Enhydra Configuration Parameters

Slobodan Vujasinovic

Table of Contents

Chapter 1. Application Configuration Parameters

This tables contain the Enhydra Application Configuration parameters.

Server Section: These options configure server attributes.

Atribute	Description	Required
ClassPath[]	Comma separated CLASSPATH directories and files used by this application. Assumes run from the output directory for debugging. If you run from the jar you must rebuild after every change to the app.	Optional
AppClass	The fully qualified name of the application class.	Required
PresentationPrefix	Prefix used to derive presentation object class names and paths from URLs. Assumes run from the output directory for debugging.	Required
AutoReload	Flag to indicate that application classes and resources should be reloaded automatically if ANY file in the CLASSPATH changes. *IMPORTANT NOTE*: This is a debugging option and may slow down the performance of your application. The CLASSPATH should NOT contain any directories (or underlying directories) that contain constantly changing files, i.e. a log file. For example, do NOT include the application's output directory in the CLASSPATH because it contains the application log files in the underlying log directory.	Optional
XMLC.AutoReload	Enable or disable automatic reloading of XMLC document classes when the clas file has changed. This is a subset of the Server.XMLC.AutoRecompilation and checks for XMLC document classes that have been loaded being out-of-date, however it doesn't check or recompile source files. If Server.XMLC.AutoRecompilatio	Optional

Atribute	**Description**	**Required**
	n is set, this option is ignored.	
XMLC.AutoRecompilation	Enable or disable recompilation of XMLC document classes when the source files are out-of-date. For recompilation to work, XMLC document classes must have been compiled using the -for-recomp flag. The application must be running with its classes (at least XMLC classes) in directories rather than in a jar. The HTML files must be on classpath in the same package as the class that was generated from them.	Optional

SessionManager Section: These options configure session attributes.

Atribute	**Description**	**Required**
Lifetime	Maximum number of minutes a user session can last.	
Class	Session Manager implementation class. If not specified com.lutris.appserver.server.sessionEnhydra.StandardSessionManager implementation class is used.	Optional
MaxIdleTime	Maximum number of minutes a user may be idle before being logged off.	Optional
MaxNoUserIdleTime	Maximum number of minutes a session that is not associated with a logged in user (perhaps login pending) may be idle before the session is deleted. Setting the value to less than or equal to 0 will give an infinite idletime. You should not set this and SessionLifetime both to zero else sessions will never expire. The default if not specified is 30 minutes.	Optional
IdleScanInterval	Seconds between idle user scans in the Session Manager. The default if not specified is every 30 seconds.	Optional
RandomizerIntervals[]	The set of varying intervals to use for adding user-generated entropy to the random number generator. It is a good idea to use several different prime numbers. These intervals are in seconds. Suitable defaults are used if this	Setting this key is only necessary in very security conscious enviroments.

Atribute	**Description**	**Required**
	is not specified.	
MemoryPersistence	Flag to indicate that SessionManager will be held in memory during restart application (default is false).	Optional
SessionEncodeUrlState	Should session ids be encoded in DOM URLs? • Auto - Only encode if browser has cookies disabled. • Never - Never use URL encoding session ids. • Always - Always use URL encoding session ids.	Optional
SessionEncodeFirstUrl	Should session id be encoded in first (default) application URL? • false - don't use URL encoding session id. • true - use URL encoding session id.	This parameter is optional. Default value is set to 'false' except when 'SessionEncodeUrlState' is set to 'Always' when parameter value is forced to true. Also, if 'SessionEncodeUrlState' is set to 'Never' parameter value will be forced to false.
SessionHome.Mode	The session home implementation to use. May be one of {BASIC, PAGE_TO_DISK, PAGE_TO_DB, CUSTOM}.	
SessionHome.MaxSessions	Maximum number of concurrent sessions.	
SessionHome.PageThreshold	The maximum number of sessions that will be held in memory simultaneously.	This setting is required by all paged (DISK, DB) implementations.
SessionHome.PageTimeThreshold	The maximum amount of time that a session will remain in memory in an inactive (passive) state. If this amount of time is exceeded then the session is paged out. The value is in seconds.	This setting is required by all paged (DISK, DB) implementations.
SessionHome.PageWait	The maximum amout of time that a thread will wait for a session to be paged back into memory. In milliseconds.	This setting is required by all paged (DISK, DB) implementations.
SessionHome.SaveOnRestart	Flag to indicate that sessions should be paged and reloaded during application restart (default is false). This parameter has no effect if use MemoryPersistence	This setting is required by all paged (DISK, DB) implementations.

