Sparksee

6.1.0

Generated by Doxygen 1.16.1

1 Hierarchical Index 1

1.1 Class Hierarchy . . . 1

2 Class Index 3

2.1 Class List . . . 3

3 Class Documentation 8

3.1 STSAttribute Class Reference . . . 8

3.1.1 Detailed Description . . . 9

3.1.2 Method Documentation . . . 9

3.2 STSAttributeList Class Reference . . . 12

3.2.1 Detailed Description . . . 13

3.2.2 Method Documentation . . . 14

3.3 STSAttributeListIterator Class Reference . . . 15

3.3.1 Detailed Description . . . 16

3.3.2 Method Documentation . . . 16

3.4 STSAttributeStatistics Class Reference . . . 17

3.4.1 Detailed Description . . . 18

3.4.2 Method Documentation . . . 18

3.5 STSBooleanList Class Reference . . . 23

3.5.1 Detailed Description . . . 24

3.5.2 Method Documentation . . . 24

3.6 STSBooleanListIterator Class Reference . . . 25

3.6.1 Detailed Description . . . 26

3.6.2 Method Documentation . . . 26

3.7 STSCommunitiesSCD Class Reference . . . 27

3.7.1 Detailed Description . . . 28

3.7.2 Method Documentation . . . 29

3.8 STSCommunityDetection Class Reference . . . 32

3.8.1 Detailed Description . . . 34

3.8.2 Method Documentation . . . 34

3.9 STSConnectedComponents Class Reference . . . 36

3.9.1 Detailed Description . . . 37

3.9.2 Method Documentation . . . 38

3.10 STSConnectivity Class Reference . . . 40

3.10.1 Detailed Description . . . 41

3.10.2 Method Documentation . . . 41

3.11 STSContext Class Reference . . . 43

3.11.1 Detailed Description . . . 45

3.11.2 Method Documentation . . . 45

3.12 STSCSVLoader Class Reference . . . 49

3.12.1 Method Documentation . . . 49

3.13 STSCSVReader Class Reference . . . 51

3.13.1 Detailed Description . . . 53
3.13.2 Method Documentation . . . 53
3.14 STSCSVWriter Class Reference . . . 57
3.14.1 Detailed Description . . . 58
3.14.2 Method Documentation . . . 59
3.15 STSDatabase Class Reference . . . 62
3.15.1 Detailed Description . . . 63
3.15.2 Method Documentation . . . 63
3.16 STSDatabaseStatistics Class Reference . . . 66
3.16.1 Detailed Description . . . 67
3.16.2 Method Documentation . . . 67
3.17 STSDefaultExport Class Reference . . . 69
3.17.1 Detailed Description . . . 72
3.17.2 Method Documentation . . . 72
3.18 STSDisjointCommunities Class Reference . . . 85
3.18.1 Detailed Description . . . 86
3.18.2 Method Documentation . . . 86
3.19 STSDisjointCommunityDetection Class Reference . . . 88
3.19.1 Detailed Description . . . 90
3.19.2 Method Documentation . . . 90
3.20 STSEdgeData Class Reference . . . 94
3.20.1 Detailed Description . . . 94
3.20.2 Method Documentation . . . 95
3.21 STSEdgeExport Class Reference . . . 96
3.21.1 Detailed Description . . . 97
3.21.2 Method Documentation . . . 98
3.22 STSEdgeTypeExporter Class Reference . . . 104
3.22.1 Detailed Description . . . 105
3.22.2 Method Documentation . . . 105
3.23 STSEdgeTypeLoader Class Reference . . . 109
3.23.1 Detailed Description . . . 111
3.23.2 Method Documentation . . . 111
3.24 STSExportManager Class Reference . . . 117
3.24.1 Detailed Description . . . 119
3.24.2 Method Documentation . . . 119
3.25 STSGraph Class Reference . . . 124
3.25.1 Detailed Description . . . 128
3.25.2 Method Documentation . . . 128
3.26 STSGraphExport Class Reference . . . 158
3.26.1 Detailed Description . . . 159
3.26.2 Method Documentation . . . 160
3.27 STSInt32List Class Reference . . . 162

3.27.1 Detailed Description . . . 163
3.27.2 Method Documentation . . . 163
3.28 STSInt32ListIterator Class Reference . . . 164
3.28.1 Detailed Description . . . 165
3.28.2 Method Documentation . . . 165
3.29 STSKeyValue Class Reference . . . 166
3.29.1 Method Documentation . . . 167
3.30 STSKeyValues Class Reference . . . 167
3.30.1 Detailed Description . . . 168
3.30.2 Method Documentation . . . 168
3.31 STSKOpt Class Reference . . . 170
3.31.1 Detailed Description . . . 170
3.31.2 Method Documentation . . . 171
3.32 STSNodeExport Class Reference . . . 171
3.32.1 Detailed Description . . . 173
3.32.2 Method Documentation . . . 173
3.33 STSNodeTypeExporter Class Reference . . . 181
3.33.1 Detailed Description . . . 182
3.33.2 Method Documentation . . . 183
3.34 STSNodeTypeLoader Class Reference . . . 185
3.34.1 Detailed Description . . . 187
3.34.2 Method Documentation . . . 187
3.35 STSObjects Class Reference . . . 191
3.35.1 Detailed Description . . . 193
3.35.2 Method Documentation . . . 194
3.36 STSObjectsIterator Class Reference . . . 201
3.36.1 Detailed Description . . . 202
3.36.2 Method Documentation . . . 202
3.37 STSOidList Class Reference . . . 203
3.37.1 Detailed Description . . . 204
3.37.2 Method Documentation . . . 204
3.38 STSOidListIterator Class Reference . . . 206
3.38.1 Detailed Description . . . 207
3.38.2 Method Documentation . . . 207
3.39 STSPageRank Class Reference . . . 208
3.39.1 Detailed Description . . . 209
3.39.2 Method Documentation . . . 209
3.40 STSPlatform Class Reference . . . 214
3.40.1 Detailed Description . . . 215
3.40.2 Method Documentation . . . 215
3.41 STSPlatformStatistics Class Reference . . . 215
3.41.1 Detailed Description . . . 216

3.41.2 Method Documentation . . . 216
3.42 STSQuery Class Reference . . . 218
3.42.1 Detailed Description . . . 219
3.42.2 Method Documentation . . . 219
3.43 STSQueryStream Class Reference . . . 220
3.43.1 Detailed Description . . . 221
3.43.2 Method Documentation . . . 222
3.44 STSQueryStreamManager Class Reference . . . 223
3.44.1 Detailed Description . . . 223
3.44.2 Method Documentation . . . 224
3.45 STSRandomWalk Class Reference . . . 225
3.45.1 Detailed Description . . . 226
3.45.2 Method Documentation . . . 227
3.46 STSResultSet Class Reference . . . 232
3.46.1 Detailed Description . . . 233
3.46.2 Method Documentation . . . 234
3.47 STSResultSetList Class Reference . . . 237
3.47.1 Detailed Description . . . 238
3.47.2 Method Documentation . . . 238
3.48 STSResultSetListIterator Class Reference . . . 239
3.48.1 Detailed Description . . . 240
3.48.2 Method Documentation . . . 240
3.49 STSRowReader Class Reference . . . 241
3.49.1 Detailed Description . . . 242
3.49.2 Method Documentation . . . 242
3.50 STSRowWriter Class Reference . . . 244
3.50.1 Detailed Description . . . 245
3.50.2 Method Documentation . . . 245
3.51 STSScriptParser Class Reference . . . 246
3.51.1 Detailed Description . . . 247
3.51.2 Method Documentation . . . 247
3.52 STSSession Class Reference . . . 249
3.52.1 Detailed Description . . . 250
3.52.2 Method Documentation . . . 251
3.53 STSShortestPath Class Reference . . . 253
3.53.1 Detailed Description . . . 254
3.53.2 Method Documentation . . . 254
3.54 STSSinglePairShortestPath Class Reference . . . 257
3.54.1 Detailed Description . . . 258
3.54.2 Method Documentation . . . 259
3.55 STSSinglePairShortestPathBFS Class Reference . . . 262
3.55.1 Detailed Description . . . 264

3.55.2 Method Documentation . . . 264
3.56 STSSinglePairShortestPathDijkstra Class Reference . . . 268
3.56.1 Detailed Description . . . 269
3.56.2 Method Documentation . . . 270
3.57 STSSinglePairShortestPathDijkstraDynamicCost Class Reference . . . 274
3.57.1 Detailed Description . . . 275
3.57.2 Method Documentation . . . 275
3.58 STSSparksee Class Reference . . . 276
3.58.1 Detailed Description . . . 277
3.58.2 Method Documentation . . . 278
3.59 STSSparkseeConfig Class Reference . . . 287
3.59.1 Detailed Description . . . 291
3.59.2 Method Documentation . . . 292
3.60 STSSparkseeProperties Class Reference . . . 314
3.60.1 Detailed Description . . . 315
3.60.2 Method Documentation . . . 315
3.61 STSStringList Class Reference . . . 318
3.61.1 Detailed Description . . . 319
3.61.2 Method Documentation . . . 319
3.62 STSStringListIterator Class Reference . . . 320
3.62.1 Detailed Description . . . 321
3.62.2 Method Documentation . . . 321
3.63 STSStrongConnectivity Class Reference . . . 322
3.63.1 Detailed Description . . . 323
3.63.2 Method Documentation . . . 324
3.64 STSStrongConnectivityGabow Class Reference . . . 326
3.64.1 Detailed Description . . . 328
3.64.2 Method Documentation . . . 329
3.65 STSTextStream Class Reference . . . 331
3.65.1 Detailed Description . . . 333
3.65.2 Method Documentation . . . 333
3.66 STSTraversal Class Reference . . . 335
3.66.1 Detailed Description . . . 336
3.66.2 Method Documentation . . . 337
3.67 STSTraversalBFS Class Reference . . . 340
3.67.1 Detailed Description . . . 341
3.67.2 Method Documentation . . . 341
3.68 STSTraversalDFS Class Reference . . . 345
3.68.1 Detailed Description . . . 347
3.68.2 Method Documentation . . . 347
3.69 STSType Class Reference . . . 351
3.69.1 Detailed Description . . . 352

3.69.2 Method Documentation . . . 352
3.70 STSTypeExporter Class Reference . . . 355
3.70.1 Detailed Description . . . 356
3.70.2 Method Documentation . . . 356
3.71 STSTypeExporterEvent Class Reference . . . 359
3.71.1 Detailed Description . . . 360
3.71.2 Method Documentation . . . 360
3.72 STSTypeExporterListener Class Reference . . . 361
3.72.1 Detailed Description . . . 362
3.72.2 Method Documentation . . . 362
3.73 STSTypeList Class Reference . . . 362
3.73.1 Detailed Description . . . 363
3.73.2 Method Documentation . . . 364
3.74 STSTypeListIterator Class Reference . . . 365
3.74.1 Detailed Description . . . 366
3.74.2 Method Documentation . . . 366
3.75 STSTypeLoader Class Reference . . . 367
3.75.1 Detailed Description . . . 368
3.75.2 Method Documentation . . . 368
3.76 STSTypeLoaderEvent Class Reference . . . 373
3.76.1 Detailed Description . . . 374
3.77 STSTypeLoaderListener Class Reference . . . 374
3.77.1 Detailed Description . . . 375
3.77.2 Method Documentation . . . 375
3.78 STSValue Class Reference . . . 375
3.78.1 Detailed Description . . . 378
3.78.2 Method Documentation . . . 378
3.79 STSValueArray Class Reference . . . 389
3.79.1 Detailed Description . . . 391
3.79.2 Method Documentation . . . 391
3.80 STSValueList Class Reference . . . 402
3.80.1 Detailed Description . . . 403
3.80.2 Method Documentation . . . 403
3.81 STSValueListIterator Class Reference . . . 404
3.81.1 Detailed Description . . . 405
3.81.2 Method Documentation . . . 405
3.82 STSValues Class Reference . . . 406
3.82.1 Detailed Description . . . 407
3.82.2 Method Documentation . . . 407
3.83 STSValuesIterator Class Reference . . . 408
3.83.1 Detailed Description . . . 409
3.83.2 Method Documentation . . . 409

3.84 STSWeakConnectivity Class Reference . . . 410
3.84.1 Detailed Description . . . 412
3.84.2 Method Documentation . . . 412
3.85 STSWeakConnectivityDFS Class Reference . . . 415
3.85.1 Detailed Description . . . 417
3.85.2 Method Documentation . . . 417
3.86 SWIGTYPE_p_p_sparksee__gdb__QueryStream Class Reference . . . 420
3.87 SWIGTYPE_p_wchar_t Class Reference . . . 421

1 Hierarchical Index

1.1 Class Hierarchy

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

NSObject

STSAttribute 8

STSAttributeList 12

STSAttributeListIterator 15

STSAttributeStatistics 17

STSBooleanList 23

STSBooleanListIterator 25

STSCSVLoader 49

STSCommunityDetection 32

STSDisjointCommunityDetection 88

STSCommunitiesSCD 27

STSConnectedComponents 36

STSConnectivity 40

STSStrongConnectivity 322

STSStrongConnectivityGabow 326

STSWeakConnectivity 410

STSWeakConnectivityDFS 415

STSContext 43

STSDatabase 62

STSDatabaseStatistics 66

STSDisjointCommunities 85

STSEdgeData 94

STSEdgeExport 96

STSExportManager 117

STSDefaultExport 69

STSGraph 124

STSGraphExport 158

STSInt32List 162

STSInt32ListIterator 164

STSKOpt 170

STSKeyValue 166

STSKeyValues 167

STSNodeExport 171

STSObjects 191

STSObjectsIterator 201

STSOidList 203

STSOidListIterator 206

STSPageRank 208

STSPlatform 214

STSPlatformStatistics 215

STSQuery 218

STSQueryStream 220

STSQueryStreamManager 223

STSResultSet 232

STSResultSetList 237

STSResultSetListIterator 239

STSRowReader 241

STSCSVReader 51

STSRowWriter 244

STSCSVWriter 57

STSScriptParser 246

STSSession 249

STSShortestPath 253

STSSinglePairShortestPath 257
STSSinglePairShortestPathBFS 262
STSSinglePairShortestPathDijkstra 268
STSSinglePairShortestPathDijkstraDynamicCost 274
STSSparksee 276
STSSparkseeConfig 287
STSSparkseeProperties 314
STSStringList 318
STSStringListIterator 320
STSTextStream 331
STSTraversal 335
STSRandomWalk 225
STSTraversalBFS 340
STSTraversalDFS 345
STSType 351
STSTypeExporter 355
STSEdgeTypeExporter 104
STSNodeTypeExporter 181
STSTypeExporterEvent 359
STSTypeExporterListener 361
STSTypeList 362
STSTypeListIterator 365
STSTypeLoader 367
STSEdgeTypeLoader 109
STSNodeTypeLoader 185
STSTypeLoaderEvent 373
STSTypeLoaderListener 374
STSValue 375
STSValueArray 389
STSValueList 402
STSValueListIterator 404
STSValues 406

STSValuesIterator 408

SWIGTYPE_p_p_sparksee__gdb__QueryStream 420

SWIGTYPE_p_wchar_t 421

2 Class Index

2.1 Class List

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

STSAttribute
Attribute data class 8

STSAttributeList
Sparksee attribute identifier list 12

STSAttributeListIterator
AttributeList iterator class 15

STSAttributeStatistics
Attribute statistics class 17

STSBooleanList
Boolean list 23

STSBooleanListIterator
BooleanList iterator class 25

STSCommunitiesSCD
CommunitiesSCD class 27

STSCommunityDetection
CommunityDetection class 32

STSConnectedComponents
ConnectedComponents class 36

STSConnectivity
Connectivity class 40

STSContext
Context class 43

STSCSVLoader 49

STSCSVReader
CSVReader interface 51

STSCSVWriter
CSVWriter interface 57

STSDatabase
Database class 62

STSDatabaseStatistics
Database statistics 66

STSDefaultExport
Default implementation for ExportManager class 69

STSDisjointCommunities
DisjointCommunities class 85

STSDisjointCommunityDetection
DisjointCommunityDetection class 88

STSEdgeData
Edge data class 94

STSEdgeExport
Stores edge exporting values 96

STSEdgeTypeExporter
EdgeTypeExporter class 104

STSEdgeTypeLoader
EdgeTypeLoader class 109

STSExportManager
Defines how to export a graph to an external format 117

STSGraph
Graph class 124

STSGraphExport
Stores the graph exporting values 158

STSInt32List
Sparksee 32-bit signed integer list 162

STSInt32ListIterator
Int32List iterator class 164

STSKeyValue 166

STSKeyValues
Value set class 167

STSKOpt
KOpt class 170

STSNodeExport
Stores the node exporting values 171

STSNodeTypeExporter
NodeTypeExporter class 181

STSNodeTypeLoader
NodeTypeLoader class 185

STSObjects
Object identifier set class 191

STSObjectsIterator
ObjectsIterator class 201

STSOidList
Sparksee object identifier list 203

STSOidListIterator
OIDList iterator class 206

STSPageRank
PageRank class 208

STSPlatform
Platform class 214

STSPlatformStatistics
Platform data and statistics 215

STSQuery
Query class 218

STSQueryStream
Query stream interface 220

STSQueryStreamManager
QueryStreamManager interface 223

STSRandomWalk
RandomWalk class 225

STSResultSet
ResultSet class 232

STSResultSetList
ResultSet list 237

STSResultSetListIterator
ResultSetList iterator class 239

STSRowReader
RowReader interface 241

STSRowWriter
RowWriter interface 244

STSScriptParser
ScriptParser 246

STSSession
Session class 249

STSShortestPath
ShortestPath class 253

STSSinglePairShortestPath
SinglePairShortestPath class 257

STSSinglePairShortestPathBFS
SinglePairShortestPathBFS class 262

STSSinglePairShortestPathDijkstra
SinglePairShortestPathDijkstra class 268

STSSinglePairShortestPathDijkstraDynamicCost
Defines how to calculate an edge weight 274

STSSparksee
Sparksee class 276

STSSparkseeConfig
Sparksee configuration class 287

STSSparkseeProperties
Sparksee properties file 314

STSStringList
String list 318

STSStringListIterator
StringList iterator class 320

STSStrongConnectivity
StrongConnectivity class 322

STSStrongConnectivityGabow
This class can be used to solve the problem of finding strongly connected components in a directed graph 326

STSTextStream
TextStream class 331

STSTraversal
Traversal class 335

STSTraversalBFS
Breadth-First Search implementation of Traversal 340

STSTraversalDFS
Depth-First Search (DFS) implementation of Traversal 345

STSType
Type data class 351

STSTypeExporter
Base TypeExporter class 355

STSTypeExporterEvent
Provides information about the progress of an TypeExproter instance 359

STSTypeExporterListener
Interface to be implemented to receive TypeExporterEvent events from a TypeExporter 361

STSTypeList
Sparksee type identifier list 362

STSTypeListIterator
TypeList iterator class 365

STSTypeLoader
Base TypeLoader class 367

STSTypeLoaderEvent
Provides information about the progress of a TypeLoader instance 373

STSTypeLoaderListener
Interface to be implemented to receive TypeLoaderEvent events from a TypeLoader 374

STSValue
Value class 375

STSValueArray
ValueArray class 389

STSValueList
Value list 402

STSValueListIterator
ValueList iterator class 404

STSValues
Value set class 406

STSValuesIterator
Values iterator class 408

STSWeakConnectivity
WeakConnectivity class 410

STSWeakConnectivityDFS
WeakConnectivityDFS class 415

SWIGTYPE_p_p_sparksee__gdb__QueryStream 420

SWIGTYPE_p_wchar_t 421

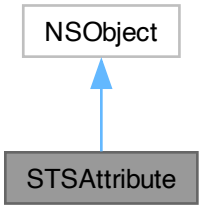
3 Class Documentation

3.1 STSAttribute Class Reference

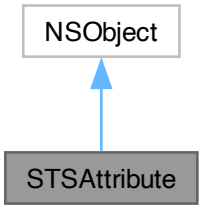
Attribute data class.

`#include <Sparksee.h>`

Inheritance diagram for STSAttribute:



Collaboration diagram for STSAttribute:



Instance Methods

- (int) - getId
  Gets the Sparksee attribute identifier.
- (int) - getTypeId
  Gets the Sparksee type identifier.
- (NSString ∗) - getName
  Gets the unique attribute name.
- (enum STSDataType) - getDataType
  Gets the data type.
- (long long) - getSize
  Gets the number of different values.
- (long long) - getCount
  Gets the number of non-NULL values.
- (enum STSAttributeKind) - getKind
  Gets the attribute kind.
- (BOOL) - isSessionAttribute
  Check if it's a session attribute or a persistent one.
- (STSValue ∗) - getDefaultValue
  Gets the default value for this attribute.
- (BOOL) - isArrayAttribute
  Check if it's an array attribute.
- (int) - getArraySize
  Gets the number of elements in the array.

Class Methods

- (int) + **getInvalidAttribute**
  Invalid attribute identifier constant.

3.1.1 Detailed Description

Attribute data class.

It contains information about an attribute.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.1.2 Method Documentation

getArraySize

```
- (int) getArraySize
```

Gets the number of elements in the array.

Returns

The size of the array

References getArraySize.

Referenced by getArraySize.

getCount

```
- (long long) getCount
```

Gets the number of non-NULL values.

Returns

The number of non-NULL values.

References getCount.

Referenced by getCount.

getDataType

```
- (enum STSDataType) getDataType
```

Gets the data type.

Returns

The DataType.

References getDataType.

Referenced by getDataType.

getDefaultValue

```
- (STSValue *) getDefaultValue
```

Gets the default value for this attribute.

Returns

A new Value instance with the default value, or NULL if none is set.

References getDefaultValue.

Referenced by getDefaultValue.

getId

```
- (int) getId
```

Gets the Sparksee attribute identifier.

Returns

The Sparksee attribute identifier.

References getId.

Referenced by getId.

getKind

```
- (enum STSAttributeKind) getKind
```

Gets the attribute kind.

Returns

The AttributeKind.

References getKind.

Referenced by getKind.

getName

```
- (NSString *) getName
```

Gets the unique attribute name.

Returns

The unique attribute name.

References getName.

Referenced by getName.

getSize

```
- (long long) getSize
```

Gets the number of different values.

Returns

The number of different values.

References getSize.

Referenced by getSize.

getTypeId

```
- (int) getTypeId
```

Gets the Sparksee type identifier.

Returns

The Sparksee type identifier.

References getTypeId.

Referenced by getTypeId.

isArrayAttribute

```
- (BOOL) isArrayAttribute
```

Check if it's an array attribute.

Returns

True if it's an array attribute, or false otherwise.

References isArrayAttribute.

Referenced by isArrayAttribute.

isSessionAttribute

```
- (BOOL) isSessionAttribute
```

Check if it's a session attribute or a persistent one.

Returns

True if it's a session attribute, or false otherwise.

References isSessionAttribute.

Referenced by isSessionAttribute.

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

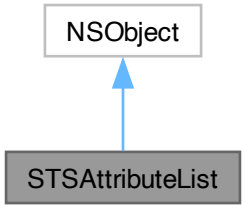
- Sparksee.h

3.2 STSAttributeList Class Reference

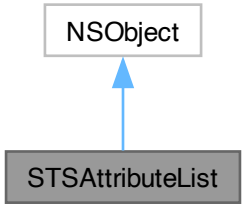
Sparksee attribute identifier list.

`#include <Sparksee.h>`

Inheritance diagram for STSAttributeList:



Collaboration diagram for STSAttributeList:



Instance Methods

- (int) - count

  Number of elements in the list.
- (id) - init

  Constructor.
- (void) - add:

  Adds a Sparksee attribute identifier at the end of the list.
- (void) - **clear**

  Clears the list.
- (id) - **initWithArray:**

  Creates a new AttributeList instance from the given array.
- (id) - **initWithNSEnumerator:**

  Creates a new AttributeList instance from the given NSEnumerator.
- (STSAttributeListIterator ∗) - **iterator**

  Gets a new AttributeListIterator.

3.2.1 Detailed Description

Sparksee attribute identifier list.

It stores a Sparksee attribute identifier list. Optionally it may also store tha corresponding attribute names. It can also optionally store a single set of values for each attribute.

Use AttributeListIterator to access all elements into this collection.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.2.2 Method Documentation

add:

```
- (void) add:
             (int) attr
```

Adds a Sparksee attribute identifier at the end of the list.

Parameters

attr	[in] Sparksee attribute identifier.

count

```
- (int) count
```

Number of elements in the list.

Returns

Number of elements in the list.

References count.

Referenced by count.

init

```
- (id) init
```

Constructor.

This creates an empty list.

References init.

Referenced by init.

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

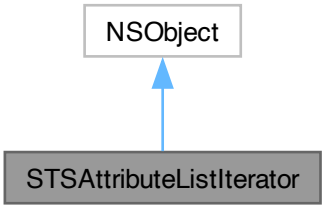
- Sparksee.h

3.3 STSAttributeListIterator Class Reference

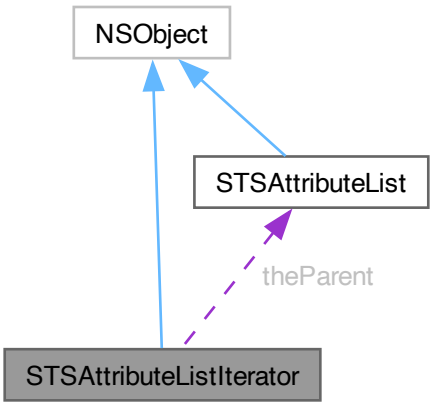
AttributeList iterator class.

`#include <Sparksee.h>`

Inheritance diagram for STSAttributeListIterator:



Collaboration diagram for STSAttributeListIterator:



Instance Methods

- (int) - next

  Moves to the next element.
- (BOOL) - hasNext

  Gets if there are more elements.
- (void) - getCurrentValue:

  Get the value for the current attribute if it's available or set it Null otherwise.
- (NSString ∗) - **getCurrentName**

  Get the current attribute Name if it's available or an empty string otherwise.

3.3.1 Detailed Description

AttributeList iterator class.

Iterator to traverse all the Sparksee attribute identifier into a AttributeList instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.3.2 Method Documentation

getCurrentValue:

```
- (void) getCurrentValue:
             (STSValue *) value
```

Get the value for the current attribute if it's available or set it Null otherwise.

Parameters

value	[in|out] The Value will be set to the current attribute value or as a Null Value.

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements.

Returns

TRUE if there are more elements, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (int) next
```

Moves to the next element.

Returns

The next element.

References next.

Referenced by next.

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

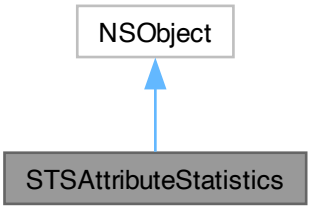
- Sparksee.h

3.4 STSAttributeStatistics Class Reference

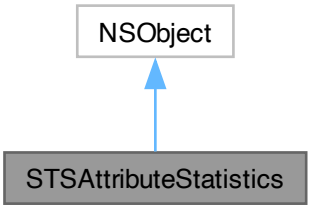
Attribute statistics class.

`#include <Sparksee.h>`

Inheritance diagram for STSAttributeStatistics:



Collaboration diagram for STSAttributeStatistics:



Instance Methods

- (long long) - getTotal

  Gets the number of objects with a non-NULL Value (BASIC statistic).
- (long long) - getNull

  Gets the number of objects NULL a Value (BASIC statistics).
- (long long) - getDistinct

  Gets the number of distinct values (BASIC statistics).
- (STSValue ∗) - getMin

  Gets the minimum existing value (BASIC statistics).
- (STSValue ∗) - getMax

  Gets the maximum existing value (BASIC statistics).
- (int) - getMaxLengthString

  Gets the maximum length.

- (int) - getMinLengthString

  Gets the minimum length.
- (double) - getAvgLengthString

  Gets the average length.
- (STSValue ∗) - getMode

  Gets the mode.
- (long long) - getModeCount

  Gets the number of objects with a Value equal to the mode.
- (double) - getMean

  Gets the mean or average.
- (double) - getVariance

  Gets the variance.
- (double) - getMedian

  Gets the median.

3.4.1 Detailed Description

Attribute statistics class.

It contains statistic data about an attribute.

Some fields are valid just for numerical attributes and others just for string attributes. Also, some statistics are considered BASIC because computing them do not require to traverse all the different values of the attribute. For each getter method the documentation tells if the statistic is BASIC or not. See the Graph class method get↩ AttributeStatistics or check out the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.4.2 Method Documentation

getAvgLengthString

```
- (double) getAvgLengthString
```

Gets the average length.

If the attribute is not an string attribute, it just returns 0.

Returns

The average length.

References getAvgLengthString.

Referenced by getAvgLengthString.

getDistinct

```
- (long long) getDistinct
```

Gets the number of distinct values (BASIC statistics).

Returns

The number of distinct values.

References getDistinct.

Referenced by getDistinct.

getMax

```
- (STSValue *) getMax
```

Gets the maximum existing value (BASIC statistics).

Returns

The maximum existing value.

References getMax.

Referenced by getMax.

getMaxLengthString

```
- (int) getMaxLengthString
```

Gets the maximum length.

If the attribute is not an string attribute, it just returns 0.

Returns

The maximum length.

References getMaxLengthString.

Referenced by getMaxLengthString.

getMean

```
- (double) getMean
```

Gets the mean or average.

Mean or average: Sum of all Values divided by the number of observations.

It is computed just for numerical attributes.

Returns

The mean.

References getMean.

Referenced by getMean.

getMedian

```
- (double) getMedian
```

Gets the median.

Median: Middle value that separates the higher half from the lower.

If a $<$ b $<$ c, then the median of the list {a, b, c} is b, and if a $<$ b $<$ c $<$ d, then the median of the list {a, b, c, d} is the mean of b and c, i.e. it is (b + c)/2

It is computed just for numerical attributes.

Returns

The median.

References getMedian.

Referenced by getMedian.

getMin

```
- (STSValue *) getMin
```

Gets the minimum existing value (BASIC statistics).

Returns

The minimum existing value.

References getMin.

Referenced by getMin.

getMinLengthString

```
- (int) getMinLengthString
```

Gets the minimum length.

If the attribute is not an string attribute, it just returns 0.

Returns

The minimum length.

References getMinLengthString.

Referenced by getMinLengthString.

getMode

```
- (STSValue *) getMode
```

Gets the mode.

Mode: Most frequent Value.

Returns

The mode.

References getMode.

Referenced by getMode.

getModeCount

```
- (long long) getModeCount
```

Gets the number of objects with a Value equal to the mode.

Returns

The number of objects with a Value equal to the mode.

References getModeCount.

Referenced by getModeCount.

getNull

```
- (long long) getNull
```

Gets the number of objects NULL a Value (BASIC statistics).

Returns

The number of objects NULL a Value.

References getNull.

Referenced by getNull.

getTotal

```
- (long long) getTotal
```

Gets the number of objects with a non-NULL Value (BASIC statistic).

Returns

The number of objects with a non-NULL Value.

References getTotal.

Referenced by getTotal.

getVariance

```
- (double) getVariance
```

Gets the variance.

It is computed just for numerical attributes.

Returns

The variance.

References getVariance.

Referenced by getVariance.

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

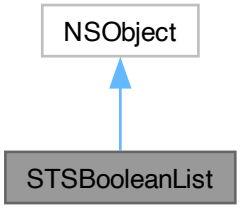
- Sparksee.h

3.5 STSBooleanList Class Reference

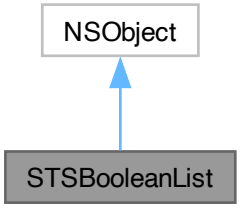
Boolean list.

`#include <Sparksee.h>`

Inheritance diagram for STSBooleanList:



Collaboration diagram for STSBooleanList:



Instance Methods

- (int) - count

  Number of elements in the list.
- (id) - init

  Constructor.
- (void) - add:

  Adds a Boolean at the end of the list.
- (void) - **clear**

  Clears the list.
- (id) - **initWithArray:**

  Creates a new BooleanList instance from the given array.
- (id) - **initWithNSEnumerator:**

  Creates a new BooleanList instance from the given NSEnumerator.
- (STSBooleanListIterator ∗) - **iterator**

  Gets a new BooleanListIterator.

3.5.1 Detailed Description

Boolean list.

It stores a Boolean list.

Use BooleanListIterator to access all elements into this collection.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.5.2 Method Documentation

add:

```
- (void) add:
             (BOOL) value
```

Adds a Boolean at the end of the list.

Parameters

value	[in] Boolean.

count

```
- (int) count
```

Number of elements in the list.

Returns

Number of elements in the list.

References count.

Referenced by count.

init

```
- (id) init
```

Constructor.

This creates an empty list.

References init.

Referenced by init.

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

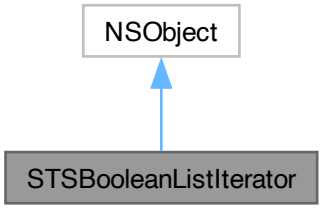
- Sparksee.h

3.6 STSBooleanListIterator Class Reference

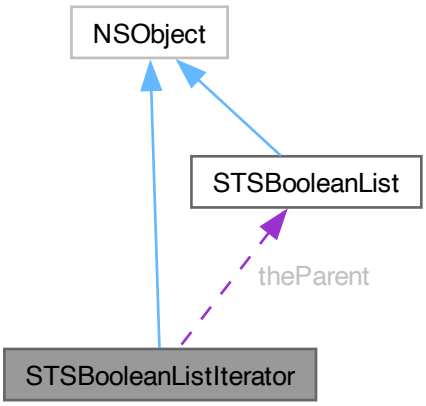
BooleanList iterator class.

`#include <Sparksee.h>`

Inheritance diagram for STSBooleanListIterator:



Collaboration diagram for STSBooleanListIterator:



Instance Methods

- (BOOL) - next

  Moves to the next element.
- (BOOL) - hasNext

  Gets if there are more elements.

3.6.1 Detailed Description

BooleanList iterator class.

Iterator to traverse all the strings into a BooleanList instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.6.2 Method Documentation

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements.

Returns

TRUE if there are more elements, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (BOOL) next
```

Moves to the next element.

Returns

The next element.

References next.

Referenced by next.

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

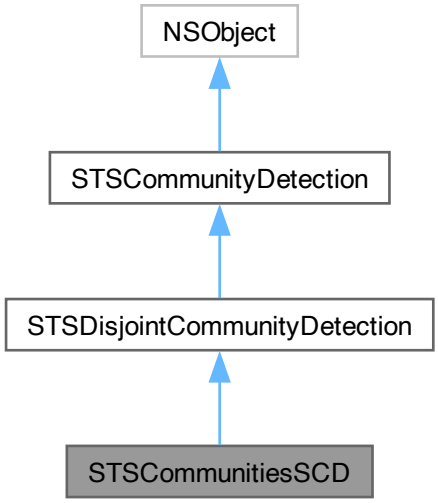
- Sparksee.h

3.7 STSCommunitiesSCD Class Reference

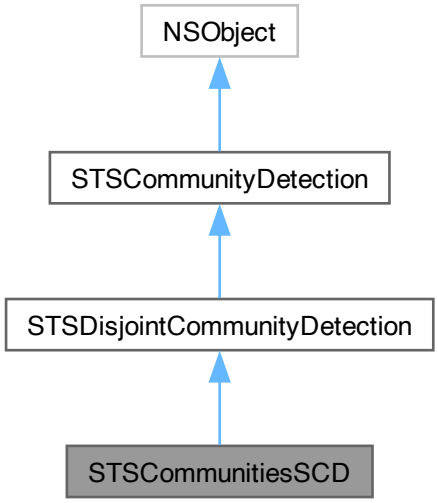
CommunitiesSCD class.

