

xapian-core Reference Manual

1.0.0

Generated by Doxygen 1.4.6

Fri May 18 00:18:31 2007

Contents

1	xapian-core Namespace Index	1
1.1	xapian-core Namespace List	1
2	xapian-core Hierarchical Index	3
2.1	xapian-core Class Hierarchy	3
3	xapian-core Class Index	5
3.1	xapian-core Class List	5
4	xapian-core File Index	7
4.1	xapian-core File List	7
5	xapian-core Page Index	9
5.1	xapian-core Related Pages	9
6	xapian-core Namespace Documentation	11
6.1	Xapian Namespace Reference	11
7	xapian-core Class Documentation	21
7.1	Xapian::BM25Weight Class Reference	21
7.2	Xapian::BoolWeight Class Reference	26
7.3	Xapian::Database Class Reference	29
7.4	Xapian::Document Class Reference	36
7.5	Xapian::Enquire Class Reference	41
7.6	Xapian::ErrorHandler Class Reference	51
7.7	Xapian::ESet Class Reference	53
7.8	Xapian::ESetIterator Class Reference	56
7.9	Xapian::ExpandDecider Class Reference	59

7.10	Xapian::ExpandDeciderAnd Class Reference	60
7.11	Xapian::ExpandDeciderFilterTerms Class Reference	62
7.12	Xapian::MatchDecider Class Reference	64
7.13	Xapian::MSet Class Reference	65
7.14	Xapian::MSetIterator Class Reference	72
7.15	Xapian::PositionIterator Class Reference	77
7.16	Xapian::PostingIterator Class Reference	79
7.17	Xapian::Query Class Reference	83
7.18	Xapian::QueryParser Class Reference	84
7.19	Xapian::RSet Class Reference	89
7.20	Xapian::SimpleStopper Class Reference	92
7.21	Xapian::Stem Class Reference	94
7.22	Xapian::Stopper Class Reference	97
7.23	Xapian::TermGenerator Class Reference	99
7.24	Xapian::TermIterator Class Reference	103
7.25	Xapian::TradWeight Class Reference	107
7.26	Xapian::Utf8Iterator Class Reference	111
7.27	Xapian::ValueIterator Class Reference	116
7.28	Xapian::Weight Class Reference	119
7.29	Xapian::WritableDatabase Class Reference	123
8	xapian-core File Documentation	131
8.1	include/xapian/database.h File Reference	131
8.2	include/xapian/dbfactory.h File Reference	134
8.3	include/xapian/document.h File Reference	136
8.4	include/xapian/enquire.h File Reference	138
8.5	include/xapian/errorhandler.h File Reference	140
8.6	include/xapian/expanddecider.h File Reference	141
8.7	include/xapian/positioniterator.h File Reference	142
8.8	include/xapian/postingiterator.h File Reference	144
8.9	include/xapian/query.h File Reference	146
8.10	include/xapian/queryparser.h File Reference	148
8.11	include/xapian/stem.h File Reference	150
8.12	include/xapian/termgenerator.h File Reference	151

8.13	include/xapian/termiterator.h File Reference	152
8.14	include/xapian/types.h File Reference	154
8.15	include/xapian/unicode.h File Reference	156
8.16	include/xapian/valueiterator.h File Reference	159
9	xapian-core Page Documentation	161
9.1	Deprecated List	161

Chapter 1

xapian-core Namespace Index

1.1 xapian-core Namespace List

Here is a list of all documented namespaces with brief descriptions:

[Xapian](#) (The [Xapian](#) library lives in the [Xapian](#) namespace) 11

Chapter 2

xapian-core Hierarchical Index

2.1 xapian-core Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Xapian::Database	29
Xapian::WritableDatabase	123
Xapian::Document	36
Xapian::Enquire	41
Xapian::ErrorHandler	51
Xapian::ESet	53
Xapian::ESetIterator	56
Xapian::ExpandDecider	59
Xapian::ExpandDeciderAnd	60
Xapian::ExpandDeciderFilterTerms	62
Xapian::MatchDecider	64
Xapian::MSet	65
Xapian::MSetIterator	72
Xapian::PositionIterator	77
Xapian::PostingIterator	79
Xapian::Query	83
Xapian::QueryParser	84
Xapian::RSet	89
Xapian::Stem	94
Xapian::Stopper	97
Xapian::SimpleStopper	92
Xapian::TermGenerator	99
Xapian::TermIterator	103
Xapian::Utf8Iterator	111
Xapian::ValueIterator	116
Xapian::Weight	119
Xapian::BM25Weight	21
Xapian::BoolWeight	26

Xapian::TradWeight	107
------------------------------	---------------------

Chapter 3

xapian-core Class Index

3.1 xapian-core Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Xapian::BM25Weight (BM25 weighting scheme)	21
Xapian::BoolWeight (Boolean weighting scheme (everything gets 0))	26
Xapian::Database (This class is used to access a database, or a group of databases)	29
Xapian::Document (A document in the database - holds data, values, terms, and postings)	36
Xapian::Enquire (This class provides an interface to the information retrieval system for the purpose of searching)	41
Xapian::ErrorHandler (Decide if a Xapian::Error exception should be ig- nored)	51
Xapian::ESet (Class representing an ordered set of expand terms (an ESet)) .	53
Xapian::ESetIterator (Iterate through terms in the ESet)	56
Xapian::ExpandDecider (Virtual base class for expand decider functor) . . .	59
Xapian::ExpandDeciderAnd (ExpandDecider subclass which rejects terms using two ExpandDeciders)	60
Xapian::ExpandDeciderFilterTerms (ExpandDecider subclass which rejects terms in a specified list)	62
Xapian::MatchDecider (Base class for matcher decision functor)	64
Xapian::MSet (A match set (MSet))	65
Xapian::MSetIterator (An iterator pointing to items in an MSet)	72
Xapian::PositionIterator (An iterator pointing to items in a list of positions) .	77
Xapian::PostingIterator (An iterator pointing to items in a list of postings) .	79
Xapian::Query (Class representing a query)	83
Xapian::QueryParser (Build a Xapian::Query object from a user query string)	84
Xapian::RSet (A relevance set (R-Set))	89
Xapian::SimpleStopper (Simple implementation of Stopper class - this will suit most users)	92
Xapian::Stem (Class representing a stemming algorithm)	94
Xapian::Stopper (Base class for stop-word decision functor)	97

Xapian::TermGenerator (Parse free text and generate terms)	99
Xapian::TermIterator (An iterator pointing to items in a list of terms)	103
Xapian::TradWeight (Traditional probabilistic weighting scheme)	107
Xapian::Utf8Iterator (An iterator which returns unicode character values from a UTF-8 encoded string)	111
Xapian::ValueIterator (An iterator pointing to values associated with a docu- ment)	116
Xapian::Weight (Abstract base class for weighting schemes)	119
Xapian::WritableDatabase (This class provides read/write access to a database)	123

Chapter 4

xapian-core File Index

4.1 xapian-core File List

Here is a list of all documented files with brief descriptions:

include/xapian.h	??
include/xapian/base.h	??
include/xapian/database.h (API for working with Xapian databases)	131
include/xapian/dbfactory.h (Factory functions for constructing Database and WritableDatabase objects)	134
include/xapian/deprecated.h	??
include/xapian/document.h (API for working with documents)	136
include/xapian/enquire.h (API for running queries)	138
include/xapian/errorhandler.h (Decide if a Xapian::Error exception should be ignored)	140
include/xapian/expanddecider.h (Allow rejection of terms during ESet generation)	141
include/xapian/positioniterator.h (Classes for iterating through position lists)	142
include/xapian/postingiterator.h (Classes for iterating through posting lists)	144
include/xapian/query.h (Classes for representing a query)	146
include/xapian/queryparser.h (Parsing a user query string to build a Xapian::Query object)	148
include/xapian/stem.h (Stemming algorithms)	150
include/xapian/termgenerator.h (Parse free text and generate terms)	151
include/xapian/termiterator.h (Classes for iterating through term lists)	152
include/xapian/types.h (Typedefs for Xapian)	154
include/xapian/unicode.h (Unicode and UTF-8 related classes and functions)	156
include/xapian/valueiterator.h (Classes for iterating through values)	159
include/xapian/visibility.h	??

Chapter 5

xapian-core Page Index

5.1 xapian-core Related Pages

Here is a list of all related documentation pages:

Deprecated List [161](#)

Chapter 6

xapian-core Namespace Documentation

6.1 Xapian Namespace Reference

The [Xapian](#) library lives in the [Xapian](#) namespace.

Classes

- class [Database](#)
This class is used to access a database, or a group of databases.
- class [WritableDatabase](#)
This class provides read/write access to a database.
- class [Document](#)
A document in the database - holds data, values, terms, and postings.
- class [MSet](#)
A match set ([MSet](#)).
- class [MSetIterator](#)
An iterator pointing to items in an [MSet](#).
- class [ESet](#)
Class representing an ordered set of expand terms (an [ESet](#)).
- class [ESetIterator](#)
Iterate through terms in the [ESet](#).
- class [RSet](#)

A relevance set (R-Set).

- class [MatchDecider](#)

Base class for matcher decision functor.

- class [Enquire](#)

This class provides an interface to the information retrieval system for the purpose of searching.

- class [Weight](#)

Abstract base class for weighting schemes.

- class [BoolWeight](#)

Boolean weighting scheme (everything gets 0).

- class [BM25Weight](#)

BM25 weighting scheme.

- class [TradWeight](#)

Traditional probabilistic weighting scheme.

- class [ErrorHandler](#)

Decide if a Xapian::Error exception should be ignored.

- class [ExpandDecider](#)

Virtual base class for expand decider functor.

- class [ExpandDeciderAnd](#)

[ExpandDecider](#) subclass which rejects terms using two [ExpandDeciders](#).

- class [ExpandDeciderFilterTerms](#)

[ExpandDecider](#) subclass which rejects terms in a specified list.

- class **[TermPosWrapper](#)**

- class [PositionIterator](#)

An iterator pointing to items in a list of positions.

- class **[DocIDWrapper](#)**

- class [PostingIterator](#)

An iterator pointing to items in a list of postings.

- class [Query](#)

Class representing a query.

- class [Stopper](#)

Base class for stop-word decision functor.

- class [SimpleStopper](#)
Simple implementation of [Stopper](#) class - this will suit most users.
- struct **ValueRangeProcessor**
- class **StringValueRangeProcessor**
- class **DateValueRangeProcessor**
- class **NumberValueRangeProcessor**
- class [QueryParser](#)
Build a Xapian::Query object from a user query string.
- class [Stem](#)
Class representing a stemming algorithm.
- class [TermGenerator](#)
Parse free text and generate terms.
- class **TermNameWrapper**
- class [TermIterator](#)
An iterator pointing to items in a list of terms.
- class [Utf8Iterator](#)
An iterator which returns unicode character values from a UTF-8 encoded string.
- class [ValueIterator](#)
An iterator pointing to values associated with a document.

Typedefs

- typedef unsigned [doccount](#)
A count of documents.
- typedef int [doccount_diff](#)
A signed difference between two counts of documents.
- typedef unsigned [docid](#)
A unique identifier for a document.
- typedef double [doclength](#)
A normalised document length.
- typedef int [percent](#)
The percentage score for a document in an [MSet](#).
- typedef unsigned [termcount](#)

A counts of terms.

- typedef int [termcount_diff](#)
A signed difference between two counts of terms.
- typedef unsigned [termpos](#)
A term position within a document or query.
- typedef int [termpos_diff](#)
A signed difference between two term positions.
- typedef unsigned [timeout](#)
A timeout value in microseconds.
- typedef unsigned [valueno](#)
The number for a value slot in a document.
- typedef int [valueno_diff](#)
A signed difference between two value slot numbers.
- typedef double [weight](#)
The weight of a document or term.

Functions

- bool **operator==** (const [MSetIterator](#) &a, const [MSetIterator](#) &b)
- bool **operator!=** (const [MSetIterator](#) &a, const [MSetIterator](#) &b)
- bool **operator==** (const [ESetIterator](#) &a, const [ESetIterator](#) &b)
- bool **operator!=** (const [ESetIterator](#) &a, const [ESetIterator](#) &b)
- bool **operator==** (const [PositionIterator](#) &a, const [PositionIterator](#) &b)
Test equality of two PositionIterators.
- bool **operator!=** (const [PositionIterator](#) &a, const [PositionIterator](#) &b)
Test inequality of two PositionIterators.
- bool **operator==** (const [PostingIterator](#) &a, const [PostingIterator](#) &b)
Test equality of two PostingIterators.
- bool **operator!=** (const [PostingIterator](#) &a, const [PostingIterator](#) &b)
Test inequality of two PostingIterators.
- bool **operator==** (const [TermIterator](#) &a, const [TermIterator](#) &b)
- bool **operator!=** (const [TermIterator](#) &a, const [TermIterator](#) &b)
- bool **operator==** (const [ValueIterator](#) &a, const [ValueIterator](#) &b)
- bool **operator!=** (const [ValueIterator](#) &a, const [ValueIterator](#) &b)

- XAPIAN_VISIBILITY_DEFAULT const char * [version_string](#) ()
Report the version string of the library which the program is linked with.
- XAPIAN_VISIBILITY_DEFAULT [XAPIAN_DEPRECATED](#) (const char *xapian_version_string())
For compatibility with [Xapian](#) 0.9.5 and earlier.
- XAPIAN_VISIBILITY_DEFAULT int [major_version](#) ()
Report the major version of the library which the program is linked to.
- XAPIAN_VISIBILITY_DEFAULT [XAPIAN_DEPRECATED](#) (int xapian_major_version())
For compatibility with [Xapian](#) 0.9.5 and earlier.
- XAPIAN_VISIBILITY_DEFAULT int [minor_version](#) ()
Report the minor version of the library which the program is linked to.
- XAPIAN_VISIBILITY_DEFAULT int [revision](#) ()
Report the revision of the library which the program is linked to.

Variables

- const int [DB_CREATE_OR_OPEN](#) = 1
Open for read/write; create if no db exists.
- const int [DB_CREATE](#) = 2
Create a new database; fail if db exists.
- const int [DB_CREATE_OR_OVERWRITE](#) = 3
Overwrite existing db; create if none exists.
- const int [DB_OPEN](#) = 4
Open for read/write; fail if no db exists.
- const [valueno](#) [BAD_VALUENO](#) = static_cast<[valueno](#)>(-1)
Reserved value to indicate "no valueno".

6.1.1 Detailed Description

The [Xapian](#) library lives in the [Xapian](#) namespace.

6.1.2 Typedef Documentation

6.1.2.1 typedef unsigned [Xapian::doccount](#)

A count of documents.

This is used to hold values such as the number of documents in a database and the frequency of a term in the database.

6.1.2.2 typedef int [Xapian::doccount_diff](#)

A signed difference between two counts of documents.

This is used by the [Xapian](#) classes which are STL containers of documents for "difference_type".

6.1.2.3 typedef unsigned [Xapian::docid](#)

A unique identifier for a document.

Docid 0 is invalid, providing an "out of range" value which can be used to mean "not a valid document".

6.1.2.4 typedef double [Xapian::doclength](#)

A normalised document length.

The normalised document length is the document length divided by the average document length in the database.

6.1.2.5 typedef int [Xapian::percent](#)

The percentage score for a document in an [MSet](#).

6.1.2.6 typedef unsigned [Xapian::termcount](#)

A counts of terms.

This is used to hold values such as the Within [Document](#) Frequency (wdf).

6.1.2.7 typedef int [Xapian::termcount_diff](#)

A signed difference between two counts of terms.

This is used by the [Xapian](#) classes which are STL containers of terms for "difference_type".

6.1.2.8 typedef unsigned [Xapian::termpos](#)

A term position within a document or query.

6.1.2.9 typedef int [Xapian::termpos_diff](#)

A signed difference between two term positions.

This is used by the [Xapian](#) classes which are STL containers of positions for "difference_type".

6.1.2.10 typedef unsigned [Xapian::timeout](#)

A timeout value in microseconds.

There are 1 million microseconds in a second, so for example, to set a timeout of 5 seconds use 5000000.

6.1.2.11 typedef unsigned [Xapian::valueno](#)

The number for a value slot in a document.

Any value slot number except [Xapian::BAD_VALUENO](#) is valid.

6.1.2.12 typedef int [Xapian::valueno_diff](#)

A signed difference between two value slot numbers.

This is used by the [Xapian](#) classes which are STL containers of values for "difference_type".

6.1.2.13 typedef double [Xapian::weight](#)

The weight of a document or term.

6.1.3 Function Documentation**6.1.3.1 [XAPIAN_VISIBILITY_DEFAULT](#) int [Xapian::major_version](#) ()**

Report the major version of the library which the program is linked to.

This may be different to the version compiled against (given by [XAPIAN_MAJOR_VERSION](#)) if shared libraries are being used.

6.1.3.2 [XAPIAN_VISIBILITY_DEFAULT](#) int [Xapian::minor_version](#) ()

Report the minor version of the library which the program is linked to.

This may be different to the version compiled against (given by XAPIAN_MINOR_VERSION) if shared libraries are being used.

6.1.3.3 `bool Xapian::operator!= (const PostingIterator & a, const PostingIterator & b) [inline]`

Test inequality of two PostingIterators.

6.1.3.4 `bool Xapian::operator!= (const PositionIterator & a, const PositionIterator & b) [inline]`

Test inequality of two PositionIterators.

6.1.3.5 `bool Xapian::operator== (const PostingIterator & a, const PostingIterator & b) [inline]`

Test equality of two PostingIterators.

6.1.3.6 `bool Xapian::operator== (const PositionIterator & a, const PositionIterator & b) [inline]`

Test equality of two PositionIterators.

6.1.3.7 `XAPIAN_VISIBILITY_DEFAULT int Xapian::revision ()`

Report the revision of the library which the program is linked to.

This may be different to the version compiled against (given by XAPIAN_REVISION) if shared libraries are being used.

6.1.3.8 `XAPIAN_VISIBILITY_DEFAULT const char* Xapian::version_string ()`

Report the version string of the library which the program is linked with.

This may be different to the version compiled against (given by XAPIAN_VERSION) if shared libraries are being used.

6.1.3.9 `XAPIAN_VISIBILITY_DEFAULT Xapian::XAPIAN_DEPRECATED (int xapian_major_version())`

For compatibility with [Xapian](#) 0.9.5 and earlier.

Deprecated

This function is now deprecated, use `Xapian::major_version()` instead.

6.1.3.10 XAPIAN_VISIBILITY_DEFAULT Xapian::XAPIAN_DEPRECATED (const char * *xapian_version_string*())

For compatibility with [Xapian](#) 0.9.5 and earlier.

Deprecated

This function is now deprecated, use [Xapian::version_string\(\)](#) instead.

6.1.4 Variable Documentation

6.1.4.1 const [valueno](#) Xapian::BAD_VALUENO = static_cast<[valueno](#)>(-1)

Reserved value to indicate "no valueno".

6.1.4.2 const int Xapian::DB_CREATE = 2

Create a new database; fail if db exists.

6.1.4.3 const int Xapian::DB_CREATE_OR_OPEN = 1

Open for read/write; create if no db exists.

6.1.4.4 const int Xapian::DB_CREATE_OR_OVERWRITE = 3

Overwrite existing db; create if none exists.

6.1.4.5 const int Xapian::DB_OPEN = 4

Open for read/write; fail if no db exists.

Chapter 7

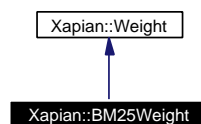
xapian-core Class Documentation

7.1 Xapian::BM25Weight Class Reference

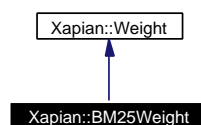
BM25 weighting scheme.

```
#include <enquire.h>
```

Inheritance diagram for Xapian::BM25Weight:



Collaboration diagram for Xapian::BM25Weight:



Public Member Functions

- [BM25Weight](#) (double k1_, double k2_, double k3_, double b_, double min_normlen_)
Construct a BM25 weight.
- [BM25Weight](#) * [clone](#) () const

Return a new weight object of this type.

- `std::string name () const`
Name of the weighting scheme.
- `std::string serialise () const`
Serialise object parameters into a string.
- `BM25Weight * unserialise (const std::string &s) const`
Create object given string serialisation returned by `serialise()`.
- `Xapian::weight get_sumpart (Xapian::termcount wdf, Xapian::doclength len) const`
Get a weight which is part of the sum over terms being performed.
- `Xapian::weight get_maxpart () const`
Gets the maximum value that `get_sumpart()` may return.
- `Xapian::weight get_sumextra (Xapian::doclength len) const`
Get an extra weight for a document to add to the sum calculated over the query terms.
- `Xapian::weight get_maxextra () const`
Gets the maximum value that `get_sumextra()` may return.
- `bool get_sumpart_needs_doclength () const`
return false if the weight object doesn't need doclength

7.1.1 Detailed Description

BM25 weighting scheme.

BM25 weighting options : The BM25 formula is

$$\frac{k_2 \cdot n_q}{1 + L_d} + \sum_t \frac{(k_3 + 1)q_t}{k_3 + q_t} \cdot \frac{(k_1 + 1)f_{t,d}}{k_1((1 - b) + bL_d) + f_{t,d}} \cdot w_t$$

where

- w_t is the termweight of term t
- $f_{t,d}$ is the within document frequency of term t in document d
- q_t is the within query frequency of term t
- L_d is the normalised length of document d
- n_q is the size of the query
- k_1 , k_2 , k_3 and b are user specified parameters

7.1.2 Constructor & Destructor Documentation

7.1.2.1 Xapian::BM25Weight::BM25Weight (double *k1_*, double *k2_*, double *k3_*, double *b_*, double *min_normlen_*) [inline]

Construct a BM25 weight.

Parameters:

- k1* governs the importance of within document frequency. Must be ≥ 0 . 0 means ignore wdf. Default is 1.
- k2* compensation factor for the high wdf values in large documents. Must be ≥ 0 . 0 means no compensation. Default is 0.
- k3* governs the importance of within query frequency. Must be ≥ 0 . 0 means ignore wqf. Default is 1.
- b* Relative importance of within document frequency and document length. Must be ≥ 0 and ≤ 1 . Default is 0.5.
- min_normlen* specifies a cutoff on the minimum value that can be used for a normalised document length - smaller values will be forced up to this cutoff. This prevents very small documents getting a huge bonus weight. Default is 0.5.

7.1.3 Member Function Documentation

7.1.3.1 [BM25Weight*](#) Xapian::BM25Weight::clone () const [virtual]

Return a new weight object of this type.

A subclass called FooWeight taking parameters param1 and param2 should implement this as:

```
virtual FooWeight * clone() const { return new FooWeight(param1, param2); }
```

Implements [Xapian::Weight](#).

7.1.3.2 [Xapian::weight](#) Xapian::BM25Weight::get_maxextra () const [virtual]

Gets the maximum value that [get_sumextra\(\)](#) may return.

This is used in optimising searches.

Implements [Xapian::Weight](#).

7.1.3.3 [Xapian::weight](#) Xapian::BM25Weight::get_maxpart () const [virtual]

Gets the maximum value that [get_sumpart\(\)](#) may return.

This is used in optimising searches, by having the postlist tree decay appropriately when parts of it can have limited, or no, further effect.

Implements [Xapian::Weight](#).

7.1.3.4 [Xapian::weight](#) [Xapian::BM25Weight::get_sumextra](#) ([Xapian::doclength](#) *len*) const [virtual]

Get an extra weight for a document to add to the sum calculated over the query terms.

This returns a weight for a given document, and is used by some weighting schemes to account for influence such as document length.

Parameters:

len the (unnormalised) document length.

Implements [Xapian::Weight](#).

7.1.3.5 [Xapian::weight](#) [Xapian::BM25Weight::get_sumpart](#) ([Xapian::termcount](#) *wdf*, [Xapian::doclength](#) *len*) const [virtual]

Get a weight which is part of the sum over terms being performed.

This returns a weight for a given term and document. These weights are summed to give a total weight for the document.

Parameters:

wdf the within document frequency of the term.

len the (unnormalised) document length.

Implements [Xapian::Weight](#).

7.1.3.6 [bool](#) [Xapian::BM25Weight::get_sumpart_needs_doclength](#) () const [virtual]

return false if the weight object doesn't need doclength

Reimplemented from [Xapian::Weight](#).

7.1.3.7 [std::string](#) [Xapian::BM25Weight::name](#) () const [virtual]

Name of the weighting scheme.

If the subclass is called FooWeight, this should return "Foo".

Implements [Xapian::Weight](#).

7.1.3.8 [std::string](#) [Xapian::BM25Weight::serialise](#) () const [virtual]

Serialise object parameters into a string.

Implements [Xapian::Weight](#).

7.1.3.9 [BM25Weight](#)* Xapian::BM25Weight::unserialise (const std::string & s) const [virtual]

Create object given string serialisation returned by [serialise\(\)](#).

Implements [Xapian::Weight](#).

The documentation for this class was generated from the following file:

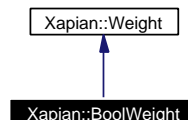
- include/xapian/[enquire.h](#)

7.2 Xapian::BoolWeight Class Reference

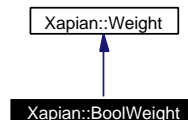
Boolean weighting scheme (everything gets 0).

```
#include <enquire.h>
```

Inheritance diagram for Xapian::BoolWeight:



Collaboration diagram for Xapian::BoolWeight:



Public Member Functions

- [BoolWeight](#) * [clone](#) () const
Return a new weight object of this type.
- std::string [name](#) () const
Name of the weighting scheme.
- std::string [serialise](#) () const
Serialise object parameters into a string.
- [BoolWeight](#) * [unserialise](#) (const std::string &s) const
Create object given string serialisation returned by [serialise\(\)](#).
- [Xapian::weight](#) [get_sumpart](#) ([Xapian::termcount](#) wdf, [Xapian::doclength](#) len) const
Get a weight which is part of the sum over terms being performed.
- [Xapian::weight](#) [get_maxpart](#) () const
Gets the maximum value that [get_sumpart\(\)](#) may return.
- [Xapian::weight](#) [get_sumextra](#) ([Xapian::doclength](#) len) const
Get an extra weight for a document to add to the sum calculated over the query terms.

- [Xapian::weight get_maxextra \(\)](#) const
Gets the maximum value that [get_sumextra\(\)](#) may return.
- [bool get_sumpart_needs_doclength \(\)](#) const
return false if the weight object doesn't need doclength

7.2.1 Detailed Description

Boolean weighting scheme (everything gets 0).

