OXPath Release Documentation

17 Nov 2011

Thank you for using OXPath. This document accompanies the OXPath release software and provides a brief overview of the API. We hope that you enjoy using OXPath. We welcome feedback, bugs, etc. at `oxpath@diadem-project.info`.

1 Copyright

OXPath is released under a BSD-style copyright license, contained in `license.txt` as well as in the header of each source file.

2 Using OXPath

OXPath is an XPath extension that facilitates data extraction from modern web applications. It is expressive enough to capture web extraction scenarios from nearly all websites, even in the presence of client-side scripting and asynchronous server communication. Details of the language are available in the `oxpath-overview.pdf` file included in this release.

OXPath is currently supported on the Linux 64 and Win 32 platforms due to the dependance on XULRunner. We are working to support additional platforms. Please contact us if you need support for another platform.

2.1 Using the provided OXPath command line tool

We provide a simple OXPath command line tool for executing expressions. It's usage is as follows:

`java -jar oxpath-1.0.jar (mode)? filename`, where mode is `--xml` for XML output or `--simple` for streaming extraction node output (default

is xml). The `filename` parameter is mandatory and contains the OXPath expression to evaluate. All output is streamed to the console.

2.2 Using the OXPath API

The OXPath API allows greater configurability. The main entry class is `OXPathNavigator`. In order to use the API to evaluate an OXPath expression, it requires four things:

1. an OXPath expression (as either a string or file)
2. a browser
3. an output stream
4. a logger

We'll comment on each of these briefly below.

2.2.1 OXPath Expression

The `evaluateOXPathQuery` inputs an OXPath expression as a string, and the `evaluateOXPathQueryFromFile` takes a file name as input.

2.2.2 Browser

Browser objects are instantiated via the DIADEM browser factory as in

```
WebBrowser browser = BrowserFactory.newWebBrowser(Engine.SWT_MOZILLA, true);
```

The first parameter is a member of the `Engine` enum and the second parameter sets if browser is displayed. As of this release, `SWT_MOZILLA` is the only supported browser, but we are working to add support for WebKit. Once finished processing, close this browser and all associated object data with `browser.shutdown();`

2.2.3 Output Stream

OXPath output handlers handle OXPath's streaming output. The available output handlers are in `uk.ac.ox.comlab.diadem.oxpath.core`. The most commonly used will be `OXPathXMLOutputHandler` and `OXPathSimpleOutputHandler`. The OXPath expression requires an `ObjectOutputStream` that is connected via a socket using standard Java library classes. The usual way to set this up is with the following boilerplate. The code below uses the XML output handler from `OXPathNavigator` which, like all block store output handlers, requires a latch to signal when the output is produced:

```
CountDownLatch latch = new CountDownLatch(1);
ServerSocket dataServer = new ServerSocket(0);
OXPathXMLOutputHandler handler =
    new OXPathXMLOutputHandler(
         "localhost",dataServer.getLocalPort(),logger,latch);
handler.start();
Socket listen = dataServer.accept();
ObjectOutputStream os = new ObjectOutputStream(listen.getOutputStream());

<OXPath invokation>

os.close();
```

2.2.4 Logger

This is a slf4j logger. If this object in `null`, one will be created by the evaluator.

3 Building OXPath project from source

The source is provided for OXPath. In order to compile from source, JavaCC is needed to generate the parser and AspectJ crosscuts additional functionality into the parser AST nodes via aspects. We recommend using Eclipse, with the JavaCC and AspectJ plugins, for developing OXPath.

4 Brief description of source packages

The source code for the core engine is written in Java. The source is written to be extensible and readable and takes full advantage of the object-oriented programming paradigm, leveraging polymorphism and encapsulation. The source is organized into the following packages:

`uk.ac.ox.comlab.diadem.oxpath.core` This package contains classes that evaluate OXPath expressions and the main API class: `OXPathNavigator`. The execution engine consists of visitors that preprocess the expression AST and evaluate the expression with the Page-At-A-Time algorithm. Iterative operations are wrapped in a dynamic proxy object that facilitates the memoization feature of the function.

`uk.ac.ox.comlab.diadem.oxpath.core.extraction` This package contains the interface and implementation for extraction via OXPath's extraction semantics to an output stream where it is collected by a listener. Also, to support OXPath's extraction semantics, this implementation is also wrapped with a dynamic proxy object facilitating memoization, so that nodes are only extracted once per label (unique extraction marker).

`uk.ac.ox.comlab.diadem.oxpath.core.state` This package defines the abstract and child classes that are passed as input data to the PAAT visitor calls. The state objects store context, page protection information, and action-free prefix configuration as well as carry values for `position()` and `last()` when appropriate. State objects are constructed using a Builder class and, once constructed, are immutable.

`uk.ac.ox.comlab.diadem.oxpath.dom` This package contains helper classes that assist with DOM processing, in particular form field identification and action simulation.

`uk.ac.ox.comlab.diadem.oxpath.model` This package specifies the data types used in OXPath, including its primitives, context sets, and extraction nodes.

`uk.ac.ox.comlab.diadem.oxpath.model.language` This package contains classes that encode OXPath language ideas such as steps, actions, selectors, etc. These objects are built into the AST by the parser.

`uk.ac.ox.comlab.diadem.oxpath.model.language.functions` This package encodes the OXPath functions, whose type and evaluation are embedded in enumeration types.

`uk.ac.ox.comlab.diadem.oxpath.model.language.operators` This package encodes the OXPath operators, whose type and evaluation are embedded in enumeration types.

`uk.ac.ox.comlab.diadem.oxpath.oxlatin` These classes are in final testing and will be included with a future release.

`uk.ac.ox.comlab.diadem.oxpath.output` This package contains output listeners that organize extraction markers into Pig Latin data types, XML, CSV files, or simple streams.

`uk.ac.ox.comlab.diadem.oxpath.parser` This package contains OXPath parser files generated by JavaCC.

`uk.ac.ox.comlab.diadem.oxpath.parser.ast` This package contains AST files generated by JJTree.

`uk.ac.ox.comlab.diadem.oxpath.parser.generated` The package contains the grammar and parser specification for OXPath. The parser contains the XPath 1.0 grammar supplemented by OXPath's additional features. OXPath Abstract Syntax Trees contain 15 types of nodes that allow for the representation of OXPath expressions. In addition, it contains Java Aspects that cross reference the AST nodes with additional functionality, rather than modify the generated files directly as appears to be common practice with JavaCC.

`uk.ac.ox.comlab.diadem.oxpath.parser.visitor` This package specifies a generic and extensible visitor pattern for OXPath ASTs.

`uk.ac.ox.comlab.diadem.oxpath.utils` This package contains several utility and helper classes for OXPath expression evaluation. In particular, a reusable dynamic proxy implementation handles the memoization features used in PAAT.