`#include <Sparksee.h>`

Inheritance diagram for STSCommunitiesSCD:



Collaboration diagram for STSCommunitiesSCD:



Instance Methods

- (id) - initWithSession:

  Creates a new instance of CommunitiesSCD.
- (void) - setLookAhead:

  Sets the size of the lookahead iterations to look (5 by default).
- (void) - **run**

  Executes the algorithm.
- (void) - addEdgeType:

  Allows connectivity through edges of the given type.
- (void) - addAllEdgeTypes

  Allows connectivity through all edge types of the graph.
- (STSDisjointCommunities *) - getCommunities

  Returns the results generated by the execution of the algorithm.
- (void) - setMaterializedAttribute:

  Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the disjoint communities found while executing this algorithm.
- (void) - addNodeType:

  Allows connectivity through nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows connectivity through all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (void) - includeNodes:

  Set additional nodes that can be used.
- (void) - includeEdges:

  Set additional edges that can be used.
- (void) - close

  Closes the CommunityDetection instance.
- (BOOL) - **isClosed**

  Check if the CommunityDetection instance is closed.

3.7.1 Detailed Description

CommunitiesSCD class.

Implementation of the community detection algorithm "Scalable Community Detection" based on the paper "High quality, scalable and parallel community detection for large real graphs" by Arnau Prat-Perez, David Dominguez-Sal, Josep-Lluis Larriba-Pey - WWW 2014.

The purpose of this algorithm is to find disjoint communities in an undirected graph or in a directed graph which will be considered as an undirected one.

It is possible to set some restrictions after constructing a new instance of this class and before running it in order to limit the results.

After the execution, we can retrieve the results stored in an instance of the DisjointCommunities class using the getCommunities method.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.7.2 Method Documentation

addAllEdgeTypes

```
- (void) addAllEdgeTypes
```

Allows connectivity through all edge types of the graph.

The edges can be used in Any direction.

References addAllEdgeTypes.

Referenced by addAllEdgeTypes.

addEdgeType:

```
- (void) addEdgeType:
             (int) type
```

Allows connectivity through edges of the given type.

The edges can be used in Any direction.

Parameters

type	[in] Edge type.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows connectivity through nodes of the given type.

Parameters

type	null

close

```
- (void) close
```

Closes the CommunityDetection instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getCommunities

```
- (STSDisjointCommunities *) getCommunities
```

Returns the results generated by the execution of the algorithm.

These results contain information related to the disjoint communities found as the number of different components, the set of nodes contained in each component or many other data.

Returns

Returns an instance of the class DisjointCommunities which contain information related to the disjoint communities found.

References getCommunities.

Referenced by getCommunities.

includeEdges:

```
- (void) includeEdges:
             (STSObjects *) edges
```

Set additional edges that can be used.

This will replace any previously specified set of include edges. Using this optional method adds valid edges to the edges of any edge type explicitly set as a valid type. Should only be used to include specific small sets of edges because it's less efficient than just using an edge type. For any edge to be used, both nodes must be also valid.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

includeNodes:

```
- (void) includeNodes:
             (STSObjects *) nodes
```

Set additional nodes that can be used.

This will replace any previously specified set of include nodes. Using this optional method adds valid nodes to the nodes of any node type explicitly set as a valid type. Should only be used to include specific small sets of nodes because it's less efficient than just using a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

initWithSession:

```
- (id) initWithSession:
             (STSSession *) session
```

Creates a new instance of CommunitiesSCD.

After creating this instance is required to indicate the set of edge types and the set of node types which will be navigated through while traversing the graph in order to find the communities.

Parameters

session	[in] Session to get the graph from and calculate the communities

setLookAhead:

```
- (void) setLookAhead:
             (int) lookahead
```

Sets the size of the lookahead iterations to look (5 by default).

Parameters

lookahead	[in] Number of iterations. It must be positive or zero.

setMaterializedAttribute:

```
- (void) setMaterializedAttribute:
             (NSString *) attributeName
```

Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the disjoint communities found while executing this algorithm.

Whenever the user wants to retrieve the results, even when the graph has been closed and opened again, it is only necessary to create a new instance of the class DisjointCommunities indicating the graph and the name of the common attribute type which stores the results. This instance will have all the information related to the disjoint communities found in the moment of the execution of the algorithm that stored this data.

It is possible to run the algorithm without specifying this parameter in order to avoid materializing the results of the execution.

Parameters

attributeName	[in] The name of the common attribute type for all node types in the graph which will store persistently the results generated by the execution of the algorithm.

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

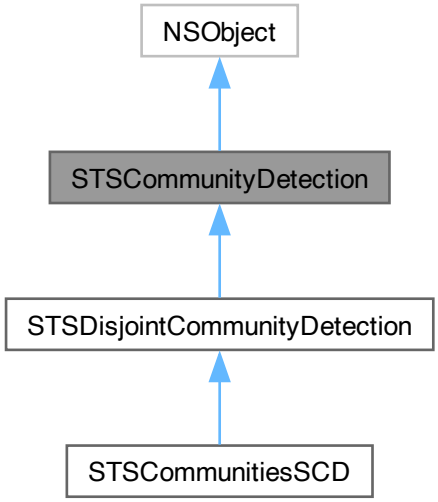
- Sparksee.h

3.8 STSCommunityDetection Class Reference

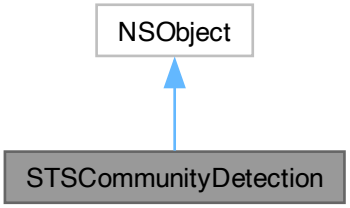
CommunityDetection class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSCommunityDetection:



Collaboration diagram for STSCommunityDetection:



Instance Methods

- (void) - addNodeType:

  Allows connectivity through nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows connectivity through all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (void) - includeNodes:

  Set additional nodes that can be used.

- (void) - includeEdges:

  Set additional edges that can be used.
- (void) - run

  Runs the algorithm in order to find the connected components.
- (void) - close

  Closes the CommunityDetection instance.
- (BOOL) - **isClosed**

  Check if the CommunityDetection instance is closed.

3.8.1 Detailed Description

CommunityDetection class.

Any class implementing this abstract class can be used to solve a problem related to graph connectivity as finding the strongly connected components, finding the weakly connected components.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.8.2 Method Documentation

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows connectivity through nodes of the given type.

Parameters

type	null

close

```
- (void) close
```

Closes the CommunityDetection instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

includeEdges:

```
- (void) includeEdges:
             (STSObjects *) edges
```

Set additional edges that can be used.

This will replace any previously specified set of include edges. Using this optional method adds valid edges to the edges of any edge type explicitly set as a valid type. Should only be used to include specific small sets of edges because it's less efficient than just using an edge type. For any edge to be used, both nodes must be also valid.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

includeNodes:

```
- (void) includeNodes:
             (STSObjects *) nodes
```

Set additional nodes that can be used.

This will replace any previously specified set of include nodes. Using this optional method adds valid nodes to the nodes of any node type explicitly set as a valid type. Should only be used to include specific small sets of nodes because it's less efficient than just using a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

run

```
- (void) run
```

Runs the algorithm in order to find the connected components.

This method can be called only once.

Implemented in STSCommunitiesSCD, and STSDisjointCommunityDetection.

References run.

Referenced by run.

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

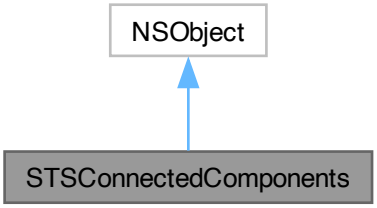
- Sparksee.h

3.9 STSConnectedComponents Class Reference

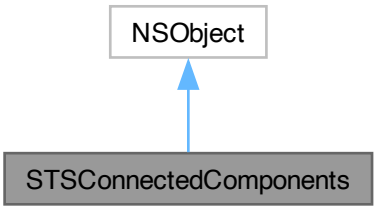
ConnectedComponents class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSConnectedComponents:



Collaboration diagram for STSConnectedComponents:



Instance Methods

- (id) - initWithS:materializedattribute:

  Creates a new instance of ConnectedComponents.
- (long long) - getConnectedComponent:

  Returns the connected component where the given node belongs to.
- (long long) - getCount

  Returns the number of connected components found in the graph.
- (STSObjects ∗) - getNodes:

  Returns the collection of nodes contained in the given connected component.
- (long long) - getSize:

  Returns the number of nodes contained in the given connected component.
- (void) - close

  Closes the ConnectedComponents instance.
- (BOOL) - **isClosed**

  Check if the ConnectedComponents instance is closed.

3.9.1 Detailed Description

ConnectedComponents class.

This class contains the results processed on a Connectivity algorithm.

These results contain information related to the connected components found. We must consider that each connected component has a number in order to identify it. These number identifiers are values from 0 to N-1, where N is the number of different connected components found.

When executing any implementation of the Connectivity, it is possible to indicate whether the results of the execution must be stored persistently using the class Connectivity setMaterializedAttribute method. In case the results are set to be materialized, users can retrieve this data whenever they want, even if the graph has been closed and opened again, just by creating a new instance of this class.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.9.2 Method Documentation

close

```
- (void) close
```

Closes the ConnectedComponents instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

getConnectedComponent:

```
- (long long) getConnectedComponent:
             (long long) idNode
```

Returns the connected component where the given node belongs to.

Parameters

idNode	[in] The node identifier for which the connected component identifier where it belongs will be returned.

Returns

The connected component identifier where the given node identifier belongs to.

getCount

```
- (long long) getCount
```

Returns the number of connected components found in the graph.

Returns

The number of connected components found in the graph.

References getCount.

Referenced by getCount.

getNodes:

```
- (STSObjects *) getNodes:
              (long long) idConnectedComponent
```

Returns the collection of nodes contained in the given connected component.

Parameters

idConnectedComponent	The connected component for which the collection of nodes contained in it will be returned.

Returns

The collection of node identifiers contained in the given connected component.

getSize:

```
- (long long) getSize:
              (long long) idConnectedComponent
```

Returns the number of nodes contained in the given connected component.

Parameters

idConnectedComponent	The connected component for which the number of nodes contained in it will be returned.

Returns

The number of nodes contained in the given connected component.

initWithS:materializedattribute:

```
- (id) initWithS:
              (STSSession *) s
             materializedattribute:(NSString *) materializedattribute
```

Creates a new instance of ConnectedComponents.

This constructor method can only be called when a previous execution of any implementation of the Connectivity class has materialized the results in a common attribute type for all the nodes in the graph. For further information about materializing the results processed on any Connectivity execution see the documentation of the Connectivity#SetMaterializedAttribute method.

Parameters

s	[in] Session to get the graph Graph on which the information will be retrieved just by getting the values contained in the given common attribute type for all the nodes in the graph and processing them.
materializedattribute	[in] The common attribute type for all the nodes in the graph where data will be retrieved in order to process the results related to the connected components found in the graph.

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

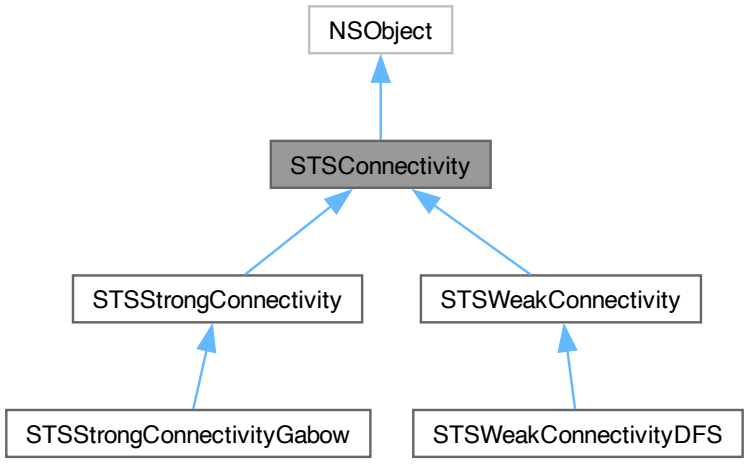
- Sparksee.h

3.10 STSConnectivity Class Reference

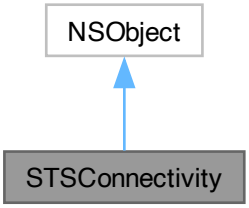
Connectivity class.

`#include <Sparksee.h>`

Inheritance diagram for STSConnectivity:



Collaboration diagram for STSConnectivity:



Instance Methods

- (void) - addNodeType:

  Allows connectivity through nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows connectivity through all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (STSConnectedComponents *) - getConnectedComponents

  Returns the results generated by the execution of the algorithm.
- (void) - run

  Runs the algorithm in order to find the connected components.
- (void) - setMaterializedAttribute:

  Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.
- (void) - close

  Closes the Connectivity instance.
- (BOOL) - **isClosed**

  Check if the Connectivity instance is closed.

3.10.1 Detailed Description

Connectivity class.

Any class implementing this abstract class can be used to solve a problem related to graph connectivity as finding the strongly connected components, finding the weakly connected components.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.10.2 Method Documentation

addNodeType:

```
- (void) addNodeType:
             (int) t
```

Allows connectivity through nodes of the given type.

Parameters

t	null

close

```
- (void) close
```

Closes the Connectivity instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getConnectedComponents

```
- (STSConnectedComponents *) getConnectedComponents
```

Returns the results generated by the execution of the algorithm.

These results contain information related to the connected components found as the number of different components, the set of nodes contained in each component or many other data.

Returns

Returns an instance of the class ConnectedComponents which contain information related to the connected components found.

References getConnectedComponents.

Referenced by getConnectedComponents.

run

```
- (void) run
```

Runs the algorithm in order to find the connected components.

This method can be called only once.

Implemented in STSStrongConnectivityGabow, and STSWeakConnectivityDFS.

References run.

Referenced by run.

setMaterializedAttribute:

```
- (void) setMaterializedAttribute:
             (NSString *) attributeName
```

Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.

Whenever the user wants to retrieve the results, even when the graph has been closed and opened again, it is only necessary to create a new instance of the class ConnectedComponents indicating the graph and the name of the common attribute type which stores the results. This instance will have all the information related to the connected components found in the moment of the execution of the algorithm that stored this data.

It is possible to run the algorithm without specifying this parameter in order to avoid materializing the results of the execution.

Parameters

attributeName	[in] The name of the common attribute type for all node types in the graph which will store persistently the results generated by the execution of the algorithm.

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

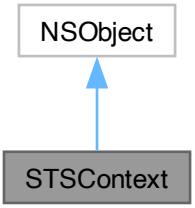
- Sparksee.h

3.11 STSContext Class Reference

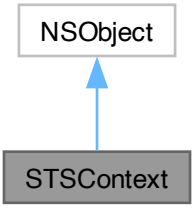
Context class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSContext:



Collaboration diagram for STSContext:



Instance Methods

- (void) - addEdgeType:d:

  Allows for traversing edges of the given type.
- (void) - addAllEdgeTypes:

  Allows for traversing all edge types of the graph.
- (void) - addNodeType:

  Allows for traversing nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows for traversing all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (STSObjects ∗) - compute

  Gets the resulting collection of nodes.
- (void) - setMaximumHops:include:

  Sets the maximum hops restriction.
- (id) - initWithSession:node:

  Creates a new instance.

- (void) - close

  Closes the Context instance.
- (BOOL) - **isClosed**

  Check if the Context instance is closed.

Class Methods

- (STSObjects *) + computeWithArguments:node:nodeTypes:edgeTypes:dir:maxhops:include:

  Helper method to easily compute a context from a node.

3.11.1 Detailed Description

Context class.

It provides a very similar functionality than the Traversal classes. The main difference is Context returns a resulting collection whereas Traversal provides an iterator behaviour.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.11.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
              (enum STSEdgesDirection) d
```

Allows for traversing all edge types of the graph.

Parameters

d	[in] Edge direction.

addEdgeType:d:

```
- (void) addEdgeType:
             (int) t
            d:(enum STSEdgesDirection) d
```

Allows for traversing edges of the given type.

Parameters

t	[in] Edge type.
d	[in] Edge direction.

addNodeType:

```
- (void) addNodeType:
             (int) t
```

Allows for traversing nodes of the given type.

Parameters

t	null

close

```
- (void) close
```

Closes the Context instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

compute

```
- (STSObjects *) compute
```

Gets the resulting collection of nodes.

Returns

The resulting collection of nodes.

References compute.

Referenced by compute.

computeWithArguments:node:nodeTypes:edgeTypes:dir:maxhops:include:

```
+ (STSObjects *) computeWithArguments:
              (STSSession *) session
              node:(long long) node
              nodeTypes:(STSTypeList *) nodeTypes
              edgeTypes:(STSTypeList *) edgeTypes
              dir:(enum STSEdgesDirection) dir
              maxhops:(int) maxhops
              include:(BOOL) include
```

Helper method to easily compute a context from a node.

Parameters

session	[in] Session to get the graph from and perform operation.
node	[in] Node to start traversal from.
nodeTypes	[in] Allowed node type list. NULL means all node types are allowed.
edgeTypes	[in] Allowed edge type list. NULL means all edge types are allowed.
dir	[in] Allowed direction for the allowed edge types.
maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
include	[in] If TRUE, the resulting collection will include those nodes at distance less or equal than the given one, otherwise it will just include those nodes at distance equal than the given one. This parameter just makes sense if maxhops is different from 0; in that case it includes all nodes no matters the distance.

Returns

Returns an Objects with the computed context of a node.

excludeEdges:

```
- (void) excludeEdges:
              (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

initWithSession:node:

```
- (id) initWithSession:
             (STSSession *) session
             node:(long long) node
```

Creates a new instance.

Parameters

session	[in] Session to get the graph from and perform operation.
node	[in] Node to start traversal from.

setMaximumHops:include:

```
- (void) setMaximumHops:
             (int) maxhops
             include:(BOOL) include
```

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored.

Parameters

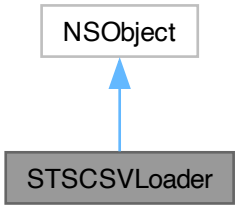
maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
include	[in] If TRUE, the resulting collection will include those nodes at distance less or equal than the given one, otherwise it will just include those nodes at distance equal than the given one. This parameter just makes sense if maxhops is different from 0; in that case it includes all nodes no matters the distance.

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

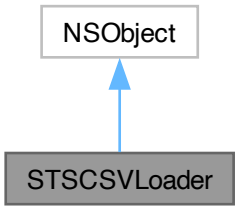
- Sparksee.h

3.12 STSCSVLoader Class Reference

Inheritance diagram for STSCSVLoader:



Collaboration diagram for STSCSVLoader:



Class Methods

- (void) + loadNodes:fileName:nodeType:separator:header:columns:attrNames:dataTypes:attrKinds:

  Loads nodes from a CSV file.
- (void) + loadEdges:fileName:edgeType:tailNodeType:headNodeType:tail:head:separator:directed:header:↩ onMissingTail:onMissingHead:columns:attrNames:dataTypes:attrKinds:

  Loads edges from a CSV file.

3.12.1 Method Documentation

loadEdges:fileName:edgeType:tailNodeType:headNodeType:tail:head:separator:directed:header:onMissingTail:onMissingH

```
+ (void) loadEdges:
             (STSGraph *) graph
             fileName:(NSString *) fileName
             edgeType:(NSString *) edgeType
```

```
              tailNodeType:(NSString *) tailNodeType
              headNodeType:(NSString *) headNodeType
              tail:(int) tail
              head:(int) head
              separator:(NSString *) separator
              directed:(BOOL) directed
              header:(BOOL) header
              onMissingTail:(enum STSMissingEndpoint) onMissingTail
              onMissingHead:(enum STSMissingEndpoint) onMissingHead
              columns:(NSArray *) columns
              attrNames:(NSArray *) attrNames
              dataTypes:(NSArray *) dataTypes
              attrKinds:(NSArray *) attrKinds
```

Loads edges from a CSV file.

nodeType[in] The type of the edge to load. If it does not exist, it creates it

Parameters

graph	[in] The graph to load the edges into
fileName	[in] The name of the file
edgeType	null
tailNodeType	[in] The type of the tail nodes. If it does not exist and onMissingTail is set to "Create", it creates it. Otherwise, an exception is thrown
headNodeType	[in] The type of the head nodes. It it does not exist and onMissingHead is set to "Create", it creates it. Otherwise, an exception is thrown
tail	[in] The tail column index. Default: 0
head	[in] The head column index. Default: 1
separator	[in] The column separator. Default: ","
directed	[in] True if this edge is directed or not. False otherwise. Default: True
header	[in] True if the CSV contains a header. False otherwise. Default: True
onMissingTail	[in] The policy to follow when a tail is missing. Default: "IsError"
onMissingHead	[in] The policy to follow when a head is missing. Default: "IsError"
columns	[in] The list of columns to load. tail and head columns must be in this list if this list is specified. If the list is empty, all columns are loaded. Default: empty
attrNames	[in] The attribute names. If this list is empty and hasHeader is set to true, the header values are used as attribute names. Default: empty
dataTypes	[in] The dataTypes of the attributes if this do not already exist. If this list is empty, the method tries to infer the data type from the first non-header row. Default: empty
attrKinds	[in] The attributeKinds of the attributes if these do not already exist. If this list is empty, the attributeKind of the created attributes are set to Basic. Default: empty

loadNodes:fileName:nodeType:separator:header:columns:attrNames:dataTypes:attrKinds:

```
+ (void) loadNodes:
              (STSGraph *) graph
              fileName:(NSString *) fileName
              nodeType:(NSString *) nodeType
              separator:(NSString *) separator
```

```
header:(BOOL) header
columns:(NSArray *) columns
attrNames:(NSArray *) attrNames
dataTypes:(NSArray *) dataTypes
attrKinds:(NSArray *) attrKinds
```

Loads nodes from a CSV file.

Parameters

graph	[in] The graph to load the nodes into
fileName	[in] The name of the file
nodeType	[in] The type of the node to load. If it does not exist, it creates it
separator	[in] The separator of the CSV file. Default: ","
header	[in] True if the CSV contains a header. False otherwise. Default: True
columns	[in] The list of columns to load. tail and head columns mus be in this list if this list is specified. Default: all columns
attrNames	[in] The attribute names. If this list is empty and hasHeader is set to true, the header values are used as attribute names. Default: empty
dataTypes	[in] The dataTypes of the attributes if this do not already exist. Default: empty If this list is empty, the method tries to infer the data type from the first non-header row
attrKinds	[in] The attributeKinds of the attributes if these do not already exist. Default: empty If this list is empty, the attributeKind of the created attributes are set to Basic. Default: empty

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

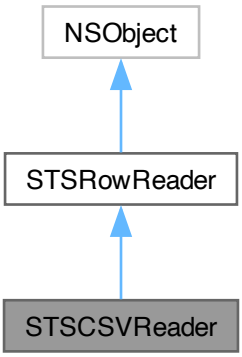
- Sparksee.h

3.13 STSCSVReader Class Reference

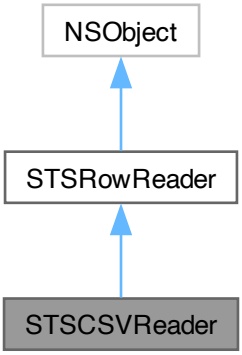
CSVReader interface.

`#include <Sparksee.h>`

Inheritance diagram for STSCSVReader:



Collaboration diagram for STSCSVReader:



Instance Methods

- (id) - **init**

  Constructs CSVReader.
- (void) - setSeparator:

  Sets the character used to separate fields in the file.
- (void) - setQuotes:

  Sets the character used to quote fields.
- (void) - setMultilines:

  Allows the use of fields with more than one line.
- (void) - **setSingleLine**

  Only allows single line fields.
- (void) - setStartLine:

  Sets the number of lines to be skiped from the beginning.
- (void) - setNumLines:

  Used to limit the number of lines that will be read.
- (void) - setLocale:

  Sets the locale that will be used to read the file.
- (void) - open:

  Opens the source file path.
- (BOOL) - reset

  Moves the reader to the beginning.
- (BOOL) - read:

  Reads the next row as a string array.
- (int) - getRow

  The row number for the current row.
- (void) - close

  Closes the reader.

3.13.1 Detailed Description

CSVReader interface.

A very simple CSV reader.

It works as any other RowReader, but open must be called once before the first read operation.

Using the format RFC 4180.

Except: leading and trailing spaces, adjacent to CSV separator character, are trimmed.

You can use your own separators and quote characters. By default the separator is the comma (,) and the quote character is the double quotes ("). Fields with multiple lines can be allowed (and the maximum lines specified), but the default is a single line. The locale string can be used to set the language, country and the file encoding. The format must be "[language_territory][.codeset]". But only the file encoding is being used in the current version. The languages supported are: "en_US", "es_ES" and "ca_ES".
 The file encodings supported are: "utf8" and "iso88591". For example: To don't change the default locales, use an empty string: "".
 To read a file in utf8 with the default language settings use ".utf8".
 To read a file in iso88591 with English language use: "en_US.iso88591".

Check out the 'Data import' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.13.2 Method Documentation

close

```
- (void) close
```

Closes the reader.

Exceptions

System.IO.IOException	If the close fails.

Implements STSRowReader.

References close.

Referenced by close.

getRow

```
- (int) getRow
```

The row number for the current row.

Returns

The current row number; 0 if there is no current row.

Exceptions

System.IO.IOException	If it fails.

Implements STSRowReader.

References getRow.

Referenced by getRow.

open:

```
- (void) open:
             (NSString *) filePath
```

Opens the source file path.

File can be optionally compressed in GZIP format.

Parameters

filePath	[in] CSV file path.

Exceptions

System.IO.IOException	If bad things happen opening the file.

read:

```
- (BOOL) read:
             (STSStringList *) row
```

Reads the next row as a string array.

Parameters

row	[out] A string list with each comma-separated element as a separate entry.

Returns

Returns true if a row had been read or false otherwise.

Exceptions

System.IO.IOException	If bad things happen during the read.

Implements STSRowReader.

reset

```
- (BOOL) reset
```

Moves the reader to the beginning.

Restarts the reader.

Returns

true if the reader can be restarted, false otherwise.

Exceptions

System.IO.IOException	If bad things happen during the restart.

Implements STSRowReader.

References reset.

Referenced by reset.

setLocale:

```
- (void) setLocale:
             (NSString *) localeStr
```

Sets the locale that will be used to read the file.

Parameters

localeStr	[in] The locale string for the file encoding.

setMultilines:

```
- (void) setMultilines:
             (int) numExtralines
```

Allows the use of fields with more than one line.

Parameters

numExtralines	[in] Maximum number of extra lines for each column (0==unlimited, N==N+1 total rows).

setNumLines:

```
- (void) setNumLines:
             (int) numLines
```

Used to limit the number of lines that will be read.

Parameters

numLines	[in] The maximum number of lines to read (0 == unlimited)

setQuotes:

```
- (void) setQuotes:
             (NSString *) quotes
```

Sets the character used to quote fields.

Parameters

quotes	[in] Quote character.

Exceptions

System.ApplicationException	For any issue

setSeparator:

```
- (void) setSeparator:
             (NSString *) sep
```

Sets the character used to separate fields in the file.

Parameters

sep	[in] Separator character.

Exceptions

System.ApplicationException	For any issue

setStartLine:

```
- (void) setStartLine:
             (int) startLine
```

Sets the number of lines to be skiped from the beginning.

Parameters

startLine	[in] The line number to skip for start reading

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

- Sparksee.h

3.14 STSCSVWriter Class Reference

CSVWriter interface.

```
#include <Sparksee.h>
```

Inheritance diagram for STSCSVWriter:

Collaboration diagram for STSCSVWriter:

Instance Methods

- (id) - **init**

  Creates a new instance.
- (void) - setSeparator:

  Sets the character used to separate fields in the file.
- (void) - setQuotes:

  Sets the character used to quote fields.
- (void) - setAutoQuotes:

  Sets on/off the automatic quote mode.
- (void) - setForcedQuotes:

  Disables the automatic quote mode and forces to be quoted those positions set to TRUE in the given vector.
- (void) - setLocale:

  Sets the locale that will be used to write the file.
- (void) - open:

  Opens the output file path.
- (void) - write:

  Writes the next row.
- (void) - close

  Closes the writer.

3.14.1 Detailed Description

CSVWriter interface.

A very simple CSV writer implementing RowWriter.

It works as any other RowWriter, but open must be called once before the first write operation.

It uses the format RFC 4180: http://tools.ietf.org/html/rfc4180

You can use your own separators and quote characters. By default the separator is the comma (,) and the quote character is the double quotes (") and autoquote is enabled.

See the CSVReader locale documentation or the SPARKSEE User Manual.

Check out the 'Data export' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.14.2 Method Documentation

close

```
- (void) close
```

Closes the writer.

ErrorFor any issue

Exceptions

System.ApplicationException	null
System.IO.IOException	If the close fails.

Implements STSRowWriter.

References close.

Referenced by close.

open:

```
- (void) open:
              (NSString *) f
```

Opens the output file path.

Parameters

f	[in] Output file path.

Exceptions

System.IO.IOException	If bad things happen opening the file.

setAutoQuotes:

```
- (void) setAutoQuotes:
             (BOOL) autoquotes
```

Sets on/off the automatic quote mode.

If there are forced quotes, setting autoquotes on will clear them. If the autoquotes is set to off and no forced quotes are provided, there will not be any quote.

Parameters

autoquotes	[in] If TRUE it enables the automatic quote mode, if FALSE it disables it.

setForcedQuotes:

```
- (void) setForcedQuotes:
             (STSBooleanList *) forcequotes
```

Disables the automatic quote mode and forces to be quoted those positions set to TRUE in the given vector.

Parameters

forcequotes	[in] A booleanList with the position for each column that must be quoted set to true.

setLocale:

```
- (void) setLocale:
             (NSString *) localeStr
```

Sets the locale that will be used to write the file.

Parameters

localeStr	[in] The locale string for the file encoding.

setQuotes:

```
- (void) setQuotes:
             (NSString *) quotes
```

Sets the character used to quote fields.

Parameters

quotes	[in] Quote character.

Exceptions

System.ApplicationException	For any issue

setSeparator:

```
- (void) setSeparator:
             (NSString *) sep
```

Sets the character used to separate fields in the file.

Parameters

sep	[in] Separator character.

Exceptions

System.ApplicationException	For any issue

write:

```
- (void) write:
             (STSStringList *) row
```

Writes the next row.

ErrorFor any issue

Parameters

row	[in] Row of data.

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen during the write.

Implements STSRowWriter.

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

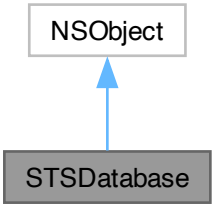
- Sparksee.h

3.15 STSDatabase Class Reference

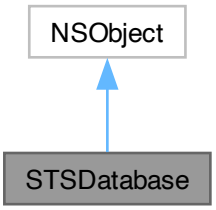
Database class.

`#include <Sparksee.h>`

Inheritance diagram for STSDatabase:



Collaboration diagram for STSDatabase:



Instance Methods

- (NSString ∗) - getAlias

  Gets the alias of the Database.
- (NSString ∗) - getPath

  Gets the path of the Database.
- (STSSession ∗) - createSession

  Creates a new Session.
- (void) - **enableRollback**

  Enables the rollback mechanism.
- (void) - **disableRollback**

  Disables the rollback mechanism.
- (void) - getStatistics:

  Gets Database statistics.

- (int) - getCacheMaxSize

  Gets the cache maximum size (in MB).
- (void) - setCacheMaxSize:

  Sets the cache maximum size (in MB).
- (void) - **fixCurrentCacheMaxSize**

  Sets the cache maximum size to the current cache size in use.
- (void) - dumpSchema:

  Dump the database schema to the given file using the Sparksee scripting format.
- (void) - redoPrecommitted:

  Redo a pending precommitted transaction recovered.
- (void) - addQueryStreamManager:

  Add a QueryStreamManager to the database.
- (void) - close

  Closes the Database instance.
- (BOOL) - **isClosed**

  Check if the Database instance is closed.

3.15.1 Detailed Description

Database class.

All the data of the Database is stored into a persistent file which just can be created or open through a Sparksee instance.

Also, all the manipulation of a Database must be done by means of a Session which can be initiated from a Database instance.

Multiple Databases do not share the memory, that is there is no negotiation among them. In those cases, memory must be prefixed for each Database. To do that, use the SPARKSEEConfig.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.15.2 Method Documentation

addQueryStreamManager:

```
- (void) addQueryStreamManager:
            (STSQueryStreamManager *) queryStreamManager
```

Add a QueryStreamManager to the database.

The user is responsible of keeping the manager alive for the duration of the Database. The Managers will be used in reverse order of addition while searching for a matching Stream.

Parameters

queryStreamManager	[In] A reference to a QueryStreamManager

close

```
- (void) close
```

Closes the Database instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

createSession

```
- (STSSession *) createSession
```

Creates a new Session.

Returns

A new Session instance.

References createSession.

Referenced by createSession.

dumpSchema:

```
- (void) dumpSchema:
             (NSString *) filePath
```

Dump the database schema to the given file using the Sparksee scripting format.

Parameters

filePath	null

getAlias

```
- (NSString *) getAlias
```

Gets the alias of the Database.

Returns

The alias of the Database.

References getAlias.

Referenced by getAlias.

getCacheMaxSize

```
- (int) getCacheMaxSize
```

Gets the cache maximum size (in MB).

Returns

Returns the current cache max size.

References getCacheMaxSize.

Referenced by getCacheMaxSize.

getPath

```
- (NSString *) getPath
```

Gets the path of the Database.

Returns

The path of the Database.

References getPath.

Referenced by getPath.

getStatistics:

```
- (void) getStatistics:
            (STSDatabaseStatistics *) stats
```

Gets Database statistics.

Parameters

stats	[out] The DatabaseStatistics instance.

redoPrecommitted:

```
- (void) redoPrecommitted:
             (long long) txId
```

Redo a pending precommitted transaction recovered.

YOU SHOULD NOT USE THIS METHOD. This method only exists for a specific service.

Parameters

txId	null

setCacheMaxSize:

```
- (void) setCacheMaxSize:
             (int) megaBytes
```

Sets the cache maximum size (in MB).

0 means unlimited which is all the physical memory of the computer minus a small margin.

Parameters

megaBytes	[in] The new cache max size.

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

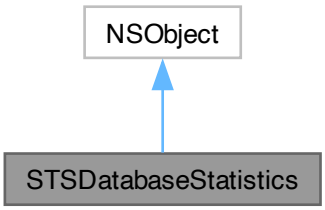
- Sparksee.h

3.16 STSDatabaseStatistics Class Reference

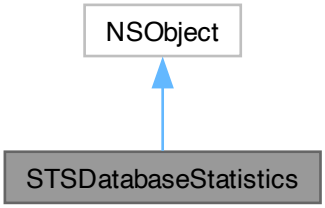
Database statistics.

```
#include <Sparksee.h>
```

Inheritance diagram for STSDatabaseStatistics:



Collaboration diagram for STSDatabaseStatistics:



Instance Methods

- (long long) - getRead
  Gets total read data in KBytes.
- (long long) - getWrite
  Gets total written data in KBytes.
- (long long) - getData
  Gets database size in KBytes.
- (long long) - getCache
  Gets cache size in KBytes.
- (long long) - getTemp
  Gets temporary storage file size in KBytes.
- (long long) - getSessions
  Gets the number of sessions.

3.16.1 Detailed Description

Database statistics.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.16.2 Method Documentation

getCache

```
- (long long) getCache
```

Gets cache size in KBytes.

Returns

Cache size in KBytes.

References getCache.

Referenced by getCache.

getData

```
- (long long) getData
```

Gets database size in KBytes.

Returns

Database size in KBytes.

References getData.

Referenced by getData.

getRead

```
- (long long) getRead
```

Gets total read data in KBytes.

Returns

Total read data in KBytes.

References getRead.

Referenced by getRead.

getSessions

```
- (long long) getSessions
```

Gets the number of sessions.

Returns

The number of sessions.

References getSessions.