7.2.2 Member Function Documentation

7.2.2.1 [BoolWeight*](#) [Xapian::BoolWeight::clone \(\)](#) const [virtual]

Return a new weight object of this type.

A subclass called FooWeight taking parameters param1 and param2 should implement this as:

```
virtual FooWeight * clone\(\) const { return new FooWeight(param1, param2); }
```

Implements [Xapian::Weight](#).

7.2.2.2 [Xapian::weight](#) [Xapian::BoolWeight::get_maxextra \(\)](#) const [virtual]

Gets the maximum value that [get_sumextra\(\)](#) may return.

This is used in optimising searches.

Implements [Xapian::Weight](#).

7.2.2.3 [Xapian::weight](#) [Xapian::BoolWeight::get_maxpart \(\)](#) const [virtual]

Gets the maximum value that [get_sumpart\(\)](#) may return.

This is used in optimising searches, by having the postlist tree decay appropriately when parts of it can have limited, or no, further effect.

Implements [Xapian::Weight](#).

7.2.2.4 [Xapian::weight](#) [Xapian::BoolWeight::get_sumextra \(Xapian::doclength len\)](#) const [virtual]

Get an extra weight for a document to add to the sum calculated over the query terms.

This returns a weight for a given document, and is used by some weighting schemes to account for influence such as document length.

Parameters:

len the (unnormalised) document length.

Implements [Xapian::Weight](#).

7.2.2.5 **`Xapian::weight Xapian::BoolWeight::get_sumpart (Xapian::termcount wdf, Xapian::doclength len) const`** [virtual]

Get a weight which is part of the sum over terms being performed.

This returns a weight for a given term and document. These weights are summed to give a total weight for the document.

Parameters:

wdf the within document frequency of the term.

len the (unnormalised) document length.

Implements [Xapian::Weight](#).

7.2.2.6 **`bool Xapian::BoolWeight::get_sumpart_needs_doclength () const`** [virtual]

return false if the weight object doesn't need doclength

Reimplemented from [Xapian::Weight](#).

7.2.2.7 **`std::string Xapian::BoolWeight::name () const`** [virtual]

Name of the weighting scheme.

If the subclass is called FooWeight, this should return "Foo".

Implements [Xapian::Weight](#).

7.2.2.8 **`std::string Xapian::BoolWeight::serialise () const`** [virtual]

Serialise object parameters into a string.

Implements [Xapian::Weight](#).

7.2.2.9 **`BoolWeight* Xapian::BoolWeight::unserialise (const std::string & s) const`** [virtual]

Create object given string serialisation returned by [serialise\(\)](#).

Implements [Xapian::Weight](#).

The documentation for this class was generated from the following file:

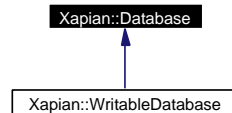
- [include/xapian/enquire.h](#)

7.3 Xapian::Database Class Reference

This class is used to access a database, or a group of databases.

```
#include <database.h>
```

Inheritance diagram for Xapian::Database:



Public Member Functions

- void [add_database](#) (const [Database](#) &database)
Add an existing database (or group of databases) to those accessed by this object.
- [Database](#) ()
Create a [Database](#) with no databases in.
- [Database](#) (const std::string &path)
Open a [Database](#), automatically determining the database backend to use.
- virtual [~Database](#) ()
Destroy this handle on the database.
- [Database](#) (const [Database](#) &other)
Copying is allowed.
- void [operator=](#) (const [Database](#) &other)
Assignment is allowed.
- void [reopen](#) ()
Re-open the database.
- virtual std::string [get_description](#) () const
Introspection method.
- [PostingIterator](#) [postlist_begin](#) (const std::string &tname) const
An iterator pointing to the start of the postlist for a given term.
- [PostingIterator](#) [postlist_end](#) (const std::string &) const
Corresponding end iterator to [postlist_begin](#)().
- [TermIterator](#) [termlist_begin](#) (Xapian::docid did) const

An iterator pointing to the start of the termlist for a given document.

- [TermIterator termlist_end](#) ([Xapian::docid](#)) const
Corresponding end iterator to [termlist_begin\(\)](#).
- bool [has_positions](#) () const
Does this database have any positional information?
- [PositionIterator positionlist_begin](#) ([Xapian::docid](#) did, const std::string &tname) const
An iterator pointing to the start of the position list for a given term in a given document.
- [PositionIterator positionlist_end](#) ([Xapian::docid](#), const std::string &) const
Corresponding end iterator to [positionlist_begin\(\)](#).
- [TermIterator allterms_begin](#) () const
An iterator which runs across all terms in the database.
- [TermIterator allterms_end](#) () const
Corresponding end iterator to [allterms_begin\(\)](#).
- [Xapian::doccount get_doccount](#) () const
Get the number of documents in the database.
- [Xapian::docid get_lastdocid](#) () const
Get the highest document id which has been used in the database.
- [Xapian::doclength get_avlength](#) () const
Get the average length of the documents in the database.
- [Xapian::doccount get_termfreq](#) (const std::string &tname) const
Get the number of documents in the database indexed by a given term.
- bool [term_exists](#) (const std::string &tname) const
Check if a given term exists in the database.
- [Xapian::termcount get_collection_freq](#) (const std::string &tname) const
Return the total number of occurrences of the given term.
- [Xapian::doclength get_doclength](#) ([Xapian::docid](#) did) const
Get the length of a document.
- void [keep_alive](#) ()
Send a "keep-alive" to remote databases to stop them timing out.

- [Xapian::Document](#) `get_document (Xapian::docid did) const`

Get a document from the database, given its document id.

7.3.1 Detailed Description

This class is used to access a database, or a group of databases.

For searching, this class is used in conjunction with an [Enquire](#) object.

Exceptions:

InvalidArgumentError will be thrown if an invalid argument is supplied, for example, an unknown database type.

DatabaseOpeningError may be thrown if the database cannot be opened (for example, a required file cannot be found).

DatabaseVersionError may be thrown if the database is in an unsupported format (for example, created by a newer version of [Xapian](#) which uses an incompatible format).

7.3.2 Constructor & Destructor Documentation

7.3.2.1 [Xapian::Database::Database](#) ()

Create a [Database](#) with no databases in.

7.3.2.2 [Xapian::Database::Database](#) (const std::string & *path*) [explicit]

Open a [Database](#), automatically determining the database backend to use.

Parameters:

path directory that the database is stored in.

7.3.2.3 virtual [Xapian::Database::~~Database](#) () [virtual]

Destroy this handle on the database.

If there are no copies of this object remaining, the database(s) will be closed.

7.3.2.4 [Xapian::Database::Database](#) (const [Database](#) & *other*)

Copying is allowed.

The internals are reference counted, so copying is cheap.

7.3.3 Member Function Documentation

7.3.3.1 `void Xapian::Database::add_database (const Database & database)`

Add an existing database (or group of databases) to those accessed by this object.

Parameters:

database the database(s) to add.

7.3.3.2 `TermIterator Xapian::Database::allterms_begin () const`

An iterator which runs across all terms in the database.

7.3.3.3 `TermIterator Xapian::Database::allterms_end () const [inline]`

Corresponding end iterator to `allterms_begin()`.

7.3.3.4 `Xapian::doclength Xapian::Database::get_avlength () const`

Get the average length of the documents in the database.

7.3.3.5 `Xapian::termcount Xapian::Database::get_collection_freq (const std::string & tname) const`

Return the total number of occurrences of the given term.

This is the sum of the number of occurrences of the term in each document it indexes: ie, the sum of the within document frequencies of the term.

Parameters:

tname The term whose collection frequency is being requested.

7.3.3.6 `virtual std::string Xapian::Database::get_description () const [virtual]`

Introspection method.

Returns:

A string describing this object.

Reimplemented in `Xapian::WritableDatabase`.

7.3.3.7 `Xapian::doccount Xapian::Database::get_doccount () const`

Get the number of documents in the database.

7.3.3.8 Xapian::doclength Xapian::Database::get_doclength (Xapian::docid *did*) const

Get the length of a document.

7.3.3.9 Xapian::Document Xapian::Database::get_document (Xapian::docid *did*) const

Get a document from the database, given its document id.

This method returns a [Xapian::Document](#) object which provides the information about a document.

Parameters:

did The document id for which to retrieve the data.

Returns:

A [Xapian::Document](#) object containing the document data

Exceptions:

Xapian::DocNotFoundError The document specified could not be found in the database.

7.3.3.10 Xapian::docid Xapian::Database::get_lastdocid () const

Get the highest document id which has been used in the database.

7.3.3.11 Xapian::doccount Xapian::Database::get_termfreq (const std::string & *tname*) const

Get the number of documents in the database indexed by a given term.

7.3.3.12 bool Xapian::Database::has_positions () const

Does this database have any positional information?

7.3.3.13 void Xapian::Database::keep_alive ()

Send a "keep-alive" to remote databases to stop them timing out.

7.3.3.14 void Xapian::Database::operator= (const Database & *other*)

Assignment is allowed.

The internals are reference counted, so assignment is cheap.

7.3.3.15 [PositionIterator](#) **Xapian::Database::positionlist_begin** ([Xapian::docid](#) *did*, const std::string & *tname*) const

An iterator pointing to the start of the position list for a given term in a given document.

7.3.3.16 [PositionIterator](#) **Xapian::Database::positionlist_end** ([Xapian::docid](#), const std::string &) const [inline]

Corresponding end iterator to [positionlist_begin\(\)](#).

7.3.3.17 [PostingIterator](#) **Xapian::Database::postlist_begin** (const std::string & *tname*) const

An iterator pointing to the start of the postlist for a given term.

If the term name is the empty string, the iterator returned will list all the documents in the database. Such an iterator will always return a WDF value of 1, since there is no obvious meaning for this quantity in this case.

7.3.3.18 [PostingIterator](#) **Xapian::Database::postlist_end** (const std::string &) const [inline]

Corresponding end iterator to [postlist_begin\(\)](#).

7.3.3.19 **void Xapian::Database::reopen ()**

Re-open the database.

This re-opens the database(s) to the latest available version(s). It can be used either to make sure the latest results are returned, or to recover from a `Xapian::DatabaseModifiedError`.

7.3.3.20 **bool Xapian::Database::term_exists** (const std::string & *tname*) const

Check if a given term exists in the database.

Return true if and only if the term exists in the database. This is the same as `(get_termfreq(tname) != 0)`, but will often be more efficient.

7.3.3.21 [TermIterator](#) **Xapian::Database::termlist_begin** ([Xapian::docid](#) *did*) const

An iterator pointing to the start of the termlist for a given document.

7.3.3.22 [TermIterator](#) Xapian::Database::termlist_end ([Xapian::docid](#)) const [inline]

Corresponding end iterator to [termlist_begin\(\)](#).

The documentation for this class was generated from the following file:

- [include/xapian/database.h](#)

7.4 Xapian::Document Class Reference

A document in the database - holds data, values, terms, and postings.

```
#include <document.h>
```

Public Member Functions

- [Document](#) (const [Document](#) &other)
Copying is allowed.
- void [operator=](#) (const [Document](#) &other)
Assignment is allowed.
- [Document](#) ()
Make a new empty [Document](#).
- [~Document](#) ()
Destructor.
- std::string [get_value](#) ([Xapian::valueno](#) value) const
Get value by number.
- void [add_value](#) ([Xapian::valueno](#) valueno, const std::string &value)
Add a new value.
- void [remove_value](#) ([Xapian::valueno](#) valueno)
Remove any value with the given number.
- void [clear_values](#) ()
Remove all values associated with the document.
- std::string [get_data](#) () const
Get data stored in the document.
- void [set_data](#) (const std::string &data)
Set data stored in the document.
- void [add_posting](#) (const std::string &tname, [Xapian::termpos](#) tpos, [Xapian::termcount](#) wdfinc=1)
Add an occurrence of a term at a particular position.
- void [add_term](#) (const std::string &tname, [Xapian::termcount](#) wdfinc=1)
Add a term to the document, without positional information.
- void [remove_posting](#) (const std::string &tname, [Xapian::termpos](#) tpos, [Xapian::termcount](#) wdfdec=1)

Remove a posting of a term from the document.

- void [remove_term](#) (const std::string &name)
Remove a term and all postings associated with it.
- void [clear_terms](#) ()
Remove all terms (and postings) from the document.
- [Xapian::termcount termlist_count](#) () const
Count the terms in this document.
- [TermIterator termlist_begin](#) () const
Iterator for the terms in this document.
- [TermIterator termlist_end](#) () const
Equivalent end iterator for [termlist_begin\(\)](#).
- [Xapian::termcount values_count](#) () const
Count the values in this document.
- [ValueIterator values_begin](#) () const
Iterator for the values in this document.
- [ValueIterator values_end](#) () const
Equivalent end iterator for [values_begin\(\)](#).
- std::string [get_description](#) () const
Introspection method.

7.4.1 Detailed Description

A document in the database - holds data, values, terms, and postings.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 Xapian::Document::Document (const [Document](#) & other)

Copying is allowed.

The internals are reference counted, so copying is cheap.

7.4.2.2 Xapian::Document::Document ()

Make a new empty [Document](#).

7.4.2.3 Xapian::Document::~~Document ()

Destructor.

7.4.3 Member Function Documentation

7.4.3.1 void Xapian::Document::add_posting (const std::string & *tname*, Xapian::termpos *tpos*, Xapian::termcount *wdfinc* = 1)

Add an occurrence of a term at a particular position.

Multiple occurrences of the term at the same position are represented only once in the positional information, but do increase the wdf.

If the term is not already in the document, it will be added to it.

Parameters:

tname The name of the term.

tpos The position of the term.

wdfinc The increment that will be applied to the wdf for this term.

7.4.3.2 void Xapian::Document::add_term (const std::string & *tname*, Xapian::termcount *wdfinc* = 1)

Add a term to the document, without positional information.

Any existing positional information for the term will be left unmodified.

Parameters:

tname The name of the term.

wdfinc The increment that will be applied to the wdf for this term.

7.4.3.3 void Xapian::Document::add_value (Xapian::valueno *valueno*, const std::string & *value*)

Add a new value.

It will replace any existing value with the same number.

7.4.3.4 void Xapian::Document::clear_terms ()

Remove all terms (and postings) from the document.

7.4.3.5 void Xapian::Document::clear_values ()

Remove all values associated with the document.

7.4.3.6 `std::string Xapian::Document::get_data () const`

Get data stored in the document.

This is a potentially expensive operation, and shouldn't normally be used in a match decider functor. Put data for use by match deciders in a value instead.

7.4.3.7 `std::string Xapian::Document::get_description () const`

Introspection method.

Returns:

A string representing this [Document](#).

7.4.3.8 `std::string Xapian::Document::get_value (Xapian::value no value) const`

Get value by number.

Returns an empty string if no value with the given number is present in the document.

Parameters:

value The number of the value.

7.4.3.9 `void Xapian::Document::operator= (const Document & other)`

Assignment is allowed.

The internals are reference counted, so assignment is cheap.

7.4.3.10 `void Xapian::Document::remove_posting (const std::string & tname, Xapian::termpos tpos, Xapian::termcount wdfdec = 1)`

Remove a posting of a term from the document.

Note that the term will still index the document even if all occurrences are removed. To remove a term from a document completely, use [remove_term\(\)](#).

Parameters:

tname The name of the term.

tpos The position of the term.

wdfdec The decrement that will be applied to the wdf when removing this posting. The wdf will not go below the value of 0.

Exceptions:

Xapian::InvalidArgumentError will be thrown if the term is not at the position specified in the position list for this term in this document.

Xapian::InvalidArgumentError will be thrown if the term is not in the document

7.4.3.11 void Xapian::Document::remove_term (const std::string & *tname*)

Remove a term and all postings associated with it.

Parameters:

tname The name of the term.

Exceptions:

Xapian::InvalidArgumentError will be thrown if the term is not in the document

7.4.3.12 void Xapian::Document::remove_value ([Xapian::valueno](#) *valueno*)

Remove any value with the given number.

7.4.3.13 void Xapian::Document::set_data (const std::string & *data*)

Set data stored in the document.

7.4.3.14 [TermIterator](#) Xapian::Document::termlist_begin () const

Iterator for the terms in this document.

7.4.3.15 [Xapian::termcount](#) Xapian::Document::termlist_count () const

Count the terms in this document.

7.4.3.16 [TermIterator](#) Xapian::Document::termlist_end () const [inline]

Equivalent end iterator for [termlist_begin\(\)](#).

7.4.3.17 [ValueIterator](#) Xapian::Document::values_begin () const

Iterator for the values in this document.

7.4.3.18 [Xapian::termcount](#) Xapian::Document::values_count () const

Count the values in this document.

7.4.3.19 [ValueIterator](#) Xapian::Document::values_end () const

Equivalent end iterator for [values_begin\(\)](#).

The documentation for this class was generated from the following file:

- include/xapian/[document.h](#)

7.5 Xapian::Enquire Class Reference

This class provides an interface to the information retrieval system for the purpose of searching.

```
#include <enquire.h>
```

Public Types

- enum **docid_order** { **ASCENDING** = 1, **DESCENDING** = 0, **DONT_CARE** = 2 }

Public Member Functions

- **Enquire** (const **Database** &database, **ErrorHandler** *errorhandler_=0)
*Create a **Xapian::Enquire** object.*
- **~Enquire** ()
*Close the **Xapian::Enquire** object.*
- void **set_query** (const **Xapian::Query** &query, **Xapian::termcount** qlen=0)
Set the query to run.
- const **Xapian::Query** & **get_query** () const
Get the query which has been set.
- void **set_weighting_scheme** (const **Weight** &weight_)
Set the weighting scheme to use for queries.
- void **set_collapse_key** (**Xapian::valueno** collapse_key)
Set the collapse key to use for queries.
- void **set_docid_order** (docid_order order)
*Set the direction in which documents are ordered by document id in the returned **MSet**.*
- void **set_cutoff** (**Xapian::percent** percent_cutoff, **Xapian::weight** weight_cutoff=0)
Set the percentage and/or weight cutoffs.
- void **set_sort_by_relevance** ()
Set the sorting to be by relevance only.
- void **set_sort_by_value** (**Xapian::valueno** sort_key, bool ascending=true)
Set the sorting to be by value only.

- void `set_sort_by_value_then_relevance` (`Xapian::valueno` sort_key, bool ascending=true)
Set the sorting to be by value, then by relevance for documents with the same value.
- void `set_sort_by_relevance_then_value` (`Xapian::valueno` sort_key, bool ascending=true)
Set the sorting to be by relevance then value.
- `MSet` `get_mset` (`Xapian::doccount` first, `Xapian::doccount` maxitems, `Xapian::doccount` checkatleast=0, const `RSet` *omrset=0, const `MatchDecider` *mdecider=0) const
Get (a portion of) the match set for the current query.
- `MSet` `get_mset` (`Xapian::doccount` first, `Xapian::doccount` maxitems, const `RSet` *omrset, const `MatchDecider` *mdecider=0) const
- `XAPIAN_DEPRECATED` (static const int include_query_terms)
Deprecated in Xapian 1.0.0, use INCLUDE_QUERY_TERMS instead.
- `XAPIAN_DEPRECATED` (static const int use_exact_termfreq)
Deprecated in Xapian 1.0.0, use USE_EXACT_TERMREQ instead.
- `ESet` `get_eset` (`Xapian::termcount` maxitems, const `RSet` &omrset, int flags=0, double k=1.0, const `Xapian::ExpandDecider` *edecider=0) const
Get the expand set for the given rset.
- `ESet` `get_eset` (`Xapian::termcount` maxitems, const `RSet` &omrset, const `Xapian::ExpandDecider` *edecider) const
Get the expand set for the given rset.
- `TermIterator` `get_matching_terms_begin` (`Xapian::docid` did) const
Get terms which match a given document, by document id.
- `TermIterator` `get_matching_terms_end` (`Xapian::docid`) const
End iterator corresponding to `get_matching_terms_begin()`.
- `TermIterator` `get_matching_terms_begin` (const `MSetIterator` &it) const
Get terms which match a given document, by match set item.
- `TermIterator` `get_matching_terms_end` (const `MSetIterator` &) const
End iterator corresponding to `get_matching_terms_begin()`.
- void `register_match_decider` (const std::string &name, const `MatchDecider` *mdecider=NULL)
Register a `MatchDecider`.
- std::string `get_description` () const
Introspection method.

Public Attributes

- Xapian::Internal::RefCntPtr< Internal > **internal**

Static Public Attributes

- static const int **INCLUDE_QUERY_TERMS** = 1
- static const int **USE_EXACT_TERMFREQ** = 2

7.5.1 Detailed Description

This class provides an interface to the information retrieval system for the purpose of searching.

Databases are usually opened lazily, so exceptions may not be thrown where you would expect them to be. You should catch Xapian::Error exceptions when calling any method in [Xapian::Enquire](#).

Exceptions:

Xapian::InvalidArgumentError will be thrown if an invalid argument is supplied, for example, an unknown database type.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 Xapian::Enquire::Enquire (const Database & database, ErrorHandler * errorhandler_ = 0) [explicit]

Create a [Xapian::Enquire](#) object.

This specification cannot be changed once the [Xapian::Enquire](#) is opened: you must create a new [Xapian::Enquire](#) object to access a different database, or set of databases.

Parameters:

database Specification of the database or databases to use.

errorhandler_ A pointer to the error handler to use. Ownership of the object pointed to is not assumed by the [Xapian::Enquire](#) object - the user should delete the [Xapian::ErrorHandler](#) object after the [Xapian::Enquire](#) object is deleted. To use no error handler, this parameter should be 0.

7.5.2.2 Xapian::Enquire::~~Enquire ()

Close the [Xapian::Enquire](#) object.

7.5.3 Member Function Documentation

7.5.3.1 `std::string Xapian::Enquire::get_description () const`

Introspection method.

Returns:

A string representing the enquire object.

7.5.3.2 `ESet Xapian::Enquire::get_eset (Xapian::termcount maxitems, const RSet & omrset, const Xapian::ExpandDecider * edecider) const [inline]`

Get the expand set for the given rset.

Parameters:

maxitems the maximum number of items to return.

omrset the relevance set to use when performing the expand operation.

edecider a decision functor to use to decide whether a given term should be put in the [ESet](#)

Returns:

An [ESet](#) object containing the results of the expand.

Exceptions:

Xapian::InvalidArgumentError See class documentation.

7.5.3.3 `ESet Xapian::Enquire::get_eset (Xapian::termcount maxitems, const RSet & omrset, int flags = 0, double k = 1.0, const Xapian::ExpandDecider * edecider = 0) const`

Get the expand set for the given rset.

Parameters:

maxitems the maximum number of items to return.

omrset the relevance set to use when performing the expand operation.

flags zero or more of these values | -ed together:

- `Xapian::Enquire::INCLUDE_QUERY_TERMS` query terms may be returned from expand
- `Xapian::Enquire::USE_EXACT_TERM_FREQ` for multi dbs, calculate the exact termfreq; otherwise an approximation is used which can greatly improve efficiency, but still returns good results.

k the parameter k in the query expansion algorithm (default is 1.0)

edecider a decision functor to use to decide whether a given term should be put in the [ESet](#)

Returns:

An [ESet](#) object containing the results of the expand.

Exceptions:

Xapian::InvalidArgumentError See class documentation.

7.5.3.4 [TermIterator](#) Xapian::Enquire::get_matching_terms_begin (const [MSetIterator](#) & *it*) const

Get terms which match a given document, by match set item.

This method returns the terms in the current query which match the given document.

If the underlying database has suitable support, using this call (rather than passing a [Xapian::docid](#)) will enable the system to ensure that the correct data is returned, and that the document has not been deleted or changed since the query was performed.

Parameters:

it The iterator for which to retrieve the matching terms.

Returns:

An iterator returning the terms which match the document. The terms will be returned (as far as this makes any sense) in the same order as the terms in the query. Terms will not occur more than once, even if they do in the query.

Exceptions:

Xapian::InvalidArgumentError See class documentation.

Xapian::DocNotFoundError The document specified could not be found in the database.

7.5.3.5 [TermIterator](#) Xapian::Enquire::get_matching_terms_begin ([Xapian::docid](#) *did*) const

Get terms which match a given document, by document id.

This method returns the terms in the current query which match the given document.

It is possible for the document to have been removed from the database between the time it is returned in an [MSet](#), and the time that this call is made. If possible, you should specify an [MSetIterator](#) instead of a [Xapian::docid](#), since this will enable database backends with suitable support to prevent this occurring.

Note that a query does not need to have been run in order to make this call.

Parameters:

did The document id for which to retrieve the matching terms.

Returns:

An iterator returning the terms which match the document. The terms will be returned (as far as this makes any sense) in the same order as the terms in the query. Terms will not occur more than once, even if they do in the query.

Exceptions:

Xapian::InvalidArgumentError See class documentation.

Xapian::DocNotFoundError The document specified could not be found in the database.

7.5.3.6 **TermIterator** **Xapian::Enquire::get_matching_terms_end** (const **MSetIterator** &) const [inline]

End iterator corresponding to [get_matching_terms_begin\(\)](#).

7.5.3.7 **TermIterator** **Xapian::Enquire::get_matching_terms_end** (**Xapian::docid**) const [inline]

End iterator corresponding to [get_matching_terms_begin\(\)](#).

7.5.3.8 **MSet** **Xapian::Enquire::get_mset** (**Xapian::doccount** *first*, **Xapian::doccount** *maxitems*, **Xapian::doccount** *checkatleast* = 0, const **RSet** * *omrset* = 0, const **MatchDecider** * *mdecider* = 0) const

Get (a portion of) the match set for the current query.

Parameters:

first the first item in the result set to return. A value of zero corresponds to the first item returned being that with the highest score. A value of 10 corresponds to the first 10 items being ignored, and the returned items starting at the eleventh.

maxitems the maximum number of items to return.

checkatleast the minimum number of items to check. Because the matcher optimises, it won't consider every document which might match, so the total number of matches is estimated. Setting *checkatleast* forces it to consider that many matches and so allows for reliable paging links.

omrset the relevance set to use when performing the query.

mdecider a decision functor to use to decide whether a given document should be put in the [MSet](#)

Returns:

A [Xapian::MSet](#) object containing the results of the query.