Atribute	**Description**	**Required**
	= true. *IMPORTANT NOTE*: Don't use same directory for pageDir key in PAGE_TO_DISK settings for different applications.	
SessionHome.PageDir	The directory where sessions will be paged.	This setting is required when using PAGE_TO_DISK.
SessionHome.DatabaseName	If the following is not specified then sessions are paged to the default database. *NOTE*: You'll need your database properly configured.	
SessionHome.DBTableName	If the following is not specified then session data is written to the table "PersistenSession". *NOTE*: You'll need your database properly configured.	
SessionUserTable.Class		Required when using custom session home interfaces.
SessionHome.Class		Required when using custom session home interfaces.

PresentationManager Configuration: These options configure presentation attributes.

Atribute	**Description**	**Required**
CacheClasses	Enables or disables caching of presentation object classes in memory.	Option, the default is true.
CacheFiles	Enables or disables caching of files (html, gif, etc) that are servered as part of the application.	Option, the default is false.
RequestExecutionLimit	Integer value (in miliseconds) produces a warning message for any PO object exceeds it. Zero value is allowed too, it producess warning messages for all PO objects, while negative value suppresses this feature.	Optional, the default is -1.
InitSessionUser	Commands to Presentation Manager whether to initialize or not session user (initialization is based on requests remote user) if user isn't allready initialized.	Option, the default is false.

Application Section: These options configure application attributes.

Atribute	**Description**	**Required**
DefaultUrl	If the URL "/" for this application is accessed, the user will be redirected to this URL. This should be a relative URL.	
PresentationTools[]	List of all available presentation tools which can be used for dynamic manipulation of pages in presentation layer of the application. For each tool, which is listed beyond, in process of application initialization, will be created factory object (for example XMLCFactory or JivanFactory) with not null value. If this parameter is omitted, the default value 'xmlc' will be assumed which means that, in presentation layer, only XMLC can be used. In current release available tools are: 'xmlc' and 'jivan'.	Optional, the default is xmlc.
Encoding	Specify character encoding which will be used by server to encode form data. If param is not specified, default encoding will be used.	Optional.

DatabaseManager Configuration: These options configure Database Manager attributes.

Many of parameters can be applied both on global level (DatabaseManager) or locally for each Database separately (database level). Also, some parameters can be applied to table directly (table level).

When they are defined on DatabaseManager level they apply on all databases defined in configuration file. Parameters defined on separate databases override those defined on global level. If any of parameters is defined for the table directly, this values is used for that table.

Only exception of this rule is readOnly parameter of table: if AllReadOnly is set on DatabaseManager or Database level to true the readOnly parameter of the tables can not override this value - in all other cases parameters behave usually.

To set parameters on DatabaseManager level, Database level and table level, use the following syntax in configuration file:

DatabaseManager.Class parameter defines class name of DatabaseManager implementation (full package name). Default value is com.lutris.appserver.server.StandardDatabaseManager.

```
DatabaseManager.defaults.<parameter_name> = parameterValue
DatabaseManager.DB.<database_name>.<parameter_name> = parameterValue
DatabaseManager.DB.<database_name>.<table_name>.<parameter_name> = parameterValue
```

Cache parameters can be set on all levels: DatabaseManager, Database and table level.

To set parameters related to cache, use the following syntax in configuration file:

```
DatabaseManager.defaults.cache.<parameter_name> = parameterValue
DatabaseManager.DB.<database_name>.cache.<parameter_name> = parameterValue
DatabaseManager.DB.<database_name>.<table_name>.cache.<parameter_name> = parameterValue
```

Atribute	**Description**	**Required**
DatabaseManager.Databases[]	The databases that are used by CSAM. Each of these databases has configuration parameters set under Database DatabaseManager.DB.<Name>.	Required
DatabaseManager.DefaultDatabase	The default database used in application.	Optional
DatabaseManager.ConfigurationDir	The path to custom configuration folder used by DODS to generate java source code and SQL scripts. Need to be set only if application is generated by DODS with same settings.	Required only in run-time if is used in DODS Generator/ Ant-Task else Optional.
TransactionCheck	If this parameter is set to true, whenever a DO without a transaction context is created in the memory, a warning is created in the log. The warning message contains information about database name, class name, oid, version and the stack trace. If the parameter is set to false or not defined, nothing happens. Level: dbManager, database. Default value is false.	Optional
DeleteCheckVersion	Before the delete of a DO, if this parameter is set to true, it is checked the DO's version. If the DO had been changed in the meantime, DODS throws an SQLException with massage ("Delete failed: Table TABLE id=OID has version DB_VERSION where object has version DO_VERSION."). Level: dbManager, database. Default value is false.	Optional
AutoWrite	If this parameter is set to true, the row will be written to the database when the update of another row occurs and before every query. If this parameter is set to false, or is not defined, no writing will be done until the transaction’s commit. Level: dbManager, database. Default value is false.	Optional
TransactionCaches	If this parameter is set to true, the transaction cache exists, otherwise does not exist. The transaction cache is cache of DOs (data objects) and is always unlimited	Optional