Referenced by getSessions.

getTemp

```
- (long long) getTemp
```

Gets temporary storage file size in KBytes.

Returns

Temporary storage file size in KBytes.

References getTemp.

Referenced by getTemp.

getWrite

```
- (long long) getWrite
```

Gets total written data in KBytes.

Returns

Total read written in KBytes.

References getWrite.

Referenced by getWrite.

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

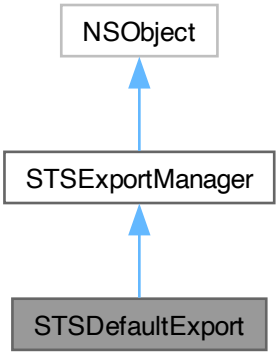
- Sparksee.h

3.17 STSDefaultExport Class Reference

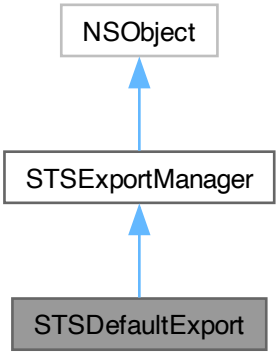
Default implementation for ExportManager class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSDefaultExport:



Collaboration diagram for STSDefaultExport:



Instance Methods

- (id) - **init**

  Creates a new instance.
- (BOOL) - subgraphNodes:

  Limit the exported nodes to the nodes in the given node set.
- (BOOL) - subgraphEdges:

  Limit the exported edges to the edges in the given edges set.
- (void) - setTitleWithAttributes:

  Include additional information including all the node/edge attributes as Title instead of the default title text (just the type name).
- (void) - setTypeColorRGB:color:

  Set the color for a specific node/edge type.
- (void) - setNodeTypeImage:image:

  Set an image for all the nodes of a specific node type.
- (void) - setNodeTypeImageAttribute:imageAttr:

  Set the attribute that stores the image for a specific node type.
- (void) - setTypeLabelAttribute:labelAttr:

  Set the attribute that stores the label for a specific node/edge type.
- (void) - setNodeTypeShape:shape:

  Set the shape for all nodes of this type.
- (BOOL) - setMaxTextCharacters:

  Sets the maximum number of Text Attribute characters to show.
- (BOOL) - setMaxArrayElements:

  Sets the maximum number of elements of an Array Attribute to show.
- (void) - prepare:

  Default implementation of the ExportManager class method Prepare.
- (void) - prepareSwigExplicitSTSDefaultExport:

  Default implementation of the ExportManager class method Prepare.
- (void) - **close**

  Default implementation of the ExportManager class method Release.

- (void) - **closeSwigExplicitSTSDefaultExport**

  Default implementation of the ExportManager class method Release.
- (BOOL) - getGraph:

  Default implementation of the ExportManager class method GetGraph.
- (BOOL) - getGraphSwigExplicitSTSDefaultExport:

  Default implementation of the ExportManager class method GetGraph.
- (BOOL) - getNodeType:nodeExport:

  Default implementation of the ExportManager class method GetNodeType.
- (BOOL) - getNodeTypeSwigExplicitSTSDefaultExport:nodeExport:

  Default implementation of the ExportManager class method GetNodeType.
- (BOOL) - getEdgeType:edgeExport:

  Default implementation of the ExportManager class method GetEdgeType.
- (BOOL) - getEdgeTypeSwigExplicitSTSDefaultExport:edgeExport:

  Default implementation of the ExportManager class method GetEdgeType.
- (BOOL) - getNode:nodeExport:

  Default implementation of the ExportManager class method GetNode.
- (BOOL) - getNodeSwigExplicitSTSDefaultExport:nodeExport:

  Default implementation of the ExportManager class method GetNode.
- (BOOL) - getEdge:edgeExport:

  Default implementation of the ExportManager class method GetEdge.
- (BOOL) - getEdgeSwigExplicitSTSDefaultExport:edgeExport:

  Default implementation of the ExportManager class method GetEdge.
- (BOOL) - enableType:

  Default implementation of the ExportManager class method EnableType.
- (BOOL) - enableTypeSwigExplicitSTSDefaultExport:

  Default implementation of the ExportManager class method EnableType.
- (BOOL) - exportAll

  Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.
- (BOOL) - exportAllSwigExplicitSTSDefaultExport

  Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.
- (long long) - nextNode

  Gets the next node to export or InvalidOID.
- (long long) - nextNodeSwigExplicitSTSDefaultExport

  Gets the next node to export or InvalidOID.
- (long long) - nextEdge

  Gets the next edge to export or InvalidOID.
- (long long) - nextEdgeSwigExplicitSTSDefaultExport

  Gets the next edge to export or InvalidOID.
- (void) - setTypeColor:red:green:blue:alpha:

  Set Set the color for a specific node/edge type with separated RGBA components.
- (BOOL) - exportAllSwigExplicitSTSExportManager

  Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.
- (long long) - nextNodeSwigExplicitSTSExportManager

  Gets the next node to export or InvalidOID.
- (long long) - nextEdgeSwigExplicitSTSExportManager

  Gets the next edge to export or InvalidOID.

3.17.1 Detailed Description

Default implementation for ExportManager class.

It uses the default values from GraphExport, NodeExport and EdgeExport to export all node and edge types.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.17.2 Method Documentation

enableType:

```
- (BOOL) enableType:
             (int) type
```

Default implementation of the ExportManager class method EnableType.

This enables all node and edge types to be exported.

Parameters

type	[in] The type to enable.

Returns

TRUE.

Implements STSExportManager.

enableTypeSwigExplicitSTSDefaultExport:

```
- (BOOL) enableTypeSwigExplicitSTSDefaultExport:
             (int) type
```

Default implementation of the ExportManager class method EnableType.

This enables all node and edge types to be exported.

Parameters

type	[in] The type to enable.

Returns

TRUE.

exportAll

```
- (BOOL) exportAll
```

Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.

Or false to use the new Next() method of this class to get the only nodes/edges that should be exported. It will return true to export all the graph unless a subgraph had been selected by calling the SubgraphNodes or SubgraphEdges methods.

Returns

bool_t Returns true to export all the graph or false otherwise.

Implements STSExportManager.

References exportAll.

Referenced by exportAll.

exportAllSwigExplicitSTSDefaultExport

```
- (BOOL) exportAllSwigExplicitSTSDefaultExport
```

Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.

Or false to use the new Next() method of this class to get the only nodes/edges that should be exported. It will return true to export all the graph unless a subgraph had been selected by calling the SubgraphNodes or SubgraphEdges methods.

Returns

bool_t Returns true to export all the graph or false otherwise.

References exportAllSwigExplicitSTSDefaultExport.

Referenced by exportAllSwigExplicitSTSDefaultExport.

exportAllSwigExplicitSTSExportManager

```
- (BOOL) exportAllSwigExplicitSTSExportManager
```

Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.

Or false to use the new NextNode() and NextEdge() methods of this class to get the only nodes/edges that should be exported.

Returns

bool_t Returns true to export all the graph or false otherwise.

References exportAllSwigExplicitSTSExportManager.

Referenced by exportAllSwigExplicitSTSExportManager.

getEdge:edgeExport:

```
- (BOOL) getEdge:
              (long long) edge
              edgeExport:(STSEdgeExport *) edgeExport
```

Default implementation of the ExportManager class method GetEdge.

This sets the default EdgeExport values and sets the OID as the label. Also, it exports the edge as directed just if the edge is directed.

Parameters

edge	[in] An edge.
edgeExport	[out] The EdgeExport that will store the information.

Returns

TRUE.

Implements STSExportManager.

getEdgeSwigExplicitSTSDefaultExport:edgeExport:

```
- (BOOL) getEdgeSwigExplicitSTSDefaultExport:
              (long long) edge
              edgeExport:(STSEdgeExport *) edgeExport
```

Default implementation of the ExportManager class method GetEdge.

This sets the default EdgeExport values and sets the OID as the label. Also, it exports the edge as directed just if the edge is directed.

Parameters

edge	[in] An edge.
edgeExport	[out] The EdgeExport that will store the information.

Returns

TRUE.

getEdgeType:edgeExport:

```
- (BOOL) getEdgeType:
             (int) type
             edgeExport:(STSEdgeExport *) edgeExport
```

Default implementation of the ExportManager class method GetEdgeType.

This sets de default EdgeExport values.

Parameters

type	[in] An edge type.
edgeExport	[out] The EdgeExport that will store the information.

Returns

TRUE.

Implements STSExportManager.

getEdgeTypeSwigExplicitSTSDefaultExport:edgeExport:

```
- (BOOL) getEdgeTypeSwigExplicitSTSDefaultExport:
             (int) type
             edgeExport:(STSEdgeExport *) edgeExport
```

Default implementation of the ExportManager class method GetEdgeType.

This sets de default EdgeExport values.

Parameters

type	[in] An edge type.
edgeExport	[out] The EdgeExport that will store the information.

Returns

TRUE.

getGraph:

```
- (BOOL) getGraph:
             (STSGraphExport *) graphExport
```

Default implementation of the ExportManager class method GetGraph.

This sets the default GraphExport values including the default background color and the database alias as the label. The default GraphExport label could be empty (because the database did not have an alias) but this method will add " graph" or " subgraph" to the label anyway to easily know if the hole graph was exported or just some nodes/edges.

Parameters

graphExport	[out] The GraphExport that will store the information.

Returns

TRUE.

Implements STSExportManager.

getGraphSwigExplicitSTSDefaultExport:

```
- (BOOL) getGraphSwigExplicitSTSDefaultExport:
             (STSGraphExport *) graphExport
```

Default implementation of the ExportManager class method GetGraph.

This sets the default GraphExport values including the default background color and the database alias as the label. The default GraphExport label could be empty (because the database did not have an alias) but this method will add " graph" or " subgraph" to the label anyway to easily know if the hole graph was exported or just some nodes/edges.

Parameters

graphExport	[out] The GraphExport that will store the information.

Returns

TRUE.

getNode:nodeExport:

```
- (BOOL) getNode:
              (long long) node
              nodeExport:(STSNodeExport *) nodeExport
```

Default implementation of the ExportManager class method GetNode.

This sets the default NodeExport values and sets the OID as the label.

Parameters

node	[in] A node.
nodeExport	[out] The NodeExport that will store the information.

Returns

TRUE.

Implements STSExportManager.

getNodeSwigExplicitSTSDefaultExport:nodeExport:

```
- (BOOL) getNodeSwigExplicitSTSDefaultExport:
              (long long) node
              nodeExport:(STSNodeExport *) nodeExport
```

Default implementation of the ExportManager class method GetNode.

This sets the default NodeExport values and sets the OID as the label.

Parameters

node	[in] A node.
nodeExport	[out] The NodeExport that will store the information.

Returns

TRUE.

getNodeType:nodeExport:

```
- (BOOL) getNodeType:
             (int) type
             nodeExport:(STSNodeExport *) nodeExport
```

Default implementation of the ExportManager class method GetNodeType.

This sets de default NodeExport values.

Parameters

type	[in] A node type.
nodeExport	[out] The NodeExport that will store the information.

Returns

TRUE.

Implements STSExportManager.

getNodeTypeSwigExplicitSTSDefaultExport:nodeExport:

```
- (BOOL) getNodeTypeSwigExplicitSTSDefaultExport:
             (int) type
             nodeExport:(STSNodeExport *) nodeExport
```

Default implementation of the ExportManager class method GetNodeType.

This sets de default NodeExport values.

Parameters

type	[in] A node type.
nodeExport	[out] The NodeExport that will store the information.

Returns

TRUE.

nextEdge

```
- (long long) nextEdge
```

Gets the next edge to export or InvalidOID.

It will return valid edge OIDs to export and InvalidOID when there is not any more edges to export. ExportAll returns false.

Returns

oid_t Return an edge OID or InvalidOID

Implements STSExportManager.

References nextEdge.

Referenced by nextEdge.

nextEdgeSwigExplicitSTSDefaultExport

```
- (long long) nextEdgeSwigExplicitSTSDefaultExport
```

Gets the next edge to export or InvalidOID.

It will return valid edge OIDs to export and InvalidOID when there is not any more edges to export. ExportAll returns false.

Returns

oid_t Return an edge OID or InvalidOID

References nextEdgeSwigExplicitSTSDefaultExport.

Referenced by nextEdgeSwigExplicitSTSDefaultExport.

nextEdgeSwigExplicitSTSExportManager

```
- (long long) nextEdgeSwigExplicitSTSExportManager
```

Gets the next edge to export or InvalidOID.

It will return valid edge OIDs to export and InvalidOID when there is not any more edges to export. ExportAll returns false.

Returns

oid_t Return an edge OID or InvalidOID

References nextEdgeSwigExplicitSTSExportManager.

Referenced by nextEdgeSwigExplicitSTSExportManager.

nextNode

```
- (long long) nextNode
```

Gets the next node to export or InvalidOID.

It will return valid node OIDs to export and InvalidOID when there is not any more nodes to export. ExportAll returns false.

Returns

oid_t Return a node OID or InvalidOID

Implements STSExportManager.

References nextNode.

Referenced by nextNode.

nextNodeSwigExplicitSTSDefaultExport

```
- (long long) nextNodeSwigExplicitSTSDefaultExport
```

Gets the next node to export or InvalidOID.

It will return valid node OIDs to export and InvalidOID when there is not any more nodes to export. ExportAll returns false.

Returns

oid_t Return a node OID or InvalidOID

References nextNodeSwigExplicitSTSDefaultExport.

Referenced by nextNodeSwigExplicitSTSDefaultExport.

nextNodeSwigExplicitSTSExportManager

```
- (long long) nextNodeSwigExplicitSTSExportManager
```

Gets the next node to export or InvalidOID.

It will return valid node OIDs to export and InvalidOID when there is not any more nodes to export. ExportAll returns false.

Returns

oid_t Return a node OID or InvalidOID

References nextNodeSwigExplicitSTSExportManager.

Referenced by nextNodeSwigExplicitSTSExportManager.

prepare:

```
- (void) prepare:
             (STSGraph *) graph
```

Default implementation of the ExportManager class method Prepare.

Parameters

graph	null

Implements STSExportManager.

prepareSwigExplicitSTSDefaultExport:

```
- (void) prepareSwigExplicitSTSDefaultExport:
             (STSGraph *) graph
```

Default implementation of the ExportManager class method Prepare.

Parameters

graph	null

setMaxArrayElements:

```
- (BOOL) setMaxArrayElements:
             (int) size
```

Sets the maximum number of elements of an Array Attribute to show.

If an Array Attribute has more than the given size elements, only the first ones will be used and "..." will be added.

Parameters

size	null

Returns

Returns True if successful or False if the give value is not valid.

setMaxTextCharacters:

```
- (BOOL) setMaxTextCharacters:
             (int) size
```

Sets the maximum number of Text Attribute characters to show.

If a Text Attribute has more than the given size characters it will be truncated and "..." will be added.

Parameters

size	null

Returns

Returns True if successful or False if the give value is not valid.

setNodeTypeImage:image:

```
- (void) setNodeTypeImage:
             (int) nodeType
             image:(NSString *) image
```

Set an image for all the nodes of a specific node type.

Parameters

nodeType	[in] A valid node type.
image	[in] A path or URL of the image to use for all nodes of this type.

setNodeTypeImageAttribute:imageAttr:

```
- (void) setNodeTypeImageAttribute:
             (int) nodeType
             imageAttr:(int) imageAttr
```

Set the attribute that stores the image for a specific node type.

Parameters

nodeType	[in] A valid node type.
imageAttr	[in] The attribute to use for nodes of this type to get the image url/path.

setNodeTypeShape:shape:

```
- (void) setNodeTypeShape:
             (int) nodeType
             shape:(enum STSNodeShape) shape
```

Set the shape for all nodes of this type.

Parameters

nodeType	[in] A valid node type.
shape	[in] The shape for all the nodes of the given type.

setTitleWithAttributes:

```
- (void) setTitleWithAttributes:
             (BOOL) encoded
```

Include additional information including all the node/edge attributes as Title instead of the default title text (just the type name).

Parameters

encoded	[in] If true the tooltip data will be added as a base64 encoded JSON instead of very basic html text format for easy parsing of the additional data.

setTypeColor:red:green:blue:alpha:

```
- (void) setTypeColor:
             (int) type
             red:(double) red
             green:(double) green
             blue:(double) blue
             alpha:(double) alpha
```

Set Set the color for a specific node/edge type with separated RGBA components.

Parameters

red	[in] The red color component ([0..1]).
green	[in] The green color component ([0..1]).
blue	[in] The blue color component ([0..1]).
alpha	[in] The alpha component ([0..1]).

setTypeColorRGB:color:

```
- (void) setTypeColorRGB:
             (int) type
             color:(int) color
```

Set the color for a specific node/edge type.

Parameters

type	[in] A valid node/edge type.
color	[in] The main color for the type. The label color will be set as a complementary color from the given color to guarantee that it will contrast.

setTypeLabelAttribute:labelAttr:

```
- (void) setTypeLabelAttribute:
             (int) type
             labelAttr:(int) labelAttr
```

Set the attribute that stores the label for a specific node/edge type.

Parameters

type	[in] A valid node/edge type.
labelAttr	[in] The attribute to use for nodes/edges of this type to get the label. If a valid label can be obttained from the attribute it will be used instead of the default OID.

subgraphEdges:

```
- (BOOL) subgraphEdges:
             (STSObjects *) edges
```

Limit the exported edges to the edges in the given edges set.

If only the subgraph edges are specified, the exported nodes will be calculated as all the nodes present in the selected edges.

Parameters

edges	[in] Subgraph edges to export. Must be kept alive by the user until the export is finished. The given Objects set can also contain nodes, but will be ignored. This is useful to use a signle Objects set to limit both subgraph nodes and edges. But if an Objects containing both nodes and edges is used, then both SubgraphNodes and SubgraphEdges methods must be called.

Returns

Returns true if successful or false otherwise.

subgraphNodes:

```
- (BOOL) subgraphNodes:
             (STSObjects *) nodes
```

Limit the exported nodes to the nodes in the given node set.

If only the subgraph nodes are specified, the exported edges will be calculated as all the edges between the selected nodes.

Parameters

nodes	[in] Subgraph nodes to export. Must be kept alive by the user until the export is finished. The given Objects set can also contain edges, but will be ignored. This is useful to use a signle Objects set to limit both subgraph nodes and edges. But if an Objects containing both nodes and edges is used, then both SubgraphNodes and SubgraphEdges methods must be called.

Returns

Returns true if successful or false otherwise.

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

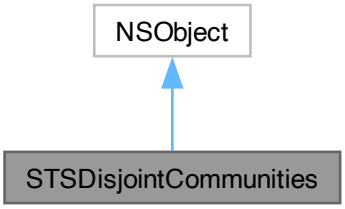
- Sparksee.h

3.18 STSDisjointCommunities Class Reference

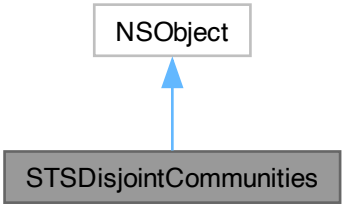
DisjointCommunities class.

`#include <Sparksee.h>`

Inheritance diagram for STSDisjointCommunities:



Collaboration diagram for STSDisjointCommunities:



Instance Methods

- (id) - initWithSession:materializedattribute:
  Creates a new instance of DisjointCommunities.
- (long long) - getCommunity:
  Returns the disjoint community where the given node belongs to.
- (long long) - getCount
  Returns the number of communities found in the graph.
- (STSObjects *) - getNodes:
  Returns the collection of nodes contained in the given community.
- (long long) - getSize:
  Returns the number of nodes contained in the given community.
- (void) - close
  Closes the DisjointCommunities instance.
- (BOOL) - **isClosed**
  Check if the DisjointCommunities instance is closed.

3.18.1 Detailed Description

DisjointCommunities class.

This class contains the results processed on a DisjointCommunityDetection algorithm.

These results contain information related to the communities found. We must consider that each community has a number in order to identify it. These number identifiers are values from 0 to N-1, where N is the number of different communities found.

When executing any implementation of the DisjointCommunityDetection, it is possible to indicate whether the results of the execution must be stored persistently using the class DisjointCommunityDetection setMaterializedAttribute method. In case the results are set to be materialized, users can retrieve this data whenever they want, even if the graph has been closed and opened again, just by creating a new instance of this class.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.18.2 Method Documentation

close

```
- (void) close
```

Closes the DisjointCommunities instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

getCommunity:

```
- (long long) getCommunity:
             (long long) idNode
```

Returns the disjoint community where the given node belongs to.

Parameters

idNode	[in] The node identifier for which the disjoint community identifier where it belongs will be returned.

Returns

The disjoint community identifier where the given node identifier belongs to.

getCount

```
- (long long) getCount
```

Returns the number of communities found in the graph.

Returns

The number of communities found in the graph.

References getCount.

Referenced by getCount.

getNodes:

```
- (STSObjects *) getNodes:
             (long long) idCommunity
```

Returns the collection of nodes contained in the given community.

Parameters

idCommunity	The community for which the collection of nodes contained in it will be returned.

Returns

The collection of node identifiers contained in the given community.

getSize:

```
- (long long) getSize:
             (long long) idCommunity
```

Returns the number of nodes contained in the given community.

Parameters

idCommunity	The community for which the number of nodes contained in it will be returned.

Returns

The number of nodes contained in the given community.

initWithSession:materializedattribute:

```
- (id) initWithSession:
             (STSSession *) session
             materializedattribute:(NSString *) materializedattribute
```

Creates a new instance of DisjointCommunities.

This constructor method can only be called when a previous execution of any implementation of the Disjoint↩ CommunityDetection class has materialized the results in a common attribute type for all the nodes in the graph. For further information about materializing the results processed on any DisjointCommunityDetection execution see the documentation of the DisjointCommunityDetection#SetMaterializedAttribute method.

Parameters

session	[in] Session to get the graph Graph on which the information will be retrieved just by getting the values contained in the given common attribute type for all the nodes in the graph and processing them.
materializedattribute	[in] The common attribute type for all the nodes in the graph where data will be retrieved in order to process the results related to the communities found in the graph.

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

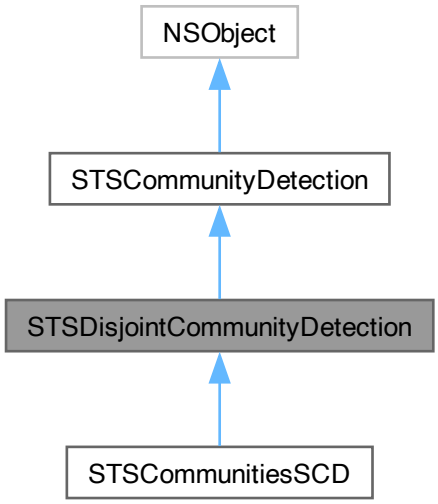
- Sparksee.h

3.19 STSDisjointCommunityDetection Class Reference

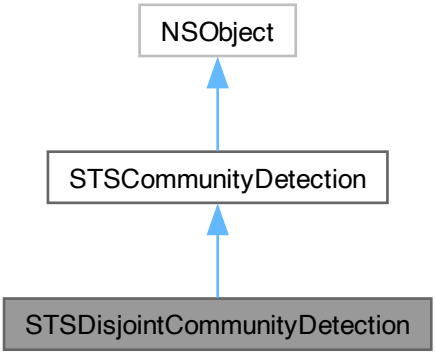
DisjointCommunityDetection class.

`#include <Sparksee.h>`

Inheritance diagram for STSDisjointCommunityDetection:



Collaboration diagram for STSDisjointCommunityDetection:



Instance Methods

- (void) - addEdgeType:

  Allows connectivity through edges of the given type.
- (void) - addAllEdgeTypes

  Allows connectivity through all edge types of the graph.
- (STSDisjointCommunities ∗) - getCommunities

  Returns the results generated by the execution of the algorithm.
- (void) - run

  Runs the algorithm in order to find the communities.
- (void) - setMaterializedAttribute:

  Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the disjoint communities found while executing this algorithm.
- (void) - addNodeType:

  Allows connectivity through nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows connectivity through all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (void) - includeNodes:

  Set additional nodes that can be used.
- (void) - includeEdges:

  Set additional edges that can be used.
- (void) - close

  Closes the CommunityDetection instance.
- (BOOL) - **isClosed**

  Check if the CommunityDetection instance is closed.

3.19.1 Detailed Description

DisjointCommunityDetection class.

Any class implementing this abstract class can be used to solve a problem related to graph connectivity as finding the strongly connected components, finding the weakly connected components.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.19.2 Method Documentation

addAllEdgeTypes

```
- (void) addAllEdgeTypes
```

Allows connectivity through all edge types of the graph.

The edges can be used in Any direction.

References addAllEdgeTypes.

Referenced by addAllEdgeTypes.

addEdgeType:

```
- (void) addEdgeType:
             (int) type
```

Allows connectivity through edges of the given type.

The edges can be used in Any direction.

Parameters

type	[in] Edge type.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows connectivity through nodes of the given type.

Parameters

type	null

close

```
- (void) close
```

Closes the CommunityDetection instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getCommunities

```
- (STSDisjointCommunities *) getCommunities
```

Returns the results generated by the execution of the algorithm.

These results contain information related to the disjoint communities found as the number of different components, the set of nodes contained in each component or many other data.

Returns

Returns an instance of the class DisjointCommunities which contain information related to the disjoint communities found.

References getCommunities.

Referenced by getCommunities.

includeEdges:

```
- (void) includeEdges:
             (STSObjects *) edges
```

Set additional edges that can be used.

This will replace any previously specified set of include edges. Using this optional method adds valid edges to the edges of any edge type explicitly set as a valid type. Should only be used to include specific small sets of edges because it's less efficient than just using an edge type. For any edge to be used, both nodes must be also valid.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

includeNodes:

```
- (void) includeNodes:
             (STSObjects *) nodes
```

Set additional nodes that can be used.

This will replace any previously specified set of include nodes. Using this optional method adds valid nodes to the nodes of any node type explicitly set as a valid type. Should only be used to include specific small sets of nodes because it's less efficient than just using a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

run

```
- (void) run
```

Runs the algorithm in order to find the communities.

This method can be called only once.

Implements STSCommunityDetection.

Implemented in STSCommunitiesSCD.

References run.

Referenced by run.

setMaterializedAttribute:

```
- (void) setMaterializedAttribute:
             (NSString *) attributeName
```

Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the disjoint communities found while executing this algorithm.

Whenever the user wants to retrieve the results, even when the graph has been closed and opened again, it is only necessary to create a new instance of the class DisjointCommunities indicating the graph and the name of the common attribute type which stores the results. This instance will have all the information related to the disjoint communities found in the moment of the execution of the algorithm that stored this data.

It is possible to run the algorithm without specifying this parameter in order to avoid materializing the results of the execution.

Parameters

attributeName	[in] The name of the common attribute type for all node types in the graph which will store persistently the results generated by the execution of the algorithm.

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

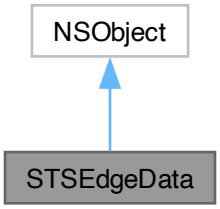
- Sparksee.h

3.20 STSEdgeData Class Reference

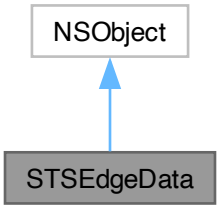
Edge data class.

`#include <Sparksee.h>`

Inheritance diagram for STSEdgeData:



Collaboration diagram for STSEdgeData:



Instance Methods

- (long long) - getEdge
  Gets the edge identifier.
- (long long) - getTail
  Gets the tail of the edge.
- (long long) - getHead
  Gets the head of the edge.

3.20.1 Detailed Description

Edge data class.

It stores the tail and the head of an edge instance.

In case of undirected eges, the tail and the head are just the two ends of the edge.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.20.2 Method Documentation

getEdge

```
- (long long) getEdge
```

Gets the edge identifier.

Returns

The Sparksee edge identifier.

References getEdge.

Referenced by getEdge.

getHead

```
- (long long) getHead
```

Gets the head of the edge.

Returns

The Sparksee edge identifier of the head of the edge.

References getHead.

Referenced by getHead.

getTail

```
- (long long) getTail
```

Gets the tail of the edge.

Returns

The Sparksee edge identifier of the tail of the edge.

References getTail.

Referenced by getTail.

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

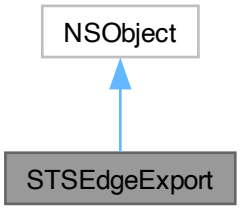
- Sparksee.h

3.21 STSEdgeExport Class Reference

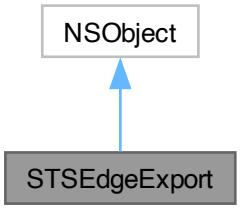
Stores edge exporting values.

`#include <Sparksee.h>`

Inheritance diagram for STSEdgeExport:



Collaboration diagram for STSEdgeExport:



Instance Methods

- (id) - **init**

  Creates a new instance.
- (void) - **setDefaults**

  Sets to default values.
- (void) - copyDefaults:

  Sets the default values using a refrerence EdgeExport.
- (NSString ∗) - getLabel

  Gets the edge label.
- (void) - setLabel:

  Sets the edge label.
- (NSString ∗) - getTitle

  Gets the edge title.

- (void) - setTitle:

  Sets the edge title.
- (BOOL) - asDirected

  Gets if the edge should be managed as directed.
- (void) - setAsDirected:

  Sets if the edge should be managed as directed.
- (int) - getColorRGB

  Gets the edge color.
- (void) - setColorRGB:

  Sets the edge color.
- (int) - getLabelColorRGB

  Gets the edge label color.
- (void) - setLabelColorRGB:

  Sets the edge label color.
- (int) - getWidth

  Gets the edge width.
- (void) - setWidth:

  Sets the edge width.
- (int) - getFontSize

  Gets the edge label font size.
- (void) - setFontSize:

  Sets the edge label font size.
- (void) - getColorRed:green:blue:alpha:

  Get the edge color separated in RGBA.
- (void) - setColorRed:green:blue:alpha:

  Set the edge color with separated RGBA components.
- (void) - getLabelColorRed:green:blue:alpha:

  Get the edge label color separated in RGBA.
- (void) - setLabelColorRed:green:blue:alpha:

  Set the edge label color with separated RGBA components.

3.21.1 Detailed Description

Stores edge exporting values.

Some properties may be ignored depending on the exportation type.

Default values are:

Label: "" (empty string).

As directed: TRUE.

Color: 13882323 (OxD3D3D3, Light gray).

Label color: 0 (Ox000000, Black).

Width: 5px.

Font size: 10.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.21.2 Method Documentation

asDirected

```
- (BOOL) asDirected
```

Gets if the edge should be managed as directed.

TRUE is the default value. If TRUE, use as directed, otherwise use as undirected.

Returns

The edge direction.

References asDirected.

Referenced by asDirected.

copyDefaults:

```
- (void) copyDefaults:
             (STSEdgeExport *) source
```

Sets the default values using a refrerence EdgeExport.

Parameters

source	null

getColorRed:green:blue:alpha:

```
- (void) getColorRed:
             (double *) red
             green:(double *) green
             blue:(double *) blue
             alpha:(double *) alpha
```

Get the edge color separated in RGBA.

Parameters

red	[out] The red color component ([0..1]).
green	[out] The green color component ([0..1]).
blue	[out] The blue color component ([0..1]).

alpha	[out] The alpha component ([0..1]).

getColorRGB

```
- (int) getColorRGB
```

Gets the edge color.

Returns

The edge color.

References getColorRGB.

Referenced by getColorRGB.

getFontSize

```
- (int) getFontSize
```

Gets the edge label font size.

Returns

The edge label font size.

References getFontSize.

Referenced by getFontSize.

getLabel

```
- (NSString *) getLabel
```

Gets the edge label.

Returns

The edge label.

References getLabel.

Referenced by getLabel.

getLabelColorRed:green:blue:alpha:

```
- (void) getLabelColorRed:
             (double *) red
            green:(double *) green
            blue:(double *) blue
            alpha:(double *) alpha
```

Get the edge label color separated in RGBA.

Parameters

red	[out] The red color component ([0..1]).
green	[out] The green color component ([0..1]).
blue	[out] The blue color component ([0..1]).
alpha	[out] The alpha component ([0..1]).

getLabelColorRGB

```
- (int) getLabelColorRGB
```

Gets the edge label color.

Returns

The edge label color.

References getLabelColorRGB.

Referenced by getLabelColorRGB.

getTitle

```
- (NSString *) getTitle
```

Gets the edge title.

Returns

The edge title.

References getTitle.

Referenced by getTitle.

getWidth

```
- (int) getWidth
```

Gets the edge width.

Returns

The edge width.

References getWidth.

Referenced by getWidth.

setAsDirected:

```
- (void) setAsDirected:
             (BOOL) directed
```

Sets if the edge should be managed as directed.

Parameters

directed	[in] If TRUE, use as directed, otherwise use as undirected.

setColorRed:green:blue:alpha:

```
- (void) setColorRed:
             (double) red
             green:(double) green
             blue:(double) blue
             alpha:(double) alpha
```

Set the edge color with separated RGBA components.

Parameters

red	[in] The red color component ([0..1]).
green	[in] The green color component ([0..1]).
blue	[in] The blue color component ([0..1]).
alpha	[in] The alpha component ([0..1]).

setColorRGB:

```
- (void) setColorRGB:
             (int) color
```

Sets the edge color.

Parameters

color	[in] The edge color.

setFontSize:

```
- (void) setFontSize:
             (int) size
```

Sets the edge label font size.

Parameters

size	[in] The edge label font size.

setLabel:

```
- (void) setLabel:
             (NSString *) label
```

Sets the edge label.

Parameters

label	[in] The edge label.

setLabelColorRed:green:blue:alpha:

```
- (void) setLabelColorRed:
             (double) red
             green:(double) green
             blue:(double) blue
             alpha:(double) alpha
```

Set the edge label color with separated RGBA components.

Parameters

red	[in] The red color component ([0..1]).

green	[in] The green color component ([0..1]).
blue	[in] The blue color component ([0..1]).
alpha	[in] The alpha component ([0..1]).

setLabelColorRGB:

```
- (void) setLabelColorRGB:
             (int) color
```

Sets the edge label color.

Parameters

color	[in] The edge label color.

setTitle:

```
- (void) setTitle:
             (NSString *) title
```

Sets the edge title.

Parameters

title	[in] The edge title.

setWidth:

```
- (void) setWidth:
             (int) width
```

Sets the edge width.

Parameters

width	[in] The edge width.

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

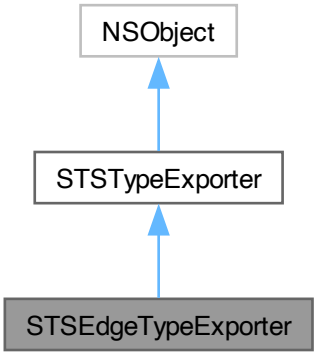
- Sparksee.h

3.22 STSEdgeTypeExporter Class Reference

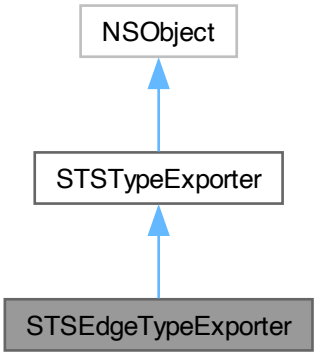
EdgeTypeExporter class.

`#include <Sparksee.h>`

Inheritance diagram for STSEdgeTypeExporter:



Collaboration diagram for STSEdgeTypeExporter:



Instance Methods

- (id) - **init**

  Creates a new instance.
- (id) - initWithRowWriter:graph:type:attrs:hPos:tPos:hAttr:tAttr:

  Creates a new instance.

