haystack-cpp

1.0

Generated by Doxygen 1.8.8

Wed Sep 24 2014 20:51:45

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Class Documentation

3.1 haystack::And Class Reference

Inheritance diagram for haystack::And:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.2 haystack::Bin Class Reference

`#include <bin.hpp>`

Inheritance diagram for haystack::Bin:

Public Member Functions

- const Type type () const
- Bin (const std::string &val)
- const std::string to_string () const
- const std::string to_zinc () const
- bool operator== (const Bin &b) const
- bool **operator!=** (const Bin &b) const
- bool **operator==** (const std::string &other) const
- bool **operator==** (const Val &other) const
- bool **operator**> (const Val &other) const
- bool **operator**< (const Val &other) const
- auto_ptr_t clone () const

Public Attributes

- const std::string value

Additional Inherited Members

3.2.1 Detailed Description

Bin models a binary file with a MIME type.

See also

Project Haystack

3.2.2 Constructor & Destructor Documentation

3.2.2.1 haystack::Bin::Bin (const std::string & *val*)

Construct from std::string

3.2.3 Member Function Documentation

3.2.3.1 auto_ptr_t haystack::Bin::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.2.3.2 bool haystack::Bin::operator== (const Bin & *b*) const

Equality is value based

3.2.3.3 const std::string haystack::Bin::to_string ( ) const [virtual]

MIME type for binary file

Reimplemented from haystack::Val.

3.2.3.4 const std::string haystack::Bin::to_zinc ( ) const [virtual]

Encode as "Bin(<mime>)"

Implements haystack::Val.

3.2.3.5 const Type haystack::Bin::type ( ) const [inline], [virtual]

Return this Val type

Implements haystack::Val.

3.2.4 Member Data Documentation

3.2.4.1 const std::string haystack::Bin::value

This string value

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

- include/bin.hpp

3.3 haystack::Bool Class Reference

`#include <bool.hpp>`

Inheritance diagram for haystack::Bool:

Public Member Functions

- const Type type () const
- **Bool** (bool val)
- const std::string to_string () const
- const std::string to_zinc () const
- bool operator== (const Bool &other) const
- bool **operator!=** (const Bool &other) const
- bool **operator==** (const Val &other) const
- bool **operator**> (const Val &other) const
- bool **operator**< (const Val &other) const
- auto_ptr_t clone () const

Public Attributes

- const bool value

Static Public Attributes

- static const Bool TRUE_VAL
- static const Bool FALSE_VAL

Additional Inherited Members

3.3.1 Detailed Description

Bool defines true/false tag values.

See also

Project Haystack

3.3.2 Member Function Documentation

3.3.2.1 auto_ptr_t haystack::Bool::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.3.2.2 bool haystack::Bool::operator== (const Bool & *other*) const

Equality

3.3.2.3 const std::string haystack::Bool::to_string ( ) const [virtual]

Encode as "true" or "false"

Reimplemented from haystack::Val.

3.3.2.4 const std::string haystack::Bool::to_zinc ( ) const [virtual]

Encode as "T" or "F"

Implements haystack::Val.

3.3.2.5 const Type haystack::Bool::type ( ) const [inline], [virtual]

Return this Val type

Implements haystack::Val.

3.3.3 Member Data Documentation

3.3.3.1 const Bool haystack::Bool::FALSE_VAL [static]

Singleton value for false

3.3.3.2 const Bool haystack::Bool::TRUE_VAL [static]

Singleton value for true

3.3.3.3 const bool haystack::Bool::value

This bool value

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

- include/bool.hpp

3.4 haystack::Client Class Reference

Inheritance diagram for haystack::Client:

Public Types

- typedef std::auto_ptr< Client > **auto_ptr_t**

Public Member Functions

- **Client** (const std::string &uri, const std::string &user, const std::string &pass)
- Client & open ()
- Dict::auto_ptr_t about () const
- Grid::auto_ptr_t ops ()
- Grid::auto_ptr_t formats ()
- Grid::auto_ptr_t **call** (const std::string &op, const Grid &req) const

Static Public Member Functions

- static Client::auto_ptr_t open (const std::string &uri, const std::string &user, const std::string &pass)

3.4.1 Member Function Documentation

3.4.1.1 Dict::auto_ptr_t haystack::Client::about ( ) const [virtual]

Call "about" to query summary info.

Implements haystack::Proj.

3.4.1.2 Grid::auto_ptr_t haystack::Client::formats ( )

Call "formats" to query which MIME formats are available.

3.4.1.3 static Client::auto_ptr_t haystack::Client::open (const std::string & *uri,* const std::string & *user,* const std::string & *pass*) [static]

Convenience for construction and call to open().

3.4.1.4 Client& haystack::Client::open ( )

Authenticate the client and return this.

3.4.1.5 Grid::auto_ptr_t haystack::Client::ops ( )

Call "ops" to query which operations are supported by server.

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

- http_client/include/client.hpp

3.5 haystack::CmpFilter Class Reference

Inheritance diagram for haystack::CmpFilter:

Protected Member Functions

- **CmpFilter** (Path::auto_ptr_t, Val::auto_ptr_t v)
- std::string **str** () const
- bool **same_type** (const Val &v) const
- virtual std::string **cmp_str** () const =0
- const Val & **val** () const

Protected Attributes

- Val::auto_ptr_t **m_val**

Additional Inherited Members

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

- include/filter.hpp

3.6 haystack::Col Class Reference

`#include <col.hpp>`

Inheritance diagram for haystack::Col:

Public Member Functions

- const std::string name () const
- const std::string dis () const
- const Dict & meta () const
- bool operator== (const Col &that)
- bool **operator!=** (const Col &that)

Public Attributes

- const size_t **m_index**
- const std::string **m_name**
- const std::auto_ptr< Dict > **m_meta**

Friends

- class **Grid**
- class **Row**

3.6.1 Detailed Description

Col is a column in a Grid.

See also

Project Haystack

3.6.2 Member Function Documentation

3.6.2.1 const std::string haystack::Col::dis ( ) const

Return display name of column which is meta.dis or name

3.6.2.2 const Dict& haystack::Col::meta ( ) const

Column meta-data tags

3.6.2.3 const std::string haystack::Col::name ( ) const

Return programatic name of column

3.6.2.4 bool haystack::Col::operator== (const Col & *that*)

Equality is name and meta

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

- include/col.hpp

3.7 haystack::CompoundFilter Class Reference

Inheritance diagram for haystack::CompoundFilter:

Protected Member Functions

- **CompoundFilter** (Filter::shared_ptr_t a, Filter::shared_ptr_t b)
- virtual std::string **keyword** () const =0
- Type **type** () const
- std::string **str** () const
- const Filter & **a** () const
- const Filter & **b** () const

Protected Attributes

- Filter::shared_ptr_t **m_a**
- Filter::shared_ptr_t **m_b**

Additional Inherited Members

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

- include/filter.hpp

3.8 haystack::const_proj_iterator Class Reference

Inheritance diagram for haystack::const_proj_iterator:

Public Member Functions

- **const_proj_iterator** (boost::ptr_map< std::string, Dict >::const_iterator it)
- void **increment** ()
- bool **equal** (const_proj_iterator const &other) const
- const Dict & **dereference** () const

Friends

- class **boost::iterator_core_access**

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

- http_server/include/server.hpp

3.9 haystack::const_row_iterator Class Reference

Inheritance diagram for haystack::const_row_iterator:

Friends

- class **Row**
- class **boost::iterator_core_access**

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

- include/row.hpp

3.10 haystack::Coord Class Reference

`#include <coord.hpp>`

Inheritance diagram for haystack::Coord:

Public Member Functions

- const Type type () const
- **Coord** (double, double)
- double lat () const
- double lng () const

- const std::string to_zinc () const
- bool operator== (const Coord &other) const
- bool **operator!=** (const Coord &other) const
- bool **operator**< (const Val &other) const
- bool **operator**> (const Val &other) const
- bool **operator==** (const std::string &other) const
- bool **operator==** (const Val &other) const
- auto_ptr_t clone () const

Static Public Member Functions

- static Coord make (const std::string &val)
- static bool is_lat (double lat)
- static bool is_lng (double lng)

Public Attributes

- const int32_t ulat
- const int32_t ulng

Additional Inherited Members

3.10.1 Detailed Description

Coord models a geographic coordinate as latitude and longitude.

See also

Project Haystack

3.10.2 Member Function Documentation

3.10.2.1 auto_ptr_t haystack::Coord::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.10.2.2 static bool haystack::Coord::is_lat (double *lat*) [static]

Return if given latitude is legal value between -90.0 and +90.0

3.10.2.3 static bool haystack::Coord::is_lng (double *lng*) [static]

Return if given is longtitude is legal value between -180.0 and +180.0

3.10.2.4 double haystack::Coord::lat ( ) const

Latitude in decimal degrees

3.10.2.5 double haystack::Coord::lng ( ) const

Longtitude in decimal degrees

3.10.2.6 static Coord haystack::Coord::make (const std::string & *val*) [static]

Parse from string fomat "C(lat,lng)" or raise runtime exception

3.10.2.7 bool haystack::Coord::operator== (const Coord & *other*) const

Equality is value based

3.10.2.8 const std::string haystack::Coord::to_zinc ( ) const [virtual]

Encode using double quotes and back slash escapes

Implements haystack::Val.

3.10.2.9 const Type haystack::Coord::type ( ) const [inline], [virtual]

Return this Val type

Implements haystack::Val.