Exceptions:

Xapian::InvalidArgumentError See class documentation.

7.5.3.9 const [Xapian::Query](#)& Xapian::Enquire::get_query () const

Get the query which has been set.

This is only valid after [set_query\(\)](#) has been called.

Exceptions:

Xapian::InvalidArgumentError will be thrown if query has not yet been set.

7.5.3.10 void Xapian::Enquire::register_match_decider (const std::string & name, const [MatchDecider](#) * mdecider = NULL)

Register a [MatchDecider](#).

Parameters:

name The name to register this matchdecider as.

mdecider The matchdecider. If omitted, then remove any matchdecider registered with this name.

7.5.3.11 void Xapian::Enquire::set_collapse_key ([Xapian::valueno](#) collapse_key)

Set the collapse key to use for queries.

Parameters:

collapse_key value number to collapse on - at most one [MSet](#) entry with each particular value will be returned.

The entry returned will be the best entry with that particular value (highest weight or highest sorting key).

An example use might be to create a value for each document containing an MD5 hash of the document contents. Then duplicate documents from different sources can be eliminated at search time (it's better to eliminate duplicates at index time, but this may not be always be possible - for example the search may be over more than one [Xapian](#) database).

Another use is to group matches in a particular category (e.g. you might collapse a mailing list search on the Subject: so that there's only one result per discussion thread). In this case you can use [get_collapse_count\(\)](#) to give the user some idea how many other results there are. And if you index the Subject: as a boolean term as well as putting it in a value, you can offer a link to a non-collapsed search restricted to that thread using a boolean filter.

(default is [Xapian::BAD_VALUENO](#) which means no collapsing).

7.5.3.12 void Xapian::Enquire::set_cutoff (Xapian::percent percent_cutoff, Xapian::weight weight_cutoff = 0)

Set the percentage and/or weight cutoffs.

Parameters:

percent_cutoff Minimum percentage score for returned documents. If a document has a lower percentage score than this, it will not appear in the [MSet](#). If your intention is to return only matches which contain all the terms in the query, then it's more efficient to use Xapian::Query::OP_AND instead of Xapian::Query::OP_OR in the query than to use set_cutoff(100). (default 0 => no percentage cut-off).

weight_cutoff Minimum weight for a document to be returned. If a document has a lower score than this, it will not appear in the [MSet](#). It is usually only possible to choose an appropriate weight for cutoff based on the results of a previous run of the same query; this is thus mainly useful for alerting operations. The other potential use is with a user specified weighting scheme. (default 0 => no weight cut-off).

7.5.3.13 void Xapian::Enquire::set_docid_order (docid_order order)

Set the direction in which documents are ordered by document id in the returned [MSet](#).

This order only has an effect on documents which would otherwise have equal rank. For a weighted probabilistic match with no sort value, this means documents with equal weight. For a boolean match, with no sort value, this means all documents. And if a sort value is used, this means documents with equal sort value (and also equal weight if ordering on relevance after the sort).

Parameters:

order This can be:

- Xapian::Enquire::ASCENDING docids sort in ascending order (default)
- Xapian::Enquire::DESCENDING docids sort in descending order
- Xapian::Enquire::DONT_CARE docids sort in whatever order is most efficient for the backend

Note: If you add documents in strict date order, then a boolean search - i.e. `set_weighting_scheme(Xapian::BoolWeight())` - with `set_docid_order(Xapian::Enquire::DESCENDING)` is a very efficient way to perform "sort by date, newest first".

7.5.3.14 void Xapian::Enquire::set_query (const Xapian::Query & query, Xapian::termcount qlen = 0)

Set the query to run.

Parameters:

query the new query to run.

qlen the query length to use in weight calculations - by default the sum of the wqf of all terms is used.

7.5.3.15 void Xapian::Enquire::set_sort_by_relevance ()

Set the sorting to be by relevance only.

This is the default.

7.5.3.16 void Xapian::Enquire::set_sort_by_relevance_then_value (Xapian::value_no sort_key, bool ascending = true)

Set the sorting to be by relevance then value.

NB sorting of values uses a string comparison, so you'll need to store numbers padded with leading zeros or spaces, or with the number of digits prepended.

Note that with the default BM25 weighting scheme parameters, non-identical documents will rarely have the same weight, so this setting will give very similar results to `set_sort_by_relevance()`. It becomes more useful with particular BM25 parameter settings (e.g. `BM25Weight(1,0,1,0,0)`) or custom weighting schemes.

Parameters:

sort_key value number to sort on.

ascending If true, documents values which sort higher by string compare are better. If false, the sort order is reversed. (default true)

7.5.3.17 void Xapian::Enquire::set_sort_by_value (Xapian::value_no sort_key, bool ascending = true)

Set the sorting to be by value only.

NB sorting of values uses a string comparison, so you'll need to store numbers padded with leading zeros or spaces, or with the number of digits prepended.

Parameters:

sort_key value number to sort on.

ascending If true, documents values which sort higher by string compare are better. If false, the sort order is reversed. (default true)

7.5.3.18 void Xapian::Enquire::set_sort_by_value_then_relevance (Xapian::value_no sort_key, bool ascending = true)

Set the sorting to be by value, then by relevance for documents with the same value.

NB sorting of values uses a string comparison, so you'll need to store numbers padded with leading zeros or spaces, or with the number of digits prepended.

Parameters:

sort_key value number to sort on.

ascending If true, documents values which sort higher by string compare are better. If false, the sort order is reversed. (default true)

7.5.3.19 void Xapian::Enquire::set_weighting_scheme (const [Weight](#) & weighting_)

Set the weighting scheme to use for queries.

Parameters:

weight_ the new weighting scheme. If no weighting scheme is specified, the default is BM25 with the default parameters.

7.5.3.20 Xapian::Enquire::XAPIAN_DEPRECATED (static const int use_exact_termfreq)

Deprecated in [Xapian](#) 1.0.0, use USE_EXACT_TERMFREQ instead.

7.5.3.21 Xapian::Enquire::XAPIAN_DEPRECATED (static const int include_query_terms)

Deprecated in [Xapian](#) 1.0.0, use INCLUDE_QUERY_TERMS instead.

The documentation for this class was generated from the following file:

- include/xapian/[enquire.h](#)

7.6 Xapian::ErrorHandler Class Reference

Decide if a Xapian::Error exception should be ignored.

```
#include <errorhandler.h>
```

Public Member Functions

- [ErrorHandler](#) ()
Default constructor.
- virtual [~ErrorHandler](#) ()
We require a virtual destructor because we have virtual methods.
- void [operator\(\)](#) (Xapian::Error &error)
Handle a Xapian::Error object.

7.6.1 Detailed Description

Decide if a Xapian::Error exception should be ignored.

You can create your own subclass of this class and pass in an instance of it when you construct a [Xapian::Enquire](#) object. Xapian::Error exceptions which happen during the match process are passed to this object and it can decide whether they should propagate or whether [Enquire](#) should attempt to continue.

The motivation is to allow searching over remote databases to handle a remote server which has died (both to allow results to be returned, and also so that such errors can be logged and dead servers temporarily removed from use).

7.6.2 Constructor & Destructor Documentation

7.6.2.1 Xapian::ErrorHandler::ErrorHandler () [inline]

Default constructor.

7.6.2.2 virtual Xapian::ErrorHandler::~~ErrorHandler () [virtual]

We require a virtual destructor because we have virtual methods.

7.6.3 Member Function Documentation

7.6.3.1 void Xapian::ErrorHandler::operator() (Xapian::Error & error)

Handle a Xapian::Error object.

This method is called when a Xapian::Error object is thrown and caught inside [Enquire](#). If this is the first [ErrorHandler](#) that the Error has been passed to, then the `handle_error()` virtual method is called, which allows the API user to decide how to handle the error.

Parameters:

error The Xapian::Error object under consideration.

The documentation for this class was generated from the following file:

- `include/xapian/errorhandler.h`

7.7 Xapian::ESet Class Reference

Class representing an ordered set of expand terms (an [ESet](#)).

```
#include <enquire.h>
```

Public Member Functions

- [ESet](#) ()
Construct an empty [ESet](#).
- [~ESet](#) ()
Destructor.
- [ESet](#) (const [ESet](#) &other)
Copying is allowed (and is cheap).
- void [operator=](#) (const [ESet](#) &other)
Assignment is allowed (and is cheap).
- [Xapian::termcount](#) [get_ebound](#) () const
A lower bound on the number of terms which are in the full set of results of the expand.
- [Xapian::termcount](#) [size](#) () const
The number of terms in this E-Set.
- [Xapian::termcount](#) [max_size](#) () const
Required to allow use as an STL container.
- bool [empty](#) () const
Test if this E-Set is empty.
- void [swap](#) ([ESet](#) &other)
Swap the E-Set we point to with another.
- [ESetIterator](#) [begin](#) () const
Iterator for the terms in this E-Set.
- [ESetIterator](#) [end](#) () const
End iterator corresponding to [begin\(\)](#).
- [ESetIterator](#) [back](#) () const
Iterator pointing to the last element of this E-Set.
- [ESetIterator](#) [operator\[\]](#) ([Xapian::termcount](#) i) const
This returns the term at position i in this E-Set.

- `std::string get_description () const`
Introspection method.

Public Attributes

- `Xapian::Internal::RefCntPtr< Internal > internal`

7.7.1 Detailed Description

Class representing an ordered set of expand terms (an [ESet](#)).

This set represents the results of an expand operation, which is performed by [Xapian::Enquire::get_eset\(\)](#).

7.7.2 Constructor & Destructor Documentation

7.7.2.1 `Xapian::ESet::ESet ()`

Construct an empty [ESet](#).

7.7.2.2 `Xapian::ESet::~~ESet ()`

Destructor.

7.7.2.3 `Xapian::ESet::ESet (const ESet & other)`

Copying is allowed (and is cheap).

7.7.3 Member Function Documentation

7.7.3.1 `ESetIterator Xapian::ESet::back () const`

Iterator pointing to the last element of this E-Set.

7.7.3.2 `ESetIterator Xapian::ESet::begin () const`

Iterator for the terms in this E-Set.

7.7.3.3 `bool Xapian::ESet::empty () const`

Test if this E-Set is empty.

7.7.3.4 [ESetIterator](#) Xapian::ESet::end () const

End iterator corresponding to [begin\(\)](#).

7.7.3.5 `std::string` Xapian::ESet::get_description () const

Introspection method.

Returns:

A string representing this [ESet](#).

7.7.3.6 [Xapian::termcount](#) Xapian::ESet::get_ebound () const

A lower bound on the number of terms which are in the full set of results of the expand.

This will be greater than or equal to [size\(\)](#)

7.7.3.7 [Xapian::termcount](#) Xapian::ESet::max_size () const `[inline]`

Required to allow use as an STL container.

7.7.3.8 `void` Xapian::ESet::operator= (const [ESet](#) & *other*)

Assignment is allowed (and is cheap).

7.7.3.9 `]`

[ESetIterator](#) Xapian::ESet::operator[] ([Xapian::termcount](#) *i*) const

This returns the term at position *i* in this E-Set.

7.7.3.10 [Xapian::termcount](#) Xapian::ESet::size () const

The number of terms in this E-Set.

7.7.3.11 `void` Xapian::ESet::swap ([ESet](#) & *other*)

Swap the E-Set we point to with another.

The documentation for this class was generated from the following file:

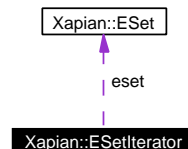
- `include/xapian/enquire.h`

7.8 Xapian::ESetIterator Class Reference

Iterate through terms in the [ESet](#).

```
#include <enquire.h>
```

Collaboration diagram for Xapian::ESetIterator:



Public Types

- typedef std::bidirectional_iterator_tag [iterator_category](#)
Allow use as an STL iterator.
- typedef std::string **value_type**
- typedef [Xapian::termcount_diff](#) **difference_type**
- typedef std::string * **pointer**
- typedef std::string & **reference**

Public Member Functions

- [ESetIterator](#) ()
Create an uninitialised iterator; this cannot be used, but is convenient syntactically.
- [ESetIterator](#) (const [ESetIterator](#) &other)
Copying is allowed (and is cheap).
- void [operator=](#) (const [ESetIterator](#) &other)
Assignment is allowed (and is cheap).
- [ESetIterator](#) & [operator++](#) ()
Advance the iterator.
- [ESetIterator](#) [operator++](#) (int)
Advance the iterator (postfix variant).
- [ESetIterator](#) & [operator--](#) ()
Decrement the iterator.
- [ESetIterator](#) [operator--](#) (int)
Decrement the iterator (postfix variant).

- `const std::string & operator * () const`
Get the term for the current position.
- `Xapian::weight get_weight () const`
Get the weight of the term at the current position.
- `std::string get_description () const`
Returns a string describing this object.

Friends

- class `ESet`
- `bool operator== (const ESetIterator &a, const ESetIterator &b)`
- `bool operator!= (const ESetIterator &a, const ESetIterator &b)`

7.8.1 Detailed Description

Iterate through terms in the `ESet`.

7.8.2 Member Typedef Documentation

7.8.2.1 `typedef std::bidirectional_iterator_tag Xapian::ESetIterator::iterator_category`

Allow use as an STL iterator.

7.8.3 Constructor & Destructor Documentation

7.8.3.1 `Xapian::ESetIterator::ESetIterator () [inline]`

Create an uninitialised iterator; this cannot be used, but is convenient syntactically.

7.8.3.2 `Xapian::ESetIterator::ESetIterator (const ESetIterator & other) [inline]`

Copying is allowed (and is cheap).

7.8.4 Member Function Documentation

7.8.4.1 `std::string Xapian::ESetIterator::get_description () const`

Returns a string describing this object.

Introspection method.

7.8.4.2 **Xapian::weight** Xapian::ESetIterator::get_weight () const

Get the weight of the term at the current position.

7.8.4.3 **const std::string&** Xapian::ESetIterator::operator * () const

Get the term for the current position.

7.8.4.4 **ESetIterator** Xapian::ESetIterator::operator++ (int) [inline]

Advance the iterator (postfix variant).

7.8.4.5 **ESetIterator&** Xapian::ESetIterator::operator++ () [inline]

Advance the iterator.

7.8.4.6 **ESetIterator** Xapian::ESetIterator::operator-- (int) [inline]

Decrement the iterator (postfix variant).

7.8.4.7 **ESetIterator&** Xapian::ESetIterator::operator-- () [inline]

Decrement the iterator.

7.8.4.8 **void** Xapian::ESetIterator::operator= (const **ESetIterator** & *other*) [inline]

Assignment is allowed (and is cheap).

The documentation for this class was generated from the following file:

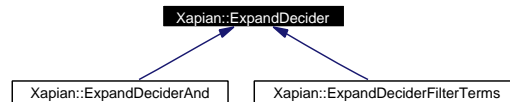
- include/xapian/[enquire.h](#)

7.9 Xapian::ExpandDecider Class Reference

Virtual base class for expand decider functor.

```
#include <expanddecider.h>
```

Inheritance diagram for Xapian::ExpandDecider:



Public Member Functions

- virtual bool [operator\(\)](#) (const std::string &term) const =0
Do we want this term in the [ESet](#)?
- virtual [~ExpandDecider](#) ()
Virtual destructor, because we have virtual methods.

7.9.1 Detailed Description

Virtual base class for expand decider functor.

7.9.2 Constructor & Destructor Documentation

7.9.2.1 virtual Xapian::ExpandDecider::~~ExpandDecider () [virtual]

Virtual destructor, because we have virtual methods.

7.9.3 Member Function Documentation

7.9.3.1 virtual bool Xapian::ExpandDecider::operator() (const std::string &term) const [pure virtual]

Do we want this term in the [ESet](#)?

Implemented in [Xapian::ExpandDeciderAnd](#), and [Xapian::ExpandDeciderFilterTerms](#).

The documentation for this class was generated from the following file:

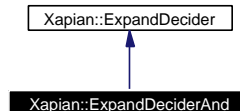
- include/xapian/[expanddecider.h](#)

7.10 Xapian::ExpandDeciderAnd Class Reference

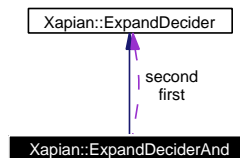
[ExpandDecider](#) subclass which rejects terms using two ExpandDeciders.

```
#include <expanddecider.h>
```

Inheritance diagram for Xapian::ExpandDeciderAnd:



Collaboration diagram for Xapian::ExpandDeciderAnd:



Public Member Functions

- [ExpandDeciderAnd](#) (const [ExpandDecider](#) &first_, const [ExpandDecider](#) &second_)

Terms will be checked with first, and if accepted, then checked with second.

- [ExpandDeciderAnd](#) (const [ExpandDecider](#) *first_, const [ExpandDecider](#) *second_)

Compatibility method.

- virtual bool [operator\(\)](#) (const std::string &term) const

Do we want this term in the [ESet](#)?

7.10.1 Detailed Description

[ExpandDecider](#) subclass which rejects terms using two ExpandDeciders.

Terms are only accepted if they are accepted by both of the specified [ExpandDecider](#) objects.

7.10.2 Constructor & Destructor Documentation

7.10.2.1 `Xapian::ExpandDeciderAnd::ExpandDeciderAnd (const ExpandDecider & first_, const ExpandDecider & second_)` [inline]

Terms will be checked with *first*, and if accepted, then checked with *second*.

7.10.2.2 `Xapian::ExpandDeciderAnd::ExpandDeciderAnd (const ExpandDecider * first_, const ExpandDecider * second_)` [inline]

Compatibility method.

7.10.3 Member Function Documentation

7.10.3.1 `virtual bool Xapian::ExpandDeciderAnd::operator() (const std::string & term) const` [virtual]

Do we want this term in the [ESet](#)?

Implements [Xapian::ExpandDecider](#).

The documentation for this class was generated from the following file:

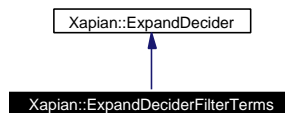
- `include/xapian/expanddecider.h`

7.11 Xapian::ExpandDeciderFilterTerms Class Reference

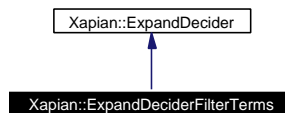
[ExpandDecider](#) subclass which rejects terms in a specified list.

```
#include <expanddecider.h>
```

Inheritance diagram for Xapian::ExpandDeciderFilterTerms:



Collaboration diagram for Xapian::ExpandDeciderFilterTerms:



Public Member Functions

- template<class Iterator> [ExpandDeciderFilterTerms](#) (Iterator reject_begin, Iterator reject_end)
The two iterators specify a list of terms to be rejected.
- virtual bool [operator\(\)](#) (const std::string &term) const
Do we want this term in the [ESet](#)?

7.11.1 Detailed Description

[ExpandDecider](#) subclass which rejects terms in a specified list.

[ExpandDeciderFilterTerms](#) provides an easy way to filter out terms from a fixed list when generating an [ESet](#).

7.11.2 Constructor & Destructor Documentation

- ##### 7.11.2.1
- template<class Iterator> Xapian::ExpandDeciderFilterTerms::ExpandDeciderFilterTerms (Iterator *reject_begin*, Iterator *reject_end*) [inline]

The two iterators specify a list of terms to be rejected.

reject_begin and *reject_end* can be any *input_iterator* type which returns `std::string` or `char *` (e.g. [TermIterator](#) or `char **`).

7.11.3 Member Function Documentation

7.11.3.1 `virtual bool Xapian::ExpandDeciderFilterTerms::operator() (const std::string & term) const` [virtual]

Do we want this term in the [ESet](#)?

Implements [Xapian::ExpandDecider](#).

The documentation for this class was generated from the following file:

- `include/xapian/expanddecider.h`

7.12 Xapian::MatchDecider Class Reference

Base class for matcher decision functor.

```
#include <enquire.h>
```

Public Member Functions

- virtual bool [operator\(\)](#) (const [Xapian::Document](#) &doc) const =0
Decide whether we want this document to be in the [MSet](#).
- virtual [~MatchDecider](#) ()
Destructor.

7.12.1 Detailed Description

Base class for matcher decision functor.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 virtual Xapian::MatchDecider::~~MatchDecider () [virtual]

Destructor.

7.12.3 Member Function Documentation

7.12.3.1 virtual bool Xapian::MatchDecider::operator() (const [Xapian::Document](#) & doc) const [pure virtual]

Decide whether we want this document to be in the [MSet](#).

The documentation for this class was generated from the following file:

- include/xapian/[enquire.h](#)

7.13 Xapian::MSet Class Reference

A match set ([MSet](#)).

```
#include <enquire.h>
```

Public Types

- typedef [MSetIterator](#) [value_type](#)
Allow use as an STL container.
- typedef [MSetIterator](#) [iterator](#)
- typedef [MSetIterator](#) [const_iterator](#)
- typedef [MSetIterator](#) & [reference](#)
- typedef [MSetIterator](#) & [const_reference](#)
- typedef [MSetIterator](#) * [pointer](#)
- typedef [Xapian::doccount_diff](#) [difference_type](#)
- typedef [Xapian::doccount](#) [size_type](#)

Public Member Functions

- [MSet](#) ([MSet::Internal](#) *internal_)
- [MSet](#) ()
Create an empty [Xapian::MSet](#).
- [~MSet](#) ()
Destroy a [Xapian::MSet](#).
- [MSet](#) (const [MSet](#) &other)
Copying is allowed (and is cheap).
- void [operator=](#) (const [MSet](#) &other)
Assignment is allowed (and is cheap).
- void [fetch](#) (const [MSetIterator](#) &begin, const [MSetIterator](#) &end) const
Fetch the document info for a set of items in the [MSet](#).
- void [fetch](#) (const [MSetIterator](#) &item) const
Fetch the single item specified.
- void [fetch](#) () const
Fetch all the items in the [MSet](#).
- [Xapian::percent convert_to_percent](#) ([Xapian::weight](#) wt) const
This converts the weight supplied to a percentage score.
- [Xapian::percent convert_to_percent](#) (const [MSetIterator](#) &it) const

Return the percentage score for a particular item.

- [Xapian::doccount get_termfreq](#) (const std::string &tname) const
Return the term frequency of the given query term.
- [Xapian::weight get_termweight](#) (const std::string &tname) const
Return the term weight of the given query term.
- [Xapian::doccount get_firstitem](#) () const
The index of the first item in the result which was put into the [MSet](#).
- [Xapian::doccount get_matches_lower_bound](#) () const
A lower bound on the number of documents in the database which match the query.
- [Xapian::doccount get_matches_estimated](#) () const
An estimate for the number of documents in the database which match the query.
- [Xapian::doccount get_matches_upper_bound](#) () const
An upper bound on the number of documents in the database which match the query.
- [Xapian::weight get_max_possible](#) () const
The maximum possible weight in the [MSet](#).
- [Xapian::weight get_max_attained](#) () const
The greatest weight which is attained by any document in the database.
- [Xapian::doccount size](#) () const
The number of items in this [MSet](#).
- [Xapian::doccount max_size](#) () const
Required to allow use as an STL container.
- bool [empty](#) () const
Test if this [MSet](#) is empty.
- void [swap](#) ([MSet](#) &other)
Swap the [MSet](#) we point to with another.
- [MSetIterator begin](#) () const
Iterator for the terms in this [MSet](#).
- [MSetIterator end](#) () const
End iterator corresponding to [begin\(\)](#).
- [MSetIterator back](#) () const
Iterator pointing to the last element of this [MSet](#).

- [MSetIterator operator\[\]](#) ([Xapian::doccount](#) i) const
This returns the document at position i in this [MSet](#) object.
- `std::string` [get_description](#) () const
Returns a string representing the [MSet](#).

Public Attributes

- [Xapian::Internal::RefCntPtr< Internal >](#) **internal**

7.13.1 Detailed Description

A match set ([MSet](#)).

This class represents (a portion of) the results of a query.

7.13.2 Member Typedef Documentation

7.13.2.1 `typedef MSetIterator Xapian::MSet::value_type`

Allow use as an STL container.

7.13.3 Constructor & Destructor Documentation

7.13.3.1 `Xapian::MSet::MSet ()`

Create an empty [Xapian::MSet](#).

7.13.3.2 `Xapian::MSet::~~MSet ()`

Destroy a [Xapian::MSet](#).

7.13.3.3 `Xapian::MSet::MSet (const MSet & other)`

Copying is allowed (and is cheap).

7.13.4 Member Function Documentation

7.13.4.1 `MSetIterator Xapian::MSet::back () const`

Iterator pointing to the last element of this [MSet](#).

7.13.4.2 [MSetIterator](#) Xapian::MSet::begin () const

Iterator for the terms in this [MSet](#).

7.13.4.3 [Xapian::percent](#) Xapian::MSet::convert_to_percent (const [MSetIterator](#) & *it*) const

Return the percentage score for a particular item.

7.13.4.4 [Xapian::percent](#) Xapian::MSet::convert_to_percent ([Xapian::weight](#) *wt*) const

This converts the weight supplied to a percentage score.

The return value will be in the range 0 to 100, and will be 0 if and only if the item did not match the query at all.

7.13.4.5 bool Xapian::MSet::empty () const

Test if this [MSet](#) is empty.

7.13.4.6 [MSetIterator](#) Xapian::MSet::end () const

End iterator corresponding to [begin\(\)](#).

7.13.4.7 void Xapian::MSet::fetch () const

Fetch all the items in the [MSet](#).

7.13.4.8 void Xapian::MSet::fetch (const [MSetIterator](#) & *item*) const

Fetch the single item specified.

7.13.4.9 void Xapian::MSet::fetch (const [MSetIterator](#) & *begin*, const [MSetIterator](#) & *end*) const

Fetch the document info for a set of items in the [MSet](#).

This method causes the documents in the range specified by the iterators to be fetched from the database, and cached in the [Xapian::MSet](#) object. This has little effect when performing a search across a local database, but will greatly speed up subsequent access to the document contents when the documents are stored in a remote database.

The iterators must be over this [Xapian::MSet](#) - undefined behaviour will result otherwise.

Parameters:

begin [MSetIterator](#) for first item to fetch.

end [MSetIterator](#) for item after last item to fetch.

7.13.4.10 `std::string Xapian::MSet::get_description () const`

Returns a string representing the [MSet](#).

Introspection method.

7.13.4.11 `Xapian::doccount Xapian::MSet::get_firstitem () const`

The index of the first item in the result which was put into the [MSet](#).