Atribute	**Description**	**Required**
	in size. Level: dbManager, database. Default value is false.	
AutoSave	If AutoSave is true and a DO belongs to a transaction and DO.setXXX(value) is done, the DO modification is automatically saved into the transaction. Deletes of DO objects is implicitly always writen to transactions , when delete occurs they are immediately written to transactions. New DOs created with createVirgin method are never auto saved, they are saved on explicit demand or after first call of DO.setXXX(value). If AutoSave is false, DO.save() is mandatory before the transaction's commit. Level: dbManager, database. Default value is false.	Optional
DefaultFetchSize	Fetch size gives the JDBC driver a hint to the number of rows that should be fetched from the database when more rows are needed. Level: dbManager, database, table. Default value: -1 (means that default fetch size in not used)	Optional
CacheLockTimeout	Time for transaction to acquire a lock on global cache (Wrapper), before it can hide certain DO's from cache. Hiding is necessary, because during commit there is a point in time some DO in cache are becoming out of sync with database. After hiding lock is released, so other transactions could hide their own DOs. Level: dbManager, database. Default value: 0	Optional
CacheLockRetryCount	This parameter contains number of transaction's retries coused by expired CacheLockTimeout while attempting to lock the Global Cache (Wrapper). Level: dbManager, database. Default value: 0	Optional
QueryTimeOut	Sets the number of seconds for which the Query should be executed If the limit is exceeded, an Exception is thrown. Value 0 means that there is no limit.	Optional

Atribute	Description	Required
	Level: dbManager, database, table. Possible values: integer values. Default value: 0.	
SelectOids	This Parameter is used in mass Updates or Deletes (in xxxUpdate, xxxDelete classes). Value true means that before SQL statement which does mass Updates(Deletes), SQL statement SELECT oid with the same WHERE clause will be executed, to pick a list of oids - which should be objects of mass Updates or Deletes. Level: dbManager, database, table. Possible values: true/false. Default value: false.	Optional
IncrementVersions	This Parameter is used in mass Updates(in xxxUpdate). Value true means that values in version column are increment during mass Updates. Level: dbManager, database, table. Possible values: true/false. Default value: true.	Optional
MainCacheLockTimeout	In unlikely case that transaction's thread dies before unlocking main cache (Wrapper), this timeout expiration would allow next transaction to gain new lock on it. Level: dbManager. Possible values: long (milliseconds). Default value: 100.	Optional
DatabaseLockRetryCount	Parameter is close related with previous (DatabaseLockTimeout). They represents number of database access retries caused database locking. Level: dbManager, database. Default value: 1	Optional
MaxConnectionUsages	Enhydra DODS uses a connection pooling mechanism. This Parameter is maximum number of how many times a connection can be reused by a new thread before it is closed and recreated. Level: Connection . Default value: -1 (never)	Optional
MaxWaitingConnections	This Parameter is maximum number of open (unused) waiting connection in pool (rest are closed on release). Level: Con-	Optional

Atribute	Description	Required
	nection . Default value: 10	
Debug	Turn on/off debugging for transactions or queries. Valid values are "true" or "false". Default value is false.	Optional
lazyLoading	Lazy loading is a mechanism that postpones the loading of DO's data (DataStruct objects) until they are needed. Parameter lazyLoading is true if all tables support lazy-loading, otherwise is false. Level: dbManager, database. Default value is false.	Optional
maxExecuteTime	The maxExecuteTime default value for all application's tables. Parameter maxExecuteTime is time for query execution. Every query that is executed longer than maxExecuteTime is printed (SQL statement and execution time) in applications log file. Level: dbManager, database. For maxExecuteTime attribute default value is 0.	Optional
AllReadOnly	IF parameter allReadOnly is true all tables are read-only and the operations: insert, update or delete on the tables are not possible. While this parameter is set to true, no table's parameter readOnly can be set to false. Level: dbManager, database. Default value is false.	Optional
readOnly	IF parameter readOnly is true, specified table is read-only. If not defined, value from table defaults is used. If parameter AllReadOnly is set to true, no table's parameter readOnly can be set to false. Level: database. Default value is false.	Optional
ChangeAutocommit	Disable DODS to change value of, database connection, 'autocommit' property. Prior to version 3.23, MySQL does not support transactions, and therefore does not support explicit commit (they use autocommit by default after eny SQL command). To use MySQL versions 3.22 and earlier, you have	Optional