3.10.3 Member Data Documentation

3.10.3.1 const int32_t haystack::Coord::ulat

Latitude in micro-degrees

3.10.3.2 const int32_t haystack::Coord::ulng

Longitude in micro-degrees

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

- include/coord.hpp

3.11 haystack::Date Class Reference

`#include <date.hpp>`

Inheritance diagram for haystack::Date:

Public Member Functions

- const Type type () const
- **Date** (int year, int month, int day)
- const std::string to_zinc () const
- Date inc_days (int) const

- Date dec_days (int) const
- int weekday () const
- std::auto_ptr< DateTime > midnight (const TimeZone &tz) const
- bool operator== (const Date &b) const
- bool **operator!=** (const Date &b) const
- bool operator< (const Val &b) const
- bool **operator**> (const Val &b) const
- bool **operator==** (const Val &other) const
- auto_ptr_t clone () const

Static Public Member Functions

- static bool is_leap_year (int year)
- static int days_in_month (int year, int mon)
- static const Date today ()

Public Attributes

- const int year
- const int month
- const int day

Friends

- class **DateTime**

Additional Inherited Members

3.11.1 Detailed Description

Date models a date (day in year) tag value.

See also

Project Haystack

3.11.2 Member Function Documentation

3.11.2.1 auto_ptr_t haystack::Date::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.11.2.2 static int haystack::Date::days_in_month (int *year*, int *mon*) [static]

Return number of days in given year (xxxx) and month (1-12)

3.11.2.3 Date haystack::Date::dec_days (int) const

Return date in past given number of days

3.11.2.4 Date haystack::Date::inc_days (int) const

Return date in future given number of days

3.11.2.5 static bool haystack::Date::is_leap_year (int *year*) [static]

Return if given year a leap year

3.11.2.6 std::auto_ptr<DateTime> haystack::Date::midnight (const TimeZone & *tz*) const

Convert this date into DateTime for midnight in given timezone.

3.11.2.7 bool haystack::Date::operator< (const Val & *b*) const [virtual]

Comparator

Implements haystack::Val.

3.11.2.8 bool haystack::Date::operator== (const Date & *b*) const

Equality

3.11.2.9 const std::string haystack::Date::to_zinc ( ) const [virtual]

Encode as "YYYY-MM-DD"

Implements haystack::Val.

3.11.2.10 static const Date haystack::Date::today ( ) [static]

Get Date for current time in default timezone.

3.11.2.11 const Type haystack::Date::type ( ) const [inline], [virtual]

Return this Val type

Implements haystack::Val.

3.11.2.12 int haystack::Date::weekday ( ) const

Return day of week: Sunday is 1, Saturday is 7

3.11.3 Member Data Documentation

3.11.3.1 const int haystack::Date::day

Day of month as 1-31

3.11.3.2 const int haystack::Date::month

Month as 1-12 (Jan is 1, Dec is 12)

3.11.3.3 const int haystack::Date::year

Four digit year such as 2014

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

- include/date.hpp

3.12 haystack::DateTime Class Reference

`#include <datetime.hpp>`

Inheritance diagram for haystack::DateTime:

Public Member Functions

- const Type type () const
- **DateTime** (int year, int month, int day, int hour, int min, int sec, const TimeZone &tz, int tzOffset)
- **DateTime** (int year, int month, int day, int hour, int min, const TimeZone &tz, int tzOffset)
- **DateTime** (const Date &date, const Time &time)
- **DateTime** (const Date &date, const Time &time, const TimeZone &tz)
- **DateTime** (const Date &date, const Time &time, const TimeZone &tz, int tzOffset)
- const std::string to_zinc () const
- bool operator== (const DateTime &) const
- bool **operator!=** (const DateTime &) const
- bool **operator==** (const Val &other) const
- bool operator< (const Val &) const
- bool **operator**> (const Val &) const
- const int64_t millis () const
- auto_ptr_t clone () const

Static Public Member Functions

- static DateTime make_time_t (const time_t &ts, const TimeZone &=TimeZone::DEFAULT)
- static DateTime make (const int64_t &time, const TimeZone &=TimeZone::DEFAULT)
- static DateTime now (const TimeZone &=TimeZone::DEFAULT)

Public Attributes

- const Date date
- const Time time
- const TimeZone tz
- const int tz_offset

Friends

- class **DateTimeRange**

Additional Inherited Members

3.12.1 Detailed Description

DateTime models a timestamp with a specific timezone.

See also

Project Haystack

3.12.2 Member Function Documentation

3.12.2.1 auto_ptr_t haystack::DateTime::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.12.2.2 static DateTime haystack::DateTime::make (const int64_t & *time*, const TimeZone & = TimeZone::DEFAULT) [static]

construct from millis

3.12.2.3 static DateTime haystack::DateTime::make_time_t (const time_t & *ts*, const TimeZone & = TimeZone::DEFAULT) [static]

construct from time_t

3.12.2.4 const int64_t haystack::DateTime::millis ( ) const

Get this date time as Java milliseconds since epoch

3.12.2.5 static DateTime haystack::DateTime::now (const TimeZone & = TimeZone::DEFAULT) [static]

Get DateTime for current time in default timezone or optionaly for given timezone

3.12.2.6 bool haystack::DateTime::operator< (const Val &) const [virtual]

Comparator

Implements haystack::Val.

3.12.2.7 bool haystack::DateTime::operator== (const DateTime &) const

Equality

3.12.2.8 const std::string haystack::DateTime::to_zinc ( ) const [virtual]

Encode as "YYYY-MM-DD'T'hh:mm:ss.FFFz zzzz"

Implements haystack::Val.

3.12.2.9 const Type haystack::DateTime::type ( ) const [inline],[virtual]

Return this Val type

Implements haystack::Val.

3.12.3 Member Data Documentation

3.12.3.1 const Date haystack::DateTime::date

Date component of the timestamp

3.12.3.2 const Time haystack::DateTime::time

Time component of the timestamp

3.12.3.3 const TimeZone haystack::DateTime::tz

Timezone as Olson database city name

3.12.3.4 const int haystack::DateTime::tz_offset

Offset in seconds from UTC including DST offset

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

- include/datetime.hpp

3.13 haystack::DateTimeRange Class Reference

`#include <datetimerange.hpp>`

Inheritance diagram for haystack::DateTimeRange:

Public Types

- enum **WeekDays** {
  SUNDAY = 1, **MONDAY**, **TUESDAY**, **WEDNESDAY**,
  THURSDAY, **FRIDAY**, **SATURDAY** }
- typedef std::auto_ptr
  < DateTimeRange > **auto_ptr_t**

Public Member Functions

- DateTimeRange (const Date &date, const TimeZone &tz)
- DateTimeRange (const DateTime &start, const DateTime &end)
- DateTimeRange (const Date &start, const Date &end, const TimeZone &tz)
- const DateTime & start () const
- const DateTime & end () const
- const std::string **to_string** () const

Static Public Member Functions

- static DateTimeRange::auto_ptr_t this_week (const TimeZone &tz)
- static DateTimeRange::auto_ptr_t this_month (const TimeZone &tz)
- static DateTimeRange::auto_ptr_t this_year (const TimeZone &tz)
- static DateTimeRange::auto_ptr_t last_week (const TimeZone &tz)
- static DateTimeRange::auto_ptr_t last_month (const TimeZone &tz)
- static DateTimeRange::auto_ptr_t last_year (const TimeZone &tz)
- static DateTimeRange::auto_ptr_t make (std::string str, const TimeZone &tz)

3.13.1 Detailed Description

DateTimeRange models a starting and ending timestamp.

See also

Project Haystack

3.13.2 Constructor & Destructor Documentation

3.13.2.1 haystack::DateTimeRange::DateTimeRange (const Date & *date,* const TimeZone & *tz*)

Make for single date within given timezone

3.13.2.2 haystack::DateTimeRange::DateTimeRange (const DateTime & *start,* const DateTime & *end*)

Make from two timestamps

3.13.2.3 haystack::DateTimeRange::DateTimeRange (const Date & *start,* const Date & *end,* const TimeZone & *tz*)

Make for inclusive dates within given timezone

3.13.3 Member Function Documentation

3.13.3.1 const DateTime& haystack::DateTimeRange::end ( ) const [inline]

Inclusive ending timestamp

3.13.3.2 static DateTimeRange::auto_ptr_t haystack::DateTimeRange::last_month (const TimeZone & *tz*) [static]

Make a range which encompasses the previous month.

3.13.3.3 static DateTimeRange::auto_ptr_t haystack::DateTimeRange::last_week (const TimeZone & *tz*) [static]

Make a range which encompasses the previous week. The week is defined as Sunday thru Saturday.

3.13.3.4 static DateTimeRange::auto_ptr_t haystack::DateTimeRange::last_year (const TimeZone & *tz*) [static]

Make a range which encompasses the previous year.

3.13.3.5 static DateTimeRange::auto_ptr_t haystack::DateTimeRange::make (std::string *str,* const TimeZone & *tz*) [static]

Parse from string using the given timezone as context for date based ranges. The formats are:

- "today"
- "yesterday"
- "{date}"
- "{date},{date}"
- "{dateTime},{dateTime}"
- "{dateTime}" // anything after given timestamp Throw exception is invalid string format.

3.13.3.6 const DateTime& haystack::DateTimeRange::start ( ) const [inline]

Inclusive starting timestamp

3.13.3.7 static DateTimeRange::auto_ptr_t haystack::DateTimeRange::this_month (const TimeZone & *tz*) [static]

Make a range which encompasses the current month.

3.13.3.8 static DateTimeRange::auto_ptr_t haystack::DateTimeRange::this_week (const TimeZone & *tz*) [static]

Make a range which encompasses the current week. The week is defined as Sunday thru Saturday.

3.13.3.9 static DateTimeRange::auto_ptr_t haystack::DateTimeRange::this_year (const TimeZone & *tz*) [static]

Make a range which encompasses the current year.