This corresponds to the parameter "first" specified in [Xapian::Enquire::get_mset\(\)](#). A value of 0 corresponds to the highest result being the first item in the [MSet](#).

7.13.4.12 `Xapian::doccount Xapian::MSet::get_matches_estimated () const`

An estimate for the number of documents in the database which match the query.

This figure takes into account collapsing of duplicates, and weighting cutoff values.

This value is returned because there is sometimes a request to display such information. However, our experience is that presenting this value to users causes them to worry about the large number of results, rather than how useful those at the top of the result set are, and is thus undesirable.

7.13.4.13 `Xapian::doccount Xapian::MSet::get_matches_lower_bound () const`

A lower bound on the number of documents in the database which match the query.

This figure takes into account collapsing of duplicates, and weighting cutoff values.

This number is usually considerably less than the actual number of documents which match the query.

7.13.4.14 `Xapian::doccount Xapian::MSet::get_matches_upper_bound () const`

An upper bound on the number of documents in the database which match the query.

This figure takes into account collapsing of duplicates, and weighting cutoff values.

This number is usually considerably greater than the actual number of documents which match the query.

7.13.4.15 `Xapian::weight Xapian::MSet::get_max_attained () const`

The greatest weight which is attained by any document in the database.

If `firstitem == 0`, this is the weight of the first entry in items.

If no documents are found by the query, this will be 0.

Note that calculation of `max_attained` requires calculation of at least one result item - therefore, if no items were requested when the query was performed (by specifying `maxitems = 0` in `Xapian::Enquire::get_mset()`), this value will be 0.

7.13.4.16 `Xapian::weight` `Xapian::MSet::get_max_possible () const`

The maximum possible weight in the `MSet`.

This weight is likely not to be attained in the set of results, but represents an upper bound on the weight which a document could attain for the given query.

7.13.4.17 `Xapian::doccount` `Xapian::MSet::get_termfreq (const std::string & tname) const`

Return the term frequency of the given query term.

Parameters:

tname The term to look for.

Exceptions:

Xapian::InvalidArgumentError is thrown if the term was not in the query.

7.13.4.18 `Xapian::weight` `Xapian::MSet::get_termweight (const std::string & tname) const`

Return the term weight of the given query term.

Parameters:

tname The term to look for.

Exceptions:

Xapian::InvalidArgumentError is thrown if the term was not in the query.

7.13.4.19 `Xapian::doccount` `Xapian::MSet::max_size () const` `[inline]`

Required to allow use as an STL container.

7.13.4.20 `void Xapian::MSet::operator= (const MSet & other)`

Assignment is allowed (and is cheap).

7.13.4.21 `operator[]`

[MSetIterator](#) Xapian::MSet::operator[] ([Xapian::doccount](#) *i*) const

This returns the document at position *i* in this [MSet](#) object.

Note that this is not the same as the document at rank *i* in the query, unless the "first" parameter to [Xapian::Enquire::get_mset](#) was 0. Rather, it is the document at rank *i* + first.

In other words, the offset is into the documents represented by this object, not into the set of documents matching the query.

7.13.4.22 `size`

The number of items in this [MSet](#).

7.13.4.23 `swap`

Swap the [MSet](#) we point to with another.

The documentation for this class was generated from the following file:

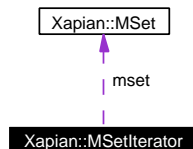
- include/xapian/[enquire.h](#)

7.14 Xapian::MSetIterator Class Reference

An iterator pointing to items in an [MSet](#).

```
#include <enquire.h>
```

Collaboration diagram for Xapian::MSetIterator:



Public Types

- typedef `std::bidirectional_iterator_tag` [iterator_category](#)
Allow use as an STL iterator.
- typedef [Xapian::docid](#) `value_type`
- typedef [Xapian::doccount_diff](#) `difference_type`
- typedef [Xapian::docid](#) * `pointer`
- typedef [Xapian::docid](#) & `reference`

Public Member Functions

- [MSetIterator](#) ()
Create an uninitialised iterator; this cannot be used, but is convenient syntactically.
- [MSetIterator](#) (const [MSetIterator](#) &other)
Copying is allowed (and is cheap).
- void `operator=` (const [MSetIterator](#) &other)
Assignment is allowed (and is cheap).
- [MSetIterator](#) & `operator++` ()
Advance the iterator.
- [MSetIterator](#) `operator++` (int)
Advance the iterator (postfix variant).
- [MSetIterator](#) & `operator--` ()
Decrement the iterator.
- [MSetIterator](#) `operator--` (int)
Decrement the iterator (postfix variant).

- [Xapian::docid operator * \(\)](#) const
Get the document ID for the current position.
- [Xapian::Document get_document \(\)](#) const
Get a [Xapian::Document](#) object for the current position.
- [Xapian::doccount get_rank \(\)](#) const
Get the rank of the document at the current position.
- [Xapian::weight get_weight \(\)](#) const
Get the weight of the document at the current position.
- [std::string get_collapse_key \(\)](#) const
Get the collapse key for this document.
- [Xapian::doccount get_collapse_count \(\)](#) const
Get an estimate of the number of documents that have been collapsed into this one.
- [Xapian::percent get_percent \(\)](#) const
This returns the weight of the document as a percentage score.
- [std::string get_description \(\)](#) const
Returns a string describing this object.

Friends

- class **MSet**
- bool **operator==** (const [MSetIterator](#) &a, const [MSetIterator](#) &b)
- bool **operator!=** (const [MSetIterator](#) &a, const [MSetIterator](#) &b)

7.14.1 Detailed Description

An iterator pointing to items in an [MSet](#).

This is used for access to individual results of a match.

7.14.2 Member Typedef Documentation

7.14.2.1 `typedef std::bidirectional_iterator_tag Xapian::MSetIterator::iterator_category`

Allow use as an STL iterator.

7.14.3 Constructor & Destructor Documentation

7.14.3.1 `Xapian::MSetIterator::MSetIterator ()` [inline]

Create an uninitialised iterator; this cannot be used, but is convenient syntactically.

7.14.3.2 `Xapian::MSetIterator::MSetIterator (const MSetIterator & other)` [inline]

Copying is allowed (and is cheap).

7.14.4 Member Function Documentation

7.14.4.1 `Xapian::doccount Xapian::MSetIterator::get_collapse_count () const`

Get an estimate of the number of documents that have been collapsed into this one.

The estimate will always be less than or equal to the actual number of other documents satisfying the match criteria with the same collapse key as this document.

This method may return 0 even though there are other documents with the same collapse key which satisfying the match criteria. However if this method returns non-zero, there definitely are other such documents. So this method may be used to inform the user that there are "at least N other matches in this group", or to control whether to offer a "show other documents in this group" feature (but note that it may not offer it in every case where it would show other documents).

7.14.4.2 `std::string Xapian::MSetIterator::get_collapse_key () const`

Get the collapse key for this document.

7.14.4.3 `std::string Xapian::MSetIterator::get_description () const`

Returns a string describing this object.

Introspection method.

7.14.4.4 `Xapian::Document Xapian::MSetIterator::get_document () const`

Get a [Xapian::Document](#) object for the current position.

This method returns a [Xapian::Document](#) object which provides the information about the document pointed to by the [MSetIterator](#).

If the underlying database has suitable support, using this call (rather than asking the database for a document based on its document ID) will enable the system to ensure that the correct data is returned, and that the document has not been deleted or changed since the query was performed.

Returns:

A [Xapian::Document](#) object containing the document data.

Exceptions:

Xapian::DocNotFoundError The document specified could not be found in the database.

7.14.4.5 [Xapian::percent](#) Xapian::MSetIterator::get_percent () const

This returns the weight of the document as a percentage score.

The return value will be in the range 0 to 100: 0 meaning that the item did not match the query at all.

7.14.4.6 [Xapian::doccount](#) Xapian::MSetIterator::get_rank () const
[inline]

Get the rank of the document at the current position.

The rank is the position that this document is at in the ordered list of results of the query. The document judged "most relevant" will have rank of 0.

7.14.4.7 [Xapian::weight](#) Xapian::MSetIterator::get_weight () const

Get the weight of the document at the current position.

7.14.4.8 [Xapian::docid](#) Xapian::MSetIterator::operator * () const

Get the document ID for the current position.

7.14.4.9 [MSetIterator](#) Xapian::MSetIterator::operator++ (int) [inline]

Advance the iterator (postfix variant).

7.14.4.10 [MSetIterator&](#) Xapian::MSetIterator::operator++ () [inline]

Advance the iterator.

7.14.4.11 [MSetIterator](#) Xapian::MSetIterator::operator-- (int) [inline]

Decrement the iterator (postfix variant).

7.14.4.12 [MSetIterator&](#) Xapian::MSetIterator::operator-- () [inline]

Decrement the iterator.

7.14.4.13 void Xapian::MSetIterator::operator= (const [MSetIterator](#) & *other*)
[inline]

Assignment is allowed (and is cheap).

The documentation for this class was generated from the following file:

- include/xapian/[enquire.h](#)

7.15 Xapian::PositionIterator Class Reference

An iterator pointing to items in a list of positions.

```
#include <positioniterator.h>
```

Public Types

- typedef std::input_iterator_tag **iterator_category**
- typedef [Xapian::termpos](#) **value_type**
- typedef [Xapian::termpos_diff](#) **difference_type**
- typedef [Xapian::termpos](#) * **pointer**
- typedef [Xapian::termpos](#) & **reference**

Public Member Functions

- **PositionIterator** (Internal *internal_)
- [PositionIterator](#) ()
Default constructor - for declaring an uninitialised iterator.
- [~PositionIterator](#) ()
Destructor.
- [PositionIterator](#) (const [PositionIterator](#) &o)
Copying is allowed.
- void [operator=](#) (const [PositionIterator](#) &o)
Assignment is allowed.
- [Xapian::termpos](#) [operator](#) * () const
- [PositionIterator](#) & [operator](#)++ ()
- TermPosWrapper [operator](#)++ (int)
- void [skip_to](#) ([Xapian::termpos](#) pos)
- std::string [get_description](#) () const
Returns a string describing this object.

Friends

- class **PostingIterator**
- class **TermIterator**
- class **Database**
- bool [operator==](#) (const [PositionIterator](#) &a, const [PositionIterator](#) &b)
Test equality of two PositionIterators.

7.15.1 Detailed Description

An iterator pointing to items in a list of positions.

7.15.2 Constructor & Destructor Documentation

7.15.2.1 Xapian::PositionIterator::PositionIterator ()

Default constructor - for declaring an uninitialised iterator.

7.15.2.2 Xapian::PositionIterator::~~PositionIterator ()

Destructor.

7.15.2.3 Xapian::PositionIterator::PositionIterator (const [PositionIterator](#) & o)

Copying is allowed.

The internals are reference counted, so copying is also cheap.

7.15.3 Member Function Documentation

7.15.3.1 std::string Xapian::PositionIterator::get_description () const

Returns a string describing this object.

Introspection method.

7.15.3.2 void Xapian::PositionIterator::operator= (const [PositionIterator](#) & o)

Assignment is allowed.

The internals are reference counted, so assignment is also cheap.

7.15.4 Friends And Related Function Documentation

7.15.4.1 bool operator== (const [PositionIterator](#) & a, const [PositionIterator](#) & b) [[friend](#)]

Test equality of two PositionIterators.

The documentation for this class was generated from the following file:

- include/xapian/[positioniterator.h](#)

7.16 Xapian::PostingIterator Class Reference

An iterator pointing to items in a list of postings.

```
#include <postingiterator.h>
```

Public Types

- typedef std::input_iterator_tag [iterator_category](#)
Allow use as an STL iterator.
- typedef [Xapian::docid](#) **value_type**
- typedef [Xapian::doccount_diff](#) **difference_type**
- typedef [Xapian::docid](#) * **pointer**
- typedef [Xapian::docid](#) & **reference**

Public Member Functions

- [PostingIterator](#) ()
Default constructor - for declaring an uninitialised iterator.
- [~PostingIterator](#) ()
Destructor.
- [PostingIterator](#) (const [PostingIterator](#) &other)
Copying is allowed.
- void [operator=](#) (const [PostingIterator](#) &other)
Assignment is allowed.
- [PostingIterator](#) & [operator++](#) ()
- DocIDWrapper [operator++](#) (int)
- void [skip_to](#) ([Xapian::docid](#) did)
Skip the iterator to document did, or the first document after did if did isn't in the list of documents being iterated.
- [Xapian::docid](#) [operator *](#) () const
Get the document id at the current position in the postlist.
- [Xapian::doclength](#) [get_doclength](#) () const
Get the length of the document at the current position in the postlist.
- [Xapian::termcount](#) [get_wdf](#) () const
Get the within document frequency of the document at the current position in the postlist.
- [PositionIterator](#) [positionlist_begin](#) () const

Return [PositionIterator](#) pointing to start of positionlist for current document.

- [PositionIterator positionlist_end](#) () const

Return [PositionIterator](#) pointing to end of positionlist for current document.

- std::string [get_description](#) () const

Returns a string describing this object.

Friends

- class **Database**
- bool [operator==](#) (const [PostingIterator](#) &a, const [PostingIterator](#) &b)

Test equality of two *PostingIterators*.

7.16.1 Detailed Description

An iterator pointing to items in a list of postings.

7.16.2 Member Typedef Documentation

7.16.2.1 typedef std::input_iterator_tag [Xapian::PostingIterator::iterator_category](#)

Allow use as an STL iterator.

7.16.3 Constructor & Destructor Documentation

7.16.3.1 [Xapian::PostingIterator::PostingIterator](#) ()

Default constructor - for declaring an uninitialised iterator.

7.16.3.2 [Xapian::PostingIterator::~~PostingIterator](#) ()

Destructor.

7.16.3.3 [Xapian::PostingIterator::PostingIterator](#) (const [PostingIterator](#) &*other*)

Copying is allowed.

The internals are reference counted, so copying is also cheap.

7.16.4 Member Function Documentation

7.16.4.1 `std::string Xapian::PostingIterator::get_description () const`

Returns a string describing this object.

Introspection method.

7.16.4.2 `Xapian::doclength Xapian::PostingIterator::get_doclength () const`

Get the length of the document at the current position in the postlist.

This information may be stored in the postlist, in which case this lookup should be extremely fast (indeed, not require further disk access). If the information is not present in the postlist, it will be retrieved from the database, at a greater performance cost.

7.16.4.3 `Xapian::termcount Xapian::PostingIterator::get_wdf () const`

Get the within document frequency of the document at the current position in the postlist.

7.16.4.4 `Xapian::docid Xapian::PostingIterator::operator * () const`

Get the document id at the current position in the postlist.

7.16.4.5 `void Xapian::PostingIterator::operator= (const PostingIterator & other)`

Assignment is allowed.

The internals are reference counted, so assignment is also cheap.

7.16.4.6 `PositionIterator Xapian::PostingIterator::positionlist_begin () const`

Return `PositionIterator` pointing to start of positionlist for current document.

7.16.4.7 `PositionIterator Xapian::PostingIterator::positionlist_end () const` [inline]

Return `PositionIterator` pointing to end of positionlist for current document.

7.16.4.8 `void Xapian::PostingIterator::skip_to (Xapian::docid did)`

Skip the iterator to document `did`, or the first document after `did` if `did` isn't in the list of documents being iterated.

7.16.5 Friends And Related Function Documentation

7.16.5.1 `bool operator==(const PostingIterator & a, const PostingIterator & b)` [friend]

Test equality of two PostingIterators.

The documentation for this class was generated from the following file:

- `include/xapian/postingiterator.h`

7.17 Xapian::Query Class Reference

Class representing a query.

```
#include <query.h>
```

7.17.1 Detailed Description

Class representing a query.

Queries are represented as a tree of objects.

The documentation for this class was generated from the following file:

- [include/xapian/query.h](#)

7.18 Xapian::QueryParser Class Reference

Build a Xapian::Query object from a user query string.

```
#include <queryparser.h>
```

Public Types

- enum [feature_flag](#) {
 [FLAG_BOOLEAN](#) = 1, [FLAG_PHRASE](#) = 2, [FLAG_LOVEHATE](#) = 4,
 [FLAG_BOOLEAN_ANY_CASE](#) = 8,
 [FLAG_WILDCARD](#) = 16, [FLAG_PURE_NOT](#) = 32, [FLAG_PARTIAL](#) = 64
}
Enum of feature flags.
- enum [stem_strategy](#) { [STEM_NONE](#), [STEM_SOME](#), [STEM_ALL](#) }

Public Member Functions

- [QueryParser](#) (const [QueryParser](#) &o)
Copy constructor.
- [QueryParser](#) & [operator=](#) (const [QueryParser](#) &o)
Assignment.
- [QueryParser](#) ()
Default constructor.
- [~QueryParser](#) ()
Destructor.
- void [set_stemmer](#) (const [Xapian::Stem](#) &stemmer)
Set the stemmer.
- void [set_stemming_strategy](#) (stem_strategy strategy)
Set the stemming strategy.
- void [set_stopper](#) (const [Stopper](#) *stop=NULL)
Set the stopper.
- void [set_default_op](#) (Query::op default_op)
Set the default boolean operator.
- Query::op [get_default_op](#) () const
Get the default boolean operator.

- void [set_database](#) (const [Database](#) &db)
Specify the database being searched.
- [Query parse_query](#) (const std::string &query_string, unsigned flags=FLAG_PHRASE|FLAG_BOOLEAN|FLAG_LOVEHATE, const std::string &default_prefix="")
Parse a query.
- void [add_prefix](#) (const std::string &field, const std::string &prefix)
Add a probabilistic term prefix.
- void [add_boolean_prefix](#) (const std::string &field, const std::string &prefix)
Add a boolean term prefix allowing the user to restrict a search with a boolean filter specified in the free text query.
- [TermIterator stoplist_begin](#) () const
Iterate over terms omitted from the query as stopwords.
- [TermIterator stoplist_end](#) () const
- [TermIterator unstem_begin](#) (const std::string &term) const
Iterate over unstemmed forms of the given (stemmed) term used in the query.
- [TermIterator unstem_end](#) (const std::string &) const
- void [add_valuerangeprocessor](#) (Xapian::ValueRangeProcessor *vrproc)
Register a ValueRangeProcessor.
- std::string [get_description](#) () const
Return a string describing this object.

7.18.1 Detailed Description

Build a Xapian::Query object from a user query string.

7.18.2 Member Enumeration Documentation

7.18.2.1 enum [Xapian::QueryParser::feature_flag](#)

Enum of feature flags.

Enumerator:

FLAG_BOOLEAN Support AND, OR, etc and bracketed subexpressions.

FLAG_PHRASE Support quoted phrases.

FLAG_LOVEHATE Support + and -.

FLAG_BOOLEAN_ANY_CASE Support AND, OR, etc even if they aren't in ALLCAPS.

FLAG_WILDCARD Support right truncation (e.g. Xap*).

NB: You need to tell the [QueryParser](#) object which database to expand wild-cards from using `set_database`.

FLAG_PURE_NOT Allow queries such as 'NOT apples'.

These require the use of a list of all documents in the database which is potentially expensive, so this feature isn't enabled by default.

FLAG_PARTIAL Enable partial matching.

Partial matching causes the parser to treat the query as a "partially entered" search. This will automatically treat the final word as a wildcarded match, unless it is followed by whitespace, to produce more stable results from interactive searches.

7.18.3 Constructor & Destructor Documentation

7.18.3.1 Xapian::QueryParser::QueryParser (const [QueryParser](#) & o)

Copy constructor.

7.18.3.2 Xapian::QueryParser::QueryParser ()

Default constructor.

7.18.3.3 Xapian::QueryParser::~~QueryParser ()

Destructor.

7.18.4 Member Function Documentation

7.18.4.1 void Xapian::QueryParser::add_boolean_prefix (const std::string & *field*, const std::string & *prefix*)

Add a boolean term prefix allowing the user to restrict a search with a boolean filter specified in the free text query.

E.g. `qp.add_boolean_prefix("site", "H");`

Allows the user to restrict a search with `site:xapian.org` which will be converted to `Hxapian.org` combined with any probabilistic query with `OP_FILTER`.

Multiple fields can be mapped to the same prefix (so you can e.g. make `site:` and `domain:` aliases for each other).

Parameters:

field The user visible field name

prefix The term prefix to map this to

7.18.4.2 void Xapian::QueryParser::add_prefix (const std::string & *field*, const std::string & *prefix*)

Add a probabilistic term prefix.

E.g. qp.add_prefix("author", "A");

Allows the user to search for author:orwell which will search for the term "Aorwel" (assuming English stemming is in use). Multiple fields can be mapped to the same prefix (so you can e.g. make title: and subject: aliases for each other).

Parameters:

field The user visible field name

prefix The term prefix to map this to

7.18.4.3 void Xapian::QueryParser::add_valuerangeprocessor (Xapian::ValueRangeProcessor * *vrproc*)

Register a ValueRangeProcessor.

7.18.4.4 Query::op Xapian::QueryParser::get_default_op () const

Get the default boolean operator.

7.18.4.5 std::string Xapian::QueryParser::get_description () const

Return a string describing this object.

7.18.4.6 QueryParser& Xapian::QueryParser::operator= (const QueryParser & *o*)

Assignment.

7.18.4.7 Query Xapian::QueryParser::parse_query (const std::string & *query_string*, unsigned *flags* = FLAG_PHRASE|FLAG_BOOLEAN|FLAG_LOVEHATE, const std::string & *default_prefix* = " ")

Parse a query.

Parameters:

query_string A free-text query as entered by a user

flags Zero or more `Query::feature_flag` specifying what features the [QueryParser](#) should support. Combine multiple values with bitwise-or (`|`).

default_prefix The default term prefix to use (default none). For example, you can pass "A" when parsing an "Author" field.

7.18.4.8 `void Xapian::QueryParser::set_database (const Database & db)`

Specify the database being searched.

7.18.4.9 `void Xapian::QueryParser::set_default_op (Query::op default_op)`

Set the default boolean operator.

7.18.4.10 `void Xapian::QueryParser::set_stemmer (const Xapian::Stem & stemmer)`

Set the stemmer.

7.18.4.11 `void Xapian::QueryParser::set_stemming_strategy (stem_strategy strategy)`

Set the stemming strategy.

7.18.4.12 `void Xapian::QueryParser::set_stopper (const Stopper * stop = NULL)`

Set the stopper.

7.18.4.13 `TermIterator Xapian::QueryParser::stoplist_begin () const`

Iterate over terms omitted from the query as stopwords.

7.18.4.14 `TermIterator Xapian::QueryParser::unstem_begin (const std::string & term) const`

Iterate over unstemmed forms of the given (stemmed) term used in the query.

The documentation for this class was generated from the following file:

- `include/xapian/queryparser.h`

7.19 Xapian::RSet Class Reference

A relevance set (R-Set).

```
#include <enquire.h>
```

Public Member Functions

- [RSet](#) (const [RSet](#) &rset)
Copy constructor.
- void [operator=](#) (const [RSet](#) &rset)
Assignment operator.
- [RSet](#) ()
Default constructor.
- [~RSet](#) ()
Destructor.
- [Xapian::doccount size](#) () const
The number of documents in this R-Set.
- bool [empty](#) () const
Test if this R-Set is empty.
- void [add_document](#) ([Xapian::docid](#) did)
Add a document to the relevance set.
- void [add_document](#) (const [Xapian::MSetIterator](#) &i)
Add a document to the relevance set.
- void [remove_document](#) ([Xapian::docid](#) did)
Remove a document from the relevance set.
- void [remove_document](#) (const [Xapian::MSetIterator](#) &i)
Remove a document from the relevance set.
- bool [contains](#) ([Xapian::docid](#) did) const
Test if a given document in the relevance set.
- bool [contains](#) (const [Xapian::MSetIterator](#) &i) const
Test if a given document in the relevance set.
- std::string [get_description](#) () const
Introspection method.

Public Attributes

- Xapian::Internal::RefCountPtr< Internal > **internal**

7.19.1 Detailed Description

A relevance set (R-Set).

This is the set of documents which are marked as relevant, for use in modifying the term weights, and in performing query expansion.

7.19.2 Constructor & Destructor Documentation

7.19.2.1 Xapian::RSet::RSet (const [RSet](#) & *rset*)

Copy constructor.

7.19.2.2 Xapian::RSet::RSet ()

Default constructor.

7.19.2.3 Xapian::RSet::~~RSet ()

Destructor.

7.19.3 Member Function Documentation

7.19.3.1 void Xapian::RSet::add_document (const [Xapian::MSetIterator](#) & *i*) [inline]

Add a document to the relevance set.

7.19.3.2 void Xapian::RSet::add_document ([Xapian::docid](#) *did*)

Add a document to the relevance set.

7.19.3.3 bool Xapian::RSet::contains (const [Xapian::MSetIterator](#) & *i*) const [inline]

Test if a given document in the relevance set.

7.19.3.4 bool Xapian::RSet::contains ([Xapian::docid](#) *did*) const

Test if a given document in the relevance set.

7.19.3.5 bool Xapian::RSet::empty () const

Test if this R-Set is empty.

7.19.3.6 std::string Xapian::RSet::get_description () const

Introspection method.

Returns:

A string representing this [RSet](#).

7.19.3.7 void Xapian::RSet::operator= (const [RSet](#) & rset)

Assignment operator.

7.19.3.8 void Xapian::RSet::remove_document (const [Xapian::MSetIterator](#) & i) [inline]

Remove a document from the relevance set.

7.19.3.9 void Xapian::RSet::remove_document ([Xapian::docid](#) did)

Remove a document from the relevance set.

7.19.3.10 [Xapian::doccount](#) Xapian::RSet::size () const

The number of documents in this R-Set.

The documentation for this class was generated from the following file:

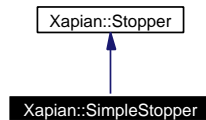
- include/xapian/[enquire.h](#)

7.20 Xapian::SimpleStopper Class Reference

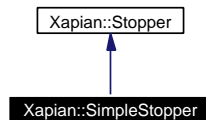
Simple implementation of [Stopper](#) class - this will suit most users.

```
#include <queryparser.h>
```

Inheritance diagram for Xapian::SimpleStopper:



Collaboration diagram for Xapian::SimpleStopper:



Public Member Functions

- [SimpleStopper](#) ()
Default constructor.
- template<class Iterator> [SimpleStopper](#) (Iterator begin, Iterator end)
Initialise from a pair of iterators.
- void [add](#) (const std::string &word)
Add a single stop word.
- virtual bool [operator\(\)](#) (const std::string &term) const
Is term a stop-word?
- virtual [~SimpleStopper](#) ()
Destructor.
- virtual std::string [get_description](#) () const
Return a string describing this object.