Atribute	**Description**	**Required**
	to make change to application configuration file. You will need to set parameter ‘ChangeAutocommit’, of logical database, to ‘false’ . Level: database.	
maxCacheSize	If these parameter is not defined, value from cache default is used.Parameter maxCacheSize contains information about maximal size of the DO cache of specified table. *NOTE*: In the following text are explained maximal cache sizes (for DO cache and query caches). The parameters maxCacheSize, maxSimpleCacheSize and maxComplexCacheSize of application's configuration file define these sizes. • maxCacheSize > 0 - This cache is limited. The maximal number of elements in the cache is maxCacheSize. • maxCacheSize = 0 - This means that there is no cache available. This value excludes cache from use. • maxCacheSize < 0 - This cache is unlimited. Context: Cache. Level: dbManager, database, table.	Optional
maxSimpleCacheSize	Default value for all application's caches. If default value for parameter is not defined, DODS uses its own program default. Parameter maxSimpleCacheSize contains information about maximal size of simple query cache for all caches. Context: Cache. Level: dbManager, database, table.	Optional
maxMultiJoinCacheSize	Default value for all multijoin query caches. If default value for parameter is not defined, DODS uses its own program default. Parameter maxMultiJoinCacheSize	Optional

Atribute	Description	Required
	contains information about maximal size of multijoin query cache. Context: Cache. Level: dbManager, database, table.	
TransactionFactory	If this parameter is set to full class name of class that implements AbstractDBTransactionFactory interface DODS will use this class to create database transaction factory, if parameter is not se DODS will use default AbstractDBTransactionFactory implementation - StandardDBTransactionFactory. Level: DatabaseManager, Database. Default value: none (DODS will use StandardTransactionFactory)	Optional
ConnectionFactory	If this parameter is set to full class name of class that implements AbstractDBConnectionFactory interface DODS will use this class to create database connection factory, if parameter is not set or wrong DODS will use default AbstractDBConnectionFactory implementation - StandardDBConnectionFactory Level: Connection. Default value: none (DODS will use StandardDBConnectionFactory)	Optional
FullCacheCountLimit	If a table has more rows then specified in 'FullCacheCountLimit', then the result shouldn't be recalculated in the memory at first time of excution The Query has to be executed at the first time against the database. The result of this query can be cached and recalculated afterwards. Level: DatabaseManager,Database,Table. Default value is -1.	Optional
InitialDSCacheSize	This parameter represents cache size during first loading data into cache - if initialCondition is enabled. Level: DatabaseManager,Database,Table. Default value is -1 (not used).	Optional
InitialCacheFetchSize	If This parameter represents value for fetch size which is used in query for cache initialization. Default value: 0 (not used). Level: cache parameter on DatabaseMan-	Optional

Atribute	**Description**	**Required**
	ager.cache,Database.cache,Table. cache level.	
InitCachesResultSetType	Possible values: TYPE_FORWARD_ONLY, TYPE_SCROLL_INSENSITIVE, or TYPE_SCROLL_SENSITIVE. Result set type. This type will be used for ResultSets created by PreparedStatemens only for SQL queries which will be executed during cache initialization. Default value is defined in dbVendor conf file - driver section - if not TYPE_FORWARD_ONLY. Level: DatabaseManager,Database,Table level.	Optional
InitCachesResultSetConcurrency	If Possible values: CONCUR_READ_ONLY or CONCUR_UPDATABLE. Concurrency type: This type will be used for ResultSets created by PreparedStatemens only for SQL queries which will be executed during cache initialization. Default value is defined in dbVendor conf file - driver section - if not CONCUR_READ_ONLY. Level: DatabaseManager,Database,Table level.	Optional
DomlClasspath	If there are no DatabaseMan-.domlF ager.DB.<databaseName>ile properties defined, CacheAdmin will look for propertie named database-.DomlClasspath DatabaseManager.DB.<Name>. This parameter represents full package path to the domlFile in classpath. Level: Database level.	Optional
ConnectionAllocator	If this parameter is set to full class name of class that implements ExtendedConnectionAllocator interface, DODS will use this class to create Connection	Optional

Atribute	Description	Required
	Allocator, if parameter is not set DODS will use default ExtendedConnectionAllocator implementation - StandardConnectionAllocator. Level: DatabaseManager, Database. Default value: none (DODS will use StandardConnectionAllocator)	
reserveFactor	Parameter reserveFactor is used in query caching. It contains information (in percents) about how many more object are taken for time evaluation. If num is number of needed results, then it is used num + DEFAULT_RESERVE_FACTOR * num of objects for estimating what is quicker: go to database for all object that are not in the DataStruct cache, or run again query on database. This value is given in percents, as number between 0 and 1 (0.25 means 25%). Context: Cache. Level: dbManager, database, table.	Optional
initialCondition	Parameter initialCondition contains "where" part of select clause. With this select clause is the DO cache of specified table initialized. If initialCondition = "*', the entire table will be added to the DO cache in DO cache initialization. Context: Cache. Level: table.	Optional
ClassList	During application startup DODS read "DODSClassList.xml" file, and for all data layer classes listed in this file initialize caches This parameter contains absolute path to "DODSClassList.xml" file. Level: dbManager, database.	Optional
initiAllCaches	If initiAllCaches parameter is set to 'true' (on "database manager" or "database" level in configuration file) then default value of initialCondition parameter for all tables are "*". This default values can also be override if initialCondition for table is set to some value. Context: Cache. Level: db-	Optional