- (void) - run

  See the TypeExporter class Run method.
- (void) - setHeadAttribute:

  Sets the attribute that will be used to get the value to be dumped for the head of the edge.
- (void) - setHeadPosition:

  Sets the position (index column) of the head attribute in the exported data.
- (void) - setTailAttribute:

  Sets the attribute that will be used to get the value to be dumped for the tail of the edge.
- (void) - setTailPosition:

  Sets the position (index column) of the tail attribute in the exported data.
- (void) - registerListener:

  Registers a new listener.
- (void) - setRowWriter:

  Sets the output data destination.
- (void) - setGraph:

  Sets the graph that will be exported.
- (void) - setType:

  Sets the type to be exported.
- (void) - setAttributes:

  Sets the list of Attributes.
- (void) - setFrequency:

  Sets the frequency of listener notification.
- (void) - setHeader:

  Sets the presence of a header row.

3.22.1 Detailed Description

EdgeTypeExporter class.

Specific TypeExporter implementation for edge types.

Check out the 'Data export' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.22.2 Method Documentation

initWithRowWriter:graph:type:attrs:hPos:tPos:hAttr:tAttr:

```
- (id) initWithRowWriter:
             (STSRowWriter *) rowWriter
             graph:(STSGraph *) graph
             type:(int) type
             attrs:(STSAttributeList *) attrs
             hPos:(int) hPos
             tPos:(int) tPos
             hAttr:(int) hAttr
             tAttr:(int) tAttr
```

Creates a new instance.

Parameters

rowWriter	[in] Output RowWriter.
graph	[in] Graph.
type	[in] Type identifier.
attrs	[in] Attribute identifiers to be exported.
hPos	[in] The position (index column) for the head value.
tPos	[in] The position (index column) for the tail value.
hAttr	[in] The attribute identifier to get the value to be dumped for the head.
tAttr	[in] The attribute identifier to get the value to be dumped for the tail.

registerListener:

```
- (void) registerListener:
            (STSTypeExporterListener *) tel
```

Registers a new listener.

Parameters

tel	[in] TypeExporterListener to be registered.

run

```
- (void) run
```

See the TypeExporter class Run method.

WrongArgumentErrorIn case of bad settings ErrorFor other issues

Exceptions

System.ApplicationException	null
System.ArgumentException	null
System.IO.IOException	If bad things happen writting to the RowWriter.

Implements STSTypeExporter.

References run.

Referenced by run.

setAttributes:

```
- (void) setAttributes:
             (STSAttributeList *) attrs
```

Sets the list of Attributes.

Parameters

attrs	[in] Attribute identifiers to be exported

setFrequency:

```
- (void) setFrequency:
             (int) freq
```

Sets the frequency of listener notification.

Parameters

freq	[in] Frequency in number of rows managed to notify progress to all listeners

setGraph:

```
- (void) setGraph:
             (STSGraph *) graph
```

Sets the graph that will be exported.

Parameters

graph	[in] Graph.

setHeadAttribute:

```
- (void) setHeadAttribute:
             (int) attr
```

Sets the attribute that will be used to get the value to be dumped for the head of the edge.

Parameters

attr	[in] Head Attribute

setHeader:

```
- (void) setHeader:
             (BOOL) header
```

Sets the presence of a header row.

Parameters

header	[in] If TRUE, a header row is dumped with the name of the attributes.

setHeadPosition:

```
- (void) setHeadPosition:
             (int) pos
```

Sets the position (index column) of the head attribute in the exported data.

Parameters

pos	[in] Head position

setRowWriter:

```
- (void) setRowWriter:
             (STSRowWriter *) rw
```

Sets the output data destination.

Parameters

rw	[in] Input RowWriter.

setTailAttribute:

```
- (void) setTailAttribute:
             (int) attr
```

Sets the attribute that will be used to get the value to be dumped for the tail of the edge.

Parameters

attr	[in] Tail Attribute

setTailPosition:

```
- (void) setTailPosition:
             (int) pos
```

Sets the position (index column) of the tail attribute in the exported data.

Parameters

pos	[in] Tail position

setType:

```
- (void) setType:
             (int) type
```

Sets the type to be exported.

Parameters

type	[in] Type identifier.

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

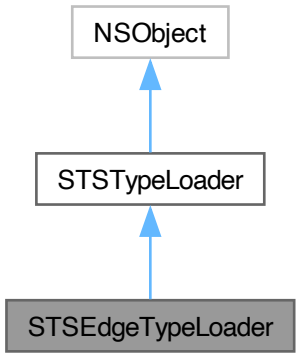
- Sparksee.h

3.23 STSEdgeTypeLoader Class Reference

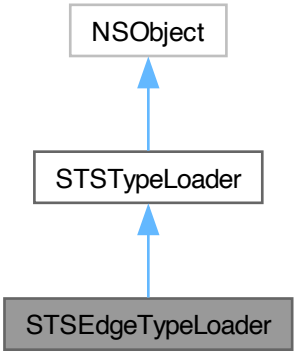
EdgeTypeLoader class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSEdgeTypeLoader:



Collaboration diagram for STSEdgeTypeLoader:



Instance Methods

- (id) - **init**

  Creates a new instance.
- (void) - run

  See the TypeLoader class Run method.
- (void) - runTwoPhases

  See the TypeLoader class RunTwoPhases method.
- (void) - runNPhases:

  See the TypeLoader class RunNPhases method.
- (void) - setHeadAttribute:

  Sets the attribute that will be used to find the head of the edge.
- (void) - setHeadPosition:

  Sets the position of the head attribute in the source data.
- (void) - setTailAttribute:

  Sets the attribute that will be used to find the tail of the edge.
- (void) - setTailPosition:

  Sets the position of the tail attribute in the source data.
- (void) - setTailMEP:

  Sets the flag to create missing tail nodes when loading nodes or edges.
- (void) - setHeadMEP:

  Sets the flag to create missing head nodes when loading nodes or edges.
- (void) - setLogError:

  Sets a log error file.
- (void) - setLogOff

  Truns off all the error reporting.
- (void) - registerListener:

  Registers a new listener.
- (void) - setRowReader:

  Sets the input data source.
- (void) - setGraph:

Sets the graph where the data will be loaded.

- (void) - setLocale:

  Sets the locale that will be used to read the data.

- (void) - setType:

  Sets the type to be loaded.

- (void) - setAttributes:

  Sets the list of Attributes.

- (void) - setAttributePositions:

  Sets the list of attribute positions.

- (void) - setTimestampFormat:

  Sets a specific timestamp format.

- (void) - setFrequency:

  Sets the frequency of listener notification.

3.23.1 Detailed Description

EdgeTypeLoader class.

Specific TypeLoader implementation for edge types.

Check out the 'Data import' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.23.2 Method Documentation

registerListener:

```
- (void) registerListener:
             (STSTypeLoaderListener *) tel
```

Registers a new listener.

Parameters

tel	TypeLoaderListener to be registered.

run

```
- (void) run
```

See the TypeLoader class Run method.

ErrorFor other issues

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen using to the RowReader.

Implements STSTypeLoader.

References run.

Referenced by run.

runNPhases:

```
- (void) runNPhases:
             (int) partitions
```

See the TypeLoader class RunNPhases method.

ErrorFor other issues

Parameters

partitions	null

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen using to the RowReader.

Implements STSTypeLoader.

runTwoPhases

```
- (void) runTwoPhases
```

See the TypeLoader class RunTwoPhases method.

ErrorFor other issues

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen using to the RowReader.

Implements STSTypeLoader.

References runTwoPhases.

Referenced by runTwoPhases.

setAttributePositions:

```
- (void) setAttributePositions:
             (STSInt32List *) attrsPos
```

Sets the list of attribute positions.

Parameters

attrsPos	[in] Attribute positions (column index >=0).

setAttributes:

```
- (void) setAttributes:
             (STSAttributeList *) attrs
```

Sets the list of Attributes.

Parameters

attrs	[in] Attribute identifiers to be loaded

setFrequency:

```
- (void) setFrequency:
             (int) freq
```

Sets the frequency of listener notification.

Parameters

freq	[in] Frequency in number of rows managed to notify progress to all listeners

setGraph:

```
- (void) setGraph:
              (STSGraph *) graph
```

Sets the graph where the data will be loaded.

Parameters

graph	[in] Graph.

setHeadAttribute:

```
- (void) setHeadAttribute:
              (int) attr
```

Sets the attribute that will be used to find the head of the edge.

This method is protected because only the Edge loaders should have it.

Parameters

attr	[in] Head Attribute

setHeadMEP:

```
- (void) setHeadMEP:
              (enum STSMissingEndpoint) mep
```

Sets the flag to create missing head nodes when loading nodes or edges.

flagTrue if the nodes need to be created. False otherwise

Parameters

mep	null

setHeadPosition:

```
- (void) setHeadPosition:
              (int) pos
```

Sets the position of the head attribute in the source data.

This method is protected because only the Edge loaders should have it.

Parameters

pos	[in] Head position

setLocale:

```
- (void) setLocale:
             (NSString *) localeStr
```

Sets the locale that will be used to read the data.

It should match the locale used in the rowreader.

Parameters

localeStr	[in] The locale string for the read data. See CSVReader.

setLogError:

```
- (void) setLogError:
             (NSString *) path
```

Sets a log error file.

By default errors are thrown as a exception and the load process ends. If a log file is set, errors are logged there and the load process does not stop.

Parameters

path	[in] The path to the error log file.

Exceptions

System.IO.IOException	If bad things happen opening the file.

setLogOff

```
- (void) setLogOff
```

Truns off all the error reporting.

The log file will not be created and no exceptions for invalid data will be thrown. If you just want to turn off the logs, but abort at the first error what you should do is not call this method and not set a logError file.

References setLogOff.

Referenced by setLogOff.

setRowReader:

```
- (void) setRowReader:
              (STSRowReader *) rr
```

Sets the input data source.

Parameters

rr	[in] Input RowReader.

setTailAttribute:

```
- (void) setTailAttribute:
              (int) attr
```

Sets the attribute that will be used to find the tail of the edge.

This method is protected because only the Edge loaders should have it.

Parameters

attr	[in] Tail Attribute

setTailMEP:

```
- (void) setTailMEP:
              (enum STSMissingEndpoint) mep
```

Sets the flag to create missing tail nodes when loading nodes or edges.

flagTrue if the nodes need to be created. False otherwise

Parameters

mep	null

setTailPosition:

```
- (void) setTailPosition:
              (int) pos
```

Sets the position of the tail attribute in the source data.

This method is protected because only the Edge loaders should have it.

Parameters

pos	[in] Tail position

setTimestampFormat:

```
- (void) setTimestampFormat:
            (NSString *) timestampFormat
```

Sets a specific timestamp format.

Parameters

timestampFormat	[in] A string with the timestamp format definition.

setType:

```
- (void) setType:
            (int) type
```

Sets the type to be loaded.

Parameters

type	[in] Type identifier.

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

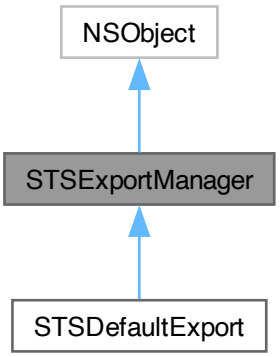
- Sparksee.h

3.24 STSExportManager Class Reference

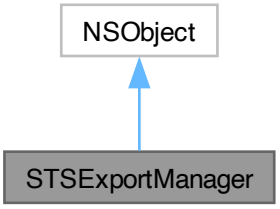
Defines how to export a graph to an external format.

`#include <Sparksee.h>`

Inheritance diagram for STSExportManager:



Collaboration diagram for STSExportManager:



Instance Methods

- (void) - prepare:

  Prepares the graph for the export process.
- (void) - close

  Ends the export process.
- (BOOL) - getGraph:

  Gets the graph export definition.
- (BOOL) - getNodeType:nodeExport:

  Gets the default node export definition for the given node type.
- (BOOL) - getEdgeType:edgeExport:

  Gets the default node export definition for the given edge type.
- (BOOL) - getNode:nodeExport:

  Gets the node export definition for the given node.
- (BOOL) - getEdge:edgeExport:

  Gets the edge export definition for the given edge.
- (BOOL) - enableType:

  Gets whether a node or edge type must be exported or not.
- (BOOL) - exportAll

  Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.
- (BOOL) - exportAllSwigExplicitSTSExportManager

  Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.
- (long long) - nextNode

  Gets the next node to export or InvalidOID.
- (long long) - nextNodeSwigExplicitSTSExportManager

  Gets the next node to export or InvalidOID.
- (long long) - nextEdge

  Gets the next edge to export or InvalidOID.
- (long long) - nextEdgeSwigExplicitSTSExportManager

  Gets the next edge to export or InvalidOID.

3.24.1 Detailed Description

Defines how to export a graph to an external format.

This is an interface which must be implemented by the user. While the export proces, a call for each node or edge type and node or edge object is done to get how to export that element.

It is possible to export a Graph to a diferent fortmats. Nowadays, available formats are defined in the ExportType enum.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.24.2 Method Documentation

close

```
- (void) close
```

Ends the export process.

It is called once after the export process.

Implemented in STSDefaultExport.

References close.

Referenced by close.

enableType:

```
- (BOOL) enableType:
             (int) type
```

Gets whether a node or edge type must be exported or not.

Parameters

type	Node or edge type identifier.

Returns

If TRUE all objects of the given type will be exported, otherwise they will not be exported.

Implemented in STSDefaultExport.

exportAll

```
- (BOOL) exportAll
```

Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.

Or false to use the new NextNode() and NextEdge() methods of this class to get the only nodes/edges that should be exported.

Returns

bool_t Returns true to export all the graph or false otherwise.

Implemented in STSDefaultExport.

References exportAll.

Referenced by exportAll.

exportAllSwigExplicitSTSExportManager

```
- (BOOL) exportAllSwigExplicitSTSExportManager
```

Get if all the graph object should be exported or only a subgraph It should return true (default option), to export all the nodes and edges in the Graph as with previous verions.

Or false to use the new NextNode() and NextEdge() methods of this class to get the only nodes/edges that should be exported.

Returns

bool_t Returns true to export all the graph or false otherwise.

References exportAllSwigExplicitSTSExportManager.

Referenced by exportAllSwigExplicitSTSExportManager.

getEdge:edgeExport:

```
- (BOOL) getEdge:
            (long long) edge
            edgeExport:(STSEdgeExport *) edgeExport
```

Gets the edge export definition for the given edge.

Parameters

edge	Edge identifier.
edgeExport	[out] The EdgeExport which defines how to export given edge.

Returns

TRUE if the given EdgeExport has been updated, otherwise FALSE will be returned and the default Edge↩ Export for the type the edge belongs to will be used.

Implemented in STSDefaultExport.

getEdgeType:edgeExport:

```
- (BOOL) getEdgeType:
              (int) type
              edgeExport:(STSEdgeExport *) edgeExport
```

Gets the default node export definition for the given edge type.

GetEdge has a higher priority than this function. That is, only if GetEdge returns FALSE, the EdgeExport of this function will be used.

Parameters

type	[in] Edge type identifier.
edgeExport	[out] The EdgeExport which defines how to export the edges of the given type.

Returns

TRUE.

Implemented in STSDefaultExport.

getGraph:

```
- (BOOL) getGraph:
              (STSGraphExport *) graphExport
```

Gets the graph export definition.

Parameters

graphExport	[out] The GraphExport which defines how to export the graph.

Returns

TRUE.

Implemented in STSDefaultExport.

getNode:nodeExport:

```
- (BOOL) getNode:
             (long long) node
             nodeExport:(STSNodeExport *) nodeExport
```

Gets the node export definition for the given node.

Parameters

node	Node identifier.
nodeExport	[out] The NodeExport which defines how to export given node.

Returns

TRUE if the given NodeExport has been updated, otherwise FALSE will be returned and the default Node↩ Export for the type the node belongs to will be used.

Implemented in STSDefaultExport.

getNodeType:nodeExport:

```
- (BOOL) getNodeType:
             (int) type
             nodeExport:(STSNodeExport *) nodeExport
```

Gets the default node export definition for the given node type.

GetNode has a higher priority than this function. That is, only if GetNode returns FALSE, the NodeExport of this function will be used.

Parameters

type	[in] Node type identifier.
nodeExport	[out] The NodeExport which defines how to export the nodes of the given type.

Returns

TRUE.

Implemented in STSDefaultExport.

nextEdge

```
- (long long) nextEdge
```

Gets the next edge to export or InvalidOID.

It will return valid edge OIDs to export and InvalidOID when there is not any more edges to export. ExportAll returns false.

Returns

oid_t Return an edge OID or InvalidOID

Implemented in STSDefaultExport.

References nextEdge.

Referenced by nextEdge.

nextEdgeSwigExplicitSTSExportManager

```
- (long long) nextEdgeSwigExplicitSTSExportManager
```

Gets the next edge to export or InvalidOID.

It will return valid edge OIDs to export and InvalidOID when there is not any more edges to export. ExportAll returns false.

Returns

oid_t Return an edge OID or InvalidOID

References nextEdgeSwigExplicitSTSExportManager.

Referenced by nextEdgeSwigExplicitSTSExportManager.

nextNode

```
- (long long) nextNode
```

Gets the next node to export or InvalidOID.

It will return valid node OIDs to export and InvalidOID when there is not any more nodes to export. ExportAll returns false.

Returns

oid_t Return a node OID or InvalidOID

Implemented in STSDefaultExport.

References nextNode.

Referenced by nextNode.

nextNodeSwigExplicitSTSExportManager

```
- (long long) nextNodeSwigExplicitSTSExportManager
```

Gets the next node to export or InvalidOID.

It will return valid node OIDs to export and InvalidOID when there is not any more nodes to export. ExportAll returns false.

Returns

oid_t Return a node OID or InvalidOID

References nextNodeSwigExplicitSTSExportManager.

Referenced by nextNodeSwigExplicitSTSExportManager.

prepare:

```
- (void) prepare:
             (STSGraph *) graph
```

Prepares the graph for the export process.

It is called once before the export process.

Parameters

graph	Graph to be exported.

Implemented in STSDefaultExport.

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

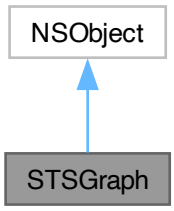
- Sparksee.h

3.25 STSGraph Class Reference

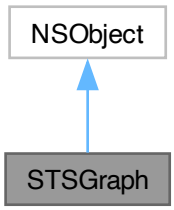
Graph class.

`#include <Sparksee.h>`

Inheritance diagram for STSGraph:



Collaboration diagram for STSGraph:



Instance Methods

- (int) - createNodeType:

  Creates a new node type.
- (long long) - createNode:

  Creates a new node instance.
- (int) - createEdgeType:directed:neighbors:

  Creates a new edge type.
- (int) - createRestrictedEdgeType:tail:head:neighbors:

  Creates a new restricted edge type.
- (void) - indexNeighbors:neighbors:

  Creates or destroys the neighbors index of an edge type.
- (long long) - createEdge:tail:head:

  Creates a new edge instance.
- (long long) - createEdgeWithAttributes:tailAttr:tailV:headAttr:headV:

  Creates a new edge instance.
- (long long) - countNodes

  Gets the number of nodes.
- (long long) - countEdges

  Gets the number of edges.
- (STSEdgeData *) - getEdgeData:

  Gets information about an edge.
- (long long) - getEdgePeer:node:

  Gets the other end for the given edge.
- (void) - drop:

  Drops the given OID.
- (void) - dropWithObjects:

  Drops all the OIDs from the given collection.
- (int) - getObjectType:

  Gets the Sparksee node or edge type identifier for the given OID.
- (int) - createAttribute:name:dt:kind:

  Creates a new attribute.
- (int) - createAttributeWithDefault:name:dt:kind:defaultValue:

  Creates a new attribute with a default value.
- (int) - createSessionAttribute:dt:kind:

  Creates a new Session attribute.
- (int) - createSessionAttributeWithDefault:dt:kind:defaultValue:

  Creates a new Session attribute with a default value.
- (int) - createArrayAttribute:name:dt:size:

  Creates a new array attribute.
- (int) - createSessionArrayAttribute:dt:size:

  Creates a new Session array attribute.
- (void) - setAttributeDefaultValue:value:

  Sets a default value for an attribute.
- (void) - indexAttribute:kind:

  Updates the index of the given attribute.
- (void) - getAttributeInValue:attr:value:

  Gets the Value for the given attribute and OID.
- (STSValue *) - getAttributeWithOid:attr:

  Gets the Value for the given attribute and OID.
- (STSTextStream *) - getAttributeText:attr:

Gets the read-only TextStream for the given text attribute and OID.

- (void) - setAttributeText:attr:tstream:

  Sets the writable TextStream for the given text attribute and OID.
- (STSValueArray *) - getArrayAttribute:attr:

  Gets the ValueArray for the given array attribute and OID or NULL if it does not exist.
- (STSValueArray *) - setArrayAttribute:attr:value:

  Sets all the elements of the ValueArray for the given array attribute and OID and returns it.
- (void) - setArrayAttributeVoid:attr:value:

  Sets all the elements of the ValueArray for the given array attribute and OID.
- (void) - setAttribute:attr:value:

  Sets the Value for the given attribute and OID.
- (STSAttributeStatistics *) - getAttributeStatistics:basic:

  Gets statistics from the given attribute.
- (long long) - getAttributeIntervalCount:lower:includeLower:higher:includeHigher:

  Gets how many objects have a value into the given range for the given attribute.
- (int) - findType:

  Gets the Sparksee type identifier for the given type name.
- (STSType *) - getType:

  Gets information about the given type.
- (void) - removeType:

  Removes the given type.
- (void) - renameTypeWithName:newName:

  Renames a type.
- (void) - renameType:newName:

  Renames a type.
- (int) - findAttribute:name:

  Gets the Sparksee attribute identifier for the given type identifier and attribute name.
- (STSAttribute *) - getAttribute:

  Gets information about the given attribute.
- (void) - removeAttribute:

  Removes the given attribute.
- (void) - renameAttribute:newName:

  Renames an attribute.
- (long long) - findObject:value:

  Finds one object having the given Value for the given attribute.
- (long long) - findOrCreateObject:value:

  Finds one object having the given Value for the attribute or it creates one does not exist any.
- (STSObjects *) - selectWithType:

  Selects all OIDs belonging to the given type.
- (STSObjects *) - selectWithAttrValue:cond:value:

  Selects all OIDs satisfying the given condition for the given attribute.
- (STSObjects *) - selectWithAttrValues:cond:lower:higher:

  Selects all OIDs satisfying the given condition for the given attribute.
- (STSObjects *) - selectWithAttrValueRestriction:cond:value:restriction:

  Selects all OIDs satisfying the given condition for the given attribute.
- (STSObjects *) - selectWithAttrValuesRestriction:cond:lower:higher:restriction:

  Selects all OIDs satisfying the given condition for the given attribute.
- (STSKeyValues *) - topkWithAttr:order:k:

  Gets a KeyValues iterator as a result of the TopK operation.
- (STSKeyValues *) - topkWithAttrValue:operation:lower:order:k:

  Gets a KeyValues iterator as a result of the TopK operation.

- (STSKeyValues *) - topkWithAttrValues:operation:lower:higher:order:k:
  Gets a KeyValues iterator as a result of the TopK operation.
- (STSKeyValues *) - topkWithAttrRestriction:order:k:restriction:
  Gets a KeyValues iterator as a result of the TopK operation.
- (STSKeyValues *) - topkWithAttrValueRestriction:operation:lower:order:k:restriction:
  Gets a KeyValues iterator as a result of the TopK operation.
- (STSKeyValues *) - topkWithAttrValuesRestriction:operation:lower:higher:order:k:restriction:
  Gets a KeyValues iterator as a result of the TopK operation.
- (STSObjects *) - explode:etype:dir:
  Selects all edges from or to the given node OID and for the given edge type.
- (STSObjects *) - explodeWithObjects:etype:dir:
  Selects all edges from or to each of the node OID in the given collection and for the given edge type.
- (long long) - degree:etype:dir:
  Gets the number of edges from or to the given node OID and for the given edge type.
- (STSObjects *) - neighbors:etype:dir:
  Selects all neighbor nodes from or to the given node OID and for the given edge type.
- (STSObjects *) - neighborsWithObjects:etype:dir:
  Selects all neighbor nodes from or to each of the node OID in the given collection and for the given edge type.
- (STSObjects *) - edges:tail:head:
  Gets all the edges of the given type between two given nodes (tail and head).
- (long long) - findEdge:tail:head:
  Gets any of the edges of the given type between two given nodes (tail and head).
- (long long) - findOrCreateEdge:tail:head:
  Gets any of the edges of the specified type between two given nodes (tail and head).
- (STSObjects *) - tails:
  Gets all the tails from the given edges collection.
- (STSObjects *) - heads:
  Gets all the heads from the given edges collection.
- (void) - tailsAndHeads:tails:heads:
  Gets all the tails and heads from the given edges collection.
- (STSTypeList *) - findNodeTypes
  Gets all existing Sparksee node type identifiers.
- (STSTypeList *) - findEdgeTypes
  Gets all existing Sparksee edge type identifiers.
- (STSTypeList *) - findTypes
  Gets all existing Sparksee node and edge type identifiers.
- (STSAttributeList *) - findAttributes:
  Gets all existing Sparksee attribute identifiers for the given type identifier.
- (STSAttributeList *) - getAttributes:
  Gets all Sparksee attribute identifiers with a non-NULL value for the given Sparksee OID.
- (STSValues *) - getValues:
  Gets the Value collection for the given attribute.
- (void) - dumpData:
  Dumps logical data to a file.
- (void) - dumpStorage:
  Dumps internal storage data to a file.
- (void) - exportGraph:type:em:
  Exports the Graph.
- (void) - backup:
  Dumps all the data to a backup file.
- (void) - encryptedBackup:keyInHex:ivInHex:
  Dumps all the data to a backup file.
- (STSSession *) - getSession
  Gets the Session this Graph belongs to.

3.25.1 Detailed Description

Graph class.

Each Database has a Graph associated, which is the persistent graph which contains all data stored into the graph database and is retrieved from a Session.

Check out the 'API' and the 'SPARKSEE graph database' sections in the SPARKSEE User Manual for more details on the use of this class.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.25.2 Method Documentation

backup:

```
- (void) backup:
             (NSString *) file
```

Dumps all the data to a backup file.

See the Sparksee class Restore methods. Errorfor other issues.

Parameters

file	[in] Output backup file path.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file cannot be created.

countEdges

```
- (long long) countEdges
```

Gets the number of edges.

Returns

The number of edges.

References countEdges.

Referenced by countEdges.

countNodes

```
- (long long) countNodes
```

Gets the number of nodes.

Returns

The number of nodes.

References countNodes.

Referenced by countNodes.

createArrayAttribute:name:dt:size:

```
- (int) createArrayAttribute:
             (int) type
             name:(NSString *) name
             dt:(enum STSDataType) dt
             size:(int) size
```

Creates a new array attribute.

Parameters

type	[in] Sparksee node or edge type identifier.
name	[in] Unique name for the new attribute.
dt	[in] Base Data type for the new array attribute elements.
size	[in] The array size.

Returns

Unique Sparksee attribute identifier.

createAttribute:name:dt:kind:

```
- (int) createAttribute:
             (int) type
             name:(NSString *) name
             dt:(enum STSDataType) dt
             kind:(enum STSAttributeKind) kind
```

Creates a new attribute.

Parameters

type	[in] Sparksee node or edge type identifier.

name	[in] Unique name for the new attribute.
dt	[in] Data type for the new attribute.
kind	[in] Attribute kind.

Returns

Unique Sparksee attribute identifier.

createAttributeWithDefault:name:dt:kind:defaultValue:

```
- (int) createAttributeWithDefault:
             (int) type
             name:(NSString *) name
             dt:(enum STSDataType) dt
             kind:(enum STSAttributeKind) kind
             defaultValue:(STSValue *) defaultValue
```

Creates a new attribute with a default value.

Parameters

type	[in] Sparksee node or edge type identifier.
name	[in] Unique name for the new attribute.
dt	[in] Data type for the new attribute.
kind	[in] Attribute kind.
defaultValue	[in] The default value to use in each new node/edge.

Returns

Unique Sparksee attribute identifier.

createEdge:tail:head:

```
- (long long) createEdge:
             (int) type
             tail:(long long) tail
             head:(long long) head
```

Creates a new edge instance.

Parameters

type	[in] Sparksee type identifier.
tail	[in] Source Sparksee OID.
head	[in] Target Sparksee OID.

Returns

Unique OID of the new edge instance.

createEdgeType:directed:neighbors:

```
- (int) createEdgeType:
              (NSString *) name
              directed:(BOOL) directed
              neighbors:(BOOL) neighbors
```

Creates a new edge type.

Parameters

name	[in] Unique name for the new edge type.
directed	[in] If TRUE, this creates a directed edge type, otherwise this creates a undirected edge type.
neighbors	[in] If TRUE, this indexes neighbor nodes, otherwise not.

Returns

Unique Sparksee type identifier.

createEdgeWithAttributes:tailAttr:tailV:headAttr:headV:

```
- (long long) createEdgeWithAttributes:
              (int) type
              tailAttr:(int) tailAttr
              tailV:(STSValue *) tailV
              headAttr:(int) headAttr
              headV:(STSValue *) headV
```

Creates a new edge instance.

The tail of the edge will be any node having the given tailV Value for the given tailAttr attribute identifier, and the head of the edge will be any node having the given headV Value for the given headAttr attribute identifier.

Parameters

type	[in] Sparksee type identifier.
tailAttr	[in] Sparksee attribute identifier.
tailV	[in] Value.
headAttr	[in] Sparksee attribute identifier.
headV	[in] Value.

Returns

Unique OID of the new edge instance.

createNode:

```
- (long long) createNode:
              (int) type
```

Creates a new node instance.

Parameters

type	[in] Sparksee type identifier.

Returns

Unique OID of the new node instance.

createNodeType:

```
- (int) createNodeType:
              (NSString *) name
```

Creates a new node type.

Parameters

name	[in] Unique name for the new node type.

Returns

Unique Sparksee type identifier.

createRestrictedEdgeType:tail:head:neighbors:

```
- (int) createRestrictedEdgeType:
              (NSString *) name
             tail:(int) tail
             head:(int) head
             neighbors:(BOOL) neighbors
```

Creates a new restricted edge type.

Parameters

name	[in] Unique name for the new edge type.
tail	[in] Tail Sparksee node type identifier.
head	[in] Head Sparksee node type identifier.
neighbors	[in] If TRUE, this indexes neighbor nodes, otherwise not.

Returns

Unique Sparksee type identifier.

createSessionArrayAttribute:dt:size:

```
- (int) createSessionArrayAttribute:
             (int) type
             dt:(enum STSDataType) dt
             size:(int) size
```

Creates a new Session array attribute.

Session attributes are exclusive for the Session (just its Session can use the attribute) and are automatically removed when the Session is closed (thus, attribute data is not persistent into the database).

Since they are not persistent, they cannot be retrieved from the database, so they do not have an identifier name.

Parameters

type	[in] Sparksee node or edge type identifier.
dt	[in] Base Data type for the new array attribute elements.
size	[in] The array size.

Returns

Unique Sparksee attribute identifier.

createSessionAttribute:dt:kind:

```
- (int) createSessionAttribute:
             (int) type
             dt:(enum STSDataType) dt
             kind:(enum STSAttributeKind) kind
```

Creates a new Session attribute.

Session attributes are exclusive for the Session (just its Session can use the attribute) and are automatically removed when the Session is closed (thus, attribute data is not persistent into the database).

Since they are not persistent, they cannot be retrieved from the database, so they do not have an identifier name.

Parameters

type	[in] Sparksee node or edge type identifier.
dt	[in] Data type for the new attribute.
kind	[in] Attribute kind.

Returns

Unique Sparksee attribute identifier.

createSessionAttributeWithDefault:dt:kind:defaultValue:

```
- (int) createSessionAttributeWithDefault:
             (int) type
             dt:(enum STSDataType) dt
             kind:(enum STSAttributeKind) kind
             defaultValue:(STSValue *) defaultValue
```

Creates a new Session attribute with a default value.

Session attributes are exclusive for the Session (just its Session can use the attribute) and are automatically removed when the Session is closed (thus, attribute data is not persistent into the database).

Since they are not persistent, they cannot be retrieved from the database, so they do not have an identifier name.

Parameters

type	[in] Sparksee node or edge type identifier.
dt	[in] Data type for the new attribute.
kind	[in] Attribute kind.
defaultValue	[in] The default value to use in each new node/edge.

Returns

Unique Sparksee attribute identifier.

degree:etype:dir:

```
- (long long) degree:
             (long long) oid
             etype:(int) etype
             dir:(enum STSEdgesDirection) dir
```

Gets the number of edges from or to the given node OID and for the given edge type.

Parameters

oid	[in] Sparksee node OID.
etype	[in] Sparksee edge type identifier.
dir	[in] Direction.

Returns

The number of edges.

drop:

```
- (void) drop:
             (long long) oid
```

Drops the given OID.

It also removes its egdges as well as its attribute values.

Parameters

oid	[in] Sparksee OID to be removed.

dropWithObjects:

```
- (void) dropWithObjects:
             (STSObjects *) objs
```

Drops all the OIDs from the given collection.

See Drop method with the single OID parameter. This performs the same operation for each object in the given set.

Parameters

objs	[in] Objects collection with the OIDs to be removed.

dumpData:

```
- (void) dumpData:
             (NSString *) file
```

Dumps logical data to a file.

Errorfor other issues.

Parameters

file	[in] Output file path.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file cannot be created.

dumpStorage:

```
- (void) dumpStorage:
             (NSString *) file
```

Dumps internal storage data to a file.

Errorfor other issues.

Parameters

file	[in] Output file path.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file cannot be created.

edges:tail:head:

```
- (STSObjects *) edges:
             (int) etype
             tail:(long long) tail
             head:(long long) head
```

Gets all the edges of the given type between two given nodes (tail and head).

Parameters

etype	[in] Sparksee edge type identifier.
tail	[in] Tail node identifier.
head	[in] Head node identifier.

Returns

Objects instance.

encryptedBackup:keyInHex:ivInHex:

```
- (void) encryptedBackup:
             (NSString *) file
             keyInHex:(NSString *) keyInHex
             ivInHex:(NSString *) ivInHex
```

Dumps all the data to a backup file.

See the Sparksee class RestoreEncryptedBackup methods. Errorfor other issues.

Parameters

file	[in] Output backup file path.
keyInHex	[In] The AES encryption Key as an hexadecimal string (8, 16 or 32 bytes).
ivInHex	[In] The AES Initialization Vector as an hexadecimal string (16 bytes).

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file cannot be created.

explode:etype:dir:

```
- (STSObjects *) explode:
             (long long) oid
            etype:(int) etype
            dir:(enum STSEdgesDirection) dir
```

Selects all edges from or to the given node OID and for the given edge type.

Parameters

oid	[in] Sparksee node OID.
etype	[in] Sparksee edge type identifier.
dir	[in] Direction.

Returns

Objects instance.

explodeWithObjects:etype:dir:

```
- (STSObjects *) explodeWithObjects:
             (STSObjects *) objs
            etype:(int) etype
            dir:(enum STSEdgesDirection) dir
```

Selects all edges from or to each of the node OID in the given collection and for the given edge type.

Parameters

objs	[in] Sparksee node OID collection.
etype	[in] Sparksee edge type identifier.
dir	[in] Direction.

Returns

Objects instance.

exportGraph:type:em:

```
- (void) exportGraph:
              (NSString *) file
             type:(enum STSExportType) type
             em:(STSExportManager *) em
```

Exports the Graph.

Parameters

file	[in] Output file.
type	[in] Export type.
em	[in] Defines how to do the export for each graph object.

Exceptions

System.IO.IOException	for errors writing to the file.

findAttribute:name:

```
- (int) findAttribute:
              (int) type
             name:(NSString *) name
```

Gets the Sparksee attribute identifier for the given type identifier and attribute name.

Parameters

type	[in] Sparksee type identifier.
name	[in] Unique attribute name.