7.20.1 Detailed Description

Simple implementation of [Stopper](#) class - this will suit most users.

7.20.2 Constructor & Destructor Documentation

7.20.2.1 Xapian::SimpleStopper::SimpleStopper () [inline]

Default constructor.

7.20.2.2 template<class Iterator> Xapian::SimpleStopper::SimpleStopper (Iterator *begin*, Iterator *end*) [inline]

Initialise from a pair of iterators.

7.20.2.3 virtual Xapian::SimpleStopper::~SimpleStopper () [inline, virtual]

Destructor.

7.20.3 Member Function Documentation

7.20.3.1 void Xapian::SimpleStopper::add (const std::string & *word*) [inline]

Add a single stop word.

7.20.3.2 virtual std::string Xapian::SimpleStopper::get_description () const [virtual]

Return a string describing this object.

Reimplemented from [Xapian::Stopper](#).

7.20.3.3 virtual bool Xapian::SimpleStopper::operator() (const std::string & *term*) const [inline, virtual]

Is term a stop-word?

Implements [Xapian::Stopper](#).

The documentation for this class was generated from the following file:

- [include/xapian/queryparser.h](#)

7.21 Xapian::Stem Class Reference

Class representing a stemming algorithm.

```
#include <stem.h>
```

Public Member Functions

- [Stem](#) (const [Stem](#) &o)
Copy constructor.
- void [operator=](#) (const [Stem](#) &o)
Assignment.
- [Stem](#) ()
Construct a [Xapian::Stem](#) object which doesn't change terms.
- [Stem](#) (const std::string &language)
Construct a [Xapian::Stem](#) object for a particular language.
- [~Stem](#) ()
Destructor.
- std::string [operator\(\)](#) (const std::string &word) const
[Stem](#) a word.
- std::string [get_description](#) () const
Return a string describing this object.

Static Public Member Functions

- static std::string [get_available_languages](#) ()
Return a list of available languages.

7.21.1 Detailed Description

Class representing a stemming algorithm.

7.21.2 Constructor & Destructor Documentation

7.21.2.1 Xapian::Stem::Stem (const [Stem](#) & o)

Copy constructor.

7.21.2.2 Xapian::Stem::Stem ()

Construct a [Xapian::Stem](#) object which doesn't change terms.

Equivalent to [Stem](#)("none").

7.21.2.3 Xapian::Stem::Stem (const std::string & *language*) [explicit]

Construct a [Xapian::Stem](#) object for a particular language.

Parameters:

language Either the English name for the language or the two letter ISO639 code.

The following language names are understood (aliases follow the name):

- none - don't stem terms
- danish (da)
- dutch (nl)
- english (en) - Martin Porter's 2002 revision of his stemmer
- english_lovins (lovins) - Lovin's stemmer
- english_porter (porter) - Porter's stemmer as described in his 1980 paper
- finnish (fi)
- french (fr)
- german (de)
- italian (it)
- norwegian (no)
- portuguese (pt)
- russian (ru)
- spanish (es)
- swedish (sv)

Exceptions:

Xapian::InvalidArgumentError is thrown if language isn't recognised.

7.21.2.4 Xapian::Stem::~~Stem ()

Destructor.

7.21.3 Member Function Documentation

7.21.3.1 `static std::string Xapian::Stem::get_available_languages ()` `[static]`

Return a list of available languages.

Each stemmer is only included once in the list (not once for each alias). The name included is the English name of the language.

The list is returned as a string, with language names separated by spaces. This is a static method, so a [Xapian::Stem](#) object is not required for this operation.

7.21.3.2 `std::string Xapian::Stem::get_description () const`

Return a string describing this object.

7.21.3.3 `std::string Xapian::Stem::operator() (const std::string & word) const`

[Stem](#) a word.

Parameters:

word a word to stem.

Returns:

the stem

7.21.3.4 `void Xapian::Stem::operator= (const Stem & o)`

Assignment.

The documentation for this class was generated from the following file:

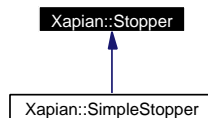
- `include/xapian/stem.h`

7.22 Xapian::Stopper Class Reference

Base class for stop-word decision functor.

```
#include <queryparser.h>
```

Inheritance diagram for Xapian::Stopper:



Public Member Functions

- virtual bool [operator\(\)](#) (const std::string &term) const =0
Is term a stop-word?
- virtual [~Stopper](#) ()
Class has virtual methods, so provide a virtual destructor.
- virtual std::string [get_description](#) () const
Return a string describing this object.

7.22.1 Detailed Description

Base class for stop-word decision functor.

7.22.2 Constructor & Destructor Documentation

7.22.2.1 virtual Xapian::Stopper::~~Stopper () [inline, virtual]

Class has virtual methods, so provide a virtual destructor.

7.22.3 Member Function Documentation

7.22.3.1 virtual std::string Xapian::Stopper::get_description () const [virtual]

Return a string describing this object.

Reimplemented in [Xapian::SimpleStopper](#).

7.22.3.2 `virtual bool Xapian::Stopper::operator() (const std::string & term) const` [pure virtual]

Is term a stop-word?

Implemented in [Xapian::SimpleStopper](#).

The documentation for this class was generated from the following file:

- `include/xapian/queryparser.h`

7.23 Xapian::TermGenerator Class Reference

Parse free text and generate terms.

```
#include <termgenerator.h>
```

Public Member Functions

- [TermGenerator](#) (const [TermGenerator](#) &o)
Copy constructor.
- [TermGenerator](#) & operator= (const [TermGenerator](#) &o)
Assignment.
- [TermGenerator](#) ()
Default constructor.
- [~TermGenerator](#) ()
Destructor.
- void [set_stemmer](#) (const [Xapian::Stem](#) &stemmer)
Set the stemmer.
- void [set_stopper](#) (const [Xapian::Stopper](#) *stop=NULL)
Set the stopper.
- void [set_document](#) (const [Xapian::Document](#) &doc)
Set the current document.
- const [Xapian::Document](#) & [get_document](#) () const
Get the current document.
- void [index_text](#) (const [Xapian::Utf8Iterator](#) &itor, [Xapian::termcount](#) weight=1, const std::string &prefix="")
Index some text.
- void [index_text](#) (const std::string &text, [Xapian::termcount](#) weight=1, const std::string &prefix="")
Index some text in a std::string.
- void [index_text_without_positions](#) (const [Xapian::Utf8Iterator](#) &itor, [Xapian::termcount](#) weight=1, const std::string &prefix="")
Index some text without positional information.
- void [index_text_without_positions](#) (const std::string &text, [Xapian::termcount](#) weight=1, const std::string &prefix="")

Index some text in a `std::string` without positional information.

- void `increase_termpos` (`Xapian::termcount` delta=100)
Increase the termpos used by `index_text` by delta.
- `Xapian::termcount` `get_termpos` () const
Get the current term position.
- void `set_termpos` (`Xapian::termcount` termpos)
Set the current term position.
- `std::string` `get_description` () const
Return a string describing this object.

7.23.1 Detailed Description

Parse free text and generate terms.

7.23.2 Constructor & Destructor Documentation

7.23.2.1 `Xapian::TermGenerator::TermGenerator (const TermGenerator & o)`

Copy constructor.

7.23.2.2 `Xapian::TermGenerator::TermGenerator ()`

Default constructor.

7.23.2.3 `Xapian::TermGenerator::~~TermGenerator ()`

Destructor.

7.23.3 Member Function Documentation

7.23.3.1 `std::string Xapian::TermGenerator::get_description () const`

Return a string describing this object.

7.23.3.2 `const Xapian::Document& Xapian::TermGenerator::get_document () const`

Get the current document.

7.23.3.3 `Xapian::termcount` `Xapian::TermGenerator::get_termpos () const`

Get the current term position.

7.23.3.4 `void Xapian::TermGenerator::increase_termpos (Xapian::termcount
delta = 100)`

Increase the termpos used by `index_text` by *delta*.

This can be used to prevent phrase searches from spanning two unconnected blocks of text (e.g. the title and body text).

7.23.3.5 `void Xapian::TermGenerator::index_text (const std::string & text,
Xapian::termcount weight = 1, const std::string & prefix = " ")
[inline]`

Index some text in a `std::string`.

7.23.3.6 `void Xapian::TermGenerator::index_text (const Xapian::Utf8Iterator
& itor, Xapian::termcount weight = 1, const std::string & prefix = " ")`

Index some text.

7.23.3.7 `void Xapian::TermGenerator::index_text_without_positions (const
std::string & text, Xapian::termcount weight = 1, const std::string &
prefix = " ") [inline]`

Index some text in a `std::string` without positional information.

7.23.3.8 `void Xapian::TermGenerator::index_text_without_positions (const
Xapian::Utf8Iterator & itor, Xapian::termcount weight = 1, const
std::string & prefix = " ")`

Index some text without positional information.

7.23.3.9 `TermGenerator& Xapian::TermGenerator::operator= (const
TermGenerator & o)`

Assignment.

7.23.3.10 `void Xapian::TermGenerator::set_document (const
Xapian::Document & doc)`

Set the current document.

7.23.3.11 void Xapian::TermGenerator::set_stemmer (const [Xapian::Stem](#) & *stemmer*)

Set the stemmer.

7.23.3.12 void Xapian::TermGenerator::set_stopper (const [Xapian::Stopper](#) * *stop* = NULL)

Set the stopper.

7.23.3.13 void Xapian::TermGenerator::set_termpos ([Xapian::termcount](#) *termpos*)

Set the current term position.

The documentation for this class was generated from the following file:

- include/xapian/[termgenerator.h](#)

7.24 Xapian::TermIterator Class Reference

An iterator pointing to items in a list of terms.

```
#include <termiterator.h>
```

Public Types

- typedef std::input_iterator_tag [iterator_category](#)
Allow use as an STL iterator.
- typedef std::string [value_type](#)
- typedef [Xapian::termcount_diff](#) [difference_type](#)
- typedef std::string * [pointer](#)
- typedef std::string & [reference](#)

Public Member Functions

- [TermIterator](#) (Internal *internal_)
- [TermIterator](#) ()
Default constructor - for declaring an uninitialised iterator.
- [~TermIterator](#) ()
Destructor.
- [TermIterator](#) (const [TermIterator](#) &other)
Copying is allowed.
- void [operator=](#) (const [TermIterator](#) &other)
Assignment is allowed.
- std::string [operator *](#) () const
Return the current term.
- [TermIterator](#) & [operator++](#) ()
- TermNameWrapper [operator++](#) (int)
- void [skip_to](#) (const std::string &tname)
Skip the iterator to term tname, or the first term after tname if tname isn't in the list of terms being iterated.
- [Xapian::termcount](#) [get_wdf](#) () const
Return the wdf of the current term (if meaningful).
- [Xapian::doccount](#) [get_termfreq](#) () const
Return the term frequency of the current term (if meaningful).
- [Xapian::termcount](#) [positionlist_count](#) () const

Return length of positionlist for current term.

- [PositionIterator positionlist_begin](#) () const

Return [PositionIterator](#) pointing to start of positionlist for current term.

- [PositionIterator positionlist_end](#) () const

Return [PositionIterator](#) pointing to end of positionlist for current term.

- std::string [get_description](#) () const

Returns a string describing this object.

Public Attributes

- Xapian::Internal::RefCntPtr< Internal > **internal**

7.24.1 Detailed Description

An iterator pointing to items in a list of terms.

7.24.2 Member Typedef Documentation

7.24.2.1 `typedef std::input_iterator_tag Xapian::TermIterator::iterator_category`

Allow use as an STL iterator.

7.24.3 Constructor & Destructor Documentation

7.24.3.1 `Xapian::TermIterator::TermIterator ()`

Default constructor - for declaring an uninitialised iterator.

7.24.3.2 `Xapian::TermIterator::~~TermIterator ()`

Destructor.

7.24.3.3 `Xapian::TermIterator::TermIterator (const TermIterator & other)`

Copying is allowed.

The internals are reference counted, so copying is also cheap.

7.24.4 Member Function Documentation

7.24.4.1 `std::string Xapian::TermIterator::get_description () const`

Returns a string describing this object.

Introspection method.

7.24.4.2 `Xapian::doccount Xapian::TermIterator::get_termfreq () const`

Return the term frequency of the current term (if meaningful).

7.24.4.3 `Xapian::termcount Xapian::TermIterator::get_wdf () const`

Return the wdf of the current term (if meaningful).

7.24.4.4 `std::string Xapian::TermIterator::operator * () const`

Return the current term.

7.24.4.5 `void Xapian::TermIterator::operator= (const TermIterator & other)`

Assignment is allowed.

The internals are reference counted, so assignment is also cheap.

7.24.4.6 `PositionIterator Xapian::TermIterator::positionlist_begin () const`

Return [PositionIterator](#) pointing to start of positionlist for current term.

7.24.4.7 `Xapian::termcount Xapian::TermIterator::positionlist_count () const`

Return length of positionlist for current term.

7.24.4.8 `PositionIterator Xapian::TermIterator::positionlist_end () const` [inline]

Return [PositionIterator](#) pointing to end of positionlist for current term.

7.24.4.9 `void Xapian::TermIterator::skip_to (const std::string & tname)`

Skip the iterator to term tname, or the first term after tname if tname isn't in the list of terms being iterated.

The documentation for this class was generated from the following file:

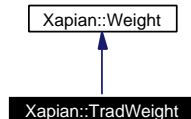
- `include/xapian/termiterator.h`

7.25 Xapian::TradWeight Class Reference

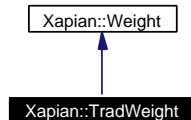
Traditional probabilistic weighting scheme.

```
#include <enquire.h>
```

Inheritance diagram for Xapian::TradWeight:



Collaboration diagram for Xapian::TradWeight:



Public Member Functions

- [TradWeight](#) (double k)
Construct a [TradWeight](#).
- [TradWeight](#) * [clone](#) () const
Return a new weight object of this type.
- std::string [name](#) () const
Name of the weighting scheme.
- std::string [serialise](#) () const
Serialise object parameters into a string.
- [TradWeight](#) * [unserialise](#) (const std::string &s) const
Create object given string serialisation returned by [serialise\(\)](#).
- [Xapian::weight](#) [get_sumpart](#) ([Xapian::termcount](#) wdf, [Xapian::doclength](#) len) const
Get a weight which is part of the sum over terms being performed.
- [Xapian::weight](#) [get_maxpart](#) () const
Gets the maximum value that [get_sumpart\(\)](#) may return.

- [Xapian::weight get_sumextra \(Xapian::doclength len\) const](#)
Get an extra weight for a document to add to the sum calculated over the query terms.
- [Xapian::weight get_maxextra \(\) const](#)
Gets the maximum value that [get_sumextra\(\)](#) may return.
- [bool get_sumpart_needs_doclength \(\) const](#)
return false if the weight object doesn't need doclength

7.25.1 Detailed Description

Traditional probabilistic weighting scheme.

This class implements the Traditional Probabilistic Weighting scheme, as described by the early papers on Probabilistic Retrieval. BM25 generally gives better results.

The Traditional weighting scheme formula is

$$\sum_t \frac{f_{t,d}}{k.L_d + f_{t,d}} . w_t$$

where

- w_t is the termweight of term t
- $f_{t,d}$ is the within document frequency of term t in document d
- L_d is the normalised length of document d
- k is a user specifiable parameter

TradWeight(k) is equivalent to BM25Weight(k , 0, 0, 1, 0), except that the latter returns weights ($k+1$) times larger.

7.25.2 Constructor & Destructor Documentation

7.25.2.1 [Xapian::TradWeight::TradWeight \(double \$k\$ \)](#) [[inline](#), [explicit](#)]

Construct a [TradWeight](#).

Parameters:

- k parameter governing the importance of within document frequency and document length - any non-negative number (0 meaning to ignore wdf and doc length when calculating weights). Default is 1.

7.25.3 Member Function Documentation

7.25.3.1 **TradWeight*** Xapian::TradWeight::clone () const [virtual]

Return a new weight object of this type.

A subclass called FooWeight taking parameters param1 and param2 should implement this as:

```
virtual FooWeight * clone() const { return new FooWeight(param1, param2); }
```

Implements [Xapian::Weight](#).

7.25.3.2 **Xapian::weight** Xapian::TradWeight::get_maxextra () const [virtual]

Gets the maximum value that [get_sumextra\(\)](#) may return.

This is used in optimising searches.

Implements [Xapian::Weight](#).

7.25.3.3 **Xapian::weight** Xapian::TradWeight::get_maxpart () const [virtual]

Gets the maximum value that [get_sumpart\(\)](#) may return.

This is used in optimising searches, by having the postlist tree decay appropriately when parts of it can have limited, or no, further effect.

Implements [Xapian::Weight](#).

7.25.3.4 **Xapian::weight** Xapian::TradWeight::get_sumextra ([Xapian::doclength](#) *len*) const [virtual]

Get an extra weight for a document to add to the sum calculated over the query terms.

This returns a weight for a given document, and is used by some weighting schemes to account for influence such as document length.

Parameters:

len the (unnormalised) document length.

Implements [Xapian::Weight](#).

7.25.3.5 **Xapian::weight** Xapian::TradWeight::get_sumpart ([Xapian::termcount](#) *wdf*, [Xapian::doclength](#) *len*) const [virtual]

Get a weight which is part of the sum over terms being performed.

This returns a weight for a given term and document. These weights are summed to give a total weight for the document.

Parameters:

wdf the within document frequency of the term.

len the (unnormalised) document length.

Implements [Xapian::Weight](#).

7.25.3.6 `bool Xapian::TradWeight::get_sumpart_needs_doclength () const`
[virtual]

return false if the weight object doesn't need doclength

Reimplemented from [Xapian::Weight](#).

7.25.3.7 `std::string Xapian::TradWeight::name () const` [virtual]

Name of the weighting scheme.

If the subclass is called FooWeight, this should return "Foo".

Implements [Xapian::Weight](#).

7.25.3.8 `std::string Xapian::TradWeight::serialise () const` [virtual]

Serialise object parameters into a string.

Implements [Xapian::Weight](#).

7.25.3.9 `TradWeight* Xapian::TradWeight::unserialise (const std::string & s)`
`const` [virtual]

Create object given string serialisation returned by [serialise\(\)](#).

Implements [Xapian::Weight](#).

The documentation for this class was generated from the following file:

- `include/xapian/enquire.h`

7.26 Xapian::Utf8Iterator Class Reference

An iterator which returns unicode character values from a UTF-8 encoded string.

```
#include <unicode.h>
```

Public Types

- typedef std::input_iterator_tag [iterator_category](#)
We implement the semantics of an STL input_iterator.
- typedef unsigned **value_type**
- typedef size_t **difference_type**
- typedef const unsigned * **pointer**
- typedef const unsigned & **reference**

Public Member Functions

- const char * [raw](#) () const
*Return the raw const char * pointer for the current position.*
- size_t [left](#) () const
Return the number of bytes left in the iterator's buffer.
- void [assign](#) (const char *p_, size_t len)
Assign a new string to the iterator.
- void [assign](#) (const std::string &s)
Assign a new string to the iterator.
- [Utf8Iterator](#) (const char *p_)
Create an iterator given a pointer to a null terminated string.
- [Utf8Iterator](#) (const char *p_, size_t len)
Create an iterator given a pointer and a length.
- [Utf8Iterator](#) (const std::string &s)
Create an iterator given a string.
- [Utf8Iterator](#) ()
Create an iterator which is at the end of its iteration.
- unsigned [operator *](#) () const
Get the current unicode character value pointed to by the iterator.
- [Utf8Iterator operator++](#) (int)

Move forward to the next unicode character.

- [Utf8Iterator](#) & [operator++](#) ()

Move forward to the next unicode character.

- bool [operator==](#) (const [Utf8Iterator](#) &other) const

Test two Utf8Iterators for equality.

- bool [operator!=](#) (const [Utf8Iterator](#) &other) const

Test two Utf8Iterators for inequality.

7.26.1 Detailed Description

An iterator which returns unicode character values from a UTF-8 encoded string.

7.26.2 Member Typedef Documentation

7.26.2.1 `typedef std::input_iterator_tag Xapian::Utf8Iterator::iterator_category`

We implement the semantics of an STL `input_iterator`.

7.26.3 Constructor & Destructor Documentation

7.26.3.1 `Xapian::Utf8Iterator::Utf8Iterator (const char * p_) [explicit]`

Create an iterator given a pointer to a null terminated string.

The iterator will return characters from the start of the string when next called. The string is not copied into the iterator, so it must remain valid while the iteration is in progress.

Parameters:

p A pointer to the start of the null terminated string to read.

7.26.3.2 `Xapian::Utf8Iterator::Utf8Iterator (const char * p_, size_t len) [inline]`

Create an iterator given a pointer and a length.

The iterator will return characters from the start of the string when next called. The string is not copied into the iterator, so it must remain valid while the iteration is in progress.

Parameters:

- p* A pointer to the start of the string to read.
- len* The length of the string to read.

7.26.3.3 Xapian::Utf8Iterator::Utf8Iterator (const std::string & s) [inline]

Create an iterator given a string.

The iterator will return characters from the start of the string when next called. The string is not copied into the iterator, so it must remain valid while the iteration is in progress.

Parameters:

- s* The string to read. Must not be modified while the iteration is in progress.

7.26.3.4 Xapian::Utf8Iterator::Utf8Iterator () [inline]

Create an iterator which is at the end of its iteration.

This can be compared to another iterator to check if the other iterator has reached its end.

7.26.4 Member Function Documentation**7.26.4.1 void Xapian::Utf8Iterator::assign (const std::string & s) [inline]**

Assign a new string to the iterator.

The iterator will forget the string it was iterating through, and return characters from the start of the new string when next called. The string is not copied into the iterator, so it must remain valid while the iteration is in progress.

Parameters:

- s* The string to read. Must not be modified while the iteration is in progress.

7.26.4.2 void Xapian::Utf8Iterator::assign (const char * p_, size_t len) [inline]

Assign a new string to the iterator.

The iterator will forget the string it was iterating through, and return characters from the start of the new string when next called. The string is not copied into the iterator, so it must remain valid while the iteration is in progress.

Parameters:

- p* A pointer to the start of the string to read.
- len* The length of the string to read.

7.26.4.3 `size_t Xapian::Utf8Iterator::left () const` `[inline]`

Return the number of bytes left in the iterator's buffer.

7.26.4.4 `unsigned Xapian::Utf8Iterator::operator * () const`

Get the current unicode character value pointed to by the iterator.

Returns unsigned(-1) if the iterator has reached the end of its buffer.

7.26.4.5 `bool Xapian::Utf8Iterator::operator!= (const Utf8Iterator & other) const` `[inline]`

Test two `Utf8Iterators` for inequality.

Returns:

true iff the iterators do not point to the same position.

7.26.4.6 `Utf8Iterator& Xapian::Utf8Iterator::operator++ ()` `[inline]`

Move forward to the next unicode character.

Returns:

A reference to this object.

7.26.4.7 `Utf8Iterator Xapian::Utf8Iterator::operator++ (int)` `[inline]`

Move forward to the next unicode character.

Returns:

An iterator pointing to the position before the move.

7.26.4.8 `bool Xapian::Utf8Iterator::operator== (const Utf8Iterator & other) const` `[inline]`

Test two `Utf8Iterators` for equality.

Returns:

true iff the iterators point to the same position.

7.26.4.9 `const char* Xapian::Utf8Iterator::raw () const` [inline]

Return the raw const char * pointer for the current position.

The documentation for this class was generated from the following file:

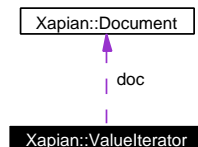
- `include/xapian/unicode.h`

7.27 Xapian::ValueIterator Class Reference

An iterator pointing to values associated with a document.

```
#include <valueiterator.h>
```

Collaboration diagram for Xapian::ValueIterator:



Public Types

- typedef std::input_iterator_tag [iterator_category](#)
Allow use as an STL iterator.
- typedef std::string **value_type**
- typedef [Xapian::value_no_diff](#) **difference_type**
- typedef std::string * **pointer**
- typedef std::string & **reference**

Public Member Functions

- [ValueIterator](#) ()
Create an uninitialised iterator; this cannot be used, but is convenient syntactically.
- [ValueIterator](#) (const [ValueIterator](#) &other)
Copying is allowed (and is cheap).
- void [operator=](#) (const [ValueIterator](#) &other)
Assignment is allowed (and is cheap).
- [ValueIterator](#) & [operator++](#) ()
Advance the iterator.
- [ValueIterator](#) [operator++](#) (int)
Advance the iterator (postfix variant).
- const std::string & [operator*](#) () const
Get the value for the current position.
- const std::string * [operator→](#) () const
Get the value for the current position.

- [Xapian::value](#) [get_valueno](#) () const
Get the number of the value at the current position.
- `std::string` [get_description](#) () const
Returns a string describing this object.

Friends

- class **Document**
- `bool` **operator==** (const [ValueIterator](#) &a, const [ValueIterator](#) &b)
- `bool` **operator!=** (const [ValueIterator](#) &a, const [ValueIterator](#) &b)

7.27.1 Detailed Description

An iterator pointing to values associated with a document.

7.27.2 Member Typedef Documentation

7.27.2.1 `typedef std::input_iterator_tag Xapian::ValueIterator::iterator_category`

Allow use as an STL iterator.

7.27.3 Constructor & Destructor Documentation

7.27.3.1 `Xapian::ValueIterator::ValueIterator () [inline]`

Create an uninitialised iterator; this cannot be used, but is convenient syntactically.

7.27.3.2 `Xapian::ValueIterator::ValueIterator (const ValueIterator & other) [inline]`

Copying is allowed (and is cheap).

7.27.4 Member Function Documentation

7.27.4.1 `std::string Xapian::ValueIterator::get_description () const`

Returns a string describing this object.

Introspection method.

7.27.4.2 [Xapian::valueno](#) **Xapian::ValueIterator::get_valueno () const**

Get the number of the value at the current position.

7.27.4.3 **const std::string& Xapian::ValueIterator::operator * () const**

Get the value for the current position.