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

- include/datetimerange.hpp

3.14 haystack::Dict Class Reference

`#include <dict.hpp>`

Inheritance diagram for haystack::Dict:

Public Types

- typedef std::auto_ptr< Dict > **auto_ptr_t**
- typedef boost::ptr_map < std::string, haystack::Val > **dict_t**
- typedef dict_t::const_iterator **const_iterator**

Public Member Functions

- virtual const bool is_empty () const
- virtual const size_t size () const
- virtual const bool has (const std::string &name) const
- virtual const bool missing (const std::string &name) const
- const Ref & id () const
- virtual const Val & get (const std::string &name, bool checked=true) const
- const_iterator begin () const
- const_iterator **end** () const
- const std::string to_string () const
- const std::string to_zinc () const
- const std::string dis () const
- bool get_bool (const std::string &name) const
- const std::string & get_str (const std::string &name) const
- const Ref & get_ref (const std::string &name) const
- long long get_int (const std::string &name) const
- double get_double (const std::string &name) const
- Dict & add (std::string name, Val::auto_ptr_t val)
- Dict & add (std::string name, const Val *val)
- Dict & add (std::string name, const Val &val)
- Dict & add (std::string name)
- Dict & add (std::string name, const std::string &val)
- Dict & add (std::string name, double val, const std::string &unit="")
- Dict & add (const Dict &other)
- virtual auto_ptr_t clone ()
- bool operator== (const Dict &b) const
- bool **operator!=** (const Dict &b) const

Static Public Member Functions

- static const bool is_tag_name (const std::string &)

Static Public Attributes

- static const Dict & EMPTY

3.14.1 Detailed Description

Dict is a map of name/Val pairs

See also

Project Haystack

3.14.2 Member Function Documentation

3.14.2.1 Dict& haystack::Dict::add (std::string *name,* Val::auto_ptr_t *val*)

Returns a dict with the value added, Val is owned by this dict

3.14.2.2 Dict& haystack::Dict::add (std::string *name,* const Val * *val*)

Returns a dict with the value added, Val* is owned by this dict

3.14.2.3 Dict& haystack::Dict::add (std::string *name,* const Val & *val*)

Returns a dict with the value added, Val& is cloned

3.14.2.4 Dict& haystack::Dict::add (std::string *name*)

Returns a dict with the Marker added

3.14.2.5 Dict& haystack::Dict::add (std::string *name,* const std::string & *val*)

Returns a dict with the Str added

3.14.2.6 Dict& haystack::Dict::add (std::string *name,* double *val,* const std::string & *unit* = " ")

Returns a dict with the Num added

3.14.2.7 Dict& haystack::Dict::add (const Dict & *other*)

Returns a dict with the Dict added

3.14.2.8 const_iterator haystack::Dict::begin ( ) const

Iteratator to walk each name / tag pair

3.14.2.9 virtual auto_ptr_t haystack::Dict::clone ( ) [virtual]

Clones this Dict and its values

3.14.2.10 const std::string haystack::Dict::dis ( ) const

Get display string for this entity:

- dis tag
- id tag

3.14.2.11 virtual const Val& haystack::Dict::get (const std::string & *name,* bool *checked* = true) const [virtual]

Get a tag by name

Reimplemented in haystack::Row.

3.14.2.12 bool haystack::Dict::get_bool (const std::string & *name*) const

Get tag as Bool or raise runtime_exception.

3.14.2.13 double haystack::Dict::get_double (const std::string & *name*) const

Get tag as Num or raise runtime_exception.

3.14.2.14 long long haystack::Dict::get_int (const std::string & *name*) const

Get tag as Num or raise runtime_exception.

3.14.2.15 const Ref& haystack::Dict::get_ref (const std::string & *name*) const

Get tag as Ref or raise runtime_exception.

3.14.2.16 const std::string& haystack::Dict::get_str (const std::string & *name*) const

Get tag as Str or raise runtime_exception.

3.14.2.17 virtual const bool haystack::Dict::has (const std::string & *name*) const [virtual]

Return if the given tag is present

3.14.2.18 const Ref& haystack::Dict::id ( ) const

Get the "id" tag as Ref

3.14.2.19 virtual const bool haystack::Dict::is_empty ( ) const [virtual]

Return true if size is zero

3.14.2.20 static const bool haystack::Dict::is_tag_name (const std::string &) [static]

Return if the given string is a legal tag name. The first char must be ASCII lower case letter. Rest of chars must be ASCII letter, digit, or underbar.

3.14.2.21 virtual const bool haystack::Dict::missing (const std::string & *name*) const [virtual]

Return if the given tag is not present

3.14.2.22 bool haystack::Dict::operator== (const Dict & *b*) const

Equality

3.14.2.23 virtual const size_t haystack::Dict::size ( ) const [virtual]

Return number of tag name / value pairs

Reimplemented in haystack::Row.

3.14.2.24 const std::string haystack::Dict::to_string ( ) const [inline]

String format is for human consumption only

3.14.2.25 const std::string haystack::Dict::to_zinc ( ) const

Encode values to zinc format

3.14.3 Member Data Documentation

3.14.3.1 const Dict& haystack::Dict::EMPTY [static]

Singleton for empty set of tags.

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

- include/dict.hpp

3.15 haystack::EmptyVal Class Reference

`#include <val.hpp>`

Inheritance diagram for haystack::EmptyVal:

Public Member Functions

- const std::string to_string () const
- const std::string to_zinc () const
- const Type type () const
- bool **operator==** (const Val &other) const
- bool **operator>** (const Val &other) const
- bool **operator<** (const Val &other) const
- auto_ptr_t clone () const

Static Public Attributes

- static const EmptyVal & **DEF**

Additional Inherited Members

3.15.1 Detailed Description

This class is the "empty" value for all types of Val

3.15.2 Member Function Documentation

3.15.2.1 auto_ptr_t haystack::EmptyVal::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.15.2.2 const std::string haystack::EmptyVal::to_string ( ) const [virtual]

String format is for human consumption only

Reimplemented from haystack::Val.

3.15.2.3 const std::string haystack::EmptyVal::to_zinc ( ) const [virtual]

Encode value to zinc format

Implements haystack::Val.

3.15.2.4 const Type haystack::EmptyVal::type ( ) const [virtual]

Return this Val type

Implements haystack::Val.

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

- include/val.hpp

3.16 haystack::Eq Class Reference

Inheritance diagram for haystack::Eq:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.17 haystack::Filter Class Reference

`#include <filter.hpp>`

Inheritance diagram for haystack::Filter:

Public Types

- enum **Type** { **NORMAL_FILTER_TYPE** = 'F', **COMPOUND_FILTER_TYPE** = 'C' }
- typedef boost::shared_ptr< Filter > **shared_ptr_t**

Public Member Functions

- shared_ptr_t AND (shared_ptr_t second)
- shared_ptr_t OR (shared_ptr_t second)
- virtual bool include (const Dict &dict, const Pather &pather) const =0
- virtual std::string **str** () const
- virtual Type **type** () const
- virtual bool **operator==** (const Filter &other)

Static Public Member Functions

- static shared_ptr_t **make** (const std::string &, bool checked=true)
- static shared_ptr_t has (const std::string &path)
- static shared_ptr_t missing (const std::string &path)
- static shared_ptr_t eq (const std::string &path, Val::auto_ptr_t val)
- static shared_ptr_t ne (const std::string &path, Val::auto_ptr_t val)
- static shared_ptr_t lt (const std::string &path, Val::auto_ptr_t val)
- static shared_ptr_t le (const std::string &path, Val::auto_ptr_t val)
- static shared_ptr_t gt (const std::string &path, Val::auto_ptr_t val)
- static shared_ptr_t ge (const std::string &path, Val::auto_ptr_t val)

3.17.1 Detailed Description

Filter models a parsed tag query string.

See also

Project Haystack

3.17.2 Member Function Documentation

3.17.2.1 shared_ptr_t haystack::Filter::AND (shared_ptr_t *second*)

Return a query which is the logical-and of this and that query.

3.17.2.2 static shared_ptr_t haystack::Filter::eq (const std::string & *path,* Val::auto_ptr_t *val*) [static]

Match records which have a tag are equal to the specified value. If the path is not defined then it is unmatched.

3.17.2.3 static shared_ptr_t haystack::Filter::ge (const std::string & *path,* Val::auto_ptr_t *val*) [static]

Match records which have tags greater than or equal to specified value. If the path is not defined then it is unmatched.

3.17.2.4 static shared_ptr_t haystack::Filter::gt (const std::string & *path,* Val::auto_ptr_t *val*) [static]

Match records which have tags greater than specified value. If the path is not defined then it is unmatched.

3.17.2.5 static shared_ptr_t haystack::Filter::has (const std::string & *path*) [static]

Match records which have the specified tag path defined.

3.17.2.6 virtual bool haystack::Filter::include (const Dict & *dict,* const Pather & *pather*) const [pure virtual]

Return if given tags entity matches this query.

Implemented in haystack::PathFilter.

3.17.2.7 static shared_ptr_t haystack::Filter::le (const std::string & *path,* Val::auto_ptr_t *val*) [static]

Match records which have tags less than or equals to specified value. If the path is not defined then it is unmatched.

3.17.2.8 static shared_ptr_t haystack::Filter::lt (const std::string & *path*, Val::auto_ptr_t *val*) [static]

Match records which have tags less than the specified value. If the path is not defined then it is unmatched.

3.17.2.9 static shared_ptr_t haystack::Filter::missing (const std::string & *path*) [static]

Match records which do not define the specified tag path.

3.17.2.10 static shared_ptr_t haystack::Filter::ne (const std::string & *path*, Val::auto_ptr_t *val*) [static]

Match records which have a tag not equal to the specified value. If the path is not defined then it is unmatched.

3.17.2.11 shared_ptr_t haystack::Filter::OR (shared_ptr_t *second*)

Return a query which is the logical-or of this and that query.