Returns

The Sparksee attribute identifier for the given type and attribute name or InvalidAttribute if there is no attribute with the given name for the given type.

findAttributes:

```
- (STSAttributeList *) findAttributes:
              (int) type
```

Gets all existing Sparksee attribute identifiers for the given type identifier.

Parameters

type	[in] Sparksee type identifier.

Returns

Sparksee attribute identifier list.

findEdge:tail:head:

```
- (long long) findEdge:
              (int) etype
             tail:(long long) tail
             head:(long long) head
```

Gets any of the edges of the given type between two given nodes (tail and head).

If there are more than one, then any of them will be returned. And in case there are no edge between the given tail and head, the Objects InvalidOID will be returned.

Parameters

etype	[in] Sparksee edge type identifier.
tail	[in] Tail node identifier.
head	[in] Head node identifier.

Returns

Any of the edges or the Objects InvalidOID.

findEdgeTypes

```
- (STSTypeList *) findEdgeTypes
```

Gets all existing Sparksee edge type identifiers.

Returns

Sparksee edge type identifier list.

References findEdgeTypes.

Referenced by findEdgeTypes.

findNodeTypes

```
- (STSTypeList *) findNodeTypes
```

Gets all existing Sparksee node type identifiers.

Returns

Sparksee node type identifier list.

References findNodeTypes.

Referenced by findNodeTypes.

findObject:value:

```
- (long long) findObject:
              (int) attr
              value:(STSValue *) value
```

Finds one object having the given Value for the given attribute.

If there are more than one, then any of them will be returned. And in case there are no object having this Value, the Objects InvalidOID will be returned.

Parameters

attr	[in] Sparksee attribute identifier.
value	[in] Value.

Returns

Sparksee OID or the Objects InvalidOID.

findOrCreateEdge:tail:head:

```
- (long long) findOrCreateEdge:
              (int) etype
              tail:(long long) tail
              head:(long long) head
```

Gets any of the edges of the specified type between two given nodes (tail and head).

If it can not find any edge of this type between them it tries to create a new one.

Parameters

etype	[in] Sparksee edge type identifier.

tail	[in] Tail node identifier.
head	[in] Head node identifier.

Returns

Any of the edges or the Objects InvalidOID.

findOrCreateObject:value:

```
- (long long) findOrCreateObject:
              (int) attr
              value:(STSValue *) value
```

Finds one object having the given Value for the attribute or it creates one does not exist any.

If the attribute is a node or edge attribute and at least one node/edge with that value is found, it returns one of them. But if it does not exist, then: If it's a node attribute it will create it and set the attribute. If it's an edge attribute it will return the InvalidOID.

Using this method with a global attribute will return the InvalidOID.

Parameters

attr	[in] Sparksee attribute identifier.
value	[in] Value.

Returns

Sparksee OID or the Objects InvalidOID.

findType:

```
- (int) findType:
              (NSString *) name
```

Gets the Sparksee type identifier for the given type name.

Parameters

name	[in] Unique type name.

Returns

The Sparksee type identifier for the given type name or the Type InvalidType if there is no type with the given name.

findTypes

```
- (STSTypeList *) findTypes
```

Gets all existing Sparksee node and edge type identifiers.

Returns

Sparksee node and edge type identifier list.

References findTypes.

Referenced by findTypes.

getArrayAttribute:attr:

```
- (STSValueArray *) getArrayAttribute:
             (long long) oid
             attr:(int) attr
```

Gets the ValueArray for the given array attribute and OID or NULL if it does not exist.

Parameters

oid	[in] Sparksee OID.
attr	[in] Sparksee array attribute identifier.

Returns

A ValueArray. This returns a ValueArray, a NULL or throws an exception.

getAttribute:

```
- (STSAttribute *) getAttribute:
             (int) attr
```

Gets information about the given attribute.

Parameters

attr	[in] Sparksee attribute identifier.

Returns

The Attribute for the given Sparksee attribute identifier.

getAttributeIntervalCount:lower:includeLower:higher:includeHigher:

```
- (long long) getAttributeIntervalCount:
              (int) attr
              lower:(STSValue *) lower
              includeLower:(BOOL) includeLower
              higher:(STSValue *) higher
              includeHigher:(BOOL) includeHigher
```

Gets how many objects have a value into the given range for the given attribute.

This only works for the attributes with the AttributeKind Indexed or Unique.

Given values must belong to the same DataType than the attribute.

Parameters

attr	[in] Sparksee attribute identifier.
lower	[in] Lower bound Value of the range.
includeLower	[in] If TRUE, include lower bound Value of the range.
higher	[in] Higher bound Value of the range.
includeHigher	[in] If TRUE, include higher bound Value of the range.

Returns

Number of objects having a value into the given range.

getAttributeInValue:attr:value:

```
- (void) getAttributeInValue:
              (long long) oid
              attr:(int) attr
              value:(STSValue *) value
```

Gets the Value for the given attribute and OID.

Parameters

oid	[in] Sparksee OID.
attr	[in] Sparksee attribute identifier.
value	[in|out] Value for the given attribute and for the given OID.

getAttributes:

```
- (STSAttributeList *) getAttributes:
             (long long) oid
```

Gets all Sparksee attribute identifiers with a non-NULL value for the given Sparksee OID.

Parameters

oid	[in] Sparksee OID.

Returns

Sparksee attribute identifier list.

getAttributeStatistics:basic:

```
- (STSAttributeStatistics *) getAttributeStatistics:
             (int) attr
             basic:(BOOL) basic
```

Gets statistics from the given attribute.

The statistics can only be obtained from Indexed or Unique attributes.

Parameters

attr	[in] Sparksee attribute identifier.
basic	[in] If FALSE all statistics are computed, if TRUE just those statistics marked as basic will be computed (see description of the AttributeStatistics class). Of course, computing just basic statistics will be faster than computing all of them.

Returns

An AttributeStatistics instace.

getAttributeText:attr:

```
- (STSTextStream *) getAttributeText:
             (long long) oid
             attr:(int) attr
```

Gets the read-only TextStream for the given text attribute and OID.

Parameters

oid	[in] Sparksee OID.

attr	[in] Sparksee attribute identifier.

Returns

A TextStream. This returns a TextStream to read.

getAttributeWithOid:attr:

```
- (STSValue *) getAttributeWithOid:
              (long long) oid
              attr:(int) attr
```

Gets the Value for the given attribute and OID.

The other version of this call, where the Value is an output parameter instead of the return, is better because it allows the user to reuse an existing Value instance, whereas this call always creates a new Value instance to be returned.

It never returns NULL. Thus, in case the OID has a NULL value for the attribute it returns a NULL Value instance.

Parameters

oid	[in] Sparksee OID.
attr	[in] Sparksee attribute identifier.

Returns

A new Value instance having the attribute value for the given OID.

getEdgeData:

```
- (STSEdgeData *) getEdgeData:
              (long long) edge
```

Gets information about an edge.

Parameters

edge	[in] Sparksee edge identifier.

Returns

An EdgeData instance.

getEdgePeer:node:

```
- (long long) getEdgePeer:
              (long long) edge
              node:(long long) node
```

Gets the other end for the given edge.

Parameters

edge	[in] Sparksee edge identifier.
node	[in] Sparksee node identifier. It must be one of the ends of the edge.

Returns

The other end of the edge.

getObjectType:

```
- (int) getObjectType:
              (long long) oid
```

Gets the Sparksee node or edge type identifier for the given OID.

Parameters

oid	[in] Sparksee OID.

Returns

Sparksee node or edge type identifier.

getSession

```
- (STSSession *) getSession
```

Gets the Session this Graph belongs to.

Returns

The owning Session instance.

References getSession.

Referenced by getSession.

getType:

```
- (STSType *) getType:
              (int) type
```

Gets information about the given type.

Parameters

type	[in] Sparksee type identifier.

Returns

The Type for the given Sparksee type identifier.

getValues:

```
- (STSValues *) getValues:
              (int) attr
```

Gets the Value collection for the given attribute.

Parameters

attr	[in] Sparksee attribute identifier.

Returns

Returns a Values object.

heads:

```
- (STSObjects *) heads:
              (STSObjects *) edges
```

Gets all the heads from the given edges collection.

Parameters

edges	[in] Sparksee edge identifier collection.

Returns

The heads collection.

indexAttribute:kind:

```
- (void) indexAttribute:
             (int) attr
             kind:(enum STSAttributeKind) kind
```

Updates the index of the given attribute.

This just works if the current index of the attribute corresponds to the AttributeKind Basic and the new one is Indexed or Unique.

Parameters

attr	[in] Sparksee attribute identifier.
kind	[in] Attribute kind.

indexNeighbors:neighbors:

```
- (void) indexNeighbors:
             (int) edgeType
             neighbors:(BOOL) neighbors
```

Creates or destroys the neighbors index of an edge type.

Parameters

edgeType	[in] Sparksee Edge type identifier.
neighbors	[in] If TRUE, it indexes the neighbor nodes, otherwise it removes the index.

neighbors:etype:dir:

```
- (STSObjects *) neighbors:
             (long long) oid
             etype:(int) etype
             dir:(enum STSEdgesDirection) dir
```

Selects all neighbor nodes from or to the given node OID and for the given edge type.

Parameters

oid	[in] Sparksee node OID.
etype	[in] Sparksee edge type identifier.
dir	[in] Direction.

Returns

Objects instance.

neighborsWithObjects:etype:dir:

```
- (STSObjects *) neighborsWithObjects:
             (STSObjects *) objs
             etype:(int) etype
             dir:(enum STSEdgesDirection) dir
```

Selects all neighbor nodes from or to each of the node OID in the given collection and for the given edge type.

Parameters

objs	[in] Sparksee node OID collection.
etype	[in] Sparksee edge type identifier.
dir	[in] Direction.

Returns

Objects instance.

removeAttribute:

```
- (void) removeAttribute:
             (int) attr
```

Removes the given attribute.

Parameters

attr	[in] Sparksee attribute identifier.

removeType:

```
- (void) removeType:
             (int) type
```

Removes the given type.

This fails if there exist attributes defined for the type or if there exist restricted edges referencing this type.

Parameters

type	[in] Sparksee type identifier.

renameAttribute:newName:

```
- (void) renameAttribute:
             (int) attr
             newName:(NSString *) newName
```

Renames an attribute.

The new name must be available.

Parameters

attr	[in] Sparksee attribute identifier.
newName	[in] The new name for the attribute.

renameType:newName:

```
- (void) renameType:
             (int) type
             newName:(NSString *) newName
```

Renames a type.

The new name must be available.

Parameters

type	[in] The type to be renamed.
newName	[in] The new name for the type.

renameTypeWithName:newName:

```
- (void) renameTypeWithName:
             (NSString *) oldName
             newName:(NSString *) newName
```

Renames a type.

The new name must be available.

Parameters

oldName	[in] The current name of the type to be renamed.
newName	[in] The new name for the type.

selectWithAttrValue:cond:value:

```
- (STSObjects *) selectWithAttrValue:
             (int) attr
             cond:(enum STSCondition) cond
             value:(STSValue *) value
```

Selects all OIDs satisfying the given condition for the given attribute.

Parameters

attr	[in] Sparksee attribute identifier.
cond	[in] Condition to be satisfied.
value	[in] Value to be satisfied.

Returns

Objects instance.

selectWithAttrValueRestriction:cond:value:restriction:

```
- (STSObjects *) selectWithAttrValueRestriction:
             (int) attr
             cond:(enum STSCondition) cond
             value:(STSValue *) value
             restriction:(STSObjects *) restriction
```

Selects all OIDs satisfying the given condition for the given attribute.

Parameters

attr	[in] Sparksee attribute identifier.
cond	[in] Condition to be satisfied.
value	[in] Value to be satisfied.
restriction	[in] Objects to limit the select in this set of objects.

Returns

Objects instance.

selectWithAttrValues:cond:lower:higher:

```
- (STSObjects *) selectWithAttrValues:
             (int) attr
             cond:(enum STSCondition) cond
             lower:(STSValue *) lower
             higher:(STSValue *) higher
```

Selects all OIDs satisfying the given condition for the given attribute.

This allows to perform the Between operation, thus it has two Value arguments.

Parameters

attr	[in] Sparksee attribute identifier.
cond	[in] Condition to be satisfied. It must be the Between Condition.
lower	[in] Lower-bound Value to be satisfied.
higher	[in] Higher-bound Value to be satisfied.

Returns

Objects instance.

selectWithAttrValuesRestriction:cond:lower:higher:restriction:

```
- (STSObjects *) selectWithAttrValuesRestriction:
             (int) attr
             cond:(enum STSCondition) cond
             lower:(STSValue *) lower
             higher:(STSValue *) higher
             restriction:(STSObjects *) restriction
```

Selects all OIDs satisfying the given condition for the given attribute.

This allows to perform the Between operation, thus it has two Value arguments.

Parameters

attr	[in] Sparksee attribute identifier.
cond	[in] Condition to be satisfied. It must be the Between Condition.
lower	[in] Lower-bound Value to be satisfied.
higher	[in] Higher-bound Value to be satisfied.
restriction	[in] Objects to limit the select in this set of objects.

Returns

Objects instance.

selectWithType:

```
- (STSObjects *) selectWithType:
             (int) type
```

Selects all OIDs belonging to the given type.

Parameters

type	[in] Sparksee type identifier.

Returns

Objects instance.

setArrayAttribute:attr:value:

```
- (STSValueArray *) setArrayAttribute:
             (long long) oid
             attr:(int) attr
             value:(STSValue *) value
```

Sets all the elements of the ValueArray for the given array attribute and OID and returns it.

Initializing the array with one of these SetArrayAttribute methods is the only way to create the array. But once created it can be updated using the ValueArray. And you can get it again with the GetArrayAttribute operation.

Parameters

oid	[in] Sparksee OID.
attr	[in] Sparksee array attribute identifier.
value	[in] Value for all the array elements of the given attribute and OID.

Returns

A ValueArray. This returns a ValueArray, a NULL or throws an exception.

setArrayAttributeVoid:attr:value:

```
- (void) setArrayAttributeVoid:
             (long long) oid
             attr:(int) attr
             value:(STSValue *) value
```

Sets all the elements of the ValueArray for the given array attribute and OID.

Initializing the array with one of these SetArrayAttribute methods is the only way to create the array. But once created it can be updated using the ValueArray which can be obtained using the GetArrayAttribute operation.

Parameters

oid	[in] Sparksee OID.
attr	[in] Sparksee array attribute identifier.
value	[in] Value for all the array elements of the given attribute and OID.

setAttribute:attr:value:

```
- (void) setAttribute:
             (long long) oid
             attr:(int) attr
             value:(STSValue *) value
```

Sets the Value for the given attribute and OID.

Parameters

oid	[in] Sparksee OID.
attr	[in] Sparksee attribute identifier.
value	[in] Value for the given attribute and for the given OID.

setAttributeDefaultValue:value:

```
- (void) setAttributeDefaultValue:
             (int) attr
             value:(STSValue *) value
```

Sets a default value for an attribute.

The default value will be applied to all the new nodes or edges.

The given value must have the same DataType as the attribute or be a NULL value to remove the current default value.

Parameters

attr	[in] The attribute.
value	[in] The default value to use for this attribute.

setAttributeText:attr:tstream:

```
- (void) setAttributeText:
             (long long) oid
             attr:(int) attr
             tstream:(STSTextStream *) tstream
```

Sets the writable TextStream for the given text attribute and OID.

Parameters

oid	[in] Sparksee OID.
attr	[in] Sparksee attribute identifier.
tstream	[in] New Text value. This corresponds to a TextStream to write.

tails:

```
- (STSObjects *) tails:
             (STSObjects *) edges
```

Gets all the tails from the given edges collection.

Parameters

edges	[in] Sparksee edge identifier collection.

Returns

The tails collection.

tailsAndHeads:tails:heads:

```
- (void) tailsAndHeads:
             (STSObjects *) edges
             tails:(STSObjects *) tails
             heads:(STSObjects *) heads
```

Gets all the tails and heads from the given edges collection.

Parameters

edges	[in] Sparksee edge identifier collection.
tails	[in|out] If not NULL, all the tails will be stored here.
heads	[in|out] If not NULL, all the heads will be stored here.

topkWithAttr:order:k:

```
- (STSKeyValues *) topkWithAttr:
             (int) attribute
             order:(enum STSOrder) order
             k:(int) k
```

Gets a KeyValues iterator as a result of the TopK operation.

Parameters

attribute	The attribute to perform the TopK on
order	The ordering semantincs of the top-k
k	The size of the TopK

Returns

A KeyValues instance

topkWithAttrRestriction:order:k:restriction:

```
- (STSKeyValues *) topkWithAttrRestriction:
              (int) attribute
              order:(enum STSOrder) order
              k:(int) k
              restriction:(STSObjects *) restriction
```

Gets a KeyValues iterator as a result of the TopK operation.

restrictonObjects to limit the topk to this set of objects

Parameters

attribute	The attribute to perform the TopK on
order	The ordering semantincs of the top-k
k	The size of the TopK
restriction	null

Returns

A KeyValues instance

topkWithAttrValue:operation:lower:order:k:

```
- (STSKeyValues *) topkWithAttrValue:
              (int) attribute
              operation:(enum STSCondition) operation
              lower:(STSValue *) lower
              order:(enum STSOrder) order
              k:(int) k
```

Gets a KeyValues iterator as a result of the TopK operation.

Parameters

attribute	The attribute to perform the TopK on
operation	The operation to perform in the top k
lower	The lower bound Value to be satisfied
order	The ordering semantincs of the top-k
k	The size of the TopK

Returns

A KeyValues instance

topkWithAttrValueRestriction:operation:lower:order:k:restriction:

```
- (STSKeyValues *) topkWithAttrValueRestriction:
             (int) attribute
             operation:(enum STSCondition) operation
             lower:(STSValue *) lower
             order:(enum STSOrder) order
             k:(int) k
             restriction:(STSObjects *) restriction
```

Gets a KeyValues iterator as a result of the TopK operation.

restrictonObjects to limit the topk to this set of objects

Parameters

attribute	The attribute to perform the TopK on
operation	The operation to perform in the top k
lower	The lower bound Value to be satisfied
order	The ordering semantincs of the top-k
k	The size of the TopK
restriction	null

Returns

A KeyValues instance

topkWithAttrValues:operation:lower:higher:order:k:

```
- (STSKeyValues *) topkWithAttrValues:
             (int) attribute
             operation:(enum STSCondition) operation
             lower:(STSValue *) lower
             higher:(STSValue *) higher
             order:(enum STSOrder) order
             k:(int) k
```

Gets a KeyValues iterator as a result of the TopK operation.

Parameters

attribute	The attribute to perform the TopK on
operation	The operation to perform in the top k
lower	The lower bound Value to be satisfied
higher	The upper bound Value to be satisfied
order	The ordering semantincs of the top-k
k	The size of the TopK

Returns

A KeyValues instance

topkWithAttrValuesRestriction:operation:lower:higher:order:k:restriction:

```
- (STSKeyValues *) topkWithAttrValuesRestriction:
              (int) attribute
              operation:(enum STSCondition) operation
              lower:(STSValue *) lower
              higher:(STSValue *) higher
              order:(enum STSOrder) order
              k:(int) k
              restriction:(STSObjects *) restriction
```

Gets a KeyValues iterator as a result of the TopK operation.

restrictonObjects to limit the topk to this set of objects

Parameters

attribute	The attribute to perform the TopK on
operation	The operation to perform in the top k
lower	The lower bound Value to be satisfied
higher	The upper bound Value to be satisfied
order	The ordering semantincs of the top-k
k	The size of the TopK
restriction	null

Returns

A KeyValues instance

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

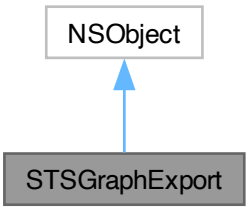
- Sparksee.h

3.26 STSGraphExport Class Reference

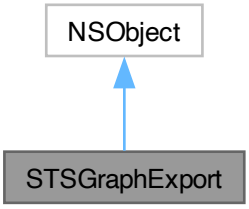
Stores the graph exporting values.

`#include <Sparksee.h>`

Inheritance diagram for STSGraphExport:



Collaboration diagram for STSGraphExport:



Instance Methods

- (id) - **init**

  Creates a new GraphExport instance.
- (void) - **setDefaults**

  Sets to default values.
- (NSString ∗) - getLabel

  Gets the graph label.
- (void) - setLabel:

  Sets the graph label.
- (int) - getColorRGB

  Gets the background color.
- (void) - setColorRGB:

  Sets the background color.
- (void) - getColorRed:green:blue:alpha:

  Get the node color separated in RGBA.
- (void) - setColorRed:green:blue:alpha:

  Set the node color with separated RGBA components.

3.26.1 Detailed Description

Stores the graph exporting values.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.26.2 Method Documentation

getColorRed:green:blue:alpha:

```
- (void) getColorRed:
             (double *) red
            green:(double *) green
            blue:(double *) blue
            alpha:(double *) alpha
```

Get the node color separated in RGBA.

Parameters

red	[out] The red color component ([0..1]).
green	[out] The green color component ([0..1]).
blue	[out] The blue color component ([0..1]).
alpha	[out] The alpha component ([0..1]).

getColorRGB

```
- (int) getColorRGB
```

Gets the background color.

Returns

The background color.

References getColorRGB.

Referenced by getColorRGB.

getLabel

```
- (NSString *) getLabel
```

Gets the graph label.

Returns

The graph label.

References getLabel.

Referenced by getLabel.

setColorRed:green:blue:alpha:

```
- (void) setColorRed:
             (double) red
             green:(double) green
             blue:(double) blue
             alpha:(double) alpha
```

Set the node color with separated RGBA components.

Parameters

red	[in] The red color component ([0..1]).
green	[in] The green color component ([0..1]).
blue	[in] The blue color component ([0..1]).
alpha	[in] The alpha component ([0..1]).

setColorRGB:

```
- (void) setColorRGB:
             (int) color
```

Sets the background color.

Parameters

color	The background color.

setLabel:

```
- (void) setLabel:
             (NSString *) label
```

Sets the graph label.

Parameters

label	[in] The graph label.

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

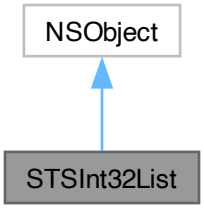
- Sparksee.h

3.27 STSInt32List Class Reference

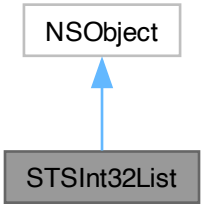
Sparksee 32-bit signed integer list.

`#include <Sparksee.h>`

Inheritance diagram for STSInt32List:



Collaboration diagram for STSInt32List:



Instance Methods

- (int) - count

  Number of elements in the list.
- (id) - init

  Constructor.
- (void) - add:

  Adds an 32-bit signed integer at the end of the list.
- (void) - **clear**

  Clears the list.
- (id) - **initWithArray:**

  Creates a new Int32List instance from the given array.
- (id) - **initWithNSEnumerator:**

  Creates a new Int32List instance from the given NSEnumerator.
- (STSInt32ListIterator ∗) - **iterator**

  Gets a new Int32ListIterator.

3.27.1 Detailed Description

Sparksee 32-bit signed integer list.

It stores a 32-bit signed integer list.

Use Int32ListIterator to access all elements into this collection.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.27.2 Method Documentation

add:

```
- (void) add:
             (int) value
```

Adds an 32-bit signed integer at the end of the list.

Parameters

value	[in] The integer.

count

```
- (int) count
```

Number of elements in the list.

Returns

Number of elements in the list.

References count.

Referenced by count.

init

```
- (id) init
```

Constructor.

This creates an empty list.

References init.

Referenced by init.

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

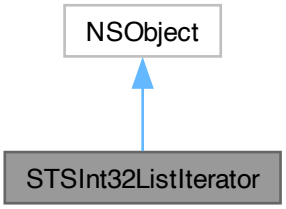
- Sparksee.h

3.28 STSInt32ListIterator Class Reference

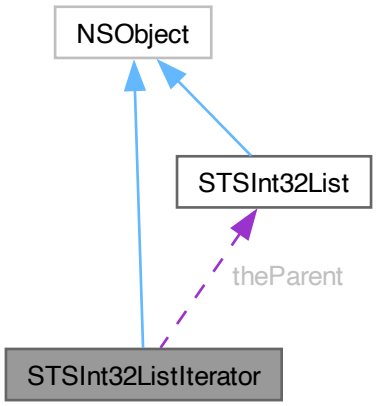
Int32List iterator class.

`#include <Sparksee.h>`

Inheritance diagram for STSInt32ListIterator:



Collaboration diagram for STSInt32ListIterator:



Instance Methods

- (int) - next

  Moves to the next element.
- (BOOL) - hasNext

  Gets if there are more elements.

3.28.1 Detailed Description

Int32List iterator class.

Iterator to traverse all the integer into a Int32List instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.28.2 Method Documentation

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements.

Returns

TRUE if there are more elements, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (int) next
```

Moves to the next element.

Returns

The next element.

References next.

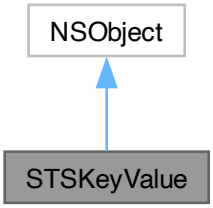
Referenced by next.

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

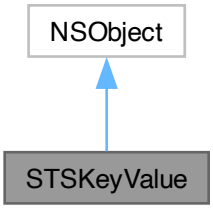
- Sparksee.h

3.29 STSKeyValue Class Reference

Inheritance diagram for STSKeyValue:



Collaboration diagram for STSKeyValue:



Instance Methods

- (STSValue ∗) - getValue

  Gets the value of this KeyValue pair.
- (long long) - getKey

  Gets the key (OID) of this KeyValue pair.
- (NSString ∗) - **stringValue**

  Returns the receiver's value as a human-readable string.
- (BOOL) - **isEqual:**

  Check if both Value instances are equal.
- (NSUInteger) - **hash**

  Get the hash value of this Value.

3.29.1 Method Documentation

getKey

```
- (long long) getKey
```

Gets the key (OID) of this KeyValue pair.

Returns

The Sparksee object identifier for this entry.

References getKey.

Referenced by getKey.

getValue

```
- (STSValue *) getValue
```

Gets the value of this KeyValue pair.

Returns

The Value associated with this entry.

References getValue.

Referenced by getValue.

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

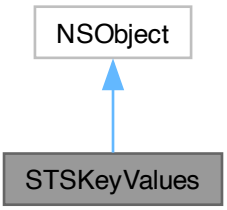
- Sparksee.h

3.30 STSKeyValues Class Reference

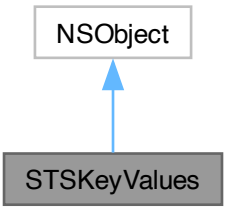
Value set class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSKeyValues:



Collaboration diagram for STSKeyValues:



Instance Methods

- (BOOL) - hasNext

  Checks if the KeyValues has more KeyValue pairs.
- (STSKeyValue ∗) - next

  Gets the next KeyValue pair.
- (void) - nextKeyValue:

  Gets the next KeyValue pair.
- (void) - close

  Closes the KeyValues instance.
- (BOOL) - **isClosed**

  Check if the KeyValues instance is closed.

3.30.1 Detailed Description

Value set class.

This is a set of Value instances, that is there is no duplicated elements.

Use a ValuesIterator to traverse all the elements into the set.

When the Values instance is closed, it closes all existing and non-closed ValuesIterator instances too.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.30.2 Method Documentation

close

```
- (void) close
```

Closes the KeyValues instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

hasNext

```
- (BOOL) hasNext
```

Checks if the KeyValues has more KeyValue pairs.

Returns

Returns ture if there are more KeyValue pairs

References hasNext.

Referenced by hasNext.

next

```
- (STSKeyValue *) next
```

Gets the next KeyValue pair.

Returns

Returns the next KeyValue pair

References next.

Referenced by next.

nextKeyValue:

```
- (void) nextKeyValue:
             (STSKeyValue *) kv
```

Gets the next KeyValue pair.

Parameters

kv	Returns the next KeyValue pair

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

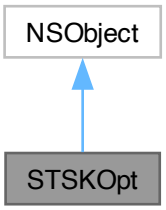
- Sparksee.h

3.31 STSKOpt Class Reference

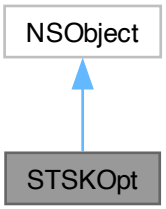
KOpt class.

`#include <Sparksee.h>`

Inheritance diagram for STSKOpt:



Collaboration diagram for STSKOpt:



Instance Methods

- (void) - close

  Closes the KOpt instance.
- (BOOL) - **isClosed**

  Check if the KOpt instance is closed.

3.31.1 Detailed Description

KOpt class.

Implements the 2-Opt and 3-Opt algorithms

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.31.2 Method Documentation

close

```
- (void) close
```

Closes the KOpt instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

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

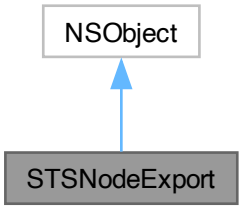
- Sparksee.h

3.32 STSNodeExport Class Reference

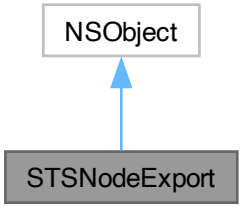
Stores the node exporting values.

`#include <Sparksee.h>`

Inheritance diagram for STSNodeExport:



Collaboration diagram for STSNodeExport:



Instance Methods

- (id) - **init**

  Creates a new instance.
- (void) - **setDefaults**

  Sets to default values.
- (void) - copyDefaults:

  Sets the default values using a refrerence NodeExport.
- (NSString ∗) - getLabel

  Gets the node label.
- (void) - setLabel:

  Sets the node label.
- (NSString ∗) - getTitle

  Gets the node title.
- (void) - setTitle:

  Sets the node title.
- (enum STSNodeShape) - getShape

  Gets the node shape.
- (void) - setShape:

  Sets the node shape.
- (int) - getColorRGB

  Gets the node color.
- (void) - setColorRGB:

  Sets the node color.
- (int) - getLabelColorRGB

  Gets the node label color.
- (void) - setLabelColorRGB:

  Sets the node label color.
- (int) - getHeight

  Gets the node height.
- (void) - setHeight:

  Sets the node height.
- (int) - getWidth

  Gets the node width.
- (void) - setWidth:

  Gets the node width.
- (BOOL) - isFit

  Gets whether the node size is fitted to the label or not.
- (void) - setFit:

  Sets the node fit property.
- (int) - getFontSize

  Gets the node label font size.
- (void) - setFontSize:

  Sets the node label font size.
- (NSString ∗) - getImage

  Gets the node image.
- (void) - setImage:

  Sets the node image.
- (void) - getColorRed:green:blue:alpha:

  Get the node color separated in RGBA.
- (void) - setColorRed:green:blue:alpha:

Set the node color with separated RGBA components.

- (void) - getLabelColorRed:green:blue:alpha:

  Get the node label color separated in RGBA.

- (void) - setLabelColorRed:green:blue:alpha:

  Set the node label color with separated RGBA components.

3.32.1 Detailed Description

Stores the node exporting values.

When 'fit' is set to TRUE, then 'height' and 'width' will be ignored.

Some properties may be ignored depending on the exportation type.

Default values are:

Label: "" (empty string).

Shape: Box.

Color: 10863606 (0xa5c3f6).

Label color: 0 (Ox000000, Black).

Height: 25px.

Width: 25px.

Fit: TRUE.

Font size: 10.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.32.2 Method Documentation

copyDefaults:

```
- (void) copyDefaults:
             (STSNodeExport *) source
```

Sets the default values using a refrerence NodeExport.

Parameters

source	null

getColorRed:green:blue:alpha:

```
- (void) getColorRed:
             (double *) red
            green:(double *) green
            blue:(double *) blue
            alpha:(double *) alpha
```

Get the node color separated in RGBA.

Parameters

red	[out] The red color component ([0..1]).
green	[out] The green color component ([0..1]).
blue	[out] The blue color component ([0..1]).
alpha	[out] The alpha component ([0..1]).

getColorRGB

```
- (int) getColorRGB
```

Gets the node color.

Returns

The node color.

References getColorRGB.

Referenced by getColorRGB.

getFontSize

```
- (int) getFontSize
```

Gets the node label font size.

Returns

The node label font size.

References getFontSize.

Referenced by getFontSize.

getHeight

```
- (int) getHeight
```

Gets the node height.

Returns

The node height in pixels.

References getHeight.

Referenced by getHeight.

getImage

```
- (NSString *) getImage
```

Gets the node image.

Returns

The node image.

References getImage.

Referenced by getImage.

getLabel

```
- (NSString *) getLabel
```

Gets the node label.

Returns

The node label.

References getLabel.

Referenced by getLabel.

getLabelColorRed:green:blue:alpha:

```
- (void) getLabelColorRed:
             (double *) red
            green:(double *) green
            blue:(double *) blue
            alpha:(double *) alpha
```

Get the node label color separated in RGBA.

Parameters

red	[out] The red color component ([0..1]).
green	[out] The green color component ([0..1]).
blue	[out] The blue color component ([0..1]).
alpha	[out] The alpha component ([0..1]).

getLabelColorRGB

```
- (int) getLabelColorRGB
```

Gets the node label color.

Returns

The node label color.

References getLabelColorRGB.

Referenced by getLabelColorRGB.

getShape

```
- (enum STSNodeShape) getShape
```

Gets the node shape.

Returns

The node shape.

References getShape.

Referenced by getShape.

getTitle

```
- (NSString *) getTitle
```

Gets the node title.

Returns

The node title.

References getTitle.

Referenced by getTitle.

getWidth

```
- (int) getWidth
```

Gets the node width.

Returns

The node width in pixels.

References getWidth.

Referenced by getWidth.

isFit

```
- (BOOL) isFit
```

Gets whether the node size is fitted to the label or not.

Returns

If TRUE, then the node size is fitted to the label, otherwise the size is fixed with the values of 'height' and 'width'.

References isFit.

Referenced by isFit.

setColorRed:green:blue:alpha:

```
- (void) setColorRed:
             (double) red
             green:(double) green
             blue:(double) blue
             alpha:(double) alpha
```

Set the node color with separated RGBA components.

Parameters

red	[in] The red color component ([0..1]).
green	[in] The green color component ([0..1]).
blue	[in] The blue color component ([0..1]).
alpha	[in] The alpha component ([0..1]).

setColorRGB:

```
- (void) setColorRGB:
             (int) color
```

Sets the node color.

Parameters

color	The node color.

setFit:

```
- (void) setFit:
             (BOOL) fit
```

Sets the node fit property.

Parameters

fit	[in] If TRUE, then the node size is fitted to the label ('height' and 'width' will be ignored), otherwise the size is fixed with the values of 'height' and 'width'.

setFontSize:

```
- (void) setFontSize:
             (int) size
```

Sets the node label font size.

Parameters

size	[in] The node label font size.

setHeight:

```
- (void) setHeight:
             (int) height
```

Sets the node height.

Parameters

height	[in] The node height in pixels.

setImage:

```
- (void) setImage:
             (NSString *) image
```

Sets the node image.

Parameters

image	[in] The node image. The image should be a path to a local file or an URL. The image usage in the export depends on the exported format and shape. For Graphviz: Graphviz expects only local image paths as images in dot files unless you plan on using something like it's "dot_url_resolve.py" script to automatically download them. The Image shape is not directly supported for Graphviz. For Html: The vis.js html exports only show the image if the node shape selected is Image or Circle (that will be translated as "circularimage").

setLabel:

```
- (void) setLabel:
             (NSString *) label
```

Sets the node label.

Parameters

label	[in] The node label.

setLabelColorRed:green:blue:alpha:

```
- (void) setLabelColorRed:
             (double) red
            green:(double) green
            blue:(double) blue
            alpha:(double) alpha
```

Set the node label color with separated RGBA components.

Parameters

red	[in] The red color component ([0..1]).
green	[in] The green color component ([0..1]).
blue	[in] The blue color component ([0..1]).
alpha	[in] The alpha component ([0..1]).

setLabelColorRGB:

```
- (void) setLabelColorRGB:
             (int) color
```

Sets the node label color.

Parameters

color	[in] The node label color.

setShape:

```
- (void) setShape:
             (enum STSNodeShape) shape
```

Sets the node shape.

Parameters

shape	[in] The node shape.

setTitle:

```
- (void) setTitle:
             (NSString *) title
```

Sets the node title.

Parameters

title	[in] The node title.

setWidth:

```
- (void) setWidth:
             (int) width
```

Gets the node width.

Parameters

width	The node width in pixels.

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

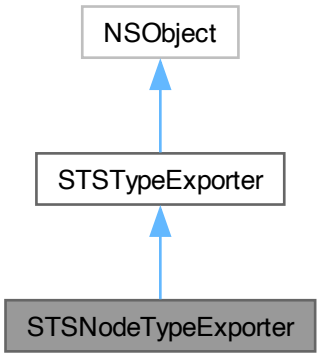
- Sparksee.h

3.33 STSNodeTypeExporter Class Reference

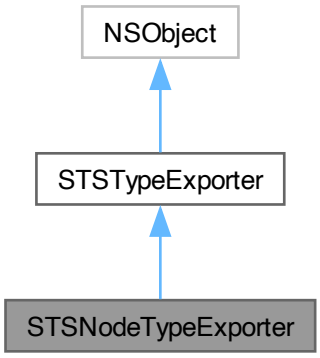
NodeTypeExporter class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSNodeTypeExporter:



Collaboration diagram for STSNodeTypeExporter:



Instance Methods

- (id) - **init**

  Creates a new instance.
- (id) - initWithRowWriter:graph:type:attrs:

  Creates a new instance.
- (void) - run

  See the TypeExporter class Run method.
- (void) - registerListener:

  Registers a new listener.
- (void) - setRowWriter:

  Sets the output data destination.
- (void) - setGraph:

  Sets the graph that will be exported.
- (void) - setType:

  Sets the type to be exported.
- (void) - setAttributes:

  Sets the list of Attributes.
- (void) - setFrequency:

  Sets the frequency of listener notification.
- (void) - setHeader:

  Sets the presence of a header row.

3.33.1 Detailed Description

NodeTypeExporter class.

Specific TypeExporter implementation for node types.

Check out the 'Data export' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.33.2 Method Documentation

initWithRowWriter:graph:type:attrs:

```
- (id) initWithRowWriter:
             (STSRowWriter *) rowWriter
             graph:(STSGraph *) graph
             type:(int) type
             attrs:(STSAttributeList *) attrs
```

Creates a new instance.

Parameters

rowWriter	[in] Output RowWriter.
graph	[in] Graph.
type	[in] Type identifier.
attrs	[in] Attribute identifiers to be exported.

registerListener:

```
- (void) registerListener:
             (STSTypeExporterListener *) tel
```

Registers a new listener.

Parameters

tel	[in] TypeExporterListener to be registered.

run

```
- (void) run
```

See the TypeExporter class Run method.

WrongArgumentErrorIn case of bad settings ErrorFor other issues

Exceptions

System.ApplicationException	null
System.ArgumentException	null
System.IO.IOException	If bad things happen writting to the RowWriter.

Implements STSTypeExporter.

References run.

Referenced by run.

setAttributes:

```
- (void) setAttributes:
             (STSAttributeList *) attrs
```

Sets the list of Attributes.

Parameters

attrs	[in] Attribute identifiers to be exported

setFrequency:

```
- (void) setFrequency:
             (int) freq
```

Sets the frequency of listener notification.

Parameters

freq	[in] Frequency in number of rows managed to notify progress to all listeners

setGraph:

```
- (void) setGraph:
             (STSGraph *) graph
```

Sets the graph that will be exported.

Parameters

graph	[in] Graph.

setHeader:

```
- (void) setHeader:
             (BOOL) header
```

Sets the presence of a header row.

Parameters

header	[in] If TRUE, a header row is dumped with the name of the attributes.

setRowWriter:

```
- (void) setRowWriter:
              (STSRowWriter *) rw
```

Sets the output data destination.

Parameters

rw	[in] Input RowWriter.

setType:

```
- (void) setType:
              (int) type
```

Sets the type to be exported.

Parameters

type	[in] Type identifier.

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

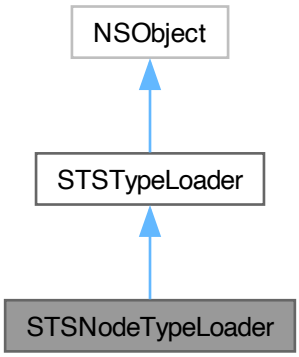
- Sparksee.h

3.34 STSNodeTypeLoader Class Reference

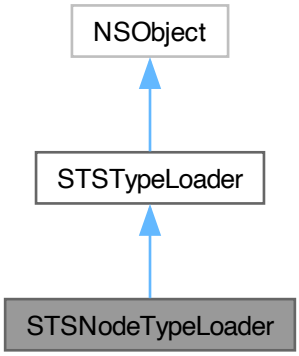
NodeTypeLoader class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSNodeTypeLoader:



Collaboration diagram for STSNodeTypeLoader:



Instance Methods

- (id) - **init**

  Creates a new instance.
- (id) - initWithRowReader:graph:type:attrs:attrsPos:

  Creates a new instance.
- (void) - run

  See the TypeLoader class Run method.
- (void) - runTwoPhases

  See the TypeLoader class RunTwoPhases method.
- (void) - runNPhases:

  See the TypeLoader class RunNPhases method.
- (void) - setLogError:

  Sets a log error file.
- (void) - setLogOff

  Truns off all the error reporting.
- (void) - registerListener:

  Registers a new listener.
- (void) - setRowReader:

  Sets the input data source.
- (void) - setGraph:

  Sets the graph where the data will be loaded.
- (void) - setLocale:

  Sets the locale that will be used to read the data.
- (void) - setType:

  Sets the type to be loaded.
- (void) - setAttributes:

  Sets the list of Attributes.
- (void) - setAttributePositions:

  Sets the list of attribute positions.
- (void) - setTimestampFormat:

  Sets a specific timestamp format.
- (void) - setFrequency:

  Sets the frequency of listener notification.

3.34.1 Detailed Description

NodeTypeLoader class.

Specific TypeLoader implementation for node types.

Check out the 'Data import' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.34.2 Method Documentation

initWithRowReader:graph:type:attrs:attrsPos:

```
- (id) initWithRowReader:
              (STSRowReader *) rowReader
              graph:(STSGraph *) graph
              type:(int) type
              attrs:(STSAttributeList *) attrs
              attrsPos:(STSInt32List *) attrsPos
```

Creates a new instance.

Parameters

rowReader	[in] Input RowReader.
graph	[in] Graph.
type	[in] Type identifier.
attrs	[in] Attribute identifiers to be loaded.
attrsPos	[in] Attribute positions (column index >=0).

registerListener:

```
- (void) registerListener:
              (STSTypeLoaderListener *) tel
```

Registers a new listener.

Parameters

tel	TypeLoaderListener to be registered.

run

```
- (void) run
```

See the TypeLoader class Run method.

ErrorFor other issues

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen using to the RowReader.

Implements STSTypeLoader.

References run.

Referenced by run.

runNPhases:

```
- (void) runNPhases:
             (int) partitions
```

See the TypeLoader class RunNPhases method.

ErrorFor other issues

Parameters

partitions	null

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen using to the RowReader.

Implements STSTypeLoader.

runTwoPhases

```
- (void) runTwoPhases
```

See the TypeLoader class RunTwoPhases method.

ErrorFor other issues

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen using to the RowReader.

Implements STSTypeLoader.

References runTwoPhases.

Referenced by runTwoPhases.

setAttributePositions:

```
- (void) setAttributePositions:
             (STSInt32List *) attrsPos
```

Sets the list of attribute positions.

Parameters

attrsPos	[in] Attribute positions (column index >=0).

setAttributes:

```
- (void) setAttributes:
             (STSAttributeList *) attrs
```

Sets the list of Attributes.

Parameters

attrs	[in] Attribute identifiers to be loaded

setFrequency:

```
- (void) setFrequency:
             (int) freq
```

Sets the frequency of listener notification.

Parameters

freq	[in] Frequency in number of rows managed to notify progress to all listeners

setGraph:

```
- (void) setGraph:
             (STSGraph *) graph
```

Sets the graph where the data will be loaded.

Parameters

graph	[in] Graph.

setLocale:

```
- (void) setLocale:
             (NSString *) localeStr
```

Sets the locale that will be used to read the data.

It should match the locale used in the rowreader.

Parameters

localeStr	[in] The locale string for the read data. See CSVReader.

setLogError:

```
- (void) setLogError:
             (NSString *) path
```

Sets a log error file.

By default errors are thrown as a exception and the load process ends. If a log file is set, errors are logged there and the load process does not stop.

Parameters

path	[in] The path to the error log file.

Exceptions

System.IO.IOException	If bad things happen opening the file.

setLogOff

```
- (void) setLogOff
```

Truns off all the error reporting.

The log file will not be created and no exceptions for invalid data will be thrown. If you just want to turn off the logs, but abort at the first error what you should do is not call this method and not set a logError file.

References setLogOff.

Referenced by setLogOff.

setRowReader:

```
- (void) setRowReader:
              (STSRowReader *) rr
```

Sets the input data source.

Parameters

rr	[in] Input RowReader.

setTimestampFormat:

```
- (void) setTimestampFormat:
              (NSString *) timestampFormat
```

Sets a specific timestamp format.

Parameters

timestampFormat	[in] A string with the timestamp format definition.

setType:

```
- (void) setType:
              (int) type
```

Sets the type to be loaded.

Parameters

type	[in] Type identifier.

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

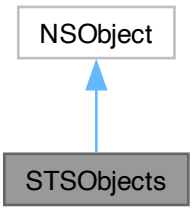
- Sparksee.h

3.35 STSObjects Class Reference

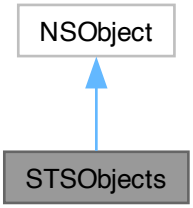
Object identifier set class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSObjects:



Collaboration diagram for STSObjects:



Instance Methods

- (STSObjects *) - clone

  Creates a new Objects instance as a copy of the given one.
- (long long) - count

  Gets the number of elements into the collection.
- (BOOL) - add:

  Adds an element into the collection.
- (BOOL) - exists:

  Gets if the given element exists into the collection.
- (long long) - any

  Gets an element from the collection.
- (BOOL) - remove:

  Removes an element from the collection.
- (void) - **clear**

  Clears the collection removing all its elements.
- (long long) - union:

  Performs the union operation.
- (long long) - intersection:

  Performs the intersection operation.

- (long long) - difference:

  Performs the difference operation.
- (BOOL) - equals:

  Checks if the given Objects contains the same information.
- (BOOL) - contains:

  Check if this objects contains the other one.
- (long long) - cloneWithObjects:

  Performs the copy operation.
- (STSObjects *) - sample:samples:

  Creates a new Objects instance which is a sample of the calling one.
- (STSObjectsIterator *) - iterator

  Gets an ObjectsIterator.
- (STSObjectsIterator *) - iteratorFromIndex:

  Gets an ObjectsIterator skipping index elements.
- (STSObjectsIterator *) - iteratorFromElement:

  Gets an ObjectsIterator starting from the given element.
- (void) - close

  Closes the Objects instance.
- (BOOL) - **isClosed**

  Check if the Objects instance is closed.
- (BOOL) - **isEqual:**

  Check if both Objects instances are equal.
- (NSUInteger) - **hash**

  Get the hash value of this Objects.

Class Methods

- (long long) + **getInvalidOID**

  Invalid OID constant.
- (STSObjects *) + combineUnion:objs2:

  Creates a new Objects instance which is the union of the two given.
- (STSObjects *) + combineIntersection:objs2:

  Creates a new Objects instance which is the intersection of the two given.
- (STSObjects *) + combineDifference:objs2:

  Creates a new Objects instance which is the difference of the two given.

3.35.1 Detailed Description

Object identifier set class.

It stores a collection of Sparksee object identifiers as a set. As a set, there is no order and no duplicated elements.

This class should be used just to store large collections. Otherwise, it is strongly recommended to use common classes from the language API.

This class is not thread-safe.

ObjectsIterator must be used to traverse all the elements into the set.

When the Objects instance is closed, it closes all existing and non-closed ObjectsIterator instances too.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.35.2 Method Documentation

add:

```
- (BOOL) add:
             (long long) e
```

Adds an element into the collection.

Parameters

e	[in] Element to be added.

Returns

TRUE if the element is added, FALSE if the element was already into the collection.

any

```
- (long long) any
```

Gets an element from the collection.

Errorfor other issues.

Returns

Any element from the collection.

Exceptions

System.ApplicationException	null
System.ApplicationException	whether the collection is empty.

References any.

Referenced by any.

clone

```
- (STSObjects *) clone
```

Creates a new Objects instance as a copy of the given one.

Returns

The new Objects instance.

References clone.

Referenced by clone.

cloneWithObjects:

```
- (long long) cloneWithObjects:
              (STSObjects *) objs
```

Performs the copy operation.

This updates the Objects calling instance and copies the given Objects instance.

Parameters

objs	[in] Objects instance.

Returns

Number of elements into the collection once the operation has been executed.

close

```
- (void) close
```

Closes the Objects instance.

It must be called before closing the Session to ensure the integrity of all data.

References close.

Referenced by close.

combineDifference:objs2:

```
+ (STSObjects *) combineDifference:
              (STSObjects *) objs1
              objs2:(STSObjects *) objs2
```

Creates a new Objects instance which is the difference of the two given.

Two given Objects belong to the same Session.

Parameters

objs1	[in] Objects instance.
objs2	[in] Objects instance.

Returns

New Objects instance.

combineIntersection:objs2:

```
+ (STSObjects *) combineIntersection:
              (STSObjects *) objs1
              objs2:(STSObjects *) objs2
```

Creates a new Objects instance which is the intersection of the two given.

Two given Objects belong to the same Session.

Parameters

objs1	[in] Objects instance.
objs2	[in] Objects instance.

Returns

New Objects instance.

combineUnion:objs2:

```
+ (STSObjects *) combineUnion:
              (STSObjects *) objs1
              objs2:(STSObjects *) objs2
```

Creates a new Objects instance which is the union of the two given.

Two given Objects belong to the same Session.

Parameters

objs1	[in] Objects instance.
objs2	[in] Objects instance.

Returns

New Objects instance.

contains:

```
- (BOOL) contains:
              (STSObjects *) objs
```

Check if this objects contains the other one.

Parameters

objs	Objects collection.

Returns

True if it contains the given object.

count

```
- (long long) count
```

Gets the number of elements into the collection.

Returns

The number of elements into the collection.

References count.

Referenced by count.

difference:

```
- (long long) difference:
              (STSObjects *) objs
```

Performs the difference operation.

This updates the Objects calling instance removing those existing elements at the given Objects instance.

Parameters

objs	[in] Objects instance.

Returns

Number of elements into the collection once the operation has been executed.

equals:

```
- (BOOL) equals:
             (STSObjects *) objs
```

Checks if the given Objects contains the same information.

Parameters

objs	[in] Objects instance.

Returns

True if the objects are equal or false otherwise.

exists:

```
- (BOOL) exists:
             (long long) e
```

Gets if the given element exists into the collection.

Parameters

e	[in] Element.

Returns

TRUE if the element exists into the collection, FALSE otherwise.

intersection:

```
- (long long) intersection:
             (STSObjects *) objs
```

Performs the intersection operation.

Updates the Objects calling instance setting those existing elements at both two collections and removing all others.

Parameters

objs	[in] Objects instance.

Returns

Number of elements into the collection once the operation has been executed.

iterator

```
- (STSObjectsIterator *) iterator
```

Gets an ObjectsIterator.

Returns

ObjectsIterator instance.

References iterator.

Referenced by iterator.

iteratorFromElement:

```
- (STSObjectsIterator *) iteratorFromElement:
             (long long) e
```

Gets an ObjectsIterator starting from the given element.

Objects collection has no order, so this method is implementation-dependent. e[in] The first element to traverse in the resulting

Parameters

e	[in] The first element to traverse in the resulting ObjectsIterator instance.

Returns

ObjectsIterator instance.

iteratorFromIndex:

```
- (STSObjectsIterator *) iteratorFromIndex:
             (long long) index
```

Gets an ObjectsIterator skipping index elements.

Objects collection has no order, so this method is implementation-dependent.

Parameters

index	[in] The number of elements to skip from the beginning. It must be in the range [0..Size).

Returns

ObjectsIterator instance.

remove:

```
- (BOOL) remove:
             (long long) e
```

Removes an element from the collection.

Parameters

e	[in] Element to be removed.

Returns

TRUE if the element is removed, FALSE if the element was not into the collection.

sample:samples:

```
- (STSObjects *) sample:
             (STSObjects *) exclude
             samples:(long long) samples
```

Creates a new Objects instance which is a sample of the calling one.

Parameters

exclude	[in] If not NULL, elements into this collection will be excluded from the resulting one.
samples	[in] Number of elements into the resulting collection.

Returns

Sample collection.

union:

```
- (long long) union:
             (STSObjects *) objs
```

Performs the union operation.

This adds all existing elements of the given Objects instance to the Objects calling instance

Parameters

objs	[in] Objects instance.

Returns

Number of elements into the collection once the operation has been executed.

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

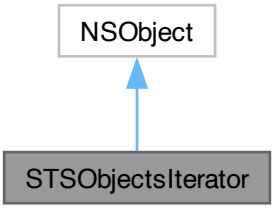
- Sparksee.h

3.36 STSObjectsIterator Class Reference

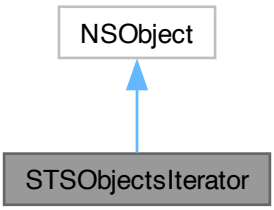
ObjectsIterator class.

`#include <Sparksee.h>`

Inheritance diagram for STSObjectsIterator:



Collaboration diagram for STSObjectsIterator:



Instance Methods

- (BOOL) - hasNext

  Gets if there are more elements to traverse.
- (long long) - next

  Gets the next element to traverse.
- (void) - close

  Closes the ObjectsIterator instance.
- (BOOL) - **isClosed**

  Check if the ObjectsIterator instance is closed.

3.36.1 Detailed Description

ObjectsIterator class.

Iterator to traverse all the object identifiers from an Objects instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.36.2 Method Documentation

close

```
- (void) close
```

Closes the ObjectsIterator instance.

It must be called before closing the parent Objects.

References close.

Referenced by close.

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements to traverse.

Returns

TRUE if there are more elements to traverse, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (long long) next
```

Gets the next element to traverse.

Returns

The next element.

References next.

Referenced by next.

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

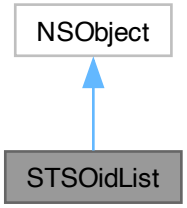
- Sparksee.h

3.37 STSOidList Class Reference

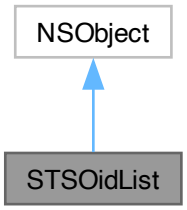
Sparksee object identifier list.

`#include <Sparksee.h>`

Inheritance diagram for STSOidList:



Collaboration diagram for STSOidList:



Instance Methods

- (int) - count

  Number of elements in the list.
- (id) - init

  Constructor.
- (id) - initWithNumInvalidOIDs:

  Constructor.
- (void) - add:

  Adds a Sparksee object identifier at the end of the list.
- (void) - set:oid:

  Sets a Sparksee object identifier at the specified position of the list.
- (void) - **clear**

  Clears the list.

- (id) - **initWithArray:**

  Creates a new OIDList instance from the given array.
- (id) - **initWithNSEnumerator:**

  Creates a new OIDList instance from the given NSEnumerator.
- (STSOidListIterator ∗) - **iterator**

  Gets a new OIDListIterator.

3.37.1 Detailed Description

Sparksee object identifier list.

It stores a Sparksee object identifier list.

Use OIDListIterator to access all elements into this collection.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.37.2 Method Documentation

add:

```
- (void) add:
             (long long) attr
```

Adds a Sparksee object identifier at the end of the list.

Parameters

attr	[in] Sparksee object identifier.

count

```
- (int) count
```

Number of elements in the list.

Returns

Number of elements in the list.

References count.

Referenced by count.

init

```
- (id) init
```

Constructor.

This creates an empty list.

References init.

Referenced by init.

initWithNumInvalidOIDs:

```
- (id) initWithNumInvalidOIDs:
              (int) numInvalidOIDs
```

Constructor.

This creates a list with N invalid oids.

Parameters

numInvalidOIDs	[in] The number of invalid oids added to the list.

set:oid:

```
- (void) set:
              (int) pos
             oid:(long long) oid
```

Sets a Sparksee object identifier at the specified position of the list.

Parameters

pos	[in] List position [0..Count()-1].
oid	[in] Sparksee object identifier.

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

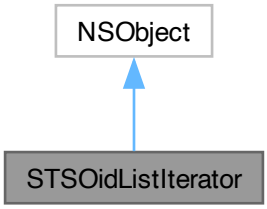
- Sparksee.h

3.38 STSOidListIterator Class Reference

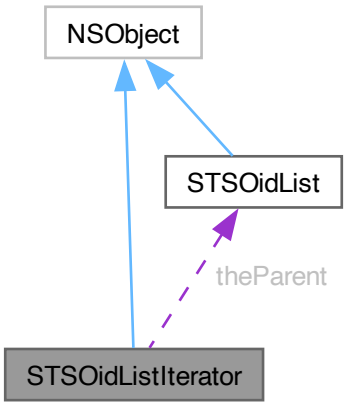
OIDList iterator class.

`#include <Sparksee.h>`

Inheritance diagram for STSOidListIterator:



Collaboration diagram for STSOidListIterator:



Instance Methods

- (long long) - next

  Moves to the next element.
- (BOOL) - hasNext

  Gets if there are more elements.

3.38.1 Detailed Description

OIDList iterator class.

Iterator to traverse all the Sparksee object identifier into a OIDList instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.38.2 Method Documentation

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements.

Returns

TRUE if there are more elements, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (long long) next
```

Moves to the next element.

Returns

The next element.

References next.

Referenced by next.

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

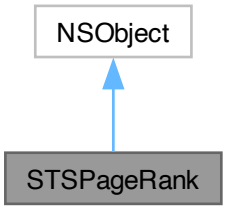
- Sparksee.h

3.39 STSPageRank Class Reference

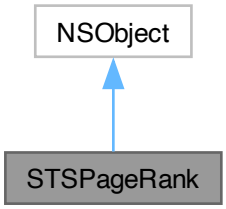
PageRank class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSPageRank:



Collaboration diagram for STSPageRank:



Instance Methods

- (id) - initWithSession:

  Builds the PageRank.
- (void) - addEdgeType:dir:

  Allows for traversing edges of the given type.
- (void) - addAllEdgeTypes:

  Allows for traversing all edge types of the graph.
- (void) - addNodeType:

  Allows for traversing nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows for traversing all node types of the graph.
- (void) - setNumIterations:

  Sets the number of iterations to run the PageRank for.

- (void) - setTolerance:

  Sets the tolerance threashold to continue computing the PageRank after each iteration.
- (void) - setDamping:

  Sets the damping value for the PageRank.
- (void) - setInitialPageRankValue:

  Sets the initial PageRank value.
- (void) - setOutputAttributeType:

  Sets the output attribute type.
- (void) - setEdgeWeightAttributeType:

  Sets the attribute to use as edge weight.
- (void) - setDefaultWeight:

  Sets the default weight for those cases when a given edge does not have a weight attribute set.
- (void) - setStartingNode:

  Sets the starting node of the page rank to compute the Personalized PageRank variant.
- (void) - run

  Runs the algorithm.
- (void) - close

  Closes the PageRank instance.
- (BOOL) - **isClosed**

  Check if the PageRank instance is closed.

3.39.1 Detailed Description

PageRank class.

Implements the PageRank algorithm

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.39.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows for traversing all edge types of the graph.

The direction is interpreted as in which direction an edge can be followed from a node to influence other nodes.

Parameters

dir	[in] Edge direction.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
             dir:(enum STSEdgesDirection) dir
```

Allows for traversing edges of the given type.

If the edge type was already added, the existing direction is overwritten The direction is interpreted as in which direction an edge can be followed from a node to influence other nodes.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

Exceptions

System.ApplicationException	null

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows for traversing nodes of the given type.

Parameters

type	null

Exceptions

System.ApplicationException	null

close

```
- (void) close
```

Closes the PageRank instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

initWithSession:

```
- (id) initWithSession:
             (STSSession *) session
```

Builds the PageRank.

Parameters

session	[in] The session to use

run

```
- (void) run
```

Runs the algorithm.

sparksee::gdb::Error

Exceptions

System.ApplicationException	null

References run.

Referenced by run.

setDamping:

```
- (void) setDamping:
             (double) damping
```

Sets the damping value for the PageRank.

Parameters

damping	[in] The damping value. Default: 0.85

setDefaultWeight:

```
- (void) setDefaultWeight:
             (double) weight
```

Sets the default weight for those cases when a given edge does not have a weight attribute set.

Default: 0.0

Parameters

weight	[in] The default weight

setEdgeWeightAttributeType:

```
- (void) setEdgeWeightAttributeType:
             (int) attr
```

Sets the attribute to use as edge weight.

If the multiple edge are set for traversal, this attribute must be of type GLOBAL_TYPE or EDGES_TYPE. Additionally, the attribute must be of type Double. Finally, negative weights are treated as non existing, so the default weight applies.

sparksee::gdb::Error

Parameters

attr	[in] The attribute type to use as a weight. Default: InvalidAttribute

Exceptions

System.ApplicationException	null

setInitialPageRankValue:

```
- (void) setInitialPageRankValue:
             (double) startValue
```

Sets the initial PageRank value.

If a starting node is set, this initial value is only set for the starting node and the rest of nodes are set to 0.0

Parameters

startValue	[in] The initial value to set. Default: 0.0

setNumIterations:

```
- (void) setNumIterations:
             (int) numIterations
```

Sets the number of iterations to run the PageRank for.

Parameters

numIterations	[in] The number of interations to set. Default: 20

setOutputAttributeType:

```
- (void) setOutputAttributeType:
             (int) attr
```

Sets the output attribute type.

If the PageRank will run on more than one node type, then the output attribute must be of type GLOBAL_TYPE or NODES_TYPE. Otherwise, it must be a valid attribute for the used node type.

Parameters

attr	[in] The attribute to store the result. Default: InvalidAttribute

Exceptions

System.ApplicationException	null

setStartingNode:

```
- (void) setStartingNode:
             (long long) startNode
```

Sets the starting node of the page rank to compute the Personalized PageRank variant.

sparksee::gdb::Error

Parameters

startNode	null

Exceptions

System.ApplicationException	null

setTolerance:

```
- (void) setTolerance:
            (double) tolerance
```

Sets the tolerance threashold to continue computing the PageRank after each iteration.

If all the changes to any PPR value after an iteration are below that tolerance threshold, the algorithm finishes.

Parameters

tolerance	[in] The tolerance to use normalized between 0 and 1. Default: 0.000001

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

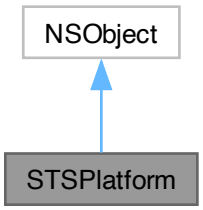
- Sparksee.h

3.40 STSPlatform Class Reference

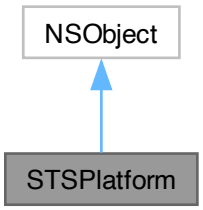
Platform class.

`#include <Sparksee.h>`

Inheritance diagram for STSPlatform:



Collaboration diagram for STSPlatform:



Class Methods

- (void) + getStatistics:

  Gets platform data and statistics.

3.40.1 Detailed Description

Platform class.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.40.2 Method Documentation

getStatistics:

```
+ (void) getStatistics:
             (STSPlatformStatistics *) stats
```

Gets platform data and statistics.

Parameters

stats	[in|out] This updates the given PlatformStatistics.

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

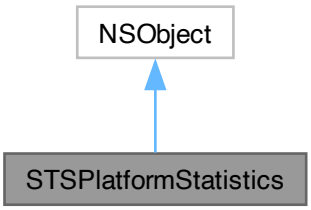
- Sparksee.h

3.41 STSPlatformStatistics Class Reference

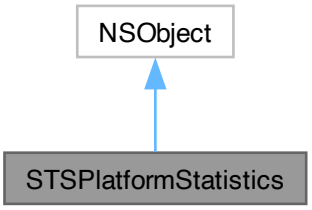
Platform data and statistics.

```
#include <Sparksee.h>
```

Inheritance diagram for STSPlatformStatistics:



Collaboration diagram for STSPlatformStatistics:



Instance Methods

- (id) - **init**

  Creates a new instance setting all values to 0.
- (int) - getNumCPUs

  Gets the number of CPUs.
- (long long) - getRealTime

  Gets time in microseconds (since epoch).
- (long long) - getUserTime

  Gets CPU user time.
- (long long) - getSystemTime

  Gets CPU system time.
- (long long) - getTotalMem

  Gets physical memory size in Bytes.
- (long long) - getAvailableMem

  Gets avialable (free) memory size in Bytes.

3.41.1 Detailed Description

Platform data and statistics.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.41.2 Method Documentation

getAvailableMem

```
- (long long) getAvailableMem
```

Gets avialable (free) memory size in Bytes.

Returns

Avialable (free) memory size in Bytes.

References getAvailableMem.

Referenced by getAvailableMem.

getNumCPUs

```
- (int) getNumCPUs
```

Gets the number of CPUs.

Returns

The number of CPUs.

References getNumCPUs.

Referenced by getNumCPUs.

getRealTime

```
- (long long) getRealTime
```

Gets time in microseconds (since epoch).

Returns

Time in microseconds (since epoch).

References getRealTime.

Referenced by getRealTime.

getSystemTime

```
- (long long) getSystemTime
```

Gets CPU system time.

Returns

CPU system time.

References getSystemTime.

Referenced by getSystemTime.

getTotalMem

```
- (long long) getTotalMem
```

Gets physical memory size in Bytes.

Returns

Physical memory size in Bytes.

References getTotalMem.

Referenced by getTotalMem.

getUserTime

```
- (long long) getUserTime
```

Gets CPU user time.

Returns

CPU user time.

References getUserTime.

Referenced by getUserTime.

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

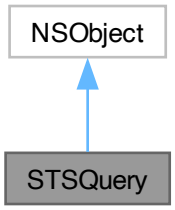
- Sparksee.h

3.42 STSQuery Class Reference

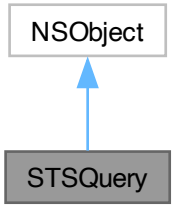
Query class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSQuery:



Collaboration diagram for STSQuery:



Instance Methods

- (STSResultSet *) - execute:reiterable:

  Executes the given statement.
- (STSQueryStream *) - setStream:handler:

  Sets a query stream handler Query streams handlers are created and destroyed by the caller.
- (void) - setDynamic:value:

  Sets the value for a dynamic paramater.
- (void) - **close**

  Closes the Query instance.
- (BOOL) - **isClosed**

  Check if the Query instance is closed.

3.42.1 Detailed Description

Query class.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.42.2 Method Documentation

execute:reiterable:

```
- (STSResultSet *) execute:
             (NSString *) stmt
             reiterable:(BOOL) reiterable
```

Executes the given statement.

Parameters

stmt	[in] Query statement.
reiterable	[in] Whether we want the resultset to be reiterable or not

Returns

A ResultSet instance with the contents of the result of the query.

setDynamic:value:

```
- (void) setDynamic:
              (NSString *) name
              value:(STSValue *) value
```

Sets the value for a dynamic paramater.

Parameters

name	[in] Parameter name
value	[in] Parameter value

setStream:handler:

```
- (STSQueryStream *) setStream:
              (NSString *) stream
              handler:(STSQueryStream *) handler
```

Sets a query stream handler Query streams handlers are created and destroyed by the caller.

DeprecatedQueryStreams should be handled by a QueryStreamManager setup at database level instead.

Parameters

stream	[in] The stream name
handler	[in] Query stream handler

Returns

The previous handler, or NULL if it does not exists

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

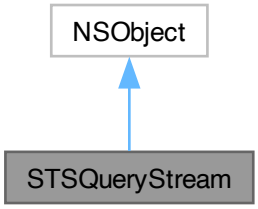
- Sparksee.h

3.43 STSQueryStream Class Reference

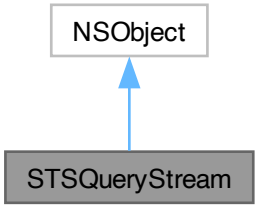
Query stream interface.

```
#include <Sparksee.h>
```

Inheritance diagram for STSQueryStream:



Collaboration diagram for STSQueryStream:



Instance Methods

- (BOOL) - prepare:

  Prepares the stream before it is started.
- (BOOL) - start:

  Starts the stream.
- (BOOL) - fetch:

  Gets the next row and moves the iterator forward.

3.43.1 Detailed Description

Query stream interface.

A QueryStream is the interface between the application and the STREAM operator. When the operator starts inside a Query, the method is prepared with query-defined arguments. Then, if there are input operations, the STREAM operator builds the ResultSets and starts the iteration. Finally, the operator fetches rows until no more are available.

Application exceptions must be cached by the subclass that implements the interface.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.43.2 Method Documentation

fetch:

```
- (BOOL) fetch:
              (STSValueList *) list
```

Gets the next row and moves the iterator forward.

The end of sequence is denoted by returning TRUE with an empty row. A valid row must contain as many values (even NULL) as expected by the query.

Parameters

list	[out] Storage for the new rows

Returns

TRUE if there is a row or end of sequence, FALSE on error

prepare:

```
- (BOOL) prepare:
              (STSValueList *) list
```

Prepares the stream before it is started.

Parameters

list	[in] Optional list of arguments

Returns

FALSE on error

start:

```
- (BOOL) start:
              (STSResultSetList *) list
```

Starts the stream.

Parameters

list	[in] Optional list of input ResultSets

Returns

FALSE on error

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

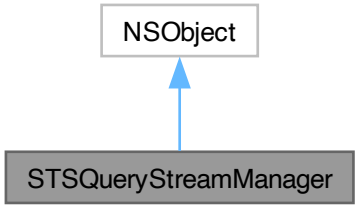
- Sparksee.h

3.44 STSQueryStreamManager Class Reference

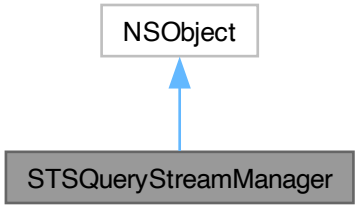
QueryStreamManager interface.

`#include <Sparksee.h>`

Inheritance diagram for STSQueryStreamManager:



Collaboration diagram for STSQueryStreamManager:



Instance Methods

- (STSQueryStream *) - getQueryStream:session:

  Returns an instance of a QueryStream with the given name or NULL.
- (void) - releaseQueryStream:

  Release a QueryStream.

3.44.1 Detailed Description

QueryStreamManager interface.

A QueryStreamManager is the interface to handle objects that will implement user defined query operations through the STREAM operator.

When a query using a custom named STREAM operator requires it it will allocate a QueryStream object using the provided QueryStreamManagers. And once it's finished it will be released through the same QueryStreamManager that was used to allocate it.

Application exceptions must be cached by the subclass that implements the interface.

Maintaining and freeing the QueryStreamManager itself is responsibility of the caller.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.44.2 Method Documentation

getQueryStream:session:

```
- (STSQueryStream *) getQueryStream:
              (NSString *) name
              session:(STSSession *) session
```

Returns an instance of a QueryStream with the given name or NULL.

If this manager handles the given name it will return an instance of a QueryStream that will provide the requested QueryStream operation.

The ReleaseQueryStream will be called once it's usage is finished. It's up to the QueryStreamManager implementation to really allocate/deallocate a new QueryStream each time or share a static one.