7.27.4.4 [ValueIterator](#) **Xapian::ValueIterator::operator++ (int) [inline]**

Advance the iterator (postfix variant).

7.27.4.5 [ValueIterator&](#) **Xapian::ValueIterator::operator++ () [inline]**

Advance the iterator.

7.27.4.6 **const std::string* Xapian::ValueIterator::operator → () const**

Get the value for the current position.

7.27.4.7 **void Xapian::ValueIterator::operator= (const [ValueIterator](#) & other)**
[inline]

Assignment is allowed (and is cheap).

The documentation for this class was generated from the following file:

- include/xapian/[valueiterator.h](#)

7.28 Xapian::Weight Class Reference

Abstract base class for weighting schemes.

```
#include <enquire.h>
```

Inheritance diagram for Xapian::Weight:



Public Member Functions

- `Weight * create (const Internal *internal_, Xapian::doclength querysize_, Xapian::termcount wqf_, const std::string &name_) const`
Create a new weight object of the same type as this and initialise it with the specified statistics.
- `virtual std::string name () const =0`
Name of the weighting scheme.
- `virtual std::string serialise () const =0`
Serialise object parameters into a string.
- `virtual Weight * unserialise (const std::string &s) const =0`
Create object given string serialisation returned by `serialise()`.
- `virtual Xapian::weight get_sumpart (Xapian::termcount wdf, Xapian::doclength len) const =0`
Get a weight which is part of the sum over terms being performed.
- `virtual Xapian::weight get_maxpart () const =0`
Gets the maximum value that `get_sumpart()` may return.
- `virtual Xapian::weight get_sumextra (Xapian::doclength len) const =0`
Get an extra weight for a document to add to the sum calculated over the query terms.
- `virtual Xapian::weight get_maxextra () const =0`
Gets the maximum value that `get_sumextra()` may return.
- `virtual bool get_sumpart_needs_doclength () const`
return false if the weight object doesn't need doclength

Protected Member Functions

- **Weight** (const [Weight](#) &)

Protected Attributes

- const Internal * **internal**
- [Xapian::doclength](#) **querysize**
- [Xapian::termcount](#) **wqf**
- std::string **tname**

Friends

- class **Enquire**
- class **::RemoteServer**

7.28.1 Detailed Description

Abstract base class for weighting schemes.

7.28.2 Member Function Documentation

7.28.2.1 [Weight](#)* **Xapian::Weight::create** (const Internal * *internal_*, [Xapian::doclength](#) *querysize_*, [Xapian::termcount](#) *wqf_*, const std::string & *tname_*) const

Create a new weight object of the same type as this and initialise it with the specified statistics.

You shouldn't call this method yourself - it's called by [Enquire](#).

Parameters:

internal_ Object to ask for collection statistics.

querysize_ Query size.

wqf_ Within query frequency of term this object is associated with.

tname_ Term which this object is associated with.

7.28.2.2 **virtual [Xapian::weight](#) Xapian::Weight::get_maxextra** () const [pure virtual]

Gets the maximum value that [get_sumextra\(\)](#) may return.

This is used in optimising searches.

Implemented in [Xapian::BoolWeight](#), [Xapian::BM25Weight](#), and [Xapian::TradWeight](#).

7.28.2.3 `virtual Xapian::weight Xapian::Weight::get_maxpart () const` [pure virtual]

Gets the maximum value that `get_sumpart()` may return.

This is used in optimising searches, by having the postlist tree decay appropriately when parts of it can have limited, or no, further effect.

Implemented in [Xapian::BoolWeight](#), [Xapian::BM25Weight](#), and [Xapian::TradWeight](#).

7.28.2.4 `virtual Xapian::weight Xapian::Weight::get_sumextra (Xapian::doclength len) const` [pure virtual]

Get an extra weight for a document to add to the sum calculated over the query terms.

This returns a weight for a given document, and is used by some weighting schemes to account for influence such as document length.

Parameters:

len the (unnormalised) document length.

Implemented in [Xapian::BoolWeight](#), [Xapian::BM25Weight](#), and [Xapian::TradWeight](#).

7.28.2.5 `virtual Xapian::weight Xapian::Weight::get_sumpart (Xapian::termcount wdf, Xapian::doclength len) const` [pure virtual]

Get a weight which is part of the sum over terms being performed.

This returns a weight for a given term and document. These weights are summed to give a total weight for the document.

Parameters:

wdf the within document frequency of the term.

len the (unnormalised) document length.

Implemented in [Xapian::BoolWeight](#), [Xapian::BM25Weight](#), and [Xapian::TradWeight](#).

7.28.2.6 `virtual bool Xapian::Weight::get_sumpart_needs_doclength () const` [virtual]

return false if the weight object doesn't need doclength

Reimplemented in [Xapian::BoolWeight](#), [Xapian::BM25Weight](#), and [Xapian::TradWeight](#).

7.28.2.7 virtual std::string Xapian::Weight::name () const [pure virtual]

Name of the weighting scheme.

If the subclass is called FooWeight, this should return "Foo".

Implemented in [Xapian::BoolWeight](#), [Xapian::BM25Weight](#), and [Xapian::TradWeight](#).

7.28.2.8 virtual std::string Xapian::Weight::serialise () const [pure virtual]

Serialise object parameters into a string.

Implemented in [Xapian::BoolWeight](#), [Xapian::BM25Weight](#), and [Xapian::TradWeight](#).

7.28.2.9 virtual [Weight](#)* Xapian::Weight::unserialise (const std::string & s) const [pure virtual]

Create object given string serialisation returned by [serialise\(\)](#).

Implemented in [Xapian::BoolWeight](#), [Xapian::BM25Weight](#), and [Xapian::TradWeight](#).

The documentation for this class was generated from the following file:

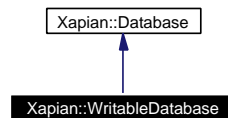
- [include/xapian/enquire.h](#)

7.29 Xapian::WritableDatabase Class Reference

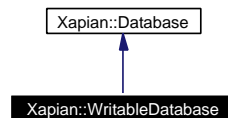
This class provides read/write access to a database.

```
#include <database.h>
```

Inheritance diagram for Xapian::WritableDatabase:



Collaboration diagram for Xapian::WritableDatabase:



Public Member Functions

- virtual [~WritableDatabase](#) ()
Destroy this handle on the database.
- [WritableDatabase](#) ()
Create an empty [WritableDatabase](#).
- [WritableDatabase](#) (const std::string &path, int action)
Open a database for update, automatically determining the database backend to use.
- [WritableDatabase](#) (const [WritableDatabase](#) &other)
Copying is allowed.
- void [operator=](#) (const [WritableDatabase](#) &other)
Assignment is allowed.
- void [flush](#) ()
Flush to disk any modifications made to the database.
- void [begin_transaction](#) (bool flushed=true)
Begin a transaction.
- void [commit_transaction](#) ()

Complete the transaction currently in progress.

- void [cancel_transaction](#) ()
Abort the transaction currently in progress, discarding the potential modifications made to the database.
- [Xapian::docid add_document](#) (const [Xapian::Document](#) &document)
Add a new document to the database.
- void [delete_document](#) ([Xapian::docid](#) did)
Delete a document from the database.
- void [delete_document](#) (const std::string &unique_term)
Delete any documents indexed by a term from the database.
- void [replace_document](#) ([Xapian::docid](#) did, const [Xapian::Document](#) &document)
Replace a given document in the database.
- [Xapian::docid replace_document](#) (const std::string &unique_term, const [Xapian::Document](#) &document)
Replace any documents matching a term.
- std::string [get_description](#) () const
Introspection method.

7.29.1 Detailed Description

This class provides read/write access to a database.

7.29.2 Constructor & Destructor Documentation

7.29.2.1 virtual [Xapian::WritableDatabase::~~WritableDatabase](#) () [virtual]

Destroy this handle on the database.

If there are no copies of this object remaining, the database will be closed. If there are any transactions in progress these will be aborted as if [cancel_transaction](#) had been called.

7.29.2.2 [Xapian::WritableDatabase::WritableDatabase](#) ()

Create an empty [WritableDatabase](#).

7.29.2.3 Xapian::WritableDatabase::WritableDatabase (const std::string & path, int action)

Open a database for update, automatically determining the database backend to use.

If the database is to be created, [Xapian](#) will try to create the directory indicated by path if it doesn't already exist (but only the leaf directory, not recursively).

Parameters:

path directory that the database is stored in.

action one of:

- [Xapian::DB_CREATE_OR_OPEN](#) open for read/write; create if no db exists
- [Xapian::DB_CREATE](#) create new database; fail if db exists
- [Xapian::DB_CREATE_OR_OVERWRITE](#) overwrite existing db; create if none exists
- [Xapian::DB_OPEN](#) open for read/write; fail if no db exists

7.29.2.4 Xapian::WritableDatabase::WritableDatabase (const WritableDatabase & other)

Copying is allowed.

The internals are reference counted, so copying is cheap.

7.29.3 Member Function Documentation

7.29.3.1 [Xapian::docid](#) Xapian::WritableDatabase::add_document (const Xapian::Document & document)

Add a new document to the database.

This method adds the specified document to the database, returning a newly allocated document ID. Automatically allocated document IDs come from a per-database monotonically increasing counter, so IDs from deleted documents won't be reused.

If you want to specify the document ID to be used, you should call [replace_document\(\)](#) instead.

Note that changes to the database won't be immediately committed to disk; see [flush\(\)](#) for more details.

As with all database modification operations, the effect is atomic: the document will either be fully added, or the document fails to be added and an exception is thrown (possibly at a later time when flush is called or the database is closed).

Parameters:

document The new document to be added.

Returns:

The document ID of the newly added document.

Exceptions:

Xapian::DatabaseError will be thrown if a problem occurs while writing to the database.

Xapian::DatabaseCorruptError will be thrown if the database is in a corrupt state.

7.29.3.2 void Xapian::WritableDatabase::begin_transaction (bool *flushed* = true)

Begin a transaction.

In [Xapian](#) a transaction is a group of modifications to the database which are linked such that either all will be applied simultaneously or none will be applied at all. Even in the case of a power failure, this characteristic should be preserved (as long as the filesystem isn't corrupted, etc).

A transaction is started with [begin_transaction\(\)](#) and can either be committed by calling [commit_transaction\(\)](#) or aborted by calling [cancel_transaction\(\)](#).

By default, a transaction implicitly calls flush before and after so that the modifications stand and fall without affecting modifications before or after.

The downside of this flushing is that small transactions cause modifications to be frequently flushed which can harm indexing performance in the same way that explicitly calling flush frequently can.

If you're applying atomic groups of changes and only wish to ensure that each group is either applied or not applied, then you can prevent the automatic flush before and after the transaction by starting the transaction with `begin_transaction(false)`. However, if `cancel_transaction` is called (or if `commit_transaction` isn't called before the [WritableDatabase](#) object is destroyed) then any changes which were pending before the transaction began will also be discarded.

Transactions aren't currently supported by the InMemory backend.

Exceptions:

Xapian::UnimplementedError will be thrown if transactions are not available for this database type.

Xapian::InvalidOperationError will be thrown if this is called at an invalid time, such as when a transaction is already in progress.

7.29.3.3 void Xapian::WritableDatabase::cancel_transaction ()

Abort the transaction currently in progress, discarding the potential modifications made to the database.

If an error occurs in this method, an exception will be thrown, but the transaction will be cancelled anyway.

Exceptions:

Xapian::DatabaseError will be thrown if a problem occurs while modifying the database.

Xapian::DatabaseCorruptError will be thrown if the database is in a corrupt state.

Xapian::InvalidOperationError will be thrown if a transaction is not currently in progress.

Xapian::UnimplementedError will be thrown if transactions are not available for this database type.

7.29.3.4 void Xapian::WritableDatabase::commit_transaction ()

Complete the transaction currently in progress.

If this method completes successfully and this is a flushed transaction, all the database modifications made during the transaction will have been committed to the database.

If an error occurs, an exception will be thrown, and none of the modifications made to the database during the transaction will have been applied to the database.

In all cases the transaction will no longer be in progress.

Exceptions:

Xapian::DatabaseError will be thrown if a problem occurs while modifying the database.

Xapian::DatabaseCorruptError will be thrown if the database is in a corrupt state.

Xapian::InvalidOperationError will be thrown if a transaction is not currently in progress.

Xapian::UnimplementedError will be thrown if transactions are not available for this database type.

7.29.3.5 void Xapian::WritableDatabase::delete_document (const std::string & unique_term)

Delete any documents indexed by a term from the database.

This method removes any documents indexed by the specified term from the database.

The intended use is to allow UIDs from another system to easily be mapped to terms in [Xapian](#), although this method probably has other uses.

Parameters:

unique_term The term to remove references to.

Exceptions:

Xapian::DatabaseError will be thrown if a problem occurs while writing to the database.

Xapian::DatabaseCorruptError will be thrown if the database is in a corrupt state.

7.29.3.6 void Xapian::WritableDatabase::delete_document (*Xapian::docid did*)

Delete a document from the database.

This method removes the document with the specified document ID from the database.

Note that changes to the database won't be immediately committed to disk; see [flush\(\)](#) for more details.

As with all database modification operations, the effect is atomic: the document will either be fully removed, or the document fails to be removed and an exception is thrown (possibly at a later time when flush is called or the database is closed).

Parameters:

did The document ID of the document to be removed.

Exceptions:

Xapian::DatabaseError will be thrown if a problem occurs while writing to the database.

Xapian::DatabaseCorruptError will be thrown if the database is in a corrupt state.

7.29.3.7 void Xapian::WritableDatabase::flush ()

Flush to disk any modifications made to the database.

For efficiency reasons, when performing multiple updates to a database it is best (indeed, almost essential) to make as many modifications as memory will permit in a single pass through the database. To ensure this, [Xapian](#) batches up modifications.

Flush may be called at any time to ensure that the modifications which have been made are written to disk: if the flush succeeds, all the preceding modifications will have been written to disk.

If any of the modifications fail, an exception will be thrown and the database will be left in a state in which each separate addition, replacement or deletion operation has either been fully performed or not performed at all: it is then up to the application to work out which operations need to be repeated.

It's not valid to call flush within a transaction.

Beware of calling flush too frequently: this will have a severe performance cost.

Note that flush need not be called explicitly: it will be called automatically when the database is closed, or when a sufficient number of modifications have been made.

Exceptions:

Xapian::DatabaseError will be thrown if a problem occurs while modifying the database.

Xapian::DatabaseCorruptError will be thrown if the database is in a corrupt state.

Xapian::DatabaseLockError will be thrown if a lock couldn't be acquired on the database.

7.29.3.8 `std::string Xapian::WritableDatabase::get_description () const` [virtual]

Introspection method.

Returns:

A string describing this object.

Reimplemented from [Xapian::Database](#).

7.29.3.9 `void Xapian::WritableDatabase::operator= (const WritableDatabase & other)`

Assignment is allowed.

The internals are reference counted, so assignment is cheap.

Note that only an [WritableDatabase](#) may be assigned to an [WritableDatabase](#): an attempt to assign a [Database](#) is caught at compile-time.

7.29.3.10 `Xapian::docid Xapian::WritableDatabase::replace_document (const std::string & unique_term, const Xapian::Document & document)`

Replace any documents matching a term.

This method replaces any documents indexed by the specified term with the specified document. If any documents are indexed by the term, the lowest document ID will be used for the document, otherwise a new document ID will be generated as for `add_document`.

The intended use is to allow UIDs from another system to easily be mapped to terms in [Xapian](#), although this method probably has other uses.

Note that changes to the database won't be immediately committed to disk; see [flush\(\)](#) for more details.

As with all database modification operations, the effect is atomic: the document(s) will either be fully replaced, or the document(s) fail to be replaced and an exception is thrown (possibly at a later time when `flush` is called or the database is closed).

Parameters:

unique_term The "unique" term.

document The new document.

Returns:

The document ID that document was given.

Exceptions:

Xapian::DatabaseError will be thrown if a problem occurs while writing to the database.

Xapian::DatabaseCorruptError will be thrown if the database is in a corrupt state.

7.29.3.11 void Xapian::WritableDatabase::replace_document (Xapian::docid *did*, const Xapian::Document & *document*)

Replace a given document in the database.

This method replaces the document with the specified document ID. If document ID *did* isn't currently used, the document will be added with document ID *did*.

The monotonic counter used for automatically allocating document IDs is increased so that the next automatically allocated document ID will be *did* + 1. Be aware that if you use this method to specify a high document ID for a new document, and also use [WritableDatabase::add_document\(\)](#), [Xapian](#) may get to a state where this counter wraps around and will be unable to automatically allocate document IDs!

Note that changes to the database won't be immediately committed to disk; see [flush\(\)](#) for more details.

As with all database modification operations, the effect is atomic: the document will either be fully replaced, or the document fails to be replaced and an exception is thrown (possibly at a later time when flush is called or the database is closed).

Parameters:

did The document ID of the document to be replaced.

document The new document.

Exceptions:

Xapian::DatabaseError will be thrown if a problem occurs while writing to the database.

Xapian::DatabaseCorruptError will be thrown if the database is in a corrupt state.

The documentation for this class was generated from the following file:

- [include/xapian/database.h](#)

Chapter 8

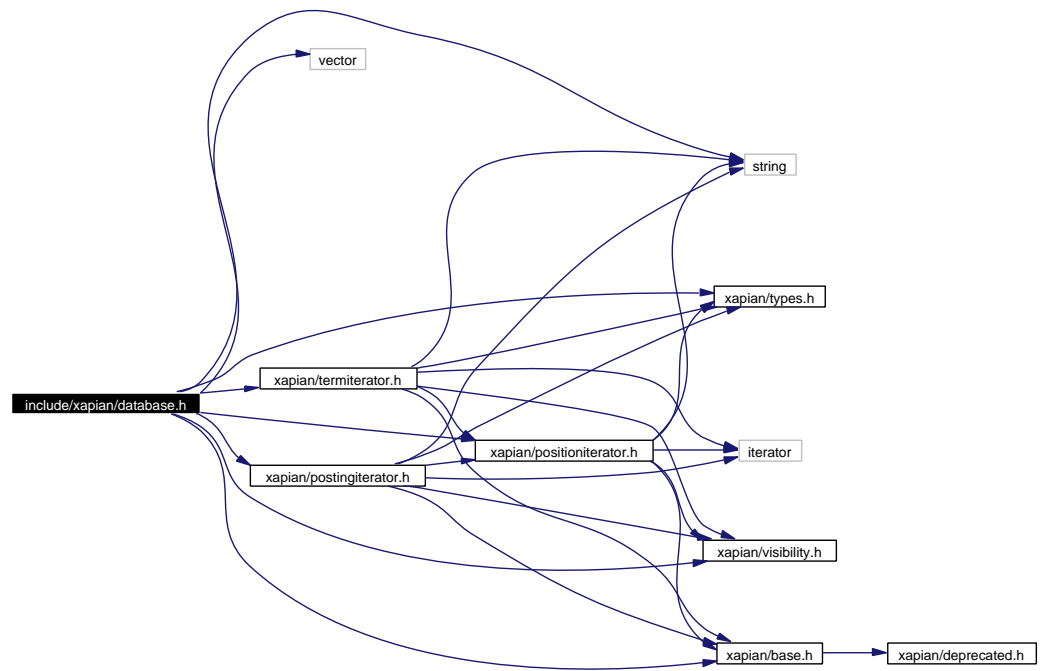
xapian-core File Documentation

8.1 include/xapian/database.h File Reference

API for working with [Xapian](#) databases.

```
#include <string>
#include <vector>
#include <xapian/base.h>
#include <xapian/types.h>
#include <xapian/positioniterator.h>
#include <xapian/postingiterator.h>
#include <xapian/termiterator.h>
#include <xapian/visibility.h>
```

Include dependency graph for database.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::Database](#)

This class is used to access a database, or a group of databases.

- class [Xapian::WritableDatabase](#)

This class provides read/write access to a database.

Variables

- const int `Xapian::DB_CREATE_OR_OPEN` = 1
Open for read/write; create if no db exists.
- const int `Xapian::DB_CREATE` = 2
Create a new database; fail if db exists.
- const int `Xapian::DB_CREATE_OR_OVERWRITE` = 3
Overwrite existing db; create if none exists.
- const int `Xapian::DB_OPEN` = 4
Open for read/write; fail if no db exists.

8.1.1 Detailed Description

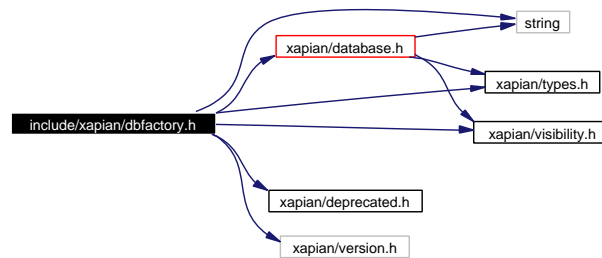
API for working with `Xapian` databases.

8.2 include/xapian/dbfactory.h File Reference

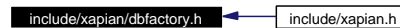
Factory functions for constructing Database and WritableDatabase objects.

```
#include <string>
#include <xapian/types.h>
#include <xapian/database.h>
#include <xapian/deprecated.h>
#include <xapian/version.h>
#include <xapian/visibility.h>
```

Include dependency graph for dbfactory.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)
- namespace **Xapian::Auto**
- namespace **Xapian::InMemory**
- namespace **Xapian::Quartz**
- namespace **Xapian::Flint**
- namespace **Xapian::Remote**

Functions

- XAPIAN_VISIBILITY_DEFAULT Database [Xapian::Auto::open_stub](#) (const std::string &file)
Construct a [Database](#) object for a stub database file.
- XAPIAN_VISIBILITY_DEFAULT WritableDatabase [Xapian::InMemory::open](#) ()

Construct a [Database](#) object for update access to an *InMemory* database.

- XAPIAN_VISIBILITY_DEFAULT [Xapian::Quartz::XAPIAN_DEPRECATED](#)
(Database open(const std::string &dir))

Construct a [Database](#) object for read-only access to a *Quartz* database.

- XAPIAN_VISIBILITY_DEFAULT [Xapian::Quartz::XAPIAN_DEPRECATED](#)
(WritableDatabase open(const std::string &dir, int action, int block_size=8192))

Construct a [Database](#) object for update access to a *Quartz* database.

- XAPIAN_VISIBILITY_DEFAULT Database [Xapian::Flint::open](#) (const std::string &dir)

Construct a [Database](#) object for read-only access to a *Flint* database.

- XAPIAN_VISIBILITY_DEFAULT WritableDatabase [Xapian::Flint::open](#)
(const std::string &dir, int action, int block_size=8192)

Construct a [Database](#) object for update access to a *Flint* database.

- XAPIAN_VISIBILITY_DEFAULT Database [Xapian::Remote::open](#) (const std::string &host, unsigned int port, [Xapian::timeout](#) timeout=10000, [Xapian::timeout](#) connect_timeout=10000)

Construct a [Database](#) object for read-only access to a remote database accessed via a *TCP* connection.

- XAPIAN_VISIBILITY_DEFAULT WritableDatabase [Xapian::Remote::open_writable](#) (const std::string &host, unsigned int port, [Xapian::timeout](#) timeout=0, [Xapian::timeout](#) connect_timeout=10000)

Construct a [WritableDatabase](#) object for update access to a remote database accessed via a *TCP* connection.

- XAPIAN_VISIBILITY_DEFAULT Database [Xapian::Remote::open](#) (const std::string &program, const std::string &args, [Xapian::timeout](#) timeout=10000)

Construct a [Database](#) object for read-only access to a remote database accessed via a *program*.

- XAPIAN_VISIBILITY_DEFAULT WritableDatabase [Xapian::Remote::open_writable](#) (const std::string &program, const std::string &args, [Xapian::timeout](#) timeout=0)

Construct a [WritableDatabase](#) object for update access to a remote database accessed via a *program*.

8.2.1 Detailed Description

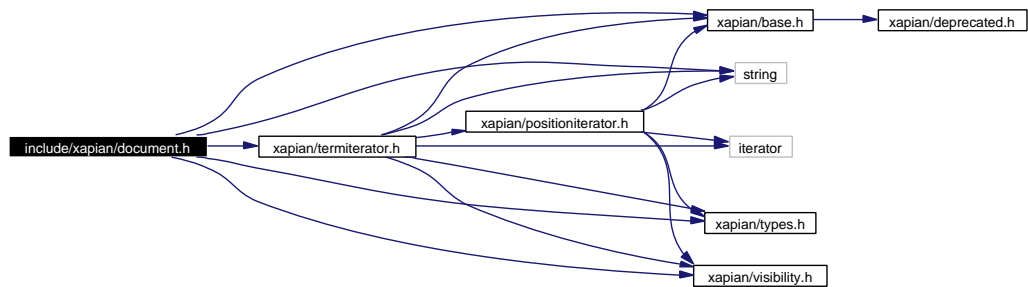
Factory functions for constructing Database and WritableDatabase objects.

8.3 include/xapian/document.h File Reference

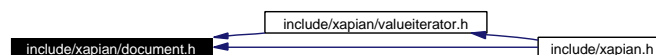
API for working with documents.

```
#include <string>
#include <xapian/base.h>
#include <xapian/types.h>
#include <xapian/termiterator.h>
#include <xapian/visibility.h>
```

Include dependency graph for document.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::Document](#)

A document in the database - holds data, values, terms, and postings.

8.3.1 Detailed Description

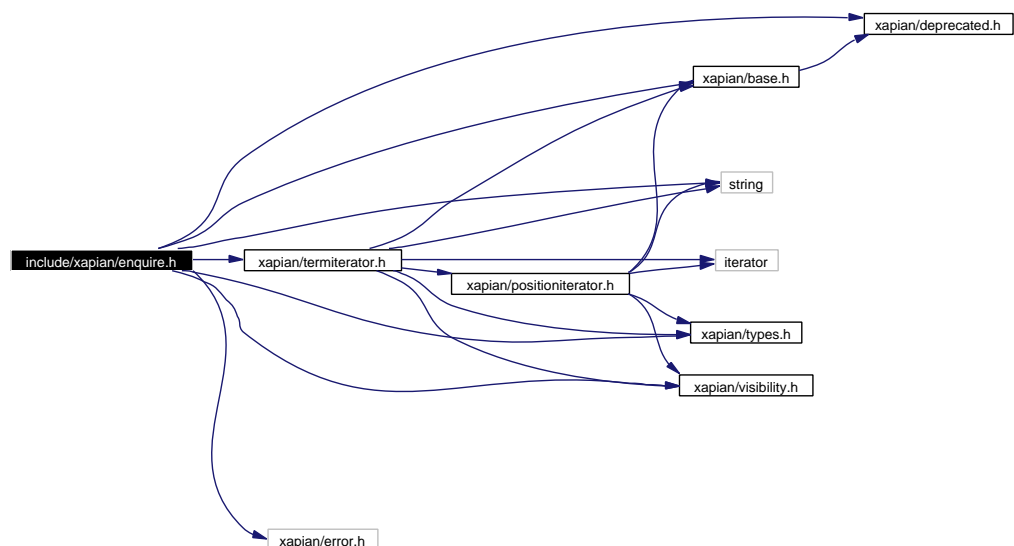
API for working with documents.