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

- include/filter.hpp

3.18 haystack::Ge Class Reference

Inheritance diagram for haystack::Ge:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.19 haystack::Grid Class Reference

`#include <grid.hpp>`

Inheritance diagram for haystack::Grid:

Public Types

- typedef boost::ptr_vector< Row > **row_vec_t**
- typedef boost::ptr_vector< Col > **col_vec_t**
- typedef std::map< std::string, size_t > **name_col_map_t**

- typedef row_vec_t::const_iterator **iterator**
- typedef row_vec_t::const_iterator **const_iterator**
- typedef std::auto_ptr< Grid > **auto_ptr_t**

Public Member Functions

- Dict & meta ()
- const Dict & **meta** () const
- const bool is_err () const
- const bool is_empty () const
- const size_t num_rows () const
- const Row & row (size_t row) const
- const size_t num_cols () const
- const Col & col (size_t index) const
- const Col *const col (const std::string &name) const
- const Col *const col (const std::string &name, bool checked) const
- const_iterator begin () const
- const_iterator **end** () const
- Dict & add_col (const std::string &name)
- Grid & add_row (Val *[], size_t count)
- void reserve_rows (size_t count)

Static Public Member Functions

- static Grid::auto_ptr_t make_err (const std::runtime_error &)
- static Grid::auto_ptr_t make (const Dict &)
- static Grid::auto_ptr_t make (const std::vector< const Dict * > &)
- static Grid::auto_ptr_t make (const boost::ptr_vector< Dict > &)

Static Public Attributes

- static const Grid & **EMPTY**

Friends

- class **const_grid_iterator**

3.19.1 Detailed Description

Grid a two dimension data structure of cols and rows.

See also

Project Haystack

3.19.2 Member Function Documentation

3.19.2.1 Dict& haystack::Grid::add_col (const std::string & *name*)

Add new column and return builder for column metadata. Columns cannot be added after adding the first row.

3.19.2.2 Grid& haystack::Grid::add_row (Val * [], size_t count)

Add new row with array of cells which correspond to column order. Return this.

3.19.2.3 const_iterator haystack::Grid::begin ( ) const

Return rows iterator

3.19.2.4 const Col& haystack::Grid::col (size_t index) const

Get a column by its index

3.19.2.5 const Col* const haystack::Grid::col (const std::string & name) const

Convenience for "col(name, true)"

3.19.2.6 const Col* const haystack::Grid::col (const std::string & name, bool checked) const

Get a column by name. If not found and checked if false then return null, otherwise throw exception

3.19.2.7 const bool haystack::Grid::is_empty ( ) const

Return if number of rows is zero

3.19.2.8 const bool haystack::Grid::is_err ( ) const

Error grid have the meta.err marker tag

3.19.2.9 static Grid::auto_ptr_t haystack::Grid::make (const Dict &) [static]

Constructs grid from Dict

3.19.2.10 static Grid::auto_ptr_t haystack::Grid::make (const std::vector< const Dict * > &) [static]

Constructs grid from Dicts* vector

3.19.2.11 static Grid::auto_ptr_t haystack::Grid::make (const boost::ptr_vector< Dict > &) [static]

Constructs grid from Dicts ptr_vector

3.19.2.12 static Grid::auto_ptr_t haystack::Grid::make_err (const std::runtime_error &) [static]

Constructs an err grid

3.19.2.13 Dict& haystack::Grid::meta ( )

Return grid level meta

3.19.2.14 const size_t haystack::Grid::num_cols ( ) const

Get number of columns

3.19.2.15 const size_t haystack::Grid::num_rows ( ) const

Return number of rows

3.19.2.16 void haystack::Grid::reserve_rows (size_t *count*)

Tell grid to allocate space for this abount of rows entries.

3.19.2.17 const Row& haystack::Grid::row (size_t *row*) const

Get a row by its zero based index

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

- include/grid.hpp

3.20 haystack::GridReader Class Reference

`#include <gridreader.hpp>`

Inheritance diagram for haystack::GridReader:

Public Member Functions

- virtual std::auto_ptr< Grid > read_grid ()=0

3.20.1 Detailed Description

GridReader is base class for reading grids from an input stream.

See also

Project Haystack

3.20.2 Member Function Documentation

3.20.2.1 virtual std::auto_ptr<Grid> haystack::GridReader::read_grid ( ) [pure virtual]

Read a grid

Implemented in haystack::ZincReader.

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

- include/gridreader.hpp

3.21 haystack::GridWriter Class Reference

`#include <gridwriter.hpp>`

Inheritance diagram for haystack::GridWriter:

Public Member Functions

- virtual void write_grid (const Grid &grid)=0

3.21.1 Detailed Description

GridWriter is base class for writing grids to an output stream.

See also

Project Haystack

3.21.2 Member Function Documentation

3.21.2.1 virtual void haystack::GridWriter::write_grid (const Grid & *grid*) [pure virtual]

Write a grid

Implemented in haystack::ZincWriter.

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

- include/gridwriter.hpp

3.22 haystack::Gt Class Reference

Inheritance diagram for haystack::Gt:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.23 haystack::Has Class Reference

Inheritance diagram for haystack::Has:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.24 haystack::HisItem Class Reference

`#include <hisitem.hpp>`

Public Member Functions

- **HisItem** (const HisItem &)
- **HisItem** (const DateTime &ts, const Val &val)
- **HisItem** (boost::shared_ptr< const DateTime > ts, boost::shared_ptr< const Val > val)

Static Public Member Functions

- static const std::vector< HisItem > grid_to_items (const Grid &grid)
- static Grid::auto_ptr_t his_items_to_grid (const Dict &meta, const std::vector< HisItem > &items)

Public Attributes

- boost::shared_ptr< const DateTime > **ts**
- boost::shared_ptr< const Val > **val**

3.24.1 Detailed Description

HisItem is used to model a timestamp/value pair.

See also

Project Haystack

3.24.2 Member Function Documentation

3.24.2.1 static const std::vector<HisItem> haystack::HisItem::grid_to_items (const Grid & *grid*) [static]

Map Grid to HisItems. Grid must have ts and val columns.

3.24.2.2 static Grid::auto_ptr_t haystack::HisItem::his_items_to_grid (const Dict & *meta*, const std::vector< HisItem > & *items*) [static]

Convenience to build grid from array of HisItem

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

- include/hisitem.hpp

3.25 haystack::Le Class Reference

Inheritance diagram for haystack::Le:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.26 haystack::Lt Class Reference

Inheritance diagram for haystack::Lt:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.27 haystack::Marker Class Reference

`#include <marker.hpp>`

Inheritance diagram for haystack::Marker:

Public Member Functions

- const Type type () const
- const std::string to_string () const
- const std::string to_zinc () const
- bool operator== (const Marker &b) const
- bool **operator==** (const Val &other) const
- bool **operator**> (const Val &other) const
- bool **operator**< (const Val &other) const
- auto_ptr_t clone () const

Static Public Attributes

- static const Marker & VAL

Additional Inherited Members

3.27.1 Detailed Description

Marker is the singleton value for a marker tag.

See also

Project Haystack

3.27.2 Member Function Documentation

3.27.2.1 auto_ptr_t haystack::Marker::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.27.2.2 bool haystack::Marker::operator== (const Marker & *b*) const

Equality

3.27.2.3 const std::string haystack::Marker::to_string ( ) const [virtual]

Encode as "marker"

Reimplemented from haystack::Val.

3.27.2.4 const std::string haystack::Marker::to_zinc ( ) const [virtual]

Encode as "M"

Implements haystack::Val.

3.27.2.5 const Type haystack::Marker::type ( ) const [inline], [virtual]

Return this Val type

Implements haystack::Val.

3.27.3 Member Data Documentation

3.27.3.1 const Marker& haystack::Marker::VAL [static]

default marker value

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

- include/marker.hpp

3.28 haystack::Missing Class Reference

Inheritance diagram for haystack::Missing:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.29 haystack::Ne Class Reference

Inheritance diagram for haystack::Ne:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.30 haystack::Num Class Reference

`#include <num.hpp>`

Inheritance diagram for haystack::Num:

Public Member Functions

- const Type type () const
- **Num** (double val, const std::string &unit)
- **Num** (double val)
- **Num** (int val, const std::string &unit)
- **Num** (int val)
- **Num** (long long val, const std::string &unit)
- **Num** (long long val)
- const std::string to_zinc () const

- bool operator== (const Num &other) const
- bool **operator!=** (const Num &other) const
- bool **operator**> (const Val &other) const
- bool **operator**< (const Val &other) const
- bool **operator==** (double other) const
- bool **operator==** (int other) const
- bool **operator==** (long long other) const
- bool **operator==** (const Val &other) const
- auto_ptr_t clone () const

Static Public Member Functions

- static bool isUnitName (const std::string &)

Public Attributes

- const double value
- const std::string unit

Static Public Attributes

- static const Num ZERO
- static const Num **POS_INF**
- static const Num **NEG_INF**
- static const Num **NaN**

Additional Inherited Members

3.30.1 Detailed Description

Num wraps a 64-bit floating point number and optional unit name.

See also

Project Haystack

3.30.2 Member Function Documentation

3.30.2.1 auto_ptr_t haystack::Num::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.30.2.2 static bool haystack::Num::isUnitName (const std::string &) [static]

check if str is a valid unit name

3.30.2.3 bool haystack::Num::operator== (const Num & *other*) const

Equality

3.30.2.4 const std::string haystack::Num::to_zinc ( ) const [virtual]

Encode value to zinc format

Implements haystack::Val.

3.30.2.5 const Type haystack::Num::type ( ) const [inline], [virtual]

Return this Val type

Implements haystack::Val.

3.30.3 Member Data Documentation

3.30.3.1 const std::string haystack::Num::unit

This unit string

3.30.3.2 const double haystack::Num::value

This double value

3.30.3.3 const Num haystack::Num::ZERO [static]

special values

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

- include/num.hpp

3.31 haystack::Op Class Reference

`#include <op.hpp>`

Inheritance diagram for haystack::Op:

Public Member Functions

- virtual const std::string name () const =0
- virtual const std::string summary () const =0
- void on_service (const Server &db, HTTPServerRequest &req, HTTPServerResponse &res)
- virtual Grid::auto_ptr_t on_service (Server &db, const Grid &req)

Protected Types

- typedef boost::ptr_vector< Ref > **refs_t**

Protected Member Functions

- refs_t **grid_to_ids** (const Server &db, const Grid &grid) const
- Val::auto_ptr_t **val_to_id** (const Server &db, const Val &val) const

3.31.1 Detailed Description

Op is the base class for server side operations exposed by the REST API. All methods on Op must be thread safe.