Atribute	Description	Required
	Manager, database.	
AutoSaveCreateVirgin	If AutoSaveCreateVirgin is true and a DO belongs to a transaction and DO.createVirgin() is done, the created DO is automatically saved into the transaction. If AutoSaveCreateVirgin is false, DO.save() is mandatory before the transaction's commit. AutoSaveCreateVirgin has no meaning for DOs without transaction. Level: dbManager, database. Default value: false	Optional
CachePercentage	Parameter CachePercentage is used for query to make decision what type of query will be executed: select t.* or select t.oid. If no lazy loading and caching is turned on and value of CachePercentage is less then currently used cache (in percents), t1.oid is used for query statement. Otherwise select t.*. Parameter value 0 means use always t1.oid if cache is turned on, -1 (default) means never if not lazyloading but cached. If lazy loading is on always is used t1.oid query. Context: Cache. Level: dbManager, database, table.	Optional
ClassType	The type of database. Normally this is "Standard". Level: database.	Required
JdbcDriver	The jdbc driver to use. Level: database.	Required
UseCursorName	When turned on, allow DODS to use named cursors, this is default value. Some jdbc compliant drivers, like Microsoft JDBC driver (msjdbc), don't implement this feature and need to disable it (set parameter to false) to avoid runtime Exceptions. Default : true. Level: DatabaseManager , Database.	Optional
CaseSensitive	When turned 'on' DODS use case sensitive version of database operation (on specified level). Level : DatabaseManager , Database,	Optional

Atribute	**Description**	**Required**
	Table.	
Connection.Url	Database url. Level: database.	Required
Connection.User	Database user name. All connection are allocated by this user. Level: database.	Optional
Connection.Password	Database user password. Level: database.	Optional
Connection.MaxPoolSize	The maximum number of connections that a connection pool will hold. If set to zero, then connections are allocated indefinitly or until the database refuses to allocate any new connections. Level: database.	Required
Connection.MaxConnectionUsages	This Parameter is maximum number of how many times a connection can be re-used by a new thread (and put in the connection pool again) before it is closed and re-created (to avoid situation where connection holds too much memory but does not release these resources again because the connection is never closed). Level: Database. Default value: -1	Optional
Connection.AllocationTimeout	Maximum amount of time that a thread will wait for a connection from the connection pool before an exception is thrown. This will prevent possible dead locks. The time out is in milliseconds. If the time out is <= zero, the allocation of connections will wait indefinitely. Level: database.	Required
Connection.DisableConnectionPool	This parameter is used only if ConnectionAllocator is DataSourceConnectionAllocator. If this parameter is set to **true** then DODS will disable internal connection pool. Default: false. Level: database.	Required
QueryCacheImplClass	If this parameter is set to full class name of class that extends DataStructCache abstract class DODS will use this class to create data struct cache for xxxDO class, if parameter is not set DODS will use class QueryCacheImpl as default.. Level: DatabaseManager , Database.	Optional

Atribute	Description	Required
SQLBatch	If this parameter is set to “true” then DODS will use SQLBatch, feature of JDBC PrepareStatement Object, during database operations. Level: DatabaseManager , Database. Default value: false	Optional
ShutDownString	Added to support new version of HSQL database (1.7.2-7 and newer). If this parameter is set to some value (string), then during application shutdown, DODS will send this string to database (as SQL command). Level: Connection. Default value: (none)	Optional
ConnectionAllocateCheckSql	The SQL query statement that will be used to validate connections from connection pool, just before returning them. If specified, this query MUST be valid SQL SELECT statement for current database/driver. Level: Connection. Default value: (none)	Optional
CaseInsensitiveDatabase	f this parameter is set to “true” then DODS will assume that database is case insensitive and will not use "LOWER" function in SQL statements (patch for MS Access).. Level: DatabaseManager , Database. Default value: false	Optional
ObjectId.PrimaryLogicalDatabase	fIn case of multibase DODS application it is possible to tell DODS to use unique (only from one logical database) objectId table to establish unique objectId(OID) across all logical databases. If "PrimaryLogicalDatabase" parameter is set to logical database name that is defined in current configuration then DODS will use objectid table from that database to create oid's for all defined logical databases. If this parameter is not set, then DODS will use separate objectId table from every database (one placed in that particular database). Context: ObjectId. Default value: not used	Optional
QueryTimeLimit	Sets the number of milliseconds for which the Query should be executed and the results readed from ResultSet. If the limit is ex-	Optional

