case of multiple selection allowed, it is possible to specify values using inner { } brackets

```
Industries
                                                                4 Industries selected
                                                                Accounting and Auditing Services
By index { {1,4,6,7} }
                                                                Advertising and PR Services
                                                                 Aerospace and Defense
                                                                Agriculture/Forestry/Fishing
                                                                  Architectural and Design Services
                                                                Automotive and Parts Mfg
By values { {"val1",..., "valn"} }
                                                                  Automotive Sales and Repair Services
                                                                 Banking
                                                                  Biotechnology/Pharmaceuticals
                                                                  Broadcasting, Music, and Film
All values { {*} } and {{regex}}
                                                                    usiness Services - Other
                                                                 Deselect all
```

Input can be also provided from external sources by special predicates such as

```
{ fromDB(db url, sql query) }
{ fromFile(url) } { fromRDF(url, sparql query) }
{ fromXML(document url, XPath query) }
```

Additional Axes for OXPath

Facilitate the navigation between form fields

- Next-Field
 - selects the next field, in document order
- Previous-Field (as inverse of Next-Field)
 - selects the previous field, in document order
- Following-Field
 - returns all the following fields, in document order.
- Preceding-Field (as inverse of Following-Field)
 - returns all the preceding fields, in document order

Additional Axes

- Support conditional predicates and node tests
 - /next-field::input[@type=radio]
 - finds the first radio button in document order
- Expressions become far shorter and more intuitively

```
www.google.com//input[@name='q']/{"Oxford"}/
following::input[@name='btnG']/{click}
```



www.google.com//next-field::*/{"Oxford"}/next-field::*/{click}

Form Field Associations

```
to 1200
Rent from:
          800
                           £'s per month
 ...//next-field::*/{ fromFile(myFile)}/
              next-field::*/ { fromFile(myFile)}/
  ...//next-field::*/{ X = fromFile(myFile)}/
               next-field::*/ { fromFile(myFile) > X }/
```

Allows for much more efficient evaluation than Cartesian products of field values

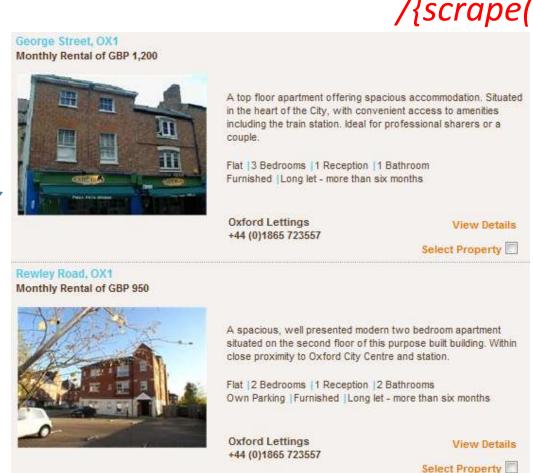
Result Extraction

..../next-field::*/{"Renting"}/.../{...}/.../{"Submit"} /foreach(...)/...
/{scrape(loc)}/...



Atomic results

regardless of presentation (list, table, etc.)



Thank you