See also

Project Haystack< / a>

3.31.2 Member Function Documentation

3.31.2.1 virtual const std::string haystack::Op::name ( ) const [pure virtual]

Programatic name of the operation.

3.31.2.2 void haystack::Op::on_service (const Server & *db,* HTTPServerRequest & *req,* HTTPServerResponse & *res*)

Service the request and return response. This method routes to "on_service(Server& db, const Grid& req)"

3.31.2.3 virtual Grid::auto_ptr_t haystack::Op::on_service (Server & *db,* const Grid & *req*) [virtual]

Service the request and return response.

3.31.2.4 virtual const std::string haystack::Op::summary ( ) const [pure virtual]

Short one line summary of what the operation does.

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

- http_server/include/op.hpp

3.32 haystack::Or Class Reference

Inheritance diagram for haystack::Or:

Friends

- class **Filter**

Additional Inherited Members

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

- include/filter.hpp

3.33 haystack::Path Class Reference

```
#include <filter.hpp>
```

Inheritance diagram for haystack::Path:

Public Types

- typedef std::auto_ptr< Path > **auto_ptr_t**

Public Member Functions

- virtual size_t size () const =0
- virtual std::string get (size_t i) const =0
- virtual bool operator== (const Path &that) const
- virtual std::string str () const =0

Static Public Member Functions

- static auto_ptr_t make (const std::string &path)

3.33.1 Detailed Description

Path is a simple name or a complex path using the "->" separator

3.33.2 Member Function Documentation

3.33.2.1 virtual std::string haystack::Path::get (size_t *i*) const [pure virtual]

Get name at given index.

3.33.2.2 static auto_ptr_t haystack::Path::make (const std::string & *path*) [static]

Construct a new Path from string or throw exception

3.33.2.3 virtual bool haystack::Path::operator== (const Path & *that*) const [inline], [virtual]

Equality is based on string.

3.33.2.4 virtual size_t haystack::Path::size ( ) const [pure virtual]

Number of names in the path.

3.33.2.5 virtual std::string haystack::Path::str ( ) const [pure virtual]

Get string encoding.

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

- include/filter.hpp

3.34 haystack::Path1 Class Reference

Inheritance diagram for haystack::Path1:

Friends

- class **Path**

Additional Inherited Members

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

- include/filter.hpp

3.35 haystack::Pather Class Reference

`#include <filter.hpp>`

Public Member Functions

- virtual const Dict & find (const std::string &ref) const =0

3.35.1 Detailed Description

Pather is a callback interface used to resolve query paths.

3.35.2 Member Function Documentation

3.35.2.1 virtual const Dict& haystack::Pather::find (const std::string & *ref*) const [pure virtual]

Given a HRef string identifier, resolve to an entity's Dict respresentation or ref is not found return null.

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

- include/filter.hpp

3.36 haystack::PathFilter Class Reference

Inheritance diagram for haystack::PathFilter:

Protected Member Functions

- **PathFilter** (Path::auto_ptr_t p)
- bool include (const Dict &dict, const Pather &pather) const
- virtual bool **do_include** (const Val &val) const =0
- std::string **str** () const
- const Path * **path** () const

Protected Attributes

- Path::auto_ptr_t **m_path**

Additional Inherited Members

3.36.1 Member Function Documentation

3.36.1.1 bool haystack::PathFilter::include (const Dict & *dict,* const Pather & *pather*) const [protected], [virtual]

Return if given tags entity matches this query.

Implements haystack::Filter.

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

- include/filter.hpp

3.37 haystack::PathN Class Reference

Inheritance diagram for haystack::PathN:

Friends

- class **Path**

Additional Inherited Members

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

- include/filter.hpp

3.38 haystack::Proj Class Reference

`#include <proj.hpp>`

Inheritance diagram for haystack::Proj:

Public Member Functions

- virtual Dict::auto_ptr_t about () const =0
- virtual Dict::auto_ptr_t read_by_id (const Ref &id) const
- virtual Dict::auto_ptr_t read_by_id (const Ref &id, bool checked) const
- virtual Grid::auto_ptr_t read_by_ids (const boost::ptr_vector< Ref > &ids) const
- virtual Grid::auto_ptr_t read_by_ids (const boost::ptr_vector< Ref > &ids, bool checked) const
- virtual Dict::auto_ptr_t on_read_by_id (const Ref &id) const =0
- virtual Grid::auto_ptr_t on_read_by_ids (const boost::ptr_vector< Ref > &ids) const =0
- virtual Dict::auto_ptr_t read (const std::string &filter) const
- virtual Dict::auto_ptr_t read (const std::string &filter, bool checked) const
- virtual Grid::auto_ptr_t read_all (const std::string &filter) const
- virtual Grid::auto_ptr_t read_all (const std::string &filter, size_t limit) const
- virtual Grid::auto_ptr_t on_read_all (const std::string &filter, size_t limit) const =0

3.38.1 Detailed Description

Proj is the common interface for Client and Server to provide access to a database tagged entity records.

See also

Project Haystack

3.38.2 Member Function Documentation

3.38.2.1 virtual Dict::auto_ptr_t haystack::Proj::about ( ) const [pure virtual]

Get the summary "about" information.

Implemented in haystack::Server, and haystack::Client.

3.38.2.2 virtual Grid::auto_ptr_t haystack::Proj::on_read_all (const std::string & *filter,* size_t *limit*) const [pure virtual]

Subclass hook for read and read_all.

Implemented in haystack::Server.

3.38.2.3 virtual Dict::auto_ptr_t haystack::Proj::on_read_by_id (const Ref & *id*) const [pure virtual]

Subclass hook for read_by_id, return null if not found.

Implemented in haystack::TestProj.

3.38.2.4 virtual Grid::auto_ptr_t haystack::Proj::on_read_by_ids (const boost::ptr_vector< Ref > & *ids*) const [pure virtual]

Subclass hook for read_by_ids, return rows with nulls cells for each id not found.

Implemented in haystack::Server.

3.38.2.5 virtual Dict::auto_ptr_t haystack::Proj::read (const std::string & *filter*) const [virtual]

Convenience for "read(filter, true)".

3.38.2.6 virtual Dict::auto_ptr_t haystack::Proj::read (const std::string & *filter,* bool *checked*) const [virtual]

Query one entity record that matches the given filter. If there is more than one record, then it is undefined which one is returned. If there are no matches than return null or raise runtime_exception based on checked flag.

3.38.2.7 virtual Grid::auto_ptr_t haystack::Proj::read_all (const std::string & *filter*) const [virtual]

Convenience for "read_all(filter, max)".

3.38.2.8 virtual Grid::auto_ptr_t haystack::Proj::read_all (const std::string & *filter,* size_t *limit*) const [virtual]

Call "read" to query every entity record that matches given filter. Clip number of results by "limit" parameter.

3.38.2.9 virtual Dict::auto_ptr_t haystack::Proj::read_by_id (const Ref & *id*) const [virtual]

Convenience for "read_by_id(id, true)"

3.38.2.10 virtual Dict::auto_ptr_t haystack::Proj::read_by_id (const Ref & *id,* bool *checked*) const [virtual]

Call "read" to lookup an entity record by it's unique identifier. If not found then return null or throw an Unknown↩ RecException based on checked.

3.38.2.11 virtual Grid::auto_ptr_t haystack::Proj::read_by_ids (const boost::ptr_vector< Ref > & *ids*) const [virtual]

Convenience for "read_by_ids(ids, true)"

3.38.2.12 virtual Grid::auto_ptr_t haystack::Proj::read_by_ids (const boost::ptr_vector< Ref > & *ids,* bool *checked*) const [virtual]

Read a list of entity records by their unique identifier. Return a grid where each row of the grid maps to the respective id array (indexes line up). If checked is true and any one of the ids cannot be resolved then raise exception for first id not resolved. If checked is false, then each id not found has a row where every cell is null.

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

- http_server/include/proj.hpp

3.39 haystack::Ref Class Reference

`#include <ref.hpp>`

Inheritance diagram for haystack::Ref:

Public Member Functions

- const Type type () const
- **Ref** (const std::string &val)
- **Ref** (const std::string &val, const std::string &unit)
- const std::string to_code () const
- const std::string to_string () const
- const std::string to_zinc () const
- const std::string dis () const
- bool operator== (const Ref &other) const
- bool **operator!=** (const Ref &other) const
- bool **operator==** (const Val &other) const
- bool **operator>** (const Val &other) const
- bool **operator<** (const Val &other) const
- auto_ptr_t clone () const

Static Public Member Functions

- static bool **is_id_char** (int c)
- static bool isId (const std::string &)

Public Attributes

- const std::string value

Additional Inherited Members

3.39.1 Detailed Description

Ref wraps a string reference identifier and a optional display name.

See also

Project Haystack

3.39.2 Member Function Documentation

3.39.2.1 auto_ptr_t haystack::Ref::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.39.2.2 const std::string haystack::Ref::dis ( ) const

Return display string which is dis field if nont empty, val field otherwise

3.39.2.3 static bool haystack::Ref::isId (const std::string &) [static]

check if str is a valid id

3.39.2.4 bool haystack::Ref::operator== (const Ref & *other*) const

Equality

3.39.2.5 const std::string haystack::Ref::to_code ( ) const [inline]

Encode as "@id"

3.39.2.6 const std::string haystack::Ref::to_string ( ) const [virtual]

Return the val string

Reimplemented from haystack::Val.

3.39.2.7 const std::string haystack::Ref::to_zinc ( ) const [virtual]

Encode value to zinc format

Implements haystack::Val.

