```
//DataStoreService.class
public int delete(String storeName, String id) throws DataStoreException
public DataBean get(String storeName, String id) throws DataStoreException
public Class<DataStoreService> getServiceInterface()
public void put(DataBean data, DataConfig config) throws DataStoreException

//DataBean.class
public byte[] getDataAsArray()
public InputStream getDataAsStream()
public Map<String, Object> getHeaders()
public void setDataAsArray(byte[] dataAsArray)
public void setDataAsStream(InputStream dataAsStream)
public void setHeaders(Map<String, Object> headers

//DataConfig.class
public Boolean doEncrypt()
public Boolean doOverwrite()
public long getExpires()
public String getId()
public String getStoreName()
public void setEncrypt(Boolean encrypt)
public void setExpires(long expires)
public void setId(String id)
public void setOverwrite(Boolean overwrite)
public void setStoreName(String storeName)
```

//DataStore.class

public int countEntries(String storeName) throws DataStoreException

public int countStores(boolean alertOnly, String excludeName) throws DataStoreException

public int delete(Long tid) throws DataStoreException

public int delete(String storeName, String id) throws DataStoreException

public int delete(String storeName, String qualifier, String id) throws DataStoreException

public int deleteExpired(int blocksize) throws DataStoreException

public int deleteStore(String storeName, String qualifier, int blocksize) throws DataStoreException

public ExtendedData get(Long tid) throws DataStoreException, MessageNotFoundException

public ExtendedData get(String storeName, String id) throws DataStoreException, MessageNotFoundException

public ExtendedData get(String storeName, String qualifier, String id) throws DataStoreException, MessageNotFoundException

public DataStoreTable getDataStoreTable()

public PlatformTransactionManager getLocalTransactionManager()

public ExtendedData getLock(Long tid) throws DataStoreException, MessageNotFoundException

public ExtendedData getLock(String storeName, String qualifier, String id) throws DataStoreException, MessageNotFoundException

public Long getTid(String storeName, String qualifier, String id) throws DataStoreException, MessageNotFoundException

public int move(String storeName, String qualifier, String oldStoreName, String oldQualifier, List<String> ids) throws DataStoreException

public void put(Data data, BaseData oldData, boolean encrypt, long alert, long expires) throws DataStoreException

public void put(Data data, boolean overwrite, boolean encrypt, long alert, long expires) throws DataStoreException

public void put(Data data, boolean encrypt, long alert, long expires) throws DataStoreException

public List<Data> select(String storeName, String qualifier, int numRows) throws DataStoreException

public List<Data> select(String storeName, int numRows) throws DataStoreException

public List<String> selectIds(String storeName, String qualifier) throws DataStoreException

public List<MetaData> selectMetaData(String storeName, String id, Date alertAt, int numRows, Long excludeRowBefore) throws DataStoreException

public List<MetaData> selectMetaData(String storeName, String qualifier, String id, Date alertAt, int numRows, Long excludeRowBefore) throws DataStoreException

public List<MetaData> selectMetaData(String storeName, String qualifier, String id, Date alertAt, int numRows, Long excludeRowBefore, boolean excludeNonRetry) throws DataStoreException

public List<DataStoreAggregate> selectStores(boolean alertOnly) throws DataStoreException

public List<DataStoreAggregate> selectStoresInternal(boolean alertOnly) throws DataStoreException

public List<Long> selectTids(String storeName, String qualifier) throws DataStoreException

public void setCipherStreamFactory(CipherStreamFactory cipherStreamFactory)

public void setDataSource(DataSource dataSource)

public void setDatabaseProperties(DatabaseProperties databaseProperties)

public void setPlatformTransactionManager(PlatformTransactionManager platformTransactionManager)

public boolean supportsSaneLocking()

public void updateRetry(Long tid, Date retryAt) throws MessageNotFoundException, DataStoreException

public void updateRetry(Long tid, Date retryAt, String mplId) throws MessageNotFoundException, DataStoreException

//Data.class

public Data(String storeName, String id, InputStream data)

public Data(String storeName, String id, InputStream data, Map<String, Object> headers)

public Data(String storeName, String id, InputStream data, Map<String, Object> headers, Integer version)

public Data(String storeName, String qualifier, String id, InputStream data)

public Data(String storeName, String qualifier, String id, InputStream data, Integer version)

public Data(String storeName, String qualifier, String id, InputStream data, Map<String, Object> headers)

public Data(String storeName, String qualifier, String id, InputStream data, Map<String, Object> headers, Integer version)

public Data(String storeName, String qualifier, String id, InputStream data, Map<String, Object> headers, String mplId, Integer version)

public Data(String storeName, String qualifier, String id, InputStream data, String mplId, Integer version)

public Data(String storeName, String qualifier, String id, byte[] data)

public Data(String storeName, String qualifier, String id, byte[] data, Integer version)

public Data(String storeName, String qualifier, String id, byte[] data, Map<String, Object> headers)

public Data(String storeName, String qualifier, String id, byte[] data, Map<String, Object> headers, Integer version)

public Data(String storeName, String qualifier, String id, byte[] data, Map<String, Object> headers, String mplId, Integer version)

public Data(String storeName, String qualifier, String id, byte[] data, String mplID, Integer version)

public Data(String storeName, String id, byte[] data)

public Data(String storeName, String id, byte[] data, Map<String, Object> headers)

public Object getData()

public byte[] getDataAsArray()

public InputStream getDataAsStream()

public Map<String, Object> getHeaders()

public void setHeaders(Map<String, Object> headers)