Indexing the Ports Tree with Xapian

NYC*BUG -- 06/05/2013

Matthew Story
Director, Axial Corps of Engineers

About Me

- Programming since 1998, professionally since 2005, with Python since 2008.

About Me

- Programming since 1998, professionally since 2005, with Python since 2008.
- B.A. Philosophy, University of Chicago

About Me

- Programming since 1998, professionally since 2005, with Python since 2008.
- B.A. Philosophy, University of Chicago
- Head Engineering at Axial since 2012.

About Me

- Programming since 1998, professionally since 2005, with Python since 2008.
- B.A. Philosophy, University of Chicago
- Head Engineering at Axial since 2012.
- Strong bias towards UNIX
  - FreeBSD contributions (xargs, find, libc)
  - File-System is my storage backend of choice
  - Write A LOT of CLI programs/filters
  - Write A LOT of network daemons (HTTP, TCP and UDP based).

About Me

- Programming since 1998, professionally since 2005, with Python since 2008.
- B.A. Philosophy, University of Chicago
- Head Engineering at Axial since 2012.
- Strong bias towards UNIX
  - FreeBSD contributions (xargs, find, libc)
  - File-System is my storage backend of choice
  - Write A LOT of CLI programs/filters
  - Write A LOT of network daemons (HTTP, TCP and UDP based).
- I Collect Beer and Vinyl

What We'll Cover

- What is Xapian?

What We'll Cover

- What is Xapian?
- The Database
  - WritableDatabase
  - Database

What We'll Cover

- What is Xapian?
- The Database
  - WritableDatabase
  - Database
- Intro to **terms**
  - Indexing
  - Stemming
  - XPREFIX **terms**

What We'll Cover

- What is Xapian?
- The Database
  - WritableDatabase
  - Database
- Intro to **terms**
  - Indexing
  - Stemming
  - XPREFIX **terms**
- Intro to Querying
  - Querying **terms**
  - Query Parser

What is Xapian?

Xapian is a Keyword Indexer and Search library

What is Xapian?

Xapian is a Keyword Indexer and Search library

- Written in C++
  - Available via ports (databases/xapian-core)

What is Xapian?

Xapian is a Keyword Indexer and Search library

- Written in C++
  - Available via ports (databases/xapian-core)
- Open Source
  - license: GPL v.2 (NB: not GPL 3, yay!)
  - xapian.org/download

What is Xapian?

Xapian is a Keyword Indexer and Search library

- Written in C++
  - Available via ports (databases/xapian-core)
- Open Source
  - license: GPL v.2 (NB: not GPL 3, yay!)
  - xapian.org/download
- Actively developed
  - current stable version: 1.2.15 (released: 4/16/2013)

What is Xapian?

Xapian is a Keyword Indexer and Search library

- Written in C++
  - Available via ports (databases/xapian-core)
- Open Source
  - license: GPL v.2 (NB: not GPL 3, yay!)
  - xapian.org/download
- Actively developed
  - current stable version: 1.2.15 (released: 4/16/2013)
- Python bindings via SWIG
  - source: http://xapian.org/docs/bindings/
  - ports: databases/xapian-bindings

What Xapian is Not

A Search Engine Appliance

What Xapian is Not

A Search Engine Appliance

- Not a server (like SOLR/ElasticSearch)

What Xapian is Not

A Search Engine Appliance

- Not a server (like SOLR/ElasticSearch)
- Limited Replication Support

What Xapian is Not

A Search Engine Appliance

- Not a server (like SOLR/ElasticSearch)
- Limited Replication Support
- More flexibility / Programmable Interface
  - Xapian::MatchDecider
  - Xapian::MatchSpy
  - Weight (custom weighting schemes)

What Xapian is Not

Written in Java

What Xapian is Not

Written in Java

- Extremely small footprint (~30MB all-in)

What Xapian is Not

Written in Java

- Extremely small footprint (~30MB all-in)
- No non-base dependencies for xapian-core
  - B-deps/R-deps:

What Xapian is Not

Written in Java

- Extremely small footprint (~30MB all-in)
- No non-base dependencies for xapian-core
  - B-deps/R-deps:
- Very few dependencies for xapian-bindings
  - B-deps/R-deps: gettext-0.18.1.1_1 libexecinfo-1.1_3 libffi-3.0.13 libiconv-1.14_1 libxml2-2.8.0_2 libyaml-0.1.4_2 pkgconf-0.9.2_1 python27-2.7.5 xapian-core-1.2.15,1

The Database

Making a DB is easy ...

```
$ # make a home for the DB
$ python
>>> import xapian as _x
>>> # open if exists, else create and open
>>> sonnet_db = _x.WritableDatabase(
...     '/usr/ports/INDEX.db',
...     _x.DB_CREATE_OR_OPEN)
```

The Database

A Xapian Database is just a directory ...

```
$ tree /usr/ports/INDEX.db
/var/xdb/INDEX.db/
├── flintlock
├── iamchert
├── postlist.baseA
├── postlist.DB
├── record.baseA
├── record.DB
├── termlist.baseA
└── termlist.DB
```

The Database

Things to know about the chert DB

- Single Writer / Multiple Reader
  - flintlock file used with flock(2)

The Database

Things to know about the chert DB

- Single Writer / Multiple Reader
  - flintlock file used with flock(2)
- WritableDatabase is **NOT** threadsafe
  - kludge warning: exec(2) to hold lock
  - Xapian::Database is threadsafe

The Database

Things to know about the chert DB

- Single Writer / Multiple Reader
  - flintlock file used with flock(2)
- WritableDatabase is **NOT** threadsafe
  - kludge warning: exec(2) to hold lock
  - Xapian::Database is threadsafe
- Database must be re-opened after modifications.
  - will raise DatabaseModified error

Indexing

Indexing:

> Parsing a block of text for individual keywords.

Indexing

Indexing:

Parsing a block of text for individual keywords.

Example:

Text: Xapian is Open Source
Terms: ['xapian', 'is', 'open', 'source']

Stemming

Stemming:

> Reducing inflected or derived words to their root.

Stemming

Stemming:

Reducing inflected or derived words to their root.

Example:

Query: probable
Matches: probabilistic, probability

Indexing and Stemming

Xapian provides an indexer with stemming support:

```
import xapian as _x
# set up an indexer with english stemming
indexer = _x.TermGenerator()
indexer.set_stemmer(_x.Stem("english"))
x_doc = _x.Document()
index.set_document(x_doc)
index.index_text('Xapian is Open Source')
```

Term Prefixes

All indexed terms are lowercase. This allows us to use uppercase prefixes to define different dimensions/facets:

```
# index the title, prefixed by 'S'
index.index_text('Indexing the Ports Tree',
                 1, 'S')
```

Term Prefix Convention

Some terms have meaning by convention:

A -- Author
Q -- ID
S -- Title
...

http://xapian.org/docs/omega/termprefixes.html

X-Prefixes

'X' is reserved by convention for custom term-prefixes, so you don't collide with once and future prefixes:

```
# add the maintainer as an X prefix
x_doc.add_term('XMAINTmatt@axial.net')
```

Indexing Demo

To play the demo, clone the portsdemo repo, and follow the portindex example.

Querying

Xapian uses the Query object to build individual queries:

```
import xapian as _x
# Query all ports with manint matt.story
x_query = _x.Query.add_term(
    'XMAINTmatt.story@axial.net')
```

Compound Queries

Xapian uses the same object to combine multiple queries:

```
joined_query = _x.Query(
    _x.Query.OP_AND, x_query, x_query2)
```

Parsing Queries

To stem a Query string, and support Google-style advanced searching, xapian provides the QueryParser class:

```
qp = _x.QueryParser()
stemmer = _x.Stem("english")
qp.set_stemmer(stemmer)
qp.set_database(x_db)
qp.add_prefix('maintainer', 'XMAINT')
x_query2 = qp.parse_query(
    '(bash OR ksh) AND maintainer:obrien*',
    0, prefix)
```

Query Demo

To play the demo, clone the portsdemo repo, and follow the portsearch instructions.

Thanks

matt.story@axial.net
github.com/matthewstory

Axial Corps of Engineers
www.axial.net/about/careers
github.com/axialmarket
axialcorps.wordpress.com