3.39.2.8 const Type haystack::Ref::type ( ) const [inline],[virtual]

Return this Val type

Implements haystack::Val.

3.39.3 Member Data Documentation

3.39.3.1 const std::string haystack::Ref::value

This string value

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

- include/ref.hpp

3.40 haystack::Row Class Reference

`#include <row.hpp>`

Inheritance diagram for haystack::Row:

Public Types

- typedef const_row_iterator **const_iterator**
- typedef boost::ptr_vector
  < boost::nullable< Val > > **val_vec_t**

Public Member Functions

- const Grid & grid () const
- const size_t size () const
- const Val & get (const std::string &name, bool checked=true) const
- const std::string get_string (const std::string &name) const
- const double get_double (const std::string &name) const
- const Val & get (const Col &col) const
- Dict::auto_ptr_t to_dict () const
- const_iterator begin () const
- const_iterator end () const

Friends

- class **Grid**

Additional Inherited Members

3.40.1 Detailed Description

Row is a row in a Grid. It implements the Dict interface also.

See also

Project Haystack

3.40.2 Member Function Documentation

3.40.2.1 const_iterator haystack::Row::begin ( ) const

Get start it

3.40.2.2 const_iterator haystack::Row::end ( ) const

Get end it

3.40.2.3 const Val& haystack::Row::get (const std::string & *name*, bool *checked* = true) const [virtual]

Get a cell by column name.

Reimplemented from haystack::Dict.

3.40.2.4 const Val& haystack::Row::get (const Col & *col*) const

Get a cell by column.

3.40.2.5 const double haystack::Row::get_double (const std::string & *name*) const

Get a int by column name.

3.40.2.6 const std::string haystack::Row::get_string (const std::string & *name*) const

Get a string by column name.

3.40.2.7 const Grid& haystack::Row::grid ( ) const

Get the grid associated with this row

3.40.2.8 const size_t haystack::Row::size ( ) const [virtual]

Number of columns in grid (which may map to null cells)

Reimplemented from haystack::Dict.

3.40.2.9 Dict::auto_ptr_t haystack::Row::to_dict ( ) const

Get new Dict for this Row.

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

- include/row.hpp

3.41 haystack::Server Class Reference

`#include <server.hpp>`

Inheritance diagram for haystack::Server:

Public Types

- typedef const_proj_iterator **iterator**
- typedef const_proj_iterator **const_iterator**

Public Member Functions

- Dict::auto_ptr_t about () const
- virtual const std::vector < const Op * > & ops ()=0
- Grid::auto_ptr_t nav (const std::string &navId) const
- Dict::auto_ptr_t nav_read_by_uri (const Uri &uri, bool checked) const
- const Watch::shared_ptr **watch_open** (const std::string &dis)
- const std::vector < Watch::shared_ptr > watches ()
- virtual Watch::shared_ptr **watch** (const std::string &id, bool checked=true)
- Grid::auto_ptr_t **point_write_array** (const Ref &id)
- void point_write (const Ref &id, int level, const Val &val, const std::string &who, const Num &dur)
- Grid::auto_ptr_t **his_read** (const Ref &id, const std::string &range)
- void **his_write** (const Ref &id, const std::vector< HisItem > &items)
- Grid::auto_ptr_t invoke_action (const Ref &id, const std::string &action, const Dict &args)

Static Public Member Functions

- static const DateTime & **boot_time** ()

Protected Member Functions

- virtual const Dict & **on_about** () const =0
- Grid::auto_ptr_t on_read_by_ids (const boost::ptr_vector< Ref > &ids) const
- Grid::auto_ptr_t on_read_all (const std::string &filter, size_t limit) const
- virtual const_iterator **begin** () const =0
- virtual const_iterator **end** () const =0
- virtual Grid::auto_ptr_t **on_nav** (const std::string &nav_id) const =0
- virtual Dict::auto_ptr_t **on_nav_read_by_uri** (const Uri &uri) const =0
- virtual Watch::shared_ptr on_watch_open (const std::string &dis)=0
- virtual const std::vector < Watch::shared_ptr > on_watches ()=0
- virtual Watch::shared_ptr on_watch (const std::string &id)=0
- virtual Grid::auto_ptr_t on_point_write_array (const Dict &rec)=0
- virtual void on_point_write (const Dict &rec, int level, const Val &val, const std::string &who, const Num &dur)=0
- virtual std::vector< HisItem > **on_his_read** (const Dict &rec, const DateTimeRange &range)=0
- virtual void on_his_write (const Dict &rec, const std::vector< HisItem > &items)=0
- virtual Grid::auto_ptr_t on_invoke_action (const Dict &rec, const std::string &action, const Dict &args)=0

3.41.1 Detailed Description

Server is the interface between HTTPRequestHandler and a database of tag based entities. All methods on HServer must be thread safe.

See also

Project Haystack

3.41.2 Member Function Documentation

3.41.2.1 Dict::auto_ptr_t haystack::Server::about ( ) const [virtual]

Get the summary "about" information.

Implements haystack::Proj.

3.41.2.2 Grid::auto_ptr_t haystack::Server::invoke_action (const Ref & *id,* const std::string & *action,* const Dict & *args*) [inline]

Invoke an action identified by id and action.

3.41.2.3 Grid::auto_ptr_t haystack::Server::nav (const std::string & *navId*) const

Return navigation children for given navId.

3.41.2.4 Dict::auto_ptr_t haystack::Server::nav_read_by_uri (const Uri & *uri,* bool *checked*) const

Read a record from the database using a navigation path. If not found then return NULL or raise runtime_exception base on checked flag.

3.41.2.5 virtual void haystack::Server::on_his_write (const Dict & *rec,* const std::vector< HisItem > & *items*) [protected], [pure virtual]

Implementation hook for onHisWrite.

Implemented in haystack::TestProj.

3.41.2.6 virtual Grid::auto_ptr_t haystack::Server::on_invoke_action (const Dict & *rec,* const std::string & *action,* const Dict & *args*) [protected], [pure virtual]

Implementation hook for invokeAction

Implemented in haystack::TestProj.

3.41.2.7 virtual void haystack::Server::on_point_write (const Dict & *rec,* int *level,* const Val & *val,* const std::string & *who,* const Num & *dur*) [protected], [pure virtual]

Implementation hook for pointWrite

Implemented in haystack::TestProj.

3.41.2.8 virtual Grid::auto_ptr_t haystack::Server::on_point_write_array (const Dict & *rec*) [protected], [pure virtual]

Implementation hook for pointWriteArray

Implemented in haystack::TestProj.

3.41.2.9 Grid::auto_ptr_t haystack::Server::on_read_all (const std::string & *filter,* size_t *limit*) const [protected], [virtual]

Subclass hook for read and read_all.

Implements haystack::Proj.

3.41.2.10 Grid::auto_ptr_t haystack::Server::on_read_by_ids (const boost::ptr_vector< Ref > & *ids*) const [protected], [virtual]

Subclass hook for read_by_ids, return rows with nulls cells for each id not found.

Implements haystack::Proj.

3.41.2.11 virtual Watch::shared_ptr haystack::Server::on_watch (const std::string & *id*) [protected], [pure virtual]

Implementation hook for watch lookup, return null if not found.

Implemented in haystack::TestProj.

3.41.2.12 virtual Watch::shared_ptr haystack::Server::on_watch_open (const std::string & *dis*) [protected], [pure virtual]

Implementation hook for on_watch_open.

Implemented in haystack::TestProj.

3.41.2.13 virtual const std::vector<Watch::shared_ptr> haystack::Server::on_watches ( ) [protected], [pure virtual]

Implementation hook for watches.

Implemented in haystack::TestProj.

3.41.2.14 virtual const std::vector<const Op*>& haystack::Server::ops ( ) [pure virtual]

Return the operations supported by this database.

Implemented in haystack::TestProj.

3.41.2.15 void haystack::Server::point_write (const Ref & *id,* int *level,* const Val & *val,* const std::string & *who,* const Num & *dur*)

Write to the given priority array level.

3.41.2.16 const std::vector<Watch::shared_ptr> haystack::Server::watches ( )

List the open watches.

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

- http_server/include/server.hpp

3.42 haystack::StdOps Class Reference

Public Types

- typedef std::map< std::string, const Op *const > **ops_map_t**

Static Public Member Functions

- static const ops_map_t & **ops_map** ()

Static Public Attributes

- static const Op & about
- static const Op & ops
- static const Op & formats
- static const Op & read
- static const Op & nav
- static const Op & watch_sub
- static const Op & watch_unsub
- static const Op & watch_poll
- static const Op & watch_list
- static const Op & point_write
- static const Op & his_read
- static const Op & his_write
- static const Op & invoke_action

3.42.1 Member Data Documentation

3.42.1.1 const Op& haystack::StdOps::about [static]

List the registered operations.

3.42.1.2 const Op& haystack::StdOps::formats [static]

List the registered grid formats.

3.42.1.3 const Op& haystack::StdOps::his_read [static]

Read time series history data.

3.42.1.4 const Op& haystack::StdOps::his_write [static]

Write time series history data.

3.42.1.5 const Op& haystack::StdOps::invoke_action [static]

Invoke action.

3.42.1.6 const Op& haystack::StdOps::nav [static]

Navigate tree structure of database.

3.42.1.7 const Op& haystack::StdOps::ops [static]

List the registered operations.

3.42.1.8 const Op& haystack::StdOps::point_write [static]

Read/write writable point priority array.

3.42.1.9 const Op& haystack::StdOps::read [static]

Read entity records in database.

3.42.1.10 const Op& haystack::StdOps::watch_list [static]

List all Watches.

3.42.1.11 const Op& haystack::StdOps::watch_poll [static]

Watch poll cov or refresh.

3.42.1.12 const Op& haystack::StdOps::watch_sub [static]

Watch subscription.

3.42.1.13 const Op& haystack::StdOps::watch_unsub [static]

Watch unsubscription.