8.4 include/xapian/enquire.h File Reference

API for running queries.

```
#include <string>
#include <xapian/base.h>
#include <xapian/deprecated.h>
#include <xapian/error.h>
#include <xapian/types.h>
#include <xapian/terminator.h>
#include <xapian/visibility.h>
```

Include dependency graph for enquire.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::MSet](#)

A match set (*MSet*).

- class [Xapian::MSetIterator](#)
An iterator pointing to items in an *MSet*.
- class [Xapian::ESet](#)
Class representing an ordered set of expand terms (an *ESet*).
- class [Xapian::ESetIterator](#)
Iterate through terms in the *ESet*.
- class [Xapian::RSet](#)
A relevance set (*R-Set*).
- class [Xapian::MatchDecider](#)
Base class for matcher decision functor.
- class [Xapian::Enquire](#)
This class provides an interface to the information retrieval system for the purpose of searching.
- class [Xapian::Weight](#)
Abstract base class for weighting schemes.
- class [Xapian::BoolWeight](#)
Boolean weighting scheme (everything gets 0).
- class [Xapian::BM25Weight](#)
BM25 weighting scheme.
- class [Xapian::TradWeight](#)
Traditional probabilistic weighting scheme.

Functions

- bool **Xapian::operator==** (const MSetIterator &a, const MSetIterator &b)
- bool **Xapian::operator!=** (const MSetIterator &a, const MSetIterator &b)
- bool **Xapian::operator==** (const ESetIterator &a, const ESetIterator &b)
- bool **Xapian::operator!=** (const ESetIterator &a, const ESetIterator &b)

8.4.1 Detailed Description

API for running queries.

8.5 include/xapian/errorhandler.h File Reference

Decide if a Xapian::Error exception should be ignored.

```
#include <xapian/error.h>
#include <xapian/visibility.h>
```

Include dependency graph for errorhandler.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::ErrorHandler](#)
Decide if a Xapian::Error exception should be ignored.

8.5.1 Detailed Description

Decide if a Xapian::Error exception should be ignored.

8.6 include/xapian/expanddecider.h File Reference

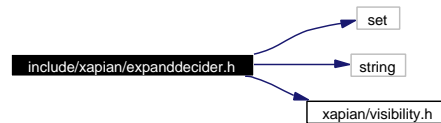
Allow rejection of terms during ESet generation.

```
#include <set>
```

```
#include <string>
```

```
#include <xapian/visibility.h>
```

Include dependency graph for expanddecider.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::ExpandDecider](#)
Virtual base class for expand decider functor.
- class [Xapian::ExpandDeciderAnd](#)
[ExpandDecider](#) subclass which rejects terms using two [ExpandDeciders](#).
- class [Xapian::ExpandDeciderFilterTerms](#)
[ExpandDecider](#) subclass which rejects terms in a specified list.

8.6.1 Detailed Description

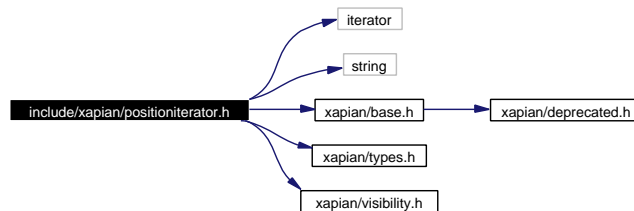
Allow rejection of terms during ESet generation.

8.7 include/xapian/positioniterator.h File Reference

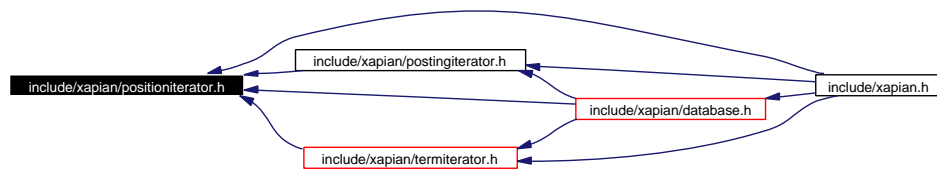
Classes for iterating through position lists.

```
#include <iterator>
#include <string>
#include <xapian/base.h>
#include <xapian/types.h>
#include <xapian/visibility.h>
```

Include dependency graph for positioniterator.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class **Xapian::TermPosWrapper**
- class [Xapian::PositionIterator](#)

An iterator pointing to items in a list of positions.

Functions

- bool [Xapian::operator==](#) (const PositionIterator &a, const PositionIterator &b)
Test equality of two PositionIterators.

- bool [Xapian::operator!=](#) (const PositionIterator &a, const PositionIterator &b)
Test inequality of two PositionIterators.

8.7.1 Detailed Description

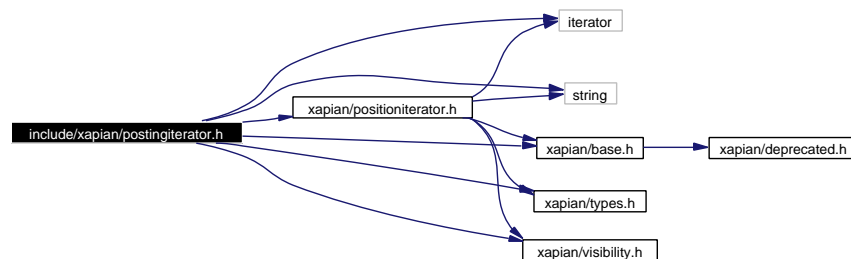
Classes for iterating through position lists.

8.8 include/xapian/postingiterator.h File Reference

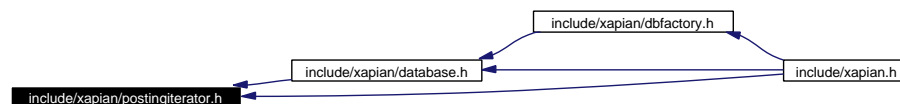
Classes for iterating through posting lists.

```
#include <iterator>
#include <string>
#include <xapian/base.h>
#include <xapian/types.h>
#include <xapian/positioniterator.h>
#include <xapian/visibility.h>
```

Include dependency graph for postingiterator.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class `Xapian::DocIDWrapper`
- class [Xapian::PostingIterator](#)

An iterator pointing to items in a list of postings.

Functions

- bool [Xapian::operator==](#) (const PostingIterator &a, const PostingIterator &b)

Test equality of two PostingIterators.

- bool [Xapian::operator!=](#) (const PostingIterator &a, const PostingIterator &b)

Test inequality of two PostingIterators.

8.8.1 Detailed Description

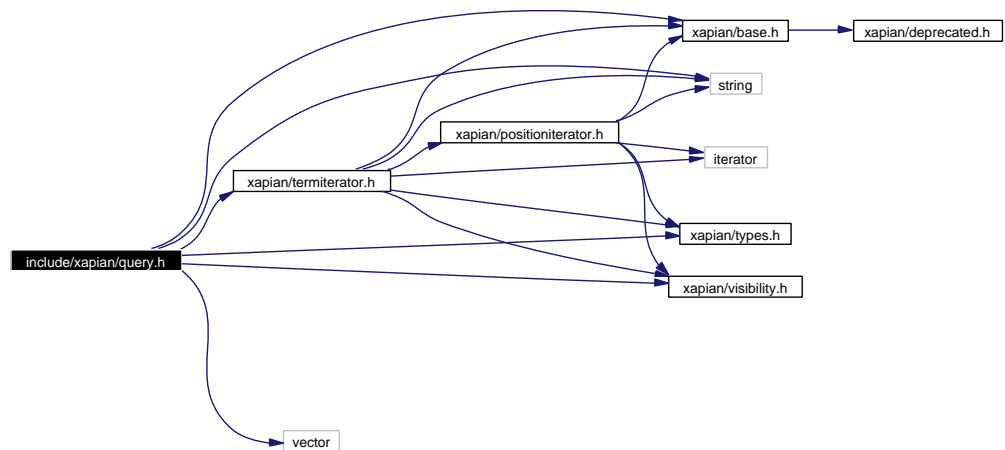
Classes for iterating through posting lists.

8.9 include/xapian/query.h File Reference

Classes for representing a query.

```
#include <string>
#include <vector>
#include <xapian/base.h>
#include <xapian/types.h>
#include <xapian/termiterator.h>
#include <xapian/visibility.h>
```

Include dependency graph for query.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::Query](#)

Class representing a query.

- class **Xapian::Query**

8.9.1 Detailed Description

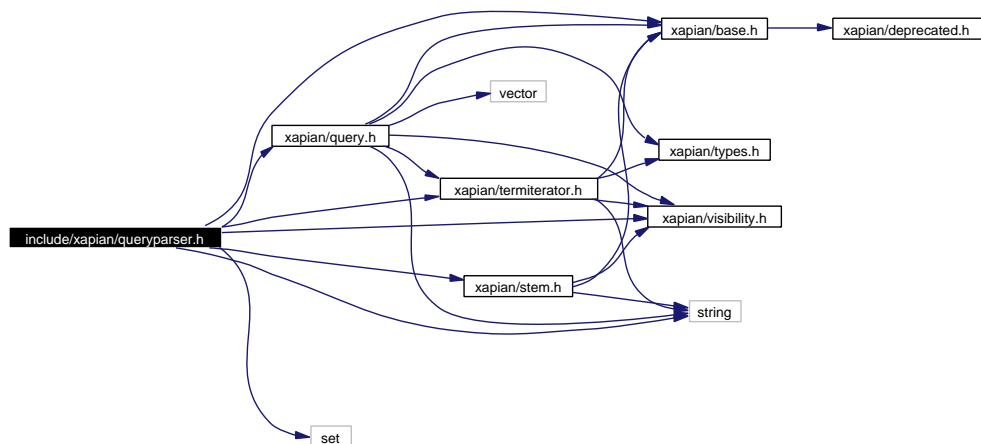
Classes for representing a query.

8.10 include/xapian/queryparser.h File Reference

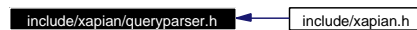
parsing a user query string to build a Xapian::Query object

```
#include <xapian/base.h>
#include <xapian/query.h>
#include <xapian/stem.h>
#include <xapian/terminator.h>
#include <xapian/visibility.h>
#include <set>
#include <string>
```

Include dependency graph for queryparser.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::Stopper](#)

Base class for stop-word decision functor.

- class [Xapian::SimpleStopper](#)
Simple implementation of [Stopper](#) class - this will suit most users.
- struct **Xapian::ValueRangeProcessor**
- class **Xapian::StringValueRangeProcessor**
- class **Xapian::DateValueRangeProcessor**
- class **Xapian::NumberValueRangeProcessor**
- class [Xapian::QueryParser](#)
Build a [Xapian::Query](#) object from a user query string.

8.10.1 Detailed Description

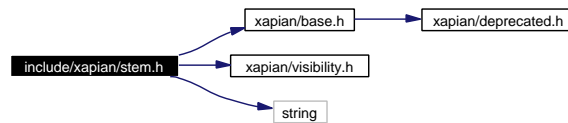
parsing a user query string to build a [Xapian::Query](#) object

8.11 include/xapian/stem.h File Reference

stemming algorithms

```
#include <xapian/base.h>
#include <xapian/visibility.h>
#include <string>
```

Include dependency graph for stem.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::Stem](#)
Class representing a stemming algorithm.

8.11.1 Detailed Description

stemming algorithms

8.12 include/xapian/termgenerator.h File Reference

parse free text and generate terms

```
#include <xapian/base.h>
```

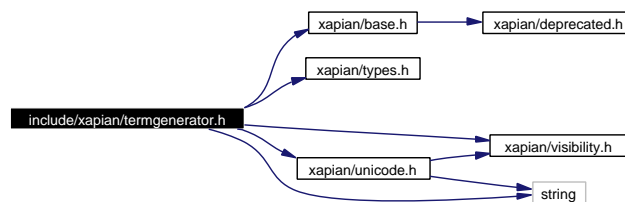
```
#include <xapian/types.h>
```

```
#include <xapian/unicode.h>
```

```
#include <xapian/visibility.h>
```

```
#include <string>
```

Include dependency graph for termgenerator.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::TermGenerator](#)
Parse free text and generate terms.

8.12.1 Detailed Description

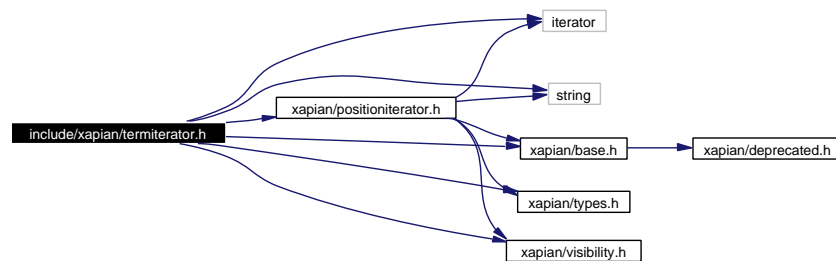
parse free text and generate terms

8.13 include/xapian/terminator.h File Reference

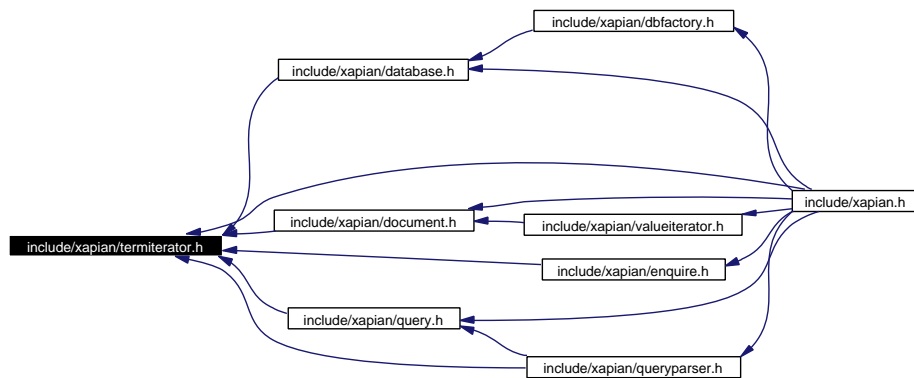
Classes for iterating through term lists.

```
#include <iterator>
#include <string>
#include <xapian/base.h>
#include <xapian/types.h>
#include <xapian/positioniterator.h>
#include <xapian/visibility.h>
```

Include dependency graph for terminator.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class `Xapian::TermNameWrapper`

- class [Xapian::TermIterator](#)

An iterator pointing to items in a list of terms.

Functions

- bool **Xapian::operator==** (const TermIterator &a, const TermIterator &b)
- bool **Xapian::operator!=** (const TermIterator &a, const TermIterator &b)

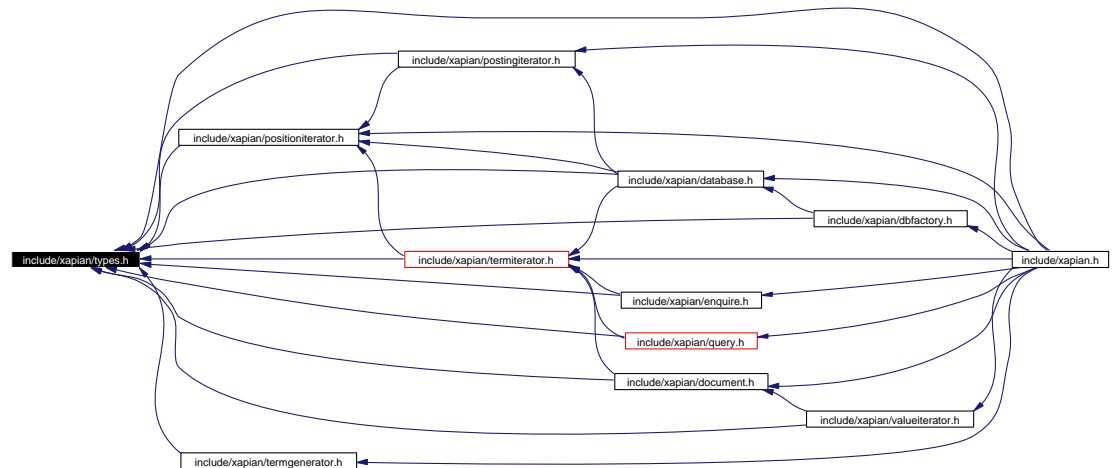
8.13.1 Detailed Description

Classes for iterating through term lists.

8.14 include/xapian/types.h File Reference

typedefs for [Xapian](#)

This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Typedefs

- typedef unsigned [Xapian::doccount](#)
A count of documents.
- typedef int [Xapian::doccount_diff](#)
A signed difference between two counts of documents.
- typedef unsigned [Xapian::docid](#)
A unique identifier for a document.
- typedef double [Xapian::doclength](#)
A normalised document length.
- typedef int [Xapian::percent](#)
The percentage score for a document in an [MSet](#).
- typedef unsigned [Xapian::termcount](#)
A counts of terms.

- typedef int [Xapian::termcount_diff](#)
A signed difference between two counts of terms.
- typedef unsigned [Xapian::termpos](#)
A term position within a document or query.
- typedef int [Xapian::termpos_diff](#)
A signed difference between two term positions.
- typedef unsigned [Xapian::timeout](#)
A timeout value in microseconds.
- typedef unsigned [Xapian::valueno](#)
The number for a value slot in a document.
- typedef int [Xapian::valueno_diff](#)
A signed difference between two value slot numbers.
- typedef double [Xapian::weight](#)
The weight of a document or term.

Variables

- const [valueno](#) [Xapian::BAD_VALUENO](#) = static_cast<[valueno](#)>(-1)
Reserved value to indicate "no valueno".

8.14.1 Detailed Description

typedefs for [Xapian](#)

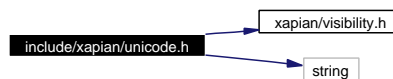
8.15 include/xapian/unicode.h File Reference

Unicode and UTF-8 related classes and functions.

```
#include <xapian/visibility.h>
```

```
#include <string>
```

Include dependency graph for unicode.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)
- namespace **Xapian::Unicode**
- namespace **Xapian::Unicode::Internal**

Classes

- class [Xapian::Utf8Iterator](#)
An iterator which returns unicode character values from a UTF-8 encoded string.

Enumerations

- enum **category** {
UNASSIGNED, UPPERCASE_LETTER, LOWERCASE_LETTER,
TITLECASE_LETTER,
MODIFIER_LETTER, OTHER_LETTER, NON_SPACING_MARK,
ENCLOSING_MARK,
COMBINING_SPACING_MARK, DECIMAL_DIGIT_NUMBER,
LETTER_NUMBER, OTHER_NUMBER,
SPACE_SEPARATOR, LINE_SEPARATOR, PARAGRAPH_
SEPARATOR, CONTROL,
FORMAT, PRIVATE_USE, SURROGATE, CONNECTOR_
PUNCTUATION,

```

DASH_PUNCTUATION,      OPEN_PUNCTUATION,      CLOSE_
PUNCTUATION, INITIAL_QUOTE_PUNCTUATION,
FINAL_QUOTE_PUNCTUATION, OTHER_PUNCTUATION, MATH_
SYMBOL, CURRENCY_SYMBOL,
MODIFIER_SYMBOL, OTHER_SYMBOL }

```

Each unicode character is in one of these categories.

Functions

- XAPIAN_VISIBILITY_DEFAULT int [Xapian::Unicode::Internal::get_character_info](#) (unsigned ch)
- int [Xapian::Unicode::Internal::get_case_type](#) (int info)
Extract how to convert the case of a unicode character from its info.
- category [Xapian::Unicode::Internal::get_category](#) (int info)
Extract the category of a unicode character from its info.
- int [Xapian::Unicode::Internal::get_delta](#) (int info)
Extract the delta to use for case conversion of a character from its info.
- XAPIAN_VISIBILITY_DEFAULT unsigned [Xapian::Unicode::nonascii_to_utf8](#) (unsigned ch, char *buf)
Convert a single non-ASCII unicode character to UTF-8.
- unsigned [Xapian::Unicode::to_utf8](#) (unsigned ch, char *buf)
Convert a single unicode character to UTF-8.
- void [Xapian::Unicode::append_utf8](#) (std::string &s, unsigned ch)
Append the UTF-8 representation of a single unicode character to a std::string.
- category [Xapian::Unicode::get_category](#) (unsigned ch)
Return the category which a given unicode character falls into.
- bool [Xapian::Unicode::is_wordchar](#) (unsigned ch)
Test if a given unicode character is a letter or number.
- bool [Xapian::Unicode::is_whitespace](#) (unsigned ch)
Test if a given unicode character is a whitespace character.
- bool [Xapian::Unicode::is_currency](#) (unsigned ch)
Test if a given unicode character is a currency symbol.
- unsigned [Xapian::Unicode::tolower](#) (unsigned ch)
Convert a unicode character to lowercase.

- `std::string Xapian::Unicode::tolower` (`const std::string &term`)
Convert a UTF-8 `std::string` to lowercase.

8.15.1 Detailed Description

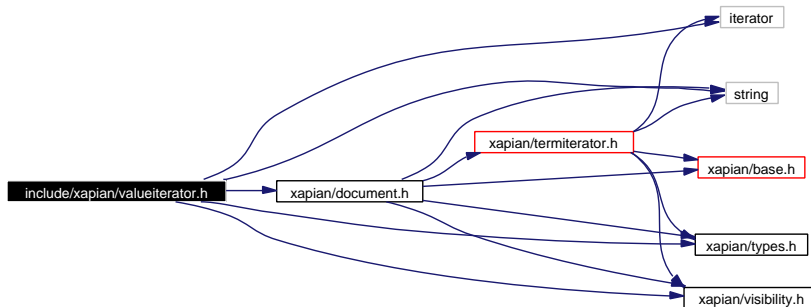
Unicode and UTF-8 related classes and functions.

8.16 include/xapian/valueiterator.h File Reference

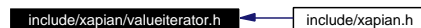
classes for iterating through values

```
#include <iterator>
#include <string>
#include <xapian/types.h>
#include <xapian/document.h>
#include <xapian/visibility.h>
```

Include dependency graph for valueiterator.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [Xapian](#)

Classes

- class [Xapian::ValueIterator](#)
An iterator pointing to values associated with a document.

Functions

- bool **Xapian::operator==** (const ValueIterator &a, const ValueIterator &b)

- bool **Xapian::operator!=**(const ValueIterator &a, const ValueIterator &b)

8.16.1 Detailed Description

classes for iterating through values

Chapter 9

xapian-core Page Documentation

9.1 Deprecated List

Member **Xapian::XAPIAN_DEPRECATED(int xapian_major_version())** This function is now deprecated, use [Xapian::major_version\(\)](#) instead.

Member **Xapian::XAPIAN_DEPRECATED(int xapian_major_version())** This function is now deprecated, use [Xapian::minor_version\(\)](#) instead.

Member **Xapian::XAPIAN_DEPRECATED(int xapian_major_version())** This function is now deprecated, use [Xapian::revision\(\)](#) instead.

Member **Xapian::XAPIAN_DEPRECATED(const char *xapian_version_string())** This function is now deprecated, use [Xapian::version_string\(\)](#) instead.