Parameters

name	[In] The name of the QueryStream that we want to get
session	[In] A pointer to the Session used in the Query and may be required by the Stream.

Returns

Returns a QueryStream or NULL

releaseQueryStream:

```
- (void) releaseQueryStream:
              (STSQueryStream *) qstream
```

Release a QueryStream.

The QueryStream must have been reserved by the same QueryStreamManager

The Manager must free the QueryStream if it was allocated or handle it as it requires for it's implementation.

Parameters

qstream	[In] The QueryStream to be released.

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

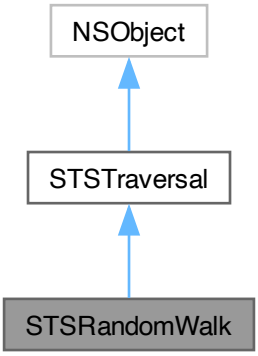
- Sparksee.h

3.45 STSRandomWalk Class Reference

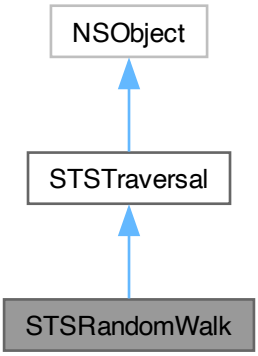
RandomWalk class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSRandomWalk:



Collaboration diagram for STSRandomWalk:



Instance Methods

- (id) - initWithSession:node:

  Builds the RandomWalk.
- (void) - addEdgeType:dir:

  Allows for traversing edges of the given type.

- (void) - addAllEdgeTypes:

  Allows for traversing all edge types of the graph.
- (void) - addNodeType:

  Allows for traversing nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows for traversing all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (long long) - next

  Gets the next object of the traversal.
- (BOOL) - hasNext

  Gets if there are more objects to be traversed.
- (int) - getCurrentDepth

  Returns the depth of the current node.
- (void) - setEdgeWeightAttributeType:

  Sets the attribute to use as edge weight.
- (void) - setDefaultWeight:

  Sets the default weight for those cases when a given edge does not have a weight attribute set.
- (void) - reset:

  Sets the starting node of the RandomWalk.
- (void) - setReturnParameter:

  Sets the return parameter of the RandomWalk.
- (void) - setInOutParameter:

  Sets the In-Out parameter of the RandomWalk.
- (void) - setSeed:

  Sets the seed of the random walk.
- (void) - setMaximumHops:

  Sets the maximum hops restriction.
- (void) - close

  Closes the Traversal instance.
- (BOOL) - **isClosed**

  Check if the Traversal instance is closed.

3.45.1 Detailed Description

RandomWalk class.

Implements the RandomWalk algorithm

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.45.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows for traversing all edge types of the graph.

Parameters

dir	[in] Edge direction.

Implements STSTraversal.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
            dir:(enum STSEdgesDirection) dir
```

Allows for traversing edges of the given type.

If the edge type was already added, the existing direction is overwritten

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

Implements STSTraversal.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows for traversing nodes of the given type.

Parameters

type	The node type to add

Implements STSTraversal.

close

```
- (void) close
```

Closes the Traversal instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

Implements STSTraversal.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

Implements STSTraversal.

getCurrentDepth

```
- (int) getCurrentDepth
```

Returns the depth of the current node.

That is, it returns the depth of the node returned in the last call to Next().

Returns

The depth of the current node.

Implements STSTraversal.

References getCurrentDepth.

Referenced by getCurrentDepth.

hasNext

```
- (BOOL) hasNext
```

Gets if there are more objects to be traversed.

Returns

TRUE if there are more objects, FALSE otherwise.

Implements STSTraversal.

References hasNext.

Referenced by hasNext.

initWithSession:node:

```
- (id) initWithSession:
            (STSSession *) session
            node:(long long) node
```

Builds the RandomWalk.

Parameters

session	[in] The session to use
node	[in] The starting node of the traversal

next

```
- (long long) next
```

Gets the next object of the traversal.

Returns

A node or edge identifier.

Implements STSTraversal.

References next.

Referenced by next.

reset:

```
- (void) reset:
             (long long) startNode
```

Sets the starting node of the RandomWalk.

This method resets the RandomWalk.

sparksee::gdb::Error

Parameters

startNode	null

setDefaultWeight:

```
- (void) setDefaultWeight:
             (double) weight
```

Sets the default weight for those cases when a given edge does not have a weight attribute set.

Default: 0.0

Parameters

weight	[in] The default weight

setEdgeWeightAttributeType:

```
- (void) setEdgeWeightAttributeType:
             (int) attr
```

Sets the attribute to use as edge weight.

If the multiple edge are set for traversal, this attribute must be of type GLOBAL_TYPE or EDGES_TYPE. Additionally, the attribute must be of type Double. Finally, negative weights are treated as non existing, so the default weight applies.

Parameters

attr	[in] The attribute type to use as a weight. Default: InvalidAttribute

setInOutParameter:

```
- (void) setInOutParameter:
             (double) val
```

Sets the In-Out parameter of the RandomWalk.

Parameters

val	The In-Out parameter to set. Default: 1.0

setMaximumHops:

```
- (void) setMaximumHops:
             (int) maxhops
```

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored.

Parameters

maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

setReturnParameter:

```
- (void) setReturnParameter:
             (double) val
```

Sets the return parameter of the RandomWalk.

Parameters

val	The return parameter to set. Default: 1.0

setSeed:

```
- (void) setSeed:
             (int) seed
```

Sets the seed of the random walk.

Parameters

seed	The seed to generate the random numbers that drive the random walk

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

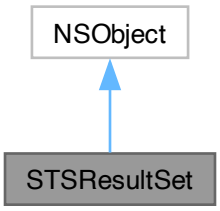
- Sparksee.h

3.46 STSResultSet Class Reference

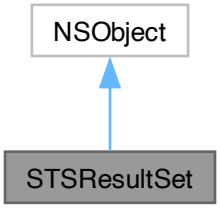
ResultSet class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSResultSet:



Collaboration diagram for STSResultSet:



Instance Methods

- (int) - getNumColumns

  Gets the number of columns.
- (NSString ∗) - getColumnName:

  Gets the name for the given column.
- (int) - getColumnIndex:

  Gets the column index for the given column name.
- (enum STSDataType) - getColumnDataType:

  Gets the datatype for the given column.
- (BOOL) - next

  Fetches the next row.
- (void) - getColumnWithValue:value:

  Gets the value for the given column.
- (STSValue ∗) - getColumn:

  Gets the value for the given column.
- (void) - **rewind**

  Positions the cursor before the first row.
- (NSString ∗) - getJSON:

  Returns rows in JSON format.
- (void) - **close**

  Closes the ResultSet instance.
- (BOOL) - **isClosed**

  Check if the ResultSet instance is closed.

3.46.1 Detailed Description

ResultSet class.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.46.2 Method Documentation

getColumn:

```
- (STSValue *) getColumn:
              (int) index
```

Gets the value for the given column.

QueryExceptionIf a database access error occurs.

Parameters

index	[in] Column index.

Returns

The Value of the given column.

getColumnDataType:

```
- (enum STSDataType) getColumnDataType:
              (int) index
```

Gets the datatype for the given column.

Parameters

index	[in] Column index.

Returns

DataType for the given column.

getColumnIndex:

```
- (int) getColumnIndex:
              (NSString *) name
```

Gets the column index for the given column name.

Parameters

name	[in] Column name.

Returns

Column index.

getColumnName:

```
- (NSString *) getColumnName:
              (int) index
```

Gets the name for the given column.

Parameters

index	[in] Column index.

Returns

Column name.

getColumnWithValue:value:

```
- (void) getColumnWithValue:
              (int) index
             value:(STSValue *) value
```

Gets the value for the given column.

QueryExceptionIf a database access error occurs.

Parameters

index	[in] Column index.
value	[in|out] Value.

getJSON:

```
- (NSString *) getJSON:
              (int) rows
```

Returns rows in JSON format.

Rows are returned from the current position.

Parameters

rows	[in] Maximum number of rows

Returns

JSON representation of the next <rows> rows in the resultset

getNumColumns

```
- (int) getNumColumns
```

Gets the number of columns.

Columns are in the range [0...COLUMNS).

Returns

The number of columns.

References getNumColumns.

Referenced by getNumColumns.

next

```
- (BOOL) next
```

Fetches the next row.

A ResultSet cursor is initially positioned before the first row; the first call to the method "Next" makes the first row the current row; the second call makes the second row the current row, and so on.

QueryExceptionIf a database access error occurs.

Returns

TRUE if the next row has been successfully fetched, FALSE otherwise.

References next.

Referenced by next.

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

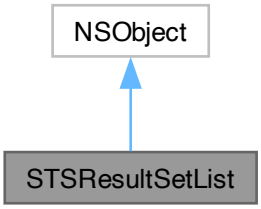
- Sparksee.h

3.47 STSResultSetList Class Reference

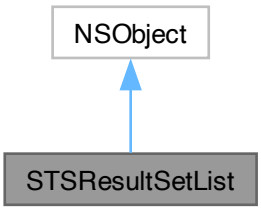
ResultSet list.

`#include <Sparksee.h>`

Inheritance diagram for STSResultSetList:



Collaboration diagram for STSResultSetList:



Instance Methods

- (int) - count

  Number of elements in the list.
- (id) - init

  Constructor.
- (void) - **clear**

  Clears the list.
- (STSResultSet ∗) - get:

  Returns the ResultSet at the specified position in the list.
- (STSResultSetListIterator ∗) - **iterator**

  Gets a new ResultSetListIterator.

3.47.1 Detailed Description

ResultSet list.

It stores a ResultSet list.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.47.2 Method Documentation

count

```
- (int) count
```

Number of elements in the list.

Returns

Number of elements in the list.

References count.

Referenced by count.

get:

```
- (STSResultSet *) get:
              (int) index
```

Returns the ResultSet at the specified position in the list.

Parameters

index	[in] Index of the element to return, starting at 0.

init

```
- (id) init
```

Constructor.

This creates an empty list.

References init.

Referenced by init.

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

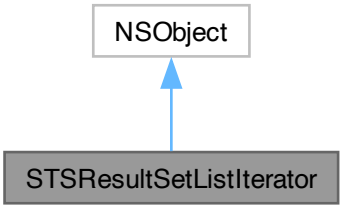
- Sparksee.h

3.48 STSResultSetListIterator Class Reference

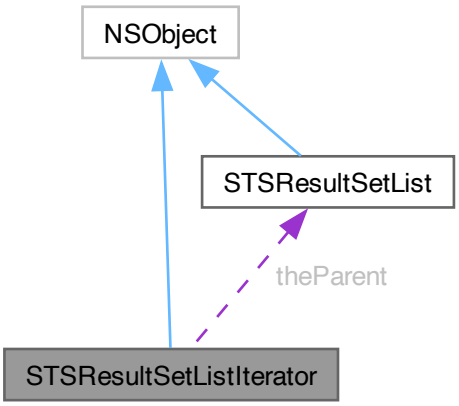
ResultSetList iterator class.

`#include <Sparksee.h>`

Inheritance diagram for STSResultSetListIterator:



Collaboration diagram for STSResultSetListIterator:



Instance Methods

- (STSResultSet *) - next

  Moves to the next element.
- (BOOL) - hasNext

  Gets if there are more elements.

3.48.1 Detailed Description

ResultSetList iterator class.

Iterator to traverse all the values into a ResultSetList instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.48.2 Method Documentation

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements.

Returns

TRUE if there are more elements, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (STSResultSet *) next
```

Moves to the next element.

Returns

The next element.

References next.

Referenced by next.

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

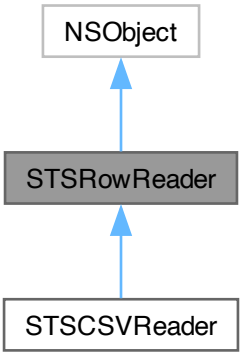
- Sparksee.h

3.49 STSRowReader Class Reference

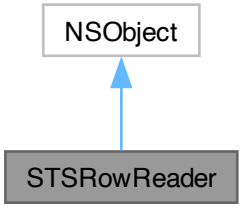
RowReader interface.

```
#include <Sparksee.h>
```

Inheritance diagram for STSRowReader:



Collaboration diagram for STSRowReader:



Instance Methods

- (BOOL) - reset

  Moves the reader to the beginning.
- (BOOL) - read:

  Reads the next row as a string array.
- (int) - getRow

  The row number for the current row.
- (void) - close

  Closes the reader.

3.49.1 Detailed Description

RowReader interface.

Common interface for those readers which get the data as an string array.

It works as follows: perform as many read operations as necessary and call close once at the end. Once close is called no more read operations can be executed.

Check out the 'Data import' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.49.2 Method Documentation

close

```
- (void) close
```

Closes the reader.

Exceptions

System.IO.IOException	If the close fails.

Implemented in STSCSVReader.

References close.

Referenced by close.

getRow

```
- (int) getRow
```

The row number for the current row.

Returns

The current row number; 0 if there is no current row.

Exceptions

System.IO.IOException	If it fails.

Implemented in STSCSVReader.

References getRow.

Referenced by getRow.

read:

```
- (BOOL) read:
              (STSStringList *) row
```

Reads the next row as a string array.

Parameters

row	[out] A string list with each comma-separated element as a separate entry.

Returns

Returns true if a row had been read or false otherwise.

Exceptions

System.IO.IOException	If bad things happen during the read.

Implemented in STSCSVReader.

reset

```
- (BOOL) reset
```

Moves the reader to the beginning.

Restarts the reader.

Returns

true if the reader can be restarted, false otherwise.

Exceptions

System.IO.IOException	If bad things happen during the restart.

Implemented in STSCSVReader.

References reset.

Referenced by reset.

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

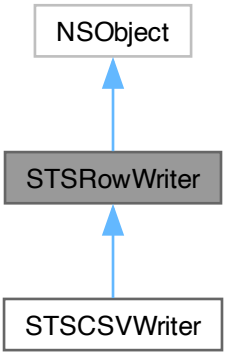
- Sparksee.h

3.50 STSRowWriter Class Reference

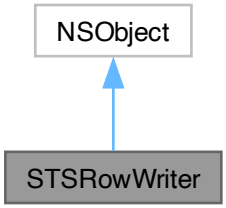
RowWriter interface.

`#include <Sparksee.h>`

Inheritance diagram for STSRowWriter:



Collaboration diagram for STSRowWriter:



Instance Methods

- (void) - write:

  Writes the next row.
- (void) - close

  Closes the writer.

3.50.1 Detailed Description

RowWriter interface.

Common interface for those writers which dump the data from an string array.

It works as follows: perform as many write operations as necessary and call close once at the end. Once close is called no more write operations can be executed.

Check out the 'Data export' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.50.2 Method Documentation

close

```
- (void) close
```

Closes the writer.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the close fails.

Implemented in STSCSVWriter.

References close.

Referenced by close.

write:

```
- (void) write:
             (STSStringList *) row
```

Writes the next row.

Parameters

row	[in] Row of data.

Exceptions

System.ApplicationException	null

System.IO.IOException	If bad things happen during the write.

Implemented in STSCSVWriter.

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

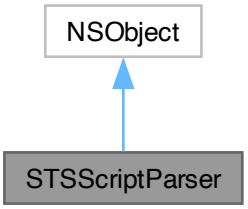
- Sparksee.h

3.51 STSScriptParser Class Reference

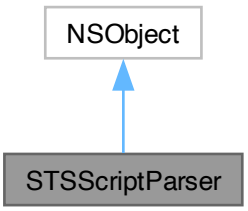
ScriptParser.

```
#include <Sparksee.h>
```

Inheritance diagram for STSScriptParser:



Collaboration diagram for STSScriptParser:



Instance Methods

- (id) - **init**

  Constructor.
- (void) - setOutputLog:

  Sets the output log.
- (void) - setErrorLog:

  Sets the error log.
- (BOOL) - parse:execute:localeStr:

  Parses the given input file.

Class Methods

- (void) + generateSchemaScript:db:

  Writes an script with the schema definition for the given database.

3.51.1 Detailed Description

ScriptParser.

The ScriptParser can create schemas and load data from a set of commands in a sparksee script.

A SPARKSEE script contains an ordered list of commands. ScriptParser will execute each one of them in order. Commands may create schemas, define nodes and edges, and load data into a previous defined SPARKSEE schema.

Check out the 'Scripting' chapter in the SPARKSEE User Manual for a comprehensive explanation on the grammar of the SPARKSEE commands and how they work.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.51.2 Method Documentation

generateSchemaScript:db:

```
+ (void) generateSchemaScript:
             (NSString *) path
             db:(STSDatabase *) db
```

Writes an script with the schema definition for the given database.

Parameters

path	[in] Filename of the script to be writen.
db	[in] Database.

Exceptions

System.IO.IOException	If bad things happen opening or writing the file.

parse:execute:localeStr:

```
- (BOOL) parse:
             (NSString *) path
             execute:(BOOL) execute
             localeStr:(NSString *) localeStr
```

Parses the given input file.

Parameters

path	[in] Input file path.
execute	[in] If TRUE the script is executed, if FALSE it is just parsed.
localeStr	[in] The locale string for reading the input file. See CSVReader.

Returns

TRUE if ok, FALSE in case of error.

Exceptions

System.IO.IOException	If bad things happen opening the file.

setErrorLog:

```
- (void) setErrorLog:
             (NSString *) path
```

Sets the error log.

If not set, error log corresponds to standard error output.

Parameters

path	[in] Path of the error log.

Exceptions

System.IO.IOException	If bad things happen opening the file.

setOutputLog:

```
- (void) setOutputLog:
              (NSString *) path
```

Sets the output log.

If not set, output log corresponds to standard output.

Parameters

path	[in] Path of the output log.

Exceptions

System.IO.IOException	If bad things happen opening the file.

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

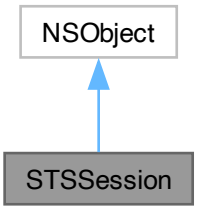
- Sparksee.h

3.52 STSSession Class Reference

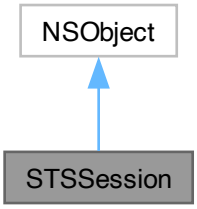
Session class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSSession:



Collaboration diagram for STSSession:



Instance Methods

- (STSGraph ∗) - getGraph
  Gets the Graph instance.
- (STSObjects ∗) - createObjects
  Creates a new Objects instance.
- (void) - **begin**
  Begins a transaction.
- (void) - **beginUpdate**
  Begins an update transaction.
- (void) - **commit**
  Commits a transaction.
- (void) - **rollback**
  Rollbacks a transaction.
- (STSQuery ∗) - createQuery:
  Creates a new Query.
- (long long) - preCommit
  PreCommits a transaction.
- (long long) - getInMemoryPoolCapacity
  Gets the capacity of the in-memory pool.
- (void) - close
  Closes the Session instance.
- (BOOL) - **isClosed**
  Check if the Session instance is closed.

3.52.1 Detailed Description

Session class.

A Session is a stateful period of activity of a user with the Database.

All the manipulation of a Database must be enclosed into a Session. A Session can be initiated from a Database instance and allows for getting a Graph instance which represents the persistent graph (the graph database).

Also, temporary data is associated to the Session, thus when a Session is closed, all the temporary data associated to the Session is removed too. Objects or Values instances or even session attributes are an example of temporary data.

Moreover, a Session is exclusive for a thread, thus if it is shared among threads results may be fatal or unexpected.

Check out the 'Processing' and 'Transactions' sections in the SPARKSEE User Manual for details about how Sessions work and the use of transactions.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.52.2 Method Documentation

close

```
- (void) close
```

Closes the Session instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

createObjects

```
- (STSObjects *) createObjects
```

Creates a new Objects instance.

Returns

The new Objects instance.

References createObjects.

Referenced by createObjects.

createQuery:

```
- (STSQuery *) createQuery:
             (enum STSQueryLanguage) lang
```

Creates a new Query.

Parameters

lang	The query language to create the query for

Returns

A new Query instance.

getGraph

```
- (STSGraph *) getGraph
```

Gets the Graph instance.

Returns

The Graph instance.

References getGraph.

Referenced by getGraph.

getInMemoryPoolCapacity

```
- (long long) getInMemoryPoolCapacity
```

Gets the capacity of the in-memory pool.

Returns

Returns the capacity of the in-memory pool

References getInMemoryPoolCapacity.

Referenced by getInMemoryPoolCapacity.

preCommit

```
- (long long) preCommit
```

PreCommits a transaction.

YOU SHOULD NOT USE THIS METHOD. This method exists for a specific service and it is used to force that the transaction is written in the recovery log even though it had not been committed yet (and may never be).

Returns

The transaction id. This has to be used in RedoPrecommited in case of a recovery process.

References preCommit.

Referenced by preCommit.

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

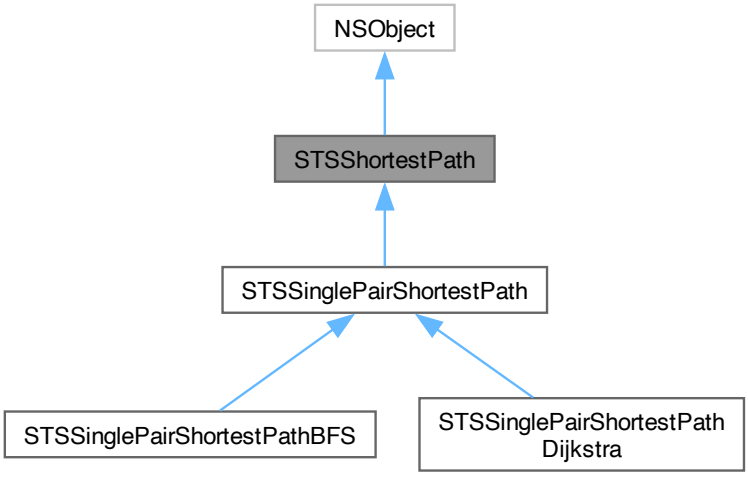
- Sparksee.h

3.53 STSShortestPath Class Reference

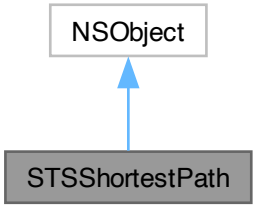
ShortestPath class.

`#include <Sparksee.h>`

Inheritance diagram for STSShortestPath:



Collaboration diagram for STSShortestPath:



Instance Methods

- (void) - setMaximumHops:

  Sets the maximum hops restriction.
- (void) - addEdgeType:dir:

  Allows for traversing edges of the given type.
- (void) - addAllEdgeTypes:

Allows for traversing all edge types of the graph.

- (void) - addNodeType:

  Allows for traversing nodes of the given type.

- (void) - **addAllNodeTypes**

  Allows for traversing all node types of the graph.

- (void) - excludeNodes:

  Set which nodes can't be used.

- (void) - excludeEdges:

  Set which edges can't be used.

- (void) - run

  Runs the algorithm.

- (void) - close

  Closes the ShortestPath instance.

- (BOOL) - **isClosed**

  Check if the ShortestPath instance is closed.

3.53.1 Detailed Description

ShortestPath class.

Classes implementing this abstract class solve the shortest path problem in a graph.

The user must set which node and edge types can be used for the traversal.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.53.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows for traversing all edge types of the graph.

Parameters

dir	[in] Edge direction.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
            dir:(enum STSEdgesDirection) dir
```

Allows for traversing edges of the given type.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows for traversing nodes of the given type.

Parameters

type	null

close

```
- (void) close
```

Closes the ShortestPath instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

run

```
- (void) run
```

Runs the algorithm.

This method can only be called once.

Implemented in STSSinglePairShortestPathBFS, and STSSinglePairShortestPathDijkstra.

References run.

Referenced by run.

setMaximumHops:

```
- (void) setMaximumHops:
             (int) maxhops
```

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored.

Parameters

maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

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

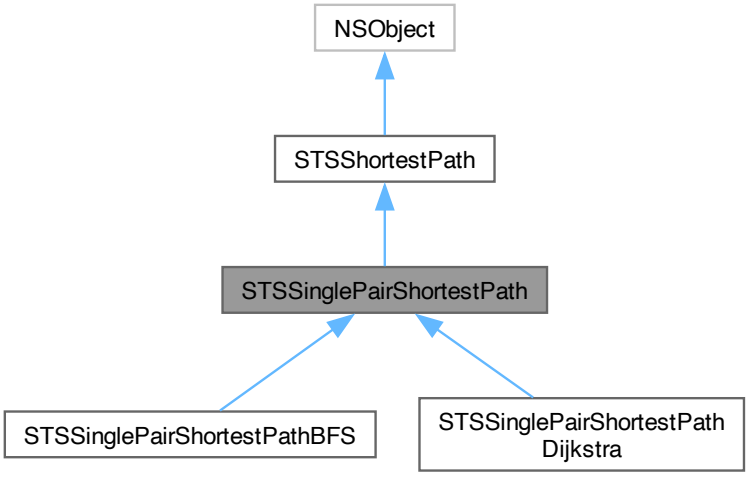
- Sparksee.h

3.54 STSSinglePairShortestPath Class Reference

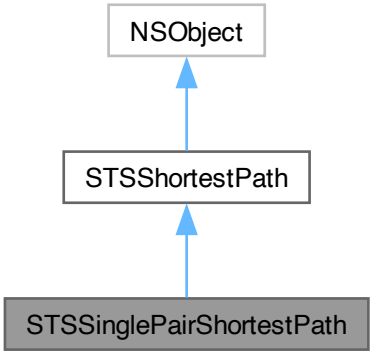
SinglePairShortestPath class.

`#include <Sparksee.h>`

Inheritance diagram for STSSinglePairShortestPath:



Collaboration diagram for STSSinglePairShortestPath:



Instance Methods

- (STSOidList *) - getPathAsNodes

Gets the shortest path between the source node and the destination node as an ordered set of nodes.

- (STSOidList *) - getPathAsEdges

  Gets the shortest path between the source node and the destination node as an ordered set of edges.

- (double) - getCost

  Gets the cost of the shortest path.

- (BOOL) - **exists**

  Returns TRUE If a path exists or FALSE otherwise.

- (void) - setMaximumHops:

  Sets the maximum hops restriction.

- (void) - addEdgeType:dir:

  Allows for traversing edges of the given type.

- (void) - addAllEdgeTypes:

  Allows for traversing all edge types of the graph.

- (void) - addNodeType:

  Allows for traversing nodes of the given type.

- (void) - **addAllNodeTypes**

  Allows for traversing all node types of the graph.

- (void) - excludeNodes:

  Set which nodes can't be used.

- (void) - excludeEdges:

  Set which edges can't be used.

- (void) - run

  Runs the algorithm.

- (void) - close

  Closes the ShortestPath instance.

- (BOOL) - **isClosed**

  Check if the ShortestPath instance is closed.

3.54.1 Detailed Description

SinglePairShortestPath class.

Classes implementing this abstract class solve the shortest path problem in a graph from a given source node and to a given destination node.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.54.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows for traversing all edge types of the graph.

Parameters

dir	[in] Edge direction.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
             dir:(enum STSEdgesDirection) dir
```

Allows for traversing edges of the given type.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows for traversing nodes of the given type.

Parameters

type	null

close

```
- (void) close
```

Closes the ShortestPath instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getCost

```
- (double) getCost
```

Gets the cost of the shortest path.

The cost for unweighted algorithms is the number of hops of the shortest path. For weighted algoritms, the cost is the sum of the costs of the edges of the shortest path.

Returns

The cost of the shortest path.

Implemented in STSSinglePairShortestPathBFS, and STSSinglePairShortestPathDijkstra.

References getCost.

Referenced by getCost.

getPathAsEdges

```
- (STSOidList *) getPathAsEdges
```

Gets the shortest path between the source node and the destination node as an ordered set of edges.

Returns

Ordered set of edge identifiers.

Implemented in STSSinglePairShortestPathBFS, and STSSinglePairShortestPathDijkstra.

References getPathAsEdges.

Referenced by getPathAsEdges.

getPathAsNodes

```
- (STSOidList *) getPathAsNodes
```

Gets the shortest path between the source node and the destination node as an ordered set of nodes.

Returns

Ordered set of node identifiers.

Implemented in STSSinglePairShortestPathBFS, and STSSinglePairShortestPathDijkstra.

References getPathAsNodes.

Referenced by getPathAsNodes.

run

```
- (void) run
```

Runs the algorithm.

This method can only be called once.

Implemented in STSSinglePairShortestPathBFS, and STSSinglePairShortestPathDijkstra.

References run.

Referenced by run.

setMaximumHops:

```
- (void) setMaximumHops:
             (int) maxhops
```

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored.

Parameters

maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

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

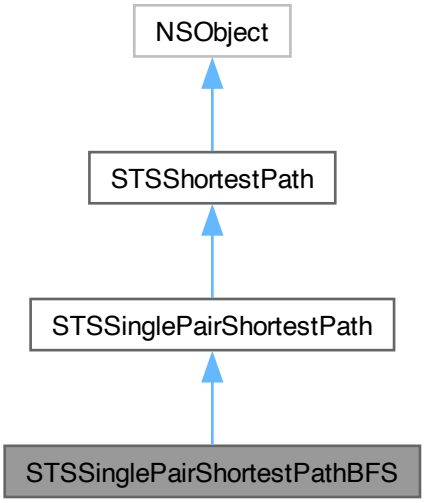
- Sparksee.h

3.55 STSSinglePairShortestPathBFS Class Reference

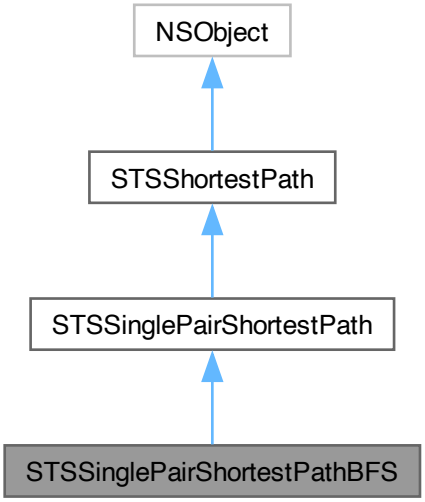
SinglePairShortestPathBFS class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSSinglePairShortestPathBFS:



Collaboration diagram for STSSinglePairShortestPathBFS:



Instance Methods

- (void) - **run**

  Executes the algorithm.
- (STSOidList ∗) - getPathAsNodes

  Gets the shortest path between the source node and the destination node as an ordered set of nodes.
- (STSOidList ∗) - getPathAsEdges

  Gets the shortest path between the source node and the destination node as an ordered set of edges.
- (double) - getCost

  Gets the cost of the shortest path.
- (id) - initWithSession:src:dst:

  Creates a new instance.
- (void) - checkOnlyExistence

  Set that only the path existence must be calculated and not the path itself.
- (BOOL) - **exists**

  Returns TRUE If a path exists or FALSE otherwise.
- (void) - setMaximumHops:

  Sets the maximum hops restriction.
- (void) - addEdgeType:dir:

  Allows for traversing edges of the given type.
- (void) - addAllEdgeTypes:

  Allows for traversing all edge types of the graph.
- (void) - addNodeType:

  Allows for traversing nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows for traversing all node types of the graph.
- (void) - excludeNodes:

Set which nodes can't be used.

- (void) - excludeEdges:

  Set which edges can't be used.

- (void) - close

  Closes the ShortestPath instance.

- (BOOL) - **isClosed**

  Check if the ShortestPath instance is closed.

3.55.1 Detailed Description

SinglePairShortestPathBFS class.

It solves the single-pair shortest path problem using a BFS-based implementation.

It is a unweighted algorithm, that is it assumes all edges have the same cost.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.55.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows for traversing all edge types of the graph.

Parameters

dir	[in] Edge direction.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
            dir:(enum STSEdgesDirection) dir
```

Allows for traversing edges of the given type.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows for traversing nodes of the given type.

Parameters

type	null

checkOnlyExistence

```
- (void) checkOnlyExistence
```

Set that only the path existence must be calculated and not the path itself.

That method should improve the performance of the algorithm, but a call to GetPathAsNodes or GetPathAsEdges will generate an exception even if the path exists.

References checkOnlyExistence.

Referenced by checkOnlyExistence.

close

```
- (void) close
```

Closes the ShortestPath instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getCost

```
- (double) getCost
```

Gets the cost of the shortest path.

The cost is the number of hops of the shortest path.

Returns

The cost of the shortest path.

Implements STSSinglePairShortestPath.

References getCost.

Referenced by getCost.

getPathAsEdges

```
- (STSOidList *) getPathAsEdges
```

Gets the shortest path between the source node and the destination node as an ordered set of edges.

Returns

Ordered set of edge identifiers.

Implements STSSinglePairShortestPath.

References getPathAsEdges.

Referenced by getPathAsEdges.

getPathAsNodes

```
- (STSOidList *) getPathAsNodes
```

Gets the shortest path between the source node and the destination node as an ordered set of nodes.

Returns

Ordered set of node identifiers.

Implements STSSinglePairShortestPath.

References getPathAsNodes.

Referenced by getPathAsNodes.

initWithSession:src:dst:

```
- (id) initWithSession:
             (STSSession *) session
             src:(long long) src
             dst:(long long) dst
```

Creates a new instance.

Parameters

session	[in] Session to get the graph from and perform traversal.
src	[in] Source node.
dst	[dst] Destination node.

setMaximumHops:

```
- (void) setMaximumHops:
             (int) maxhops
```

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored.

Parameters

maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

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

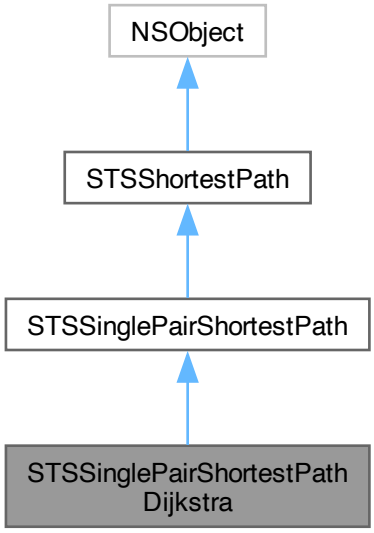
- Sparksee.h

3.56 STSSinglePairShortestPathDijkstra Class Reference

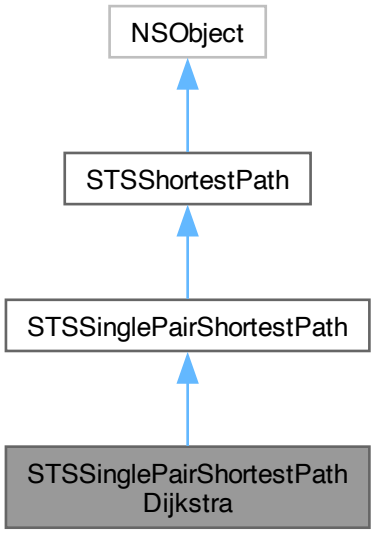
SinglePairShortestPathDijkstra class.

`#include <Sparksee.h>`

Inheritance diagram for STSSinglePairShortestPathDijkstra:



Collaboration diagram for STSSinglePairShortestPathDijkstra:



Instance Methods

- (void) - **run**
  Executes the algorithm.
- (STSOidList *) - getPathAsNodes
  Gets the shortest path between the source node and the destination node as an ordered set of nodes.
- (STSOidList *) - getPathAsEdges
  Gets the shortest path between the source node and the destination node as an ordered set of edges.
- (double) - getCost
  Gets the cost of the shortest path.
- (id) - initWithSession:src:dst:
  Creates a new instance.
- (void) - addWeightedEdgeType:dir:attr:
  Allows for traversing edges of the given type using the given attribute as the weight.
- (void) - setUnweightedEdgeCost:
  Sets the weight assigned to the unweighted edges.
- (void) - setDynamicEdgeCostCallback:
  Set a class callback to dynamically calculate the cost of the edges.
- (BOOL) - **exists**
  Returns TRUE If a path exists or FALSE otherwise.
- (void) - setMaximumHops:
  Sets the maximum hops restriction.
- (void) - addEdgeType:dir:
  Allows for traversing edges of the given type.
- (void) - addAllEdgeTypes:
  Allows for traversing all edge types of the graph.
- (void) - addNodeType:
  Allows for traversing nodes of the given type.
- (void) - **addAllNodeTypes**
  Allows for traversing all node types of the graph.
- (void) - excludeNodes:
  Set which nodes can't be used.
- (void) - excludeEdges:
  Set which edges can't be used.
- (void) - close
  Closes the ShortestPath instance.
- (BOOL) - **isClosed**
  Check if the ShortestPath instance is closed.

3.56.1 Detailed Description

SinglePairShortestPathDijkstra class.

It solves the single-pair shortest path problem using a Dijkstra-based implementation.

It is a weighted algorithm, so it takes into account the cost of the edges to compute a minimum-cost shortest path. That is, the user may set for each edge type which attribute should be used to retrieve the cost of the edge. If no attribute is given for an edge type, this will assume the edge has a fixed cost (the default is 1). Only numerical attribute can be set as weight attributes (that is Long, Integer or Double attributes are allowed).

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.56.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows for traversing all edge types of the graph.

Parameters

dir	[in] Edge direction.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
             dir:(enum STSEdgesDirection) dir
```

Allows for traversing edges of the given type.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows for traversing nodes of the given type.