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

- http_server/include/op.hpp

3.43 haystack::Str Class Reference

`#include <str.hpp>`

Inheritance diagram for haystack::Str:

Public Member Functions

- const Type type () const
- **Str** (const std::string &val)
- const std::string to_string () const
- const std::string to_zinc () const
- bool operator== (const Str &other) const
- bool **operator!=** (const Str &other) const
- bool **operator**< (const Val &other) const
- bool **operator**> (const Val &other) const
- bool **operator==** (const std::string &other) const
- bool **operator==** (const Val &other) const
- auto_ptr_t clone () const

Public Attributes

- const std::string value

Static Public Attributes

- static const Str & **EMPTY**

Friends

- class **Ref**

Additional Inherited Members

3.43.1 Detailed Description

Str wraps a std::string as a tag value.

See also

Project Haystack

3.43.2 Member Function Documentation

3.43.2.1 auto_ptr_t haystack::Str::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.43.2.2 bool haystack::Str::operator== (const Str & *other*) const

Equality is value based

3.43.2.3 const std::string haystack::Str::to_string ( ) const [virtual]

Return value string.

Reimplemented from haystack::Val.

3.43.2.4 const std::string haystack::Str::to_zinc ( ) const [virtual]

Encode using double quotes and back slash escapes

Implements haystack::Val.

3.43.2.5 const Type haystack::Str::type ( ) const [inline], [virtual]

Return this Val type

Implements haystack::Val.

3.43.3 Member Data Documentation

3.43.3.1 const std::string haystack::Str::value

This string value

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

- include/str.hpp

3.44 haystack::TestProj Class Reference

Inheritance diagram for haystack::TestProj:

Public Types

- typedef boost::ptr_map < std::string, Dict > **recs_t**
- typedef std::map< std::string, Watch::shared_ptr > **watches_t**

Public Member Functions

- const std::vector< const Op * > & ops ()
- const Op *const **op** (const std::string &name, bool checked=true) const
- const Dict & **on_about** () const
- const_iterator **begin** () const
- const_iterator **end** () const

Protected Member Functions

- Dict::auto_ptr_t on_read_by_id (const Ref &id) const
- Grid::auto_ptr_t **on_nav** (const std::string &nav_id) const
- Dict::auto_ptr_t **on_nav_read_by_uri** (const Uri &uri) const
- Watch::shared_ptr on_watch_open (const std::string &dis)
- const std::vector < Watch::shared_ptr > on_watches ()
- Watch::shared_ptr on_watch (const std::string &id)
- Grid::auto_ptr_t on_point_write_array (const Dict &rec)

- void on_point_write (const Dict &rec, int level, const Val &val, const std::string &who, const Num &dur)
- std::vector< HisItem > **on_his_read** (const Dict &entity, const DateTimeRange &range)
- void on_his_write (const Dict &rec, const std::vector< HisItem > &items)
- Grid::auto_ptr_t on_invoke_action (const Dict &rec, const std::string &action, const Dict &args)

Friends

- class **TestWatch**

Additional Inherited Members

3.44.1 Member Function Documentation

3.44.1.1 void haystack::TestProj::on_his_write (const Dict & *rec,* const std::vector< HisItem > & *items*) [protected], [virtual]

Implementation hook for onHisWrite.

Implements haystack::Server.

3.44.1.2 Grid::auto_ptr_t haystack::TestProj::on_invoke_action (const Dict & *rec,* const std::string & *action,* const Dict & *args*) [protected], [virtual]

Implementation hook for invokeAction

Implements haystack::Server.

3.44.1.3 void haystack::TestProj::on_point_write (const Dict & *rec,* int *level,* const Val & *val,* const std::string & *who,* const Num & *dur*) [protected], [virtual]

Implementation hook for pointWrite

Implements haystack::Server.

3.44.1.4 Grid::auto_ptr_t haystack::TestProj::on_point_write_array (const Dict & *rec*) [protected], [virtual]

Implementation hook for pointWriteArray

Implements haystack::Server.

3.44.1.5 Dict::auto_ptr_t haystack::TestProj::on_read_by_id (const Ref & *id*) const [protected], [virtual]

Subclass hook for read_by_id, return null if not found.

Implements haystack::Proj.

3.44.1.6 Watch::shared_ptr haystack::TestProj::on_watch (const std::string & *id*) [protected], [virtual]

Implementation hook for watch lookup, return null if not found.

Implements haystack::Server.

3.44.1.7 Watch::shared_ptr haystack::TestProj::on_watch_open (const std::string & *dis*) [protected], [virtual]

Implementation hook for on_watch_open.

Implements haystack::Server.

3.44.1.8 const std::vector<Watch::shared_ptr> haystack::TestProj::on_watches ( ) [protected], [virtual]

Implementation hook for watches.

Implements haystack::Server.

3.44.1.9 const std::vector<const Op*>& haystack::TestProj::ops ( ) [virtual]

Return the operations supported by this database.

Implements haystack::Server.

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

- http_server/include/testproj.hpp

3.45 haystack::TestWatch Class Reference

Inheritance diagram for haystack::TestWatch:

Public Member Functions

- **TestWatch** (const TestProj &server, const std::string &dis)
- const std::string id () const
- const std::string dis () const
- const int lease () const
- void **lease** (int)
- Grid::auto_ptr_t sub (const refs_t &ids, bool checked=true)
- void unsub (const refs_t &ids)
- Grid::auto_ptr_t poll_changes ()
- Grid::auto_ptr_t poll_refresh ()
- void close ()
- bool is_open () const

Additional Inherited Members

3.45.1 Member Function Documentation

3.45.1.1 void haystack::TestWatch::close ( ) [virtual]

Close the watch and free up any state resources.

Implements haystack::Watch.

3.45.1.2 const std::string haystack::TestWatch::dis ( ) const [virtual]

Debug display string used during "Proj.watch_open"

Implements haystack::Watch.

3.45.1.3 const std::string haystack::TestWatch::id ( ) const [virtual]

Unique watch identifier within a project database. The id may not be assigned until after the first call to "sub", in which case return "".

Implements haystack::Watch.

3.45.1.4 bool haystack::TestWatch::is_open ( ) const [virtual]

Return whether this watch is currently open.

Implements haystack::Watch.

3.45.1.5 const int haystack::TestWatch::lease ( ) const [virtual]

Lease period or int::min if watch has not been opened yet.

Implements haystack::Watch.

3.45.1.6 Grid::auto_ptr_t haystack::TestWatch::poll_changes ( ) [virtual]

Poll for any changes to the subscribed records.

Implements haystack::Watch.

3.45.1.7 Grid::auto_ptr_t haystack::TestWatch::poll_refresh ( ) [virtual]

Poll all the subscribed records even if there have been no changes.

Implements haystack::Watch.

3.45.1.8 Grid::auto_ptr_t haystack::TestWatch::sub (const refs_t & *ids*, bool *checked* = true) [virtual]

Add a list of records to the subscription list and return their current representation. If checked is true and any one of the ids cannot be resolved then raise runtime_exception for first id not resolved. If checked is false, then each id not found has a row where every cell is null. The Grid that is returned must contain metadata entries for 'watchId' and 'lease'.

Implements haystack::Watch.

3.45.1.9 void haystack::TestWatch::unsub (const refs_t & *ids*) [virtual]

Remove a list of records from watch. Silently ignore any invalid ids.

Implements haystack::Watch.

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

- http_server/include/testproj.hpp

3.46 haystack::Time Class Reference

`#include <time.hpp>`

Inheritance diagram for haystack::Time:

Public Member Functions

- const Type type () const
- **Time** (int hour, int minutes, int sec, int ms)
- **Time** (int hour, int minutes, int sec)
- **Time** (int hour, int minutes)
- const std::string to_zinc () const
- bool operator== (const Time &b) const
- bool **operator!=** (const Time &b) const
- bool **operator==** (const Val &other) const
- bool operator< (const Val &b) const
- bool **operator**> (const Val &b) const
- auto_ptr_t clone () const

Public Attributes

- const int hour
- const int minutes
- const int sec
- const int ms

Static Public Attributes

- static const Time & MIDNIGHT

Friends

- class **DateTime**

Additional Inherited Members

3.46.1 Detailed Description

Time models a time of day tag value.

See also

Project Haystack

3.46.2 Member Function Documentation

3.46.2.1 auto_ptr_t haystack::Time::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.46.2.2 bool haystack::Time::operator< (const Val & *b*) const [virtual]

Comparator

Implements haystack::Val.

3.46.2.3 bool haystack::Time::operator== (const Time & *b*) const

Equality

3.46.2.4 const std::string haystack::Time::to_zinc ( ) const [virtual]

Encode as "hh:mm:ss.FFF"

Implements haystack::Val.

3.46.2.5 const Type haystack::Time::type ( ) const [inline], [virtual]

Return this Val type

Implements haystack::Val.

3.46.3 Member Data Documentation

3.46.3.1 const int haystack::Time::hour

Hour of day as 0-23

3.46.3.2 const Time& haystack::Time::MIDNIGHT [static]

constant for midnight

3.46.3.3 const int haystack::Time::minutes

Minute of hour as 0-59

3.46.3.4 const int haystack::Time::ms

Fractional seconds in milliseconds 0-999

3.46.3.5 const int haystack::Time::sec

Second of minute as 0-59

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

- include/time.hpp

3.47 haystack::TimeZone Class Reference

`#include <timezone.hpp>`

Public Member Functions

- **TimeZone** (const std::string &name)
- **TimeZone** (const std::string &name, const int offset)
- **TimeZone** (const TimeZone &other)
- bool operator== (const TimeZone &other) const
- bool **operator!=** (const TimeZone &other) const

Public Attributes

- const std::string name
- const int offset

Static Public Attributes

- static const TimeZone UTC
- static const TimeZone DEFAULT

3.47.1 Detailed Description

TimeZone.

See also

Project Haystack

3.47.2 Member Function Documentation

3.47.2.1 bool haystack::TimeZone::operator== (const TimeZone & *other*) const

Equality

3.47.3 Member Data Documentation

3.47.3.1 const TimeZone haystack::TimeZone::DEFAULT [static]

Default timezone for the machine

3.47.3.2 const std::string haystack::TimeZone::name

This string name

3.47.3.3 const int haystack::TimeZone::offset

This int offset

3.47.3.4 const TimeZone haystack::TimeZone::UTC [static]

UTC timezone

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

- include/timezone.hpp

3.48 haystack::Uri Class Reference

`#include <uri.hpp>`

Inheritance diagram for haystack::Uri:

Public Member Functions

- const Type type () const
- **Uri** (const std::string &val)
- const std::string to_string () const
- const std::string to_zinc () const
- bool operator== (const Uri &other) const
- bool **operator!=** (const Uri &other) const
- bool **operator**< (const Val &other) const
- bool **operator**> (const Val &other) const
- bool **operator==** (const std::string &other) const
- bool **operator==** (const Val &other) const
- auto_ptr_t clone () const

Public Attributes

- const std::string value

Static Public Attributes

- static const Uri **EMPTY**

Additional Inherited Members

3.48.1 Detailed Description

Uri models a URI as a string.

See also

Project Haystack

3.48.2 Member Function Documentation

3.48.2.1 auto_ptr_t haystack::Uri::clone ( ) const [virtual]

creates an auto_ptr pointer to the cloned Val

Implements haystack::Val.

3.48.2.2 bool haystack::Uri::operator== (const Uri & *other*) const

Equality is value based

3.48.2.3 const std::string haystack::Uri::to_string ( ) const [virtual]

Return value string.

Reimplemented from haystack::Val.

3.48.2.4 const std::string haystack::Uri::to_zinc ( ) const [virtual]

Encode using double quotes and back slash escapes

Implements haystack::Val.

3.48.2.5 const Type haystack::Uri::type ( ) const [inline], [virtual]

Return this Val type

Implements haystack::Val.

3.48.3 Member Data Documentation

3.48.3.1 const std::string haystack::Uri::value

This value

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

- include/uri.hpp

3.49 haystack::Val Class Reference

`#include <val.hpp>`

Inheritance diagram for haystack::Val:

Public Types

- enum **Type** {
  BOOL_TYPE = 'B', **BIN_TYPE** = 'b', **COORD_TYPE** = 'C', **DATE_TIME_TYPE** = 'd',
  DATE_TYPE = 'D', **MARKER_TYPE** = 'M', **NUM_TYPE** = 'N', **REF_TYPE** = 'R',
  STR_TYPE = 'S', **TIME_TYPE** = 'T', **URI_TYPE** = 'U', **EMPTY_TYPE** = '|' }
- typedef std::auto_ptr< Val > **auto_ptr_t**

Public Member Functions

- virtual const std::string to_string () const
- virtual const std::string to_zinc () const =0
- virtual const Type type () const =0
- virtual bool **operator==** (const Val &other) const =0
- virtual bool **operator>** (const Val &other) const =0
- virtual bool **operator<** (const Val &other) const =0
- template<class Type >
  const Type & as () const
- virtual auto_ptr_t clone () const =0
- const bool is_empty () const

3.49.1 Detailed Description

Val is the base class for representing haystack tag scalar values as an immutable class.

See also

Project Haystack

3.49.2 Member Function Documentation

3.49.2.1 template<class Type > const Type& haystack::Val::as ( ) const [inline]

Converts to a concrete type

3.49.2.2 virtual auto_ptr_t haystack::Val::clone ( ) const [pure virtual]

creates an auto_ptr pointer to the cloned Val

Implemented in haystack::Date, haystack::DateTime, haystack::EmptyVal, haystack::Coord, haystack::Time, haystack::Num, haystack::Ref, haystack::Bool, haystack::Bin, haystack::Str, haystack::Uri, and haystack::Marker.

3.49.2.3 const bool haystack::Val::is_empty ( ) const [inline]

Check if this Val's Type is EMPTY_TYPE

3.49.2.4 virtual const std::string haystack::Val::to_string ( ) const [inline], [virtual]

String format is for human consumption only

Reimplemented in haystack::EmptyVal, haystack::Bool, haystack::Ref, haystack::Bin, haystack::Str, haystack::Uri, and haystack::Marker.

3.49.2.5 virtual const std::string haystack::Val::to_zinc ( ) const [pure virtual]

Encode value to zinc format

Implemented in haystack::EmptyVal, haystack::DateTime, haystack::Coord, haystack::Time, haystack::Num, haystack::Date, haystack::Bool, haystack::Ref, haystack::Bin, haystack::Str, haystack::Uri, and haystack::Marker.

3.49.2.6 virtual const Type haystack::Val::type ( ) const [pure virtual]

Return this Val type

Implemented in haystack::EmptyVal, haystack::DateTime, haystack::Date, haystack::Time, haystack::Coord, haystack::Marker, haystack::Bin, haystack::Num, haystack::Str, haystack::Bool, haystack::Ref, and haystack::Uri.

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

- include/val.hpp

3.50 haystack::Watch Class Reference

#include <watch.hpp>

Inheritance diagram for haystack::Watch:

Public Types

- typedef boost::shared_ptr< Watch > **shared_ptr**
- typedef boost::ptr_vector< Ref > **refs_t**

Public Member Functions

- virtual const std::string id () const =0
- virtual const std::string dis () const =0
- virtual const int lease () const =0
- virtual Grid::auto_ptr_t sub (const refs_t &ids, bool checked=true)=0
- virtual void unsub (const refs_t &ids)=0
- virtual Grid::auto_ptr_t poll_changes ()=0
- virtual Grid::auto_ptr_t poll_refresh ()=0
- virtual void close ()=0
- virtual bool is_open () const =0

3.50.1 Detailed Description

Watch models a subscription to a list of entity records. Use Proj::watch_open to create a new watch.

See also

`Project Haystack< / a>`

3.50.2 Member Function Documentation

3.50.2.1 virtual void haystack::Watch::close ( ) [pure virtual]

Close the watch and free up any state resources.

Implemented in haystack::TestWatch.

3.50.2.2 virtual const std::string haystack::Watch::dis ( ) const [pure virtual]

Debug display string used during "Proj.watch_open"

Implemented in haystack::TestWatch.

3.50.2.3 virtual const std::string haystack::Watch::id ( ) const [pure virtual]

Unique watch identifier within a project database. The id may not be assigned until after the first call to "sub", in which case return "".

Implemented in haystack::TestWatch.

3.50.2.4 virtual bool haystack::Watch::is_open ( ) const [pure virtual]

Return whether this watch is currently open.

Implemented in haystack::TestWatch.

3.50.2.5 virtual const int haystack::Watch::lease ( ) const [pure virtual]

Lease period or int::min if watch has not been opened yet.

Implemented in haystack::TestWatch.

3.50.2.6 virtual Grid::auto_ptr_t haystack::Watch::poll_changes ( ) [pure virtual]

Poll for any changes to the subscribed records.

Implemented in haystack::TestWatch.

3.50.2.7 virtual Grid::auto_ptr_t haystack::Watch::poll_refresh ( ) [pure virtual]

Poll all the subscribed records even if there have been no changes.

Implemented in haystack::TestWatch.

3.50.2.8 virtual Grid::auto_ptr_t haystack::Watch::sub (const refs_t & *ids*, bool *checked* = true) [pure virtual]

Add a list of records to the subscription list and return their current representation. If checked is true and any one of the ids cannot be resolved then raise runtime_exception for first id not resolved. If checked is false, then each id not found has a row where every cell is null. The Grid that is returned must contain metadata entries for 'watchId' and 'lease'.

Implemented in haystack::TestWatch.

3.50.2.9 virtual void haystack::Watch::unsub (const refs_t & *ids*) [pure virtual]

Remove a list of records from watch. Silently ignore any invalid ids.

Implemented in haystack::TestWatch.

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

- http_server/include/watch.hpp

3.51 haystack::ZincReader Class Reference

`#include <zincreader.hpp>`

Inheritance diagram for haystack::ZincReader:

Public Member Functions

- **ZincReader** (std::istream &is)
- std::auto_ptr< Grid > read_grid ()
- Filter::shared_ptr_t read_filter ()
- std::auto_ptr< Dict > read_dict ()
- Val::auto_ptr_t read_scalar ()

Static Public Member Functions

- static std::auto_ptr< ZincReader > **make** (const std::string &s)

3.51.1 Detailed Description

ZincReader reads grids using the Zinc format.

See also

Project Haystack

3.51.2 Member Function Documentation

3.51.2.1 std::auto_ptr<Dict> haystack::ZincReader::read_dict ( )

Read set of name/value tags as dictionary

3.51.2.2 Filter::shared_ptr_t haystack::ZincReader::read_filter ( )

Parses a filter

3.51.2.3 std::auto_ptr<Grid> haystack::ZincReader::read_grid ( ) [virtual]

Read a grid

Implements haystack::GridReader.

3.51.2.4 Val::auto_ptr_t haystack::ZincReader::read_scalar ( )

Read scalar value.

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

- include/zincreader.hpp

3.52 haystack::ZincWriter Class Reference

`#include <zincwriter.hpp>`

Inheritance diagram for haystack::ZincWriter:

Public Member Functions

- **ZincWriter** (std::ostream &os)
- void write_grid (const Grid &grid)

Static Public Member Functions

- static const std::string grid_to_string (const Grid &grid)

3.52.1 Detailed Description

ZincWriter is used to write grids in the Zinc format.

See also

Project Haystack

3.52.2 Member Function Documentation

3.52.2.1 static const std::string haystack::ZincWriter::grid_to_string (const Grid & *grid*) [static]

Write a grid to string

3.52.2.2 void haystack::ZincWriter::write_grid (const Grid & *grid*) [virtual]

Write a grid

Implements haystack::GridWriter.

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

- include/zincwriter.hpp