Index

- ~Database
 - Xapian::Database, [31](#)
- ~Document
 - Xapian::Document, [37](#)
- ~ESet
 - Xapian::ESet, [54](#)
- ~Enquire
 - Xapian::Enquire, [43](#)
- ~ErrorHandler
 - Xapian::ErrorHandler, [51](#)
- ~ExpandDecider
 - Xapian::ExpandDecider, [59](#)
- ~MSet
 - Xapian::MSet, [67](#)
- ~MatchDecider
 - Xapian::MatchDecider, [64](#)
- ~PositionIterator
 - Xapian::PositionIterator, [78](#)
- ~PostingIterator
 - Xapian::PostingIterator, [80](#)
- ~QueryParser
 - Xapian::QueryParser, [86](#)
- ~RSet
 - Xapian::RSet, [90](#)
- ~SimpleStopper
 - Xapian::SimpleStopper, [93](#)
- ~Stem
 - Xapian::Stem, [95](#)
- ~Stopper
 - Xapian::Stopper, [97](#)
- ~TermGenerator
 - Xapian::TermGenerator, [100](#)
- ~TermIterator
 - Xapian::TermIterator, [104](#)
- ~WritableDatabase
 - Xapian::WritableDatabase, [124](#)
- add
 - Xapian::SimpleStopper, [93](#)
- add_boolean_prefix
 - Xapian::QueryParser, [86](#)
- add_database
 - Xapian::Database, [32](#)
- add_document
 - Xapian::RSet, [90](#)
 - Xapian::WritableDatabase, [125](#)
- add_posting
 - Xapian::Document, [38](#)
- add_prefix
 - Xapian::QueryParser, [87](#)
- add_term
 - Xapian::Document, [38](#)
- add_value
 - Xapian::Document, [38](#)
- add_valuerangeprocessor
 - Xapian::QueryParser, [87](#)
- allterms_begin
 - Xapian::Database, [32](#)
- allterms_end
 - Xapian::Database, [32](#)
- assign
 - Xapian::Utf8Iterator, [113](#)
- back
 - Xapian::ESet, [54](#)
 - Xapian::MSet, [67](#)
- BAD_VALUENO
 - Xapian, [19](#)
- begin
 - Xapian::ESet, [54](#)
 - Xapian::MSet, [67](#)
- begin_transaction
 - Xapian::WritableDatabase, [126](#)
- BM25Weight
 - Xapian::BM25Weight, [23](#)
- cancel_transaction
 - Xapian::WritableDatabase, [126](#)
- clear_terms
 - Xapian::Document, [38](#)
- clear_values
 - Xapian::Document, [38](#)

- clone
 - Xapian::BM25Weight, [23](#)
 - Xapian::BoolWeight, [27](#)
 - Xapian::TradWeight, [109](#)
- commit_transaction
 - Xapian::WritableDatabase, [127](#)
- contains
 - Xapian::RSet, [90](#)
- convert_to_percent
 - Xapian::MSet, [68](#)
- create
 - Xapian::Weight, [120](#)
- Database
 - Xapian::Database, [31](#)
- DB_CREATE
 - Xapian, [19](#)
- DB_CREATE_OR_OPEN
 - Xapian, [19](#)
- DB_CREATE_OR_OVERWRITE
 - Xapian, [19](#)
- DB_OPEN
 - Xapian, [19](#)
- delete_document
 - Xapian::WritableDatabase, [127](#), [128](#)
- doccount
 - Xapian, [16](#)
- doccount_diff
 - Xapian, [16](#)
- docid
 - Xapian, [16](#)
- doclength
 - Xapian, [16](#)
- Document
 - Xapian::Document, [37](#)
- empty
 - Xapian::ESet, [54](#)
 - Xapian::MSet, [68](#)
 - Xapian::RSet, [90](#)
- end
 - Xapian::ESet, [54](#)
 - Xapian::MSet, [68](#)
- Enquire
 - Xapian::Enquire, [43](#)
- ErrorHandler
 - Xapian::ErrorHandler, [51](#)
- ESet
 - Xapian::ESet, [54](#)
- ESetIterator
 - Xapian::ESetIterator, [57](#)
- ExpandDeciderAnd
 - Xapian::ExpandDeciderAnd, [61](#)
- ExpandDeciderFilterTerms
 - Xapian::ExpandDeciderFilterTerms, [62](#)
- feature_flag
 - Xapian::QueryParser, [85](#)
- fetch
 - Xapian::MSet, [68](#)
- FLAG_BOOLEAN
 - Xapian::QueryParser, [85](#)
- FLAG_BOOLEAN_ANY_CASE
 - Xapian::QueryParser, [85](#)
- FLAG_LOVEHATE
 - Xapian::QueryParser, [85](#)
- FLAG_PARTIAL
 - Xapian::QueryParser, [86](#)
- FLAG_PHRASE
 - Xapian::QueryParser, [85](#)
- FLAG_PURE_NOT
 - Xapian::QueryParser, [86](#)
- FLAG_WILDCARD
 - Xapian::QueryParser, [86](#)
- flush
 - Xapian::WritableDatabase, [128](#)
- get_available_languages
 - Xapian::Stem, [96](#)
- get_avlength
 - Xapian::Database, [32](#)
- get_collapse_count
 - Xapian::MSetIterator, [74](#)
- get_collapse_key
 - Xapian::MSetIterator, [74](#)
- get_collection_freq
 - Xapian::Database, [32](#)
- get_data
 - Xapian::Document, [38](#)
- get_default_op
 - Xapian::QueryParser, [87](#)
- get_description
 - Xapian::Database, [32](#)
 - Xapian::Document, [39](#)
 - Xapian::Enquire, [44](#)
 - Xapian::ESet, [55](#)
 - Xapian::ESetIterator, [57](#)
 - Xapian::MSet, [69](#)

- Xapian::MSetIterator, 74
- Xapian::PositionIterator, 78
- Xapian::PostingIterator, 81
- Xapian::QueryParser, 87
- Xapian::RSet, 91
- Xapian::SimpleStopper, 93
- Xapian::Stem, 96
- Xapian::Stopper, 97
- Xapian::TermGenerator, 100
- Xapian::TermIterator, 105
- Xapian::ValueIterator, 117
- Xapian::WritableDatabase, 129
- get_doccount
 - Xapian::Database, 32
- get_doclength
 - Xapian::Database, 32
 - Xapian::PostingIterator, 81
- get_document
 - Xapian::Database, 33
 - Xapian::MSetIterator, 74
 - Xapian::TermGenerator, 100
- get_ebound
 - Xapian::ESet, 55
- get_eset
 - Xapian::Enquire, 44
- get_firstitem
 - Xapian::MSet, 69
- get_lastdocid
 - Xapian::Database, 33
- get_matches_estimated
 - Xapian::MSet, 69
- get_matches_lower_bound
 - Xapian::MSet, 69
- get_matches_upper_bound
 - Xapian::MSet, 69
- get_matching_terms_begin
 - Xapian::Enquire, 45
- get_matching_terms_end
 - Xapian::Enquire, 46
- get_max_attained
 - Xapian::MSet, 69
- get_max_possible
 - Xapian::MSet, 70
- get_maxextra
 - Xapian::BM25Weight, 23
 - Xapian::BoolWeight, 27
 - Xapian::TradWeight, 109
 - Xapian::Weight, 120
- get_maxpart
 - Xapian::BM25Weight, 23
- Xapian::BoolWeight, 27
- Xapian::TradWeight, 109
- Xapian::Weight, 120
- get_mset
 - Xapian::Enquire, 46
- get_percent
 - Xapian::MSetIterator, 75
- get_query
 - Xapian::Enquire, 46
- get_rank
 - Xapian::MSetIterator, 75
- get_sumextra
 - Xapian::BM25Weight, 24
 - Xapian::BoolWeight, 27
 - Xapian::TradWeight, 109
 - Xapian::Weight, 121
- get_sumpart
 - Xapian::BM25Weight, 24
 - Xapian::BoolWeight, 28
 - Xapian::TradWeight, 109
 - Xapian::Weight, 121
- get_sumpart_needs_doclength
 - Xapian::BM25Weight, 24
 - Xapian::BoolWeight, 28
 - Xapian::TradWeight, 110
 - Xapian::Weight, 121
- get_termfreq
 - Xapian::Database, 33
 - Xapian::MSet, 70
 - Xapian::TermIterator, 105
- get_termpos
 - Xapian::TermGenerator, 100
- get_termweight
 - Xapian::MSet, 70
- get_value
 - Xapian::Document, 39
- get_valueno
 - Xapian::ValueIterator, 117
- get_wdf
 - Xapian::PostingIterator, 81
 - Xapian::TermIterator, 105
- get_weight
 - Xapian::ESetIterator, 58
 - Xapian::MSetIterator, 75
- has_positions
 - Xapian::Database, 33
- include/xapian/database.h, 131
- include/xapian/dbfactory.h, 134

- include/xapian/document.h, [136](#)
- include/xapian/enquire.h, [138](#)
- include/xapian/errorhandler.h, [140](#)
- include/xapian/expanddecider.h, [141](#)
- include/xapian/positioniterator.h, [142](#)
- include/xapian/postingiterator.h, [144](#)
- include/xapian/query.h, [146](#)
- include/xapian/queryparser.h, [148](#)
- include/xapian/stem.h, [150](#)
- include/xapian/termgenerator.h, [151](#)
- include/xapian/termiterator.h, [152](#)
- include/xapian/types.h, [154](#)
- include/xapian/unicode.h, [156](#)
- include/xapian/valueiterator.h, [159](#)
- increase_termpos
 - Xapian::TermGenerator, [101](#)
- index_text
 - Xapian::TermGenerator, [101](#)
- index_text_without_positions
 - Xapian::TermGenerator, [101](#)
- iterator_category
 - Xapian::ESetIterator, [57](#)
 - Xapian::MSetIterator, [73](#)
 - Xapian::PostingIterator, [80](#)
 - Xapian::TermIterator, [104](#)
 - Xapian::Utf8Iterator, [112](#)
 - Xapian::ValueIterator, [117](#)
- keep_alive
 - Xapian::Database, [33](#)
- left
 - Xapian::Utf8Iterator, [113](#)
- major_version
 - Xapian, [17](#)
- max_size
 - Xapian::ESet, [55](#)
 - Xapian::MSet, [70](#)
- minor_version
 - Xapian, [17](#)
- MSet
 - Xapian::MSet, [67](#)
- MSetIterator
 - Xapian::MSetIterator, [74](#)
- name
 - Xapian::BM25Weight, [24](#)
 - Xapian::BoolWeight, [28](#)
 - Xapian::TradWeight, [110](#)
 - Xapian::Weight, [121](#)
- operator *
 - Xapian::ESetIterator, [58](#)
 - Xapian::MSetIterator, [75](#)
 - Xapian::PostingIterator, [81](#)
 - Xapian::TermIterator, [105](#)
 - Xapian::Utf8Iterator, [114](#)
 - Xapian::ValueIterator, [118](#)
- operator!=
 - Xapian, [18](#)
 - Xapian::Utf8Iterator, [114](#)
- operator()
 - Xapian::ErrorHandler, [51](#)
 - Xapian::ExpandDecider, [59](#)
 - Xapian::ExpandDeciderAnd, [61](#)
 - Xapian::ExpandDeciderFilterTerms, [63](#)
 - Xapian::MatchDecider, [64](#)
 - Xapian::SimpleStopper, [93](#)
 - Xapian::Stem, [96](#)
 - Xapian::Stopper, [97](#)
- operator++
 - Xapian::ESetIterator, [58](#)
 - Xapian::MSetIterator, [75](#)
 - Xapian::Utf8Iterator, [114](#)
 - Xapian::ValueIterator, [118](#)
- operator--
 - Xapian::ESetIterator, [58](#)
 - Xapian::MSetIterator, [75](#)
- operator->
 - Xapian::ValueIterator, [118](#)
- operator=
 - Xapian::Database, [33](#)
 - Xapian::Document, [39](#)
 - Xapian::ESet, [55](#)
 - Xapian::ESetIterator, [58](#)
 - Xapian::MSet, [70](#)
 - Xapian::MSetIterator, [75](#)
 - Xapian::PositionIterator, [78](#)
 - Xapian::PostingIterator, [81](#)
 - Xapian::QueryParser, [87](#)
 - Xapian::RSet, [91](#)
 - Xapian::Stem, [96](#)
 - Xapian::TermGenerator, [101](#)
 - Xapian::TermIterator, [105](#)
 - Xapian::ValueIterator, [118](#)
 - Xapian::WritableDatabase, [129](#)
- operator==
 - Xapian, [18](#)

- Xapian::PositionIterator, [78](#)
- Xapian::PostingIterator, [82](#)
- Xapian::Utf8Iterator, [114](#)
- operator[]
 - Xapian::ESet, [55](#)
 - Xapian::MSet, [70](#)
- parse_query
 - Xapian::QueryParser, [87](#)
- percent
 - Xapian, [16](#)
- PositionIterator
 - Xapian::PositionIterator, [78](#)
- positionlist_begin
 - Xapian::Database, [33](#)
 - Xapian::PostingIterator, [81](#)
 - Xapian::TermIterator, [105](#)
- positionlist_count
 - Xapian::TermIterator, [105](#)
- positionlist_end
 - Xapian::Database, [34](#)
 - Xapian::PostingIterator, [81](#)
 - Xapian::TermIterator, [105](#)
- PostingIterator
 - Xapian::PostingIterator, [80](#)
- postlist_begin
 - Xapian::Database, [34](#)
- postlist_end
 - Xapian::Database, [34](#)
- QueryParser
 - Xapian::QueryParser, [86](#)
- raw
 - Xapian::Utf8Iterator, [114](#)
- register_match_decider
 - Xapian::Enquire, [47](#)
- remove_document
 - Xapian::RSet, [91](#)
- remove_posting
 - Xapian::Document, [39](#)
- remove_term
 - Xapian::Document, [39](#)
- remove_value
 - Xapian::Document, [40](#)
- reopen
 - Xapian::Database, [34](#)
- replace_document
 - Xapian::WritableDatabase, [129](#), [130](#)
- revision
 - Xapian, [18](#)
- RSet
 - Xapian::RSet, [90](#)
- serialise
 - Xapian::BM25Weight, [24](#)
 - Xapian::BoolWeight, [28](#)
 - Xapian::TradWeight, [110](#)
 - Xapian::Weight, [122](#)
- set_collapse_key
 - Xapian::Enquire, [47](#)
- set_cutoff
 - Xapian::Enquire, [47](#)
- set_data
 - Xapian::Document, [40](#)
- set_database
 - Xapian::QueryParser, [88](#)
- set_default_op
 - Xapian::QueryParser, [88](#)
- set_docid_order
 - Xapian::Enquire, [48](#)
- set_document
 - Xapian::TermGenerator, [101](#)
- set_query
 - Xapian::Enquire, [48](#)
- set_sort_by_relevance
 - Xapian::Enquire, [49](#)
- set_sort_by_relevance_then_value
 - Xapian::Enquire, [49](#)
- set_sort_by_value
 - Xapian::Enquire, [49](#)
- set_sort_by_value_then_relevance
 - Xapian::Enquire, [49](#)
- set_stemmer
 - Xapian::QueryParser, [88](#)
 - Xapian::TermGenerator, [101](#)
- set_stemming_strategy
 - Xapian::QueryParser, [88](#)
- set_stopper
 - Xapian::QueryParser, [88](#)
 - Xapian::TermGenerator, [102](#)
- set_termpos
 - Xapian::TermGenerator, [102](#)
- set_weighting_scheme
 - Xapian::Enquire, [50](#)
- SimpleStopper
 - Xapian::SimpleStopper, [93](#)
- size
 - Xapian::ESet, [55](#)

- Xapian::MSet, [71](#)
 - Xapian::RSet, [91](#)
- skip_to
 - Xapian::PostingIterator, [81](#)
 - Xapian::TermIterator, [105](#)
- Stem
 - Xapian::Stem, [94](#), [95](#)
- stoplist_begin
 - Xapian::QueryParser, [88](#)
- swap
 - Xapian::ESet, [55](#)
 - Xapian::MSet, [71](#)
- term_exists
 - Xapian::Database, [34](#)
- termcount
 - Xapian, [16](#)
- termcount_diff
 - Xapian, [16](#)
- TermGenerator
 - Xapian::TermGenerator, [100](#)
- TermIterator
 - Xapian::TermIterator, [104](#)
- termlist_begin
 - Xapian::Database, [34](#)
 - Xapian::Document, [40](#)
- termlist_count
 - Xapian::Document, [40](#)
- termlist_end
 - Xapian::Database, [34](#)
 - Xapian::Document, [40](#)
- termpos
 - Xapian, [16](#)
- termpos_diff
 - Xapian, [17](#)
- timeout
 - Xapian, [17](#)
- TradWeight
 - Xapian::TradWeight, [108](#)
- unserialise
 - Xapian::BM25Weight, [24](#)
 - Xapian::BoolWeight, [28](#)
 - Xapian::TradWeight, [110](#)
 - Xapian::Weight, [122](#)
- unstem_begin
 - Xapian::QueryParser, [88](#)
- Utf8Iterator
 - Xapian::Utf8Iterator, [112](#), [113](#)
- value_type
 - Xapian::MSet, [67](#)
- ValueIterator
 - Xapian::ValueIterator, [117](#)
- valueno
 - Xapian, [17](#)
- valueno_diff
 - Xapian, [17](#)
- values_begin
 - Xapian::Document, [40](#)
- values_count
 - Xapian::Document, [40](#)
- values_end
 - Xapian::Document, [40](#)
- version_string
 - Xapian, [18](#)
- weight
 - Xapian, [17](#)
- WritableDatabase
 - Xapian::WritableDatabase, [124](#), [125](#)
- Xapian, [11](#)
 - BAD_VALUENO, [19](#)
 - DB_CREATE, [19](#)
 - DB_CREATE_OR_OPEN, [19](#)
 - DB_CREATE_OR_OVERWRITE, [19](#)
 - DB_OPEN, [19](#)
 - doccount, [16](#)
 - doccount_diff, [16](#)
 - docid, [16](#)
 - doclength, [16](#)
 - major_version, [17](#)
 - minor_version, [17](#)
 - operator!=, [18](#)
 - operator==, [18](#)
 - percent, [16](#)
 - revision, [18](#)
 - termcount, [16](#)
 - termcount_diff, [16](#)
 - termpos, [16](#)
 - termpos_diff, [17](#)
 - timeout, [17](#)
 - valueno, [17](#)
 - valueno_diff, [17](#)
 - version_string, [18](#)
 - weight, [17](#)
 - XAPIAN_DEPRECATED, [18](#)
- Xapian::BM25Weight, [21](#)

- BM25Weight, 23
- clone, 23
- get_maxextra, 23
- get_maxpart, 23
- get_sumextra, 24
- get_sumpart, 24
- get_sumpart_needs_doclength, 24
- name, 24
- serialise, 24
- unserialise, 24
- Xapian::BoolWeight, 26
- Xapian::BoolWeight
 - clone, 27
 - get_maxextra, 27
 - get_maxpart, 27
 - get_sumextra, 27
 - get_sumpart, 28
 - get_sumpart_needs_doclength, 28
 - name, 28
 - serialise, 28
 - unserialise, 28
- Xapian::Database, 29
 - ~Database, 31
 - add_database, 32
 - allterms_begin, 32
 - allterms_end, 32
 - Database, 31
 - get_avlength, 32
 - get_collection_freq, 32
 - get_description, 32
 - get_doccount, 32
 - get_doclength, 32
 - get_document, 33
 - get_lastdocid, 33
 - get_termfreq, 33
 - has_positions, 33
 - keep_alive, 33
 - operator=, 33
 - positionlist_begin, 33
 - positionlist_end, 34
 - postlist_begin, 34
 - postlist_end, 34
 - reopen, 34
 - term_exists, 34
 - termlist_begin, 34
 - termlist_end, 34
- Xapian::Document, 36
 - ~Document, 37
 - add_posting, 38
 - add_term, 38
 - add_value, 38
 - clear_terms, 38
 - clear_values, 38
 - Document, 37
 - get_data, 38
 - get_description, 39
 - get_value, 39
 - operator=, 39
 - remove_posting, 39
 - remove_term, 39
 - remove_value, 40
 - set_data, 40
 - termlist_begin, 40
 - termlist_count, 40
 - termlist_end, 40
 - values_begin, 40
 - values_count, 40
 - values_end, 40
- Xapian::Enquire, 41
 - ~Enquire, 43
 - Enquire, 43
 - get_description, 44
 - get_eset, 44
 - get_matching_terms_begin, 45
 - get_matching_terms_end, 46
 - get_mset, 46
 - get_query, 46
 - register_match_decider, 47
 - set_collapse_key, 47
 - set_cutoff, 47
 - set_docid_order, 48
 - set_query, 48
 - set_sort_by_relevance, 49
 - set_sort_by_relevance_then_value, 49
 - set_sort_by_value, 49
 - set_sort_by_value_then_relevance, 49
 - set_weighting_scheme, 50
 - XAPIAN_DEPRECATED, 50
- Xapian::ErrorHandler, 51
- Xapian::ErrorHandler
 - ~ErrorHandler, 51
 - ErrorHandler, 51
 - operator(), 51
- Xapian::ESet, 53
 - ~ESet, 54
 - back, 54

- begin, [54](#)
- empty, [54](#)
- end, [54](#)
- ESet, [54](#)
- get_description, [55](#)
- get_ebound, [55](#)
- max_size, [55](#)
- operator=, [55](#)
- operator[], [55](#)
- size, [55](#)
- swap, [55](#)
- Xapian::ESetIterator, [56](#)
- Xapian::ESetIterator
 - ESetIterator, [57](#)
 - get_description, [57](#)
 - get_weight, [58](#)
 - iterator_category, [57](#)
 - operator*, [58](#)
 - operator++, [58](#)
 - operator--, [58](#)
 - operator=, [58](#)
- Xapian::ExpandDecider, [59](#)
- Xapian::ExpandDecider
 - ~ExpandDecider, [59](#)
 - operator(), [59](#)
- Xapian::ExpandDeciderAnd, [60](#)
- Xapian::ExpandDeciderAnd
 - ExpandDeciderAnd, [61](#)
 - operator(), [61](#)
- Xapian::ExpandDeciderFilterTerms, [62](#)
- Xapian::ExpandDeciderFilterTerms
 - ExpandDeciderFilterTerms, [62](#)
 - operator(), [63](#)
- Xapian::MatchDecider, [64](#)
- Xapian::MatchDecider
 - ~MatchDecider, [64](#)
 - operator(), [64](#)
- Xapian::MSet, [65](#)
- Xapian::MSet
 - ~MSet, [67](#)
 - back, [67](#)
 - begin, [67](#)
 - convert_to_percent, [68](#)
 - empty, [68](#)
 - end, [68](#)
 - fetch, [68](#)
 - get_description, [69](#)
 - get_firstitem, [69](#)
 - get_matches_estimated, [69](#)
 - get_matches_lower_bound, [69](#)
 - get_matches_upper_bound, [69](#)
 - get_max_attained, [69](#)
 - get_max_possible, [70](#)
 - get_termfreq, [70](#)
 - get_termweight, [70](#)
 - max_size, [70](#)
 - MSet, [67](#)
 - operator=, [70](#)
 - operator[], [70](#)
 - size, [71](#)
 - swap, [71](#)
 - value_type, [67](#)
- Xapian::MSetIterator, [72](#)
- Xapian::MSetIterator
 - get_collapse_count, [74](#)
 - get_collapse_key, [74](#)
 - get_description, [74](#)
 - get_document, [74](#)
 - get_percent, [75](#)
 - get_rank, [75](#)
 - get_weight, [75](#)
 - iterator_category, [73](#)
 - MSetIterator, [74](#)
 - operator*, [75](#)
 - operator++, [75](#)
 - operator--, [75](#)
 - operator=, [75](#)
- Xapian::PositionIterator, [77](#)
- Xapian::PositionIterator
 - ~PositionIterator, [78](#)
 - get_description, [78](#)
 - operator=, [78](#)
 - operator==, [78](#)
 - PositionIterator, [78](#)
- Xapian::PostingIterator, [79](#)
- Xapian::PostingIterator
 - ~PostingIterator, [80](#)
 - get_description, [81](#)
 - get_doclength, [81](#)
 - get_wdf, [81](#)
 - iterator_category, [80](#)
 - operator*, [81](#)
 - operator=, [81](#)
 - operator==, [82](#)
 - positionlist_begin, [81](#)
 - positionlist_end, [81](#)
 - PostingIterator, [80](#)
 - skip_to, [81](#)
- Xapian::Query, [83](#)
- Xapian::QueryParser, [84](#)

- FLAG_BOOLEAN, 85
- FLAG_BOOLEAN_ANY_CASE, 85
- FLAG_LOVEHATE, 85
- FLAG_PARTIAL, 86
- FLAG_PHRASE, 85
- FLAG_PURE_NOT, 86
- FLAG_WILDCARD, 86
- Xapian::QueryParser
 - ~QueryParser, 86
 - add_boolean_prefix, 86
 - add_prefix, 87
 - add_valuerangeprocessor, 87
 - feature_flag, 85
 - get_default_op, 87
 - get_description, 87
 - operator=, 87
 - parse_query, 87
 - QueryParser, 86
 - set_database, 88
 - set_default_op, 88
 - set_stemmer, 88
 - set_stemming_strategy, 88
 - set_stopper, 88
 - stoplist_begin, 88
 - unstem_begin, 88
- Xapian::RSet, 89
 - ~RSet, 90
 - add_document, 90
 - contains, 90
 - empty, 90
 - get_description, 91
 - operator=, 91
 - remove_document, 91
 - RSet, 90
 - size, 91
- Xapian::SimpleStopper, 92
- Xapian::SimpleStopper
 - ~SimpleStopper, 93
 - add, 93
 - get_description, 93
 - operator(), 93
 - SimpleStopper, 93
- Xapian::Stem, 94
 - ~Stem, 95
 - get_available_languages, 96
 - get_description, 96
 - operator(), 96
 - operator=, 96
 - Stem, 94, 95
- Xapian::Stopper, 97
 - ~Stopper, 97
 - get_description, 97
 - operator(), 97
- Xapian::TermGenerator, 99
- Xapian::TermGenerator
 - ~TermGenerator, 100
 - get_description, 100
 - get_document, 100
 - get_termpos, 100
 - increase_termpos, 101
 - index_text, 101
 - index_text_without_positions, 101
 - operator=, 101
 - set_document, 101
 - set_stemmer, 101
 - set_stopper, 102
 - set_termpos, 102
 - TermGenerator, 100
- Xapian::TermIterator, 103
- Xapian::TermIterator
 - ~TermIterator, 104
 - get_description, 105
 - get_termfreq, 105
 - get_wdf, 105
 - iterator_category, 104
 - operator *, 105
 - operator=, 105
 - positionlist_begin, 105
 - positionlist_count, 105
 - positionlist_end, 105
 - skip_to, 105
 - TermIterator, 104
- Xapian::TradWeight, 107
- Xapian::TradWeight
 - clone, 109
 - get_maxextra, 109
 - get_maxpart, 109
 - get_sumextra, 109
 - get_sumpart, 109
 - get_sumpart_needs_doclength, 110
 - name, 110
 - serialise, 110
 - TradWeight, 108
 - unserialise, 110
- Xapian::Utf8Iterator, 111
 - assign, 113
 - iterator_category, 112

- left, [113](#)
- operator *, [114](#)
- operator!=, [114](#)
- operator++, [114](#)
- operator==, [114](#)
- raw, [114](#)
- Utf8Iterator, [112](#), [113](#)
- Xapian::ValueIterator, [116](#)
- Xapian::ValueIterator
 - get_description, [117](#)
 - get_valueno, [117](#)
 - iterator_category, [117](#)
 - operator *, [118](#)
 - operator++, [118](#)
 - operator->, [118](#)
 - operator=, [118](#)
 - ValueIterator, [117](#)
- Xapian::Weight, [119](#)
 - create, [120](#)
 - get_maxextra, [120](#)
 - get_maxpart, [120](#)
 - get_sumextra, [121](#)
 - get_sumpart, [121](#)
 - get_sumpart_needs_doclength, [121](#)
 - name, [121](#)
 - serialise, [122](#)
 - unserialise, [122](#)
- Xapian::WritableDatabase, [123](#)
- Xapian::WritableDatabase
 - ~WritableDatabase, [124](#)
 - add_document, [125](#)
 - begin_transaction, [126](#)
 - cancel_transaction, [126](#)
 - commit_transaction, [127](#)
 - delete_document, [127](#), [128](#)
 - flush, [128](#)
 - get_description, [129](#)
 - operator=, [129](#)
 - replace_document, [129](#), [130](#)
 - WritableDatabase, [124](#), [125](#)
- XAPIAN_DEPRECATED
 - Xapian, [18](#)
 - Xapian::Enquire, [50](#)