Parameters

type	null

addWeightedEdgeType:dir:attr:

```
- (void) addWeightedEdgeType:
             (int) type
             dir:(enum STSEdgesDirection) dir
             attr:(int) attr
```

Allows for traversing edges of the given type using the given attribute as the weight.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.
attr	[in] Attribute to be used as the weight. It must be a global attribute or an attribute of the given edge type.

close

```
- (void) close
```

Closes the ShortestPath instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getCost

```
- (double) getCost
```

Gets the cost of the shortest path.

The cost is the sum of the weights of the edges in the shortest path.

Returns

The cost of the shortest path.

Implements STSSinglePairShortestPath.

References getCost.

Referenced by getCost.

getPathAsEdges

```
- (STSOidList *) getPathAsEdges
```

Gets the shortest path between the source node and the destination node as an ordered set of edges.

Returns

Ordered set of edge identifiers.

Implements STSSinglePairShortestPath.

References getPathAsEdges.

Referenced by getPathAsEdges.

getPathAsNodes

```
- (STSOidList *) getPathAsNodes
```

Gets the shortest path between the source node and the destination node as an ordered set of nodes.

Returns

Ordered set of node identifiers.

Implements STSSinglePairShortestPath.

References getPathAsNodes.

Referenced by getPathAsNodes.

initWithSession:src:dst:

```
- (id) initWithSession:
              (STSSession *) session
              src:(long long) src
              dst:(long long) dst
```

Creates a new instance.

Parameters

session	[in] Session to get the graph from and perform traversal.
src	[in] Source node.
dst	[dst] Destination node.

setDynamicEdgeCostCallback:

```
- (void) setDynamicEdgeCostCallback:
              (STSSinglePairShortestPathDijkstraDynamicCost *) dynCostCalculator
```

Set a class callback to dynamically calculate the cost of the edges.

The callback can be set to NULL (the default) to use the normal attribute based cost weights. The given class must be kept alive by the user for as long as the algorithm is running.

Parameters

dynCostCalculator	[in] Class callback to calculate the edge costs

setMaximumHops:

```
- (void) setMaximumHops:
              (int) maxhops
```

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored.

Parameters

maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

setUnweightedEdgeCost:

```
- (void) setUnweightedEdgeCost:
            (double) weight
```

Sets the weight assigned to the unweighted edges.

All the edges from the types added without an explicit weight attribute will get this weight. The default weight for this edges is 1.

Parameters

weight	[in] The weight value for unweighted edges.

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

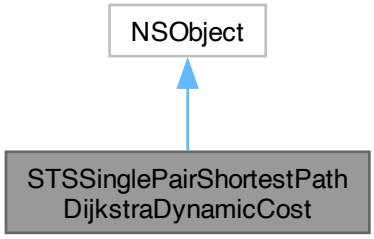
- Sparksee.h

3.57 STSSinglePairShortestPathDijkstraDynamicCost Class Reference

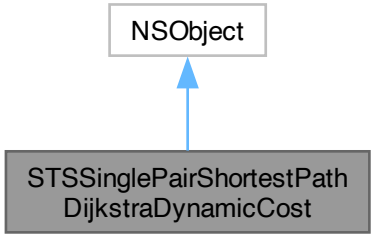
Defines how to calculate an edge weight.

`#include <Sparksee.h>`

Inheritance diagram for STSSinglePairShortestPathDijkstraDynamicCost:



Collaboration diagram for STSSinglePairShortestPathDijkstraDynamicCost:



Instance Methods

- (double) - calculateEdgeCost:sourceCost:sourceLevel:targetNode:edge:edgeWeightAttr:

  Dynamic calculation of an edge weight.

3.57.1 Detailed Description

Defines how to calculate an edge weight.

This is an interface which must be implemented by the user. While the algorithm is running, one or more calls for each edge that can be in the shortest path will be issued to get the weight of the edge that could change based on the predecessor node and the current minimum path from the source.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.57.2 Method Documentation

calculateEdgeCost:sourceCost:sourceLevel:targetNode:edge:edgeWeightAttr:

```
- (double) calculateEdgeCost:
              (long long) sourceNode
              sourceCost:(double) sourceCost
              sourceLevel:(int) sourceLevel
              targetNode:(long long) targetNode
              edge:(long long) edge
              edgeWeightAttr:(int) edgeWeightAttr
```

Dynamic calculation of an edge weight.

This method will be called several times during the algorithm execution. It can be called for the same edge multiple times because the path to the source node (and as a consequence the source weight and levels) might change while the algorithm is running. This method will have the normal default implementation that only uses the specified edge attribute to get the weight or the unweighted edge cost if the edge doesn't have the specified cost attribute. But it can be overriden in a derived user class to implement a dinamic edge weight, that may need the sourceCost and sourceLevels to calculate the weight.

Parameters

sourceNode	[in] The current node
sourceCost	[in] Current total cost up to the source node
sourceLevel	[in] Current path size up to the source node
targetNode	[in] The next node
edge	[in] The edge to the next node that this methos must weight
edgeWeightAttr	[in] The attribute that stores the edge weight or InvalidAttribute

Returns

Returns the current weight of the edge ($>=0$) or <0 if the edge can not be used.

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

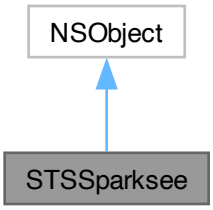
- Sparksee.h

3.58 STSSparksee Class Reference

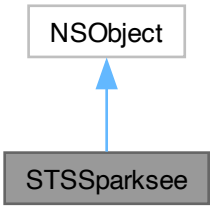
Sparksee class.

`#include <Sparksee.h>`

Inheritance diagram for STSSparksee:



Collaboration diagram for STSSparksee:



Instance Methods

- (id) - initWithConfig:
  Creates a new instance.
- (STSDatabase *) - create:alias:
  Creates a new Database instance.
- (STSDatabase *) - createWithConfig:alias:config:
  Creates a new Database instance.
- (STSDatabase *) - open:readOnly:
  Opens an existing Database instance.
- (STSDatabase *) - openWithConfig:readOnly:config:
  Opens an existing Database instance.
- (STSDatabase *) - restore:backupFile:
  Restores a Database from a backup file.

- (STSDatabase *) - restoreWithConfig:backupFile:config:

  Restores a Database from a backup file.
- (STSDatabase *) - restoreEncryptedBackup:backupFile:keyInHex:ivInHex:

  Restores a Database from an encrypted backup file.
- (STSDatabase *) - restoreEncryptedBackupWithConfig:backupFile:config:keyInHex:ivInHex:

  Restores a Database from an encrypted backup file.
- (BOOL) - verifyChecksums:

  Verifies the Checksum integrity of the given image file and it's recovery log file.
- (BOOL) - addChecksums:

  Converts a database WITHOUT checksums into a database WITH checksums.
- (BOOL) - removeChecksums:

  Converts a database WITH checksums into a database WITHOUT checksums.
- (BOOL) - encrypt:

  Converts a database WITHOUT encryption into an encrypted database.
- (BOOL) - decrypt:

  Converts a database WITH encryption into a database WITHOUT encryption.
- (void) - resizeInMemoryPool:

  Resizes the in memory pool allocator.
- (long long) - getInMemoryPoolCapacity

  Gets the capacity of the in-memory pool (in Megabytes).
- (void) - rebuildDatabase:config:minimizeDisk:

  Rebuild the database from scratch with the same data.
- (void) - **close**

  Closes the Sparksee instance.
- (BOOL) - **isClosed**

  Check if the Sparksee instance is closed.

Class Methods

- (NSString *) + **getVersion**

  Sparksee version.

3.58.1 Detailed Description

Sparksee class.

All Sparksee programs must have one single Sparksee instance to manage one or more Database instances.

This class allows for the creation of new Databases or open an existing one.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.58.2 Method Documentation

addChecksums:

```
- (BOOL) addChecksums:
             (NSString *) path
```

Converts a database WITHOUT checksums into a database WITH checksums.

Any error found is logged. The database (and it's recovery log if present) does NOT have checksums. sparksee::↩ gdb::FileNotFoundExceptionsparksee::gdb::Error

Parameters

path	The path to the database image file

Returns

Returns true if the checksum could be added or false otherwise.

Exceptions

System.ApplicationException	null
System.IO.IOException	null

create:alias:

```
- (STSDatabase *) create:
             (NSString *) path
            alias:(NSString *) alias
```

Creates a new Database instance.

Errorfor other issues.

Parameters

path	[in] Database storage file.
alias	[in] Database alias name.

Returns

A Database instance.

Exceptions

System.ApplicationException	null

System.IO.IOException	If the given file cannot be created.

createWithConfig:alias:config:

```
- (STSDatabase *) createWithConfig:
              (NSString *) path
              alias:(NSString *) alias
              config:(STSSparkseeConfig *) config
```

Creates a new Database instance.

Errorfor other issues.

Parameters

path	[in] Database storage file.
alias	[in] Database alias name.
config	[in] Use a specific configuration instead of the one in this Sparksee

Returns

A Database instance.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file cannot be created.

decrypt:

```
- (BOOL) decrypt:
              (NSString *) path
```

Converts a database WITH encryption into a database WITHOUT encryption.

Any error found is logged. The database (and it's recovery log if present) is encrypted. The current encryption had been configured using SparkseeConfig. sparksee::gdb::FileNotFoundExceptionsparksee::gdb::Error

Parameters

path	The path to the database image file

Returns

Returns true if the database could be unencrypted. The errors are returned with an exception.

Exceptions

System.ApplicationException	null
System.IO.IOException	null

encrypt:

```
- (BOOL) encrypt:
              (NSString *) path
```

Converts a database WITHOUT encryption into an encrypted database.

Any error found is logged. The database (and it's recovery log if present) is NOT encrypted. The encryption had already been configured using SparkseeConfig. sparksee::gdb::FileNotFoundExceptionsparksee::gdb::Error

Parameters

path	The path to the database image file

Returns

Returns true if the database could be encrypted. The errors are returned with an exception.

Exceptions

System.ApplicationException	null
System.IO.IOException	null

getInMemoryPoolCapacity

```
- (long long) getInMemoryPoolCapacity
```

Gets the capacity of the in-memory pool (in Megabytes).

Returns

Returns the capacity of the in memory pool

References getInMemoryPoolCapacity.

Referenced by getInMemoryPoolCapacity.

initWithConfig:

```
- (id) initWithConfig:
              (STSSparkseeConfig *) config
```

Creates a new instance.

Parameters

config	[in/out] Sparksee configuration (may be updated).

open:readOnly:

```
- (STSDatabase *) open:
              (NSString *) path
              readOnly:(BOOL) readOnly
```

Opens an existing Database instance.

Errorfor other issues.

Parameters

path	[in] Database storage file.
readOnly	[in] If TRUE, open Database in read-only mode.

Returns

A Database instance.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file does not exist.

openWithConfig:readOnly:config:

```
- (STSDatabase *) openWithConfig:
              (NSString *) path
              readOnly:(BOOL) readOnly
              config:(STSSparkseeConfig *) config
```

Opens an existing Database instance.

The provided configuration will be used for all the database settings but not for the license, which must already be properly setup in this class. Errorfor other issues.

Parameters

path	[in] Database storage file.
readOnly	[in] If TRUE, open Database in read-only mode.
config	[in] Use a specific configuration instead of the one in this Sparksee

Returns

A Database instance.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file does not exist.

rebuildDatabase:config:minimizeDisk:

```
- (void) rebuildDatabase:
             (NSString *) path
             config:(STSSparkseeConfig *) config
             minimizeDisk:(BOOL) minimizeDisk
```

Rebuild the database from scratch with the same data.

It will create a new database on the same file with the same data but rebuilt from scratch, so the internal OIDs may change and the storage usage may be compacted.

It's the equivalent to extract a backup of the data and restore it. This operation can take a long time and the result file should be smaller than the original if there had been data removals or the data was not loaded in the most optimal order. But the file size could also end up being slightly bigger the the original in some cases (the data creation in the database was optimal and nothing had beed removed).

WARNING: If anything goes wrong while the database is being rebuilt you may lose the database. The data should not be lost, but it may require an additional process to recover it.

To avoid data loss, the next open after a failed rebuild operation will attempt to automatically recover the database by the fastest solution available (restoring the original file, finishing the almost completed rebuild or restoring a backup).

The source database is also kept until the rebuilt database is available to reduce the recovery time after a rebuild failure and avoid data loss risks. But if reducing the temporary disk usage is more important, you can use the minimizeDisk to delete it as soon as possible instead.

The final rebuilt database size will be the same whether you use the minimizeDisk argument or not.

Parameters

path	[In] The Sparksee database to rebuild.
config	[in] Use a specific configuration instead of the one in this Sparksee.
minimizeDisk	[in] Use less temporary disk space.

Exceptions

System.ApplicationException	null
System.IO.IOException	null

removeChecksums:

```
- (BOOL) removeChecksums:
             (NSString *) path
```

Converts a database WITH checksums into a database WITHOUT checksums.

Any error found is logged. The database (and it's recovery log if present) have checksums. sparksee::gdb::File↩ NotFoundExceptionsparksee::gdb::Error

Parameters

path	The path to the database image file

Returns

Returns true if the checksum could be removed. The errors are returned with an exception.

Exceptions

System.ApplicationException	null
System.IO.IOException	null

resizeInMemoryPool:

```
- (void) resizeInMemoryPool:
             (long long) newCapacity
```

Resizes the in memory pool allocator.

sparksee::gdb::SystemError

Parameters

newCapacity	The new capacity of the in memory pool allocator

Returns

Returns true if was successful

Exceptions

System.SystemException	null

restore:backupFile:

```
- (STSDatabase *) restore:
             (NSString *) path
            backupFile:(NSString *) backupFile
```

Restores a Database from a backup file.

See the Graph class Backup method. Errorfor other issues.

Parameters

path	[in] Database storage file.
backupFile	[in] The Backup file to be restored.

Returns

A Database instance.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file cannot be created, or the exported data file does not exists.

restoreEncryptedBackup:backupFile:keyInHex:ivInHex:

```
- (STSDatabase *) restoreEncryptedBackup:
             (NSString *) path
            backupFile:(NSString *) backupFile
            keyInHex:(NSString *) keyInHex
            ivInHex:(NSString *) ivInHex
```

Restores a Database from an encrypted backup file.

See the Graph class EncryptedBackup method. Errorfor other issues.

Parameters

path	[in] Database storage file.
backupFile	[in] The Backup file to be restored.

keyInHex	[In] The AES encryption Key of the backup file as an hexadecimal string (8, 16 or 32 bytes).
ivInHex	[In] The AES Initialization Vector of the backup file as an hexadecimal string (16 bytes).

Returns

A Database instance.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file cannot be created, or the exported data file does not exists.

restoreEncryptedBackupWithConfig:backupFile:config:keyInHex:ivInHex:

```
- (STSDatabase *) restoreEncryptedBackupWithConfig:
              (NSString *) path
             backupFile:(NSString *) backupFile
             config:(STSSparkseeConfig *) config
             keyInHex:(NSString *) keyInHex
             ivInHex:(NSString *) ivInHex
```

Restores a Database from an encrypted backup file.

See the Graph class EncryptedBackup method. Errorfor other issues.

Parameters

path	[in] Database storage file.
backupFile	[in] The Backup file to be restored.
config	[in] Use a specific configuration instead of the one in this Sparksee
keyInHex	[In] The AES encryption Key of the backup file as an hexadecimal string (8, 16 or 32 bytes).
ivInHex	[In] The AES Initialization Vector of the backup file as an hexadecimal string (16 bytes).

Returns

A Database instance.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file cannot be created, or the exported data file does not exists.

restoreWithConfig:backupFile:config:

```
- (STSDatabase *) restoreWithConfig:
              (NSString *) path
             backupFile:(NSString *) backupFile
             config:(STSSparkseeConfig *) config
```

Restores a Database from a backup file.

See the Graph class Backup method. Errorfor other issues.

Parameters

path	[in] Database storage file.
backupFile	[in] The Backup file to be restored.
config	[in] Use a specific configuration instead of the one in this Sparksee

Returns

A Database instance.

Exceptions

System.ApplicationException	null
System.IO.IOException	If the given file cannot be created, or the exported data file does not exists.

verifyChecksums:

```
- (BOOL) verifyChecksums:
              (NSString *) path
```

Verifies the Checksum integrity of the given image file and it's recovery log file.

The main database file checksums are always checked.

When the recovery log file exists, the recovery log file is also checked for checksum and format errors.

When the checksums of the main file or the recovery file are incorrect, the funcion returns false, in any other error it throws an exception.

Any error found is logged as a WARNING.

sparksee::gdb::FileNotFoundExceptionsparksee::gdb::Error

Parameters

path	The path to the image file

Returns

Returns true if the checksum verification succeeded. Returns false if an invalid checksum was detected.

Exceptions

System.ApplicationException	null
System.IO.IOException	null

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

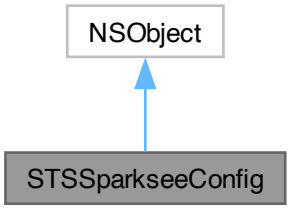
- Sparksee.h

3.59 STSSparkseeConfig Class Reference

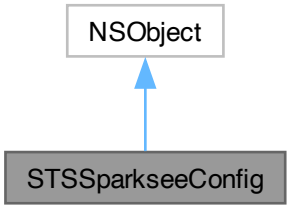
Sparksee configuration class.

`#include <Sparksee.h>`

Inheritance diagram for STSSparkseeConfig:



Collaboration diagram for STSSparkseeConfig:



Instance Methods

- (id) - init

  Creates a new instance.
- (id) - initWithPath:

  Creates a new instance with a specific config file.
- (id) - initWithPath:clientId:licenseId:

  Creates a new instance with a specific config file and IDs.
- (int) - getExtentSize

  Gets the size of a extent.
- (void) - setExtentSize:

  Sets the size of the extents in KB.
- (int) - getExtentPages

  Gets the number of pages per extent.
- (void) - setExtentPages:

  Sets the number of pages per extent.
- (int) - getPoolFrameSize

  Gets the size of a pool frame in number of extents.
- (void) - setPoolFrameSize:

  Sets the size of a pool frame in number of extents.
- (int) - getPoolPersistentMinSize

  Gets the minimum size for the persistent pool in number of frames.
- (void) - setPoolPersistentMinSize:

  Sets the minimum size for the persistent pool in number of frames.
- (int) - getPoolPersistentMaxSize

  Gets the maximum size for the persistent pool in number of frames.
- (void) - setPoolPersistentMaxSize:

  Sets the maximum size for the persistent pool in number of frames.
- (int) - getPoolTemporaryMinSize

  Gets the minimum size for the temporary pool in number of frames.
- (void) - setPoolTemporaryMinSize:

  Sets the minimum size for the temporary pool in number of frames.
- (int) - getPoolTemporaryMaxSize

  Gets the maximum size for the temporary pool in number of frames.
- (void) - setPoolTemporaryMaxSize:

  Sets the maximum size for the temporary pool in number of frames.
- (int) - getPoolPartitions

  Gets the number of partitions in each PartitionedPool.
- (void) - setPoolPartitions:

  Sets the number of pools in each PartitionedPool.
- (int) - getCacheMaxSize

  Gets the maximum size for the cache (all pools) in MB.
- (void) - setCacheMaxSize:

  Sets the maximum size for the cache (all pools) in MB.
- (NSString ∗) - getLicense

  Gets the license key.
- (int) - setLicense:

  Sets the license key.
- (NSString ∗) - getLogFile

  Gets the log file.
- (void) - setLogFile:

Sets the log file.

- (enum STSLogLevel) - getLogLevel

  Gets the log level.

- (void) - setLogLevel:

  Sets the log level.

- (BOOL) - getCacheStatisticsEnabled

  Gets whether cache statistics are enabled or disabled.

- (void) - setCacheStatisticsEnabled:

  Enables or disables cache statistics.

- (NSString *) - getCacheStatisticsFile

  Gets the cache statistics log file.

- (void) - setCacheStatisticsFile:

  Sets the cache statistics log file.

- (long long) - getCacheStatisticsSnapshotTime

  Gets the cache statistics snapshot time in microseconds.

- (void) - setCacheStatisticsSnapshotTime:

  Sets the cache statistics snapshot time.

- (BOOL) - getRollbackEnabled

  Gets whether the rollback is enabled or disabled.

- (void) - setRollbackEnabled:

  Enables or disables the rollback.

- (BOOL) - getRecoveryEnabled

  Gets whether the recovery is enabled or disabled.

- (void) - setRecoveryEnabled:

  Enables or disables the recovery.

- (NSString *) - getRecoveryLogFile

  Gets the recovery log file.

- (void) - setRecoveryLogFile:

  Sets the recovery log file.

- (int) - getRecoveryCacheMaxSize

  Gets the maximum size for the recovery log cache in extents.

- (void) - setRecoveryCacheMaxSize:

  Sets the maximum size for the recovery log cache in extents.

- (long long) - getRecoveryCheckpointTime

  Gets the delay time (in microseconds) between automatic checkpoints.

- (void) - setRecoveryCheckpointTime:

  Sets the delay time (in microseconds) between automatic checkpoints.

- (BOOL) - getChecksumEnabled

  Gets whether the storage checksum usage is enabled or disabled.

- (void) - setChecksumEnabled:

  Enables or disables the storage checksum usage.

- (BOOL) - getHighAvailabilityEnabled

  Gets whether high availability mode is enabled or disabled.

- (void) - setHighAvailabilityEnabled:

  Enables or disables high availability mode.

- (NSString *) - getHighAvailabilityIP

  Gets the IP address and port of the instance.

- (void) - setHighAvailabilityIP:

  Sets the IP address and port of the instance.

- (NSString *) - getHighAvailabilityCoordinators

  Gets the coordinators address and port list.

- (void) - setHighAvailabilityCoordinators:

  Sets the coordinators address and port list.
- (long long) - getHighAvailabilitySynchronization

  Gets the synchronization polling time.
- (void) - setHighAvailabilitySynchronization:

  Sets the synchronization polling time.
- (long long) - getHighAvailabilityMasterHistory

  Gets the master's history log.
- (void) - setHighAvailabilityMasterHistory:

  Sets the master's history log.
- (BOOL) - getCallStackDump

  Gets whether the signals will be captured to dump the call stack or not.
- (void) - setCallStackDump:

  Sets whether the signals will be captured to dump the call stack or not.
- (NSString ∗) - getClientId

  Gets the client identifier.
- (void) - setClientId:

  Set the client identifier.
- (NSString ∗) - getLicenseId

  Gets the license identifier.
- (void) - setLicenseId:

  Set the license identifier.
- (int) - getLicensePreDownloadDays

  Get the number of days before expiration when a new license will be downloaded.
- (void) - setLicensePreDownloadDays:

  Start trying to automatically download a new license at the specified number of days before expiration.
- (int) - downloadLicense

  Try to download a license key Can be used to explicitly download a license when the autodownload is disabled (LicensePreDownloadDays == -1).
- (NSString ∗) - getDownloadStatus

  Gets a message with the license download result.
- (BOOL) - **downloadExpected**

  Check if a new license will be automatically downloaded with the current settings.
- (NSString ∗) - **getLicenseRequest**

  Get information useful to manually request a license.
- (NSString ∗) - getTmpFolder

  Gets the temporary folder used for temporary staging.
- (void) - setTmpFolder:

  Sets the temporary folder used for temporary staging.
- (BOOL) - getTmpEnabled

  Gets whether using temporary storage for computations is enabled.
- (void) - setTmpEnabled:

  Sets whether to use temporary storage for computations or not.
- (long long) - getInMemAllocSize

  Gets the in-memory allocator size.
- (void) - setInMemAllocSize:

  Sets the in-memory allocator size.
- (NSString ∗) - getPluginsDir

  Gets the Plugins directory.
- (void) - setPluginsDir:

  Sets the Plugins directory.

- (BOOL) - save

  Save the current configuration in the specified config file.
- (BOOL) - saveAll

  Save all the current configuration settings in the specified config file.
- (BOOL) - getEncryptionEnabled

  Gets whether the storage encryption is enabled or disabled.
- (BOOL) - setAESEncryptionEnabled:ivInHex:

  Enables storage encryption using AES with the given key and iv.
- (NSString ∗) - getAESKey

  Get the AES encryption key in a hexadecimal encoded string.
- (NSString ∗) - getAESIV

  Get the AES initialization vector in a hexadecimal encoded string.
- (void) - **setEncryptionDisabled**

  Disables storage encryption.
- (void) - setSparkseeConfigFile:

  Sets the config file path.
- (NSString ∗) - getSparkseeConfigFile

  Gets the config file path.

3.59.1 Detailed Description

Sparksee configuration class.

If not specified, 0 means unlimited which is the maximum available. For the pools that's the total cache size. For the cache unlimited means nearly all the physical memory of the computer.

For each field, there is a default value. This value can be overrided with values from a properties file (see Sparksee↩ Properties class). Also, this settings can be overrided calling a specific setter.

For each field, it is shown its default value and the property to override this value:

Extent size: 4KB ('sparksee.storage.extentsize' at SparkseeProperties).

Pages per extent: 1 page ('sparksee.storage.extentpages' at SparkseeProperties).

Checksums enabled: true ('sparksee.storage.checksum' at SparkseeProperties).

Pool frame size: 1 extent ('sparksee.io.pool.frame.size' at SparkseeProperties).

Minimum size for the persistent pool: 64 frames ('sparksee.io.pool.persistent.minsize' at SparkseeProperties).

Maximum size for the persistent pool: 0 frames ('sparksee.io.pool.persistent.maxsize' at SparkseeProperties).

Minimum size for the temporary pool: 16 frames ('sparksee.io.pool.temporal.minsize' at SparkseeProperties).

Maximum size for the temporary pool: 0 frames ('sparksee.io.pool.temporal.maxsize' at SparkseeProperties).

Number of pools in the pool cluster: 0 pools ('sparksee.io.pool.clustersize' at SparkseeProperties). 0 or 1 means the clustering is disabled.

Maximum size for the cache (all pools): 0 MB ('sparksee.io.cache.maxsize' at SparkseeProperties).

License code: "" ('sparksee.license' at SparkseeProperties). No license code means evaluation license.

Log level: Info ('sparksee.log.level' at SparkseeProperties).

Log file: "sparksee.log" ('sparksee.log.file' at SparkseeProperties).

Cache statistics: false (disabled) ('sparksee.cache.statistics' at SparkseeProperties).

Cache statistics log file: "statistics.log" ('sparksee.cache.statisticsFile' at SparkseeProperties).

Cache statistics snapshot time: 1000 msecs [TimeUnit] ('sparksee.cache.statisticsSnapshotTime' at Sparksee↩ Properties).

Recovery enabled: false ('sparksee.io.recovery' at SparkseeProperties).

Recovery log file: "" ('sparksee.io.recovery.logfile' at SparkseeProperties).

Recovery cache max size: 1MB ('sparksee.io.recovery.cachesize' at SparkseeProperties).

Recovery checkpoint time: 60 seconds [TimeUnit] ('sparksee.io.recovery.checkpointTime' at SparkseeProperties).

High-availability: false (disabled) ('sparksee.ha' at SparkseeProperties).

High-availability coordinators: "" ('sparksee.ha.coordinators' at SparkseeProperties).

High-availability IP: "" ('sparksee.ha.ip' at SparkseeProperties).

High-availability sync polling: 0 (disabled) [TimeUnit] ('sparksee.ha.sync' at SparkseeProperties).

High-availability master history: 1D (1 day) [TimeUnit] ('sparksee.ha.master.history' at SparkseeProperties).

Use of TimeUnit:

Those variables using TimeUnit allow for:

<X>[D|H|M|S|s|m|u]

where <X> is a number followed by an optional character which represents the unit: D for days, H for hours, M for minutes, S or s for seconds, m for milliseconds and u for microseconds. If no unit character is given, seconds are assumed.

Capture abort signals to dump the call stack ('sparksee.callstackdump' at SparkseeProperties) is enabled by default on most platforms.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.59.2 Method Documentation

downloadLicense

```
- (int) downloadLicense
```

Try to download a license key Can be used to explicitly download a license when the autodownload is disabled (LicensePreDownloadDays == -1).

Returns

Returns -1 if a valid license key couldn't be downloaded, 0 if the same license was downloaded or 1 if a different license is downloaded.

References downloadLicense.

Referenced by downloadLicense.

getAESIV

```
- (NSString *) getAESIV
```

Get the AES initialization vector in a hexadecimal encoded string.

Returns

The AES initialization vector as an hexadecimal string.

References getAESIV.

Referenced by getAESIV.

getAESKey

```
- (NSString *) getAESKey
```

Get the AES encryption key in a hexadecimal encoded string.

Returns

The AES encryption Key as an hexadecimal string.

References getAESKey.

Referenced by getAESKey.

getCacheMaxSize

```
- (int) getCacheMaxSize
```

Gets the maximum size for the cache (all pools) in MB.

Returns

The maximum size for the cache (all pools) in MB.

References getCacheMaxSize.

Referenced by getCacheMaxSize.

getCacheStatisticsEnabled

```
- (BOOL) getCacheStatisticsEnabled
```

Gets whether cache statistics are enabled or disabled.

Returns

TRUE if cache statistics are enabled, FALSE otherwise.

References getCacheStatisticsEnabled.

Referenced by getCacheStatisticsEnabled.

getCacheStatisticsFile

```
- (NSString *) getCacheStatisticsFile
```

Gets the cache statistics log file.

Useless if cache statistics are disabled.

Returns

The cache statistics log file.

References getCacheStatisticsFile.

Referenced by getCacheStatisticsFile.

getCacheStatisticsSnapshotTime

```
- (long long) getCacheStatisticsSnapshotTime
```

Gets the cache statistics snapshot time in microseconds.

Useless if cache statistics are disabled.

Returns

The cache statistics snapshot time in microseconds.

References getCacheStatisticsSnapshotTime.

Referenced by getCacheStatisticsSnapshotTime.

getCallStackDump

```
- (BOOL) getCallStackDump
```

Gets whether the signals will be captured to dump the call stack or not.

Returns

TRUE if the signals must be captured, FALSE otherwise.

References getCallStackDump.

Referenced by getCallStackDump.

getChecksumEnabled

```
- (BOOL) getChecksumEnabled
```

Gets whether the storage checksum usage is enabled or disabled.

Returns

TRUE if the checksum is enabled, FALSE otherwise.

References getChecksumEnabled.

Referenced by getChecksumEnabled.

getClientId

```
- (NSString *) getClientId
```

Gets the client identifier.

Returns

The client identifier.

References getClientId.

Referenced by getClientId.

getDownloadStatus

```
- (NSString *) getDownloadStatus
```

Gets a message with the license download result.

It can be used when the DownloadLicense call failed.

Returns

The log download result.

References getDownloadStatus.

Referenced by getDownloadStatus.

getEncryptionEnabled

```
- (BOOL) getEncryptionEnabled
```

Gets whether the storage encryption is enabled or disabled.

Returns

TRUE if the encryption is enabled, FALSE otherwise.

References getEncryptionEnabled.

Referenced by getEncryptionEnabled.

getExtentPages

```
- (int) getExtentPages
```

Gets the number of pages per extent.

Returns

The number of pages per extent.

References getExtentPages.

Referenced by getExtentPages.

getExtentSize

```
- (int) getExtentSize
```

Gets the size of a extent.

Returns

The size of a extent in KB.

References getExtentSize.

Referenced by getExtentSize.

getHighAvailabilityCoordinators

```
- (NSString *) getHighAvailabilityCoordinators
```

Gets the coordinators address and port list.

Returns

The coordinators address and port list.

References getHighAvailabilityCoordinators.

Referenced by getHighAvailabilityCoordinators.

getHighAvailabilityEnabled

```
- (BOOL) getHighAvailabilityEnabled
```

Gets whether high availability mode is enabled or disabled.

Returns

TRUE if high availability mode is enabled, FALSE otherwise.

References getHighAvailabilityEnabled.

Referenced by getHighAvailabilityEnabled.

getHighAvailabilityIP

```
- (NSString *) getHighAvailabilityIP
```

Gets the IP address and port of the instance.

Returns

The IP address and port of the instance.

References getHighAvailabilityIP.

Referenced by getHighAvailabilityIP.

getHighAvailabilityMasterHistory

```
- (long long) getHighAvailabilityMasterHistory
```

Gets the master's history log.

Returns

The master's history log.

References getHighAvailabilityMasterHistory.

Referenced by getHighAvailabilityMasterHistory.

getHighAvailabilitySynchronization

```
- (long long) getHighAvailabilitySynchronization
```

Gets the synchronization polling time.

Returns

The Synchronization polling time.

References getHighAvailabilitySynchronization.

Referenced by getHighAvailabilitySynchronization.

getInMemAllocSize

```
- (long long) getInMemAllocSize
```

Gets the in-memory allocator size.

Returns

Returns the in-memory allocator size

References getInMemAllocSize.

Referenced by getInMemAllocSize.

getLicense

```
- (NSString *) getLicense
```

Gets the license key.

Returns

The license key.

References getLicense.

Referenced by getLicense.

getLicenseId

```
- (NSString *) getLicenseId
```

Gets the license identifier.

Returns

The license identifier.

References getLicenseId.

Referenced by getLicenseId.

getLicensePreDownloadDays

```
- (int) getLicensePreDownloadDays
```

Get the number of days before expiration when a new license will be downloaded.

Returns

Returns the number of days or -1 if it will never be automatically downloaded.

References getLicensePreDownloadDays.

Referenced by getLicensePreDownloadDays.

getLogFile

```
- (NSString *) getLogFile
```

Gets the log file.

Returns

The log file.

References getLogFile.

Referenced by getLogFile.

getLogLevel

```
- (enum STSLogLevel) getLogLevel
```

Gets the log level.

Returns

The LogLevel.

References getLogLevel.

Referenced by getLogLevel.

getPluginsDir

```
- (NSString *) getPluginsDir
```

Gets the Plugins directory.

Returns

The pulgins directory.

References getPluginsDir.

Referenced by getPluginsDir.

getPoolFrameSize

```
- (int) getPoolFrameSize
```

Gets the size of a pool frame in number of extents.

Returns

The size of a pool frame in number of extents.

References getPoolFrameSize.

Referenced by getPoolFrameSize.

getPoolPartitions

```
- (int) getPoolPartitions
```

Gets the number of partitions in each PartitionedPool.

Returns

The number of partitions in each PartitionedPool.

References getPoolPartitions.

Referenced by getPoolPartitions.

getPoolPersistentMaxSize

```
- (int) getPoolPersistentMaxSize
```

Gets the maximum size