Atribute	Description	Required
	ceeded, an Exception is thrown. Value 0 means that there is no limit. Context: DatabaseManager, Database, Table Default value: 0	
DODSCacheFactory	If this parameter is set to full class name of class that implements DODSCacheAbstractFactory interface DODS will use this class to creates internal, cache, data storage objects used in DataStructCache, QueryCache and TransactionQueryCache implementation to store cache items. Context: cache parameter on DatabaseManager and LogicalDatabase level. Default value: **org.enhydra.dods.cache.lru.DODSLRUCacheFactory**	Optional
Connection.Logging	Used to log database (SQL) activity. Level: database.	Required
Connection.DataSourceName	Parameter define JNDI name (in **jndi:<dataSourceName>** formath) of externally defined DataSource that can be used by DODS to establish connection to database. This parameter is used by DODS only if ConnectionAllocator parameter is set to use DataSource ConnectionAllocator or SimpleDataSource ConnectionAllocator Level: connection.	Optional
ObjectId.CacheSize	Cache size for table objectid. Level: database.	Required
ObjectId.MinValue	The number of object identifiers that are allocated as a group and held in memory. These identifiers are assigned to new data objects that are inserted into the database. Level: database.	Required
ObjectId.NextWithPrefix	If this parametre set to true following statament will be used for update object id table: update OID_TABLE set OID_TABLE.next = ? where OID_TABLE.next = ?. Default value for this parameter is false. Level: database.	Optional
ObjectId.OidTableName	Name of table in database that contains next available value for "oid". Default value for this parameter is "objectid". Level: database.	Optional
ObjectId.NextColumnName	Column name in OidTable-	Optional

Atribute	Description	Required
	Name> that contains next available value for "oid . Default value for this parameter is "next". Level: database.	
DatabaseManager.DB.User.userWildcard	Defines user wildcard sign.	Optional
DatabaseManager.DB.User.userWildcardEscape	Defines user wildcard escape sign.	Optional
DatabaseManager.DB.User.userSingleWildcard	Defines user single wildcard sign.	Optional
DatabaseManager.DB.User.userSingleWildcardEscape	Defines user single wildcard escape sign.	Optional

Logging section: These options configure logging application attributes.

These options have no prefix (eg. not Logging.LogClassName but only LogClassName).

Atribute	Description	Required
LogClassName	Fully qualified name of class implementing Logger. Enhydra ships with three: com.lutris.logging.StandardLogger, com.lutris.logging.Log4jLogger and com.lutris.logging.MonologLogger. Default is com.lutris.logging.StandardLogger	Optional
LogFile	This is the file where the server log is written. Parameter is required for StandardLogger implementation.	Optional (see LogClassName)
LogToFile[]	This is a comma separated list of message types to send to the log file specified in LogFile (uper option). Parameter is required for StandardLogger implementation.	Optional (see LogClassName)
LogToStderr[]	This is a comma separated list of message types to send to standard error. Parameter is required for StandardLogger implementation.	Optional (see LogClassName)
Monolog	Pathname of monolog configuration (property) file.	Optional (see LogClassName)
Log4j	Pathname of Log4j XML configuration file.	Optional (see LogClassName)

Chapter 2. Enhydra DODS Configuration Parameters

Values of DODS configuration parameters are contained in XML documents (not in *.conf files).

Dods Configuration

The dodsConf is composed of one "TemplateDir" element (location of templates), one "CacheJar" element, one "CacheClassName" element, one "DbVendorConfDir" element and one "Database" element.

TemplateDir element: User can make its own template sets. All template sets (standard and users) are placed in one directory. The name and location of this directory can be set (or changed) in dods.properties file (in <dods _root> directory). Within this directory, every template set is placed in its own subdirectory.

Database element: Composed of one or more "Vendor" elements.

Vendor element: Contains the relativ path (reletiv to "DbVendorConfDir") to vendor configuration file Contains the following atribute.

Atribute	Description	Required
name	The vendors name.	This field is required.

Vendor Cofiguration: Database Vendor's configuration file contains information about that database (type of ObjectId, column name for oid and version, information about DeleteCascade, constraints, quotes,comments, characterd for like and wildcard, mapping JDBC types to vendor-specific data types,...).

Element	Description	Required
ClassPath	If tags not mentioned, standard code is generated for that database vendor. If database is specific, path to jar file for that database vendor is in tag.	Optional
ClassName	If tags not mentioned, standard code is generated for that database vendor. If database is specific, path to its main class for that database vendor is in tag.	Optional
OidDbType	The Database type isused to store ObjectId's.	Required
OidJdbcType	The JDBC type is used to store ObjectId's.	Required
OidDbColumnName	The OidDbColumnName is used for generating the SQL CREATE TABLE statements in the create_tables.sql and `SQL.sql` files	Required
VersionDbColumnName	The VersionDbColumnName is	Required

Element	**Description**	**Required**
	used for generating the SQL CREATE TABLE statements in the create_tables.sql and `SQL.sql files`	
DeleteCascade	Some vendors support the ON DELETE CASCADE row attribute. So if DODS generates .sql files containing the following table definitions: • create table dept (deptid INT); • create table employee (deptid INT REFERENCES(dept) ON DELETE CASCADE); then deleting a dept row will cause the deletion of employee rows which refer to that dept. This saves having to write the most common of database triggers.	Required
NamedConstraint	Some vendors support named referential constraints. So if DODS generates .sql files containing the following table definitions: • create table dept (deptid INT); • create table employee (deptid INT CONSTRAINT emp_refs_dept REFERENCES(dept)); then if an employee row is committed with an invalid deptid the error will report a problem with constraint 'emp_refs_dept' rather than some unhelpful constraint name chosen by the database.	Required
StringQuoteCharacter	The StringQuoteCharacter was added by Chris Ryan (cryan@plugged.net.au). There does not appear to be a standard string quote character amongst the DB vendors. Most appear to support ["] but Sybase really objects to this. It requires ['].	Required
CommentStart	Maping acording to the vendor.	Required
CommentEnd	Maping acording to the vendor.	Required

Element	Description	Required
LikeKeyword	Maping acording to the vendor.	Required
Wildcard	Maping acording to the vendor.	Required
SingleWildcard	Maping acording to the vendor.	Required
SingleWildcardEscape	Maping acording to the vendor.	Required
WildcardEscape	Maping acording to the vendor.	Required
WildcardEscapeClause	Maping acording to the vendor.	Required
JDBCtype	Here we map JDBC types to vendor-specific data types.	Required

JDBCtype element: For almost all vendors, the JDBC types BIT, TINYINT, SMALLINT and BIGINT are mapped to the data type INTEGER. For most vendors, these types are actually INTEGER. The following entries can be changed to impose size restrictions. For example, to specify an actual size for a TINYINT, make the change: Database.JDBCtype.TINYINT.Oracle= "DECIMAL(4,0)".

NOTE: The "set" methods in the DO classes generated by DODS do not test for size restrictions on integer values. So, altering the mappings here requires additional range checking in the business logic of the classes using the DOs.

Element	Description	Required
BIT	JDBC type maping acording to Vendor.	This element is required.
TINYINT	JDBC type maping acording to Vendor.	This element is required.
SMALLINT	JDBC type maping acording to Vendor.	This element is required.
INTEGER	JDBC type maping acording to Vendor.	This element is required.
BIGINT	JDBC type maping acording to Vendor.	This element is required.
REAL	JDBC type maping acording to Vendor.	This element is required.
FLOAT	JDBC type maping acording to Vendor.	This element is required.
DOUBLE	JDBC type maping acording to Vendor.	This element is required.
NUMERIC	JDBC type maping acording to Vendor.	This element is required.
DECIMAL	JDBC type maping acording to Vendor.	This element is required.
CHAR	JDBC type maping acording to Vendor.	This element is required.
VARCHAR	JDBC type maping acording to Vendor.	This element is required.
LONGVARCHAR	JDBC type maping acording to Vendor.	This element is required.
BINARY	JDBC type maping acording to	This element is required.

Element	Description	Required
	Vendor.	
VARBINARY	JDBC type maping acording to Vendor.	This element is required.
LONGVARBINARY	JDBC type maping acording to Vendor.	This element is required.
DATE	JDBC type maping acording to Vendor.	This element is required.
TIME	JDBC type maping acording to Vendor.	This element is required.
TIMESTAMP	JDBC type maping acording to Vendor.	This element is required.

Chapter 3. Enhydra DOML Parameters

Values of DOML configuration parameters are contained in XML documents (not in *.conf files). There fore we are giving you an DTD ovewiev of this document.

doml element: The topmost container for the DOA structure is the doml. This will contain one database heirarchy that contains all the packages.

Subelements	**Description**	**Required**
database	Explained in the following text.	
author	Informations abouth author.	
projectname	Project informations.	

database element: This will contain all the package heirarchies that contain all the tables. The only thing that the database needs to actually specify is what sort of a database this is planned for. It has one "package" subelement.

Atributes	**Description**	**Required**
database	Values: Standard, InstantDB, Oracle, Informix, MSQL, Sybase, PostgreSQL, DB2, QED, MySQL. The defoult value is Standard.	
templateset		

package element: Each package can contain a sub-package or a table structures. The naming format for the id's includes the parent's package id. So if package "A" had a child package, it's id would be "A.B".

Atributes	**Description**	**Required**
id		This field is required.

table element: Each table can contain a column or a index structures. Each database table is defined by the following attributes within DODS.

Atributes	**Description**	**Required**
id	The "id" param is much like the package id param, in that it is composed of it's parent packages prefexed before it (so if table C lived in pacakge B which lived in package A, the id would be "A.B.C").	This field is required.
dbTableName	The "dbTableName" param is the actual SQL table name, which is by default the same as the table's name (the "id" minus the parent prefix).	This field is required.

Atributes	Description	Required
isView	Values: true/false. The default value is false. The parameter has yet to be implemented by the current DODs, but is left in place for future growth.	
generateSecure	Values: true/false. The default value is false.	Optional
generateInsecure	Values: true/false. The default value is true.	Optional
multidb	Values: true/false. The default value is false. DODS has the possibility of working with more than one database at the same time. This means that, when the application is started, you don't have to stop it in order to change the database the application uses. For this DODS possibility, you should set attribute multidb to true in doml file for every table that will use multi databases.	Optional
dirtyDOs	Values: Compatible/Deprecate/Omit. The default value is Compatible. Optionaly mark "dirty" methods (that can create DOs in memory without transactions) as "deprecated" or even do not generate them at all. So we have also a possible compile-time check for clean transaction handling of old code.	Optional

column element: Each table column is described with the following attributes and sub-elements javadoc, error, referenceObject, type and initialValue. Each sub-element can exists once and only once.

Atributes	Description	Required
id	The "id" param is the name of the column in the SQL Table (there is no parential prefix required).	This field is required.
usedForQuery	Values: true/false. The default value is false.	
isConstant	Values: true/false. The default value is false.	
generateSecure	Values: true/false. The default value is false.	Optional
generateInsecure	Values: true/false. The default value is true.	Optional

error element: This is not currently implemented by the SourceBuilder side of DODS, but this attribute is in place so that the *DO.set*() methods can complain about illegal values appropriately.

javadoc element: If javadoc is to be included with the column's information, then this is the text that will be spat out.

referenceObject element: If the column is a reference to another row in some other table, that table is specified with this atributes.

Atributes	**Description**	**Required**
constraint	Values: true/false. The default value is false. The "constraint" param is "true" if the other row must exist.	This field is optional.
reference		This field is required.

type element: The type of a column dictates the form of the data stored. If not present in the column then default values are for all attributes of the type.

Atributes	**Description**	**Required**
size	Parameter indicates the number of data width of types that are commonly measured in width, such as VARCHARS. This must be an integer.	
dbType	Parameter dictates internal SQL datatype the database will be using to store values of this column.	
javaType	Parameter dictates what datatype the DO will return to the user when querying this attribute of a Data Object.	This field is required.
canBeNull	Values: true/false. The default value is false.	

initialValue element: If the column's data should always be some default initial value, that would be specified in the span of this element. This was made a sub-element rather than an attribute for possible future expansion.

index element: Each table index is described with the following attributes and can contain indexColumn sub-elements.

Atributes	**Description**	**Required**
id	The "id" param is the name of the index in the SQL Table.	This field is required.
unique	Values: true/false. The default value is false.	Optional
clustered	Values: true/false. The default value is false.	Optional

indexColumn element: Each table index is described with the following attributes.

Atributes	**Description**	**Required**
id	The "id" param is the name of the index column.	This field is required.

Chapter 4. Enhydra EnTray Configuration Parameters

This tables contain the Enhydra EnTray Configuration parameters. These parameters are ment to be specified in entray.properties file placed in EnTay's configuration directory /en-(<enhydra_root>tray/conf).

We can divide these parameters in two sections:

EnTray Core

Atribute	**Description**	**Required**
enhydra.systray.servicenames	Comma separated list of Enhydra Services that are to be administrated thro EnTray entity (multiservice mode of work is aoutomaticly started if more then one service neme defined)	Required
enhydra.systray.root	Path to the EnTray root directory. Parameter is used for internal EnTray administration.	Required

EnTray Service

These parameters are to be repeatedly defined for every Enhydra Service defined in 'enhydra.systray.servicenames' (EnTray Core) parameter value.

Atribute	**Description**	**Required**
enhydra.systray.[service_name].namingport	Naming port of actual Enhydra service (installation option). EnTray is communicating with Enhydra service thro this port (gathering the informations over RMI). The default if not specified is 1099.	Optional
enhydra.systray.[service_name].cmd.start	Command line (command and parameters) that is to be executed in order to start actual Enhydra service.	Required

Atribute	**Description**	**Required**
enhydra.systray.[service_name].cmd.stop	Command line (command and parameters) that is to be executed in order to stop actual Enhydra service.	Required
enhydra.systray.[service_name].username	Username used as connection (to MBean server - if t supports username/password autentication) attribute.	Optional. If not specified, default value 'admin' is aplied.
enhydra.systray.[service_name].password	Password used as connection (to MBean server - if t supports username/password autentication) attribute.	Optional. If not specified, default value 'enhydra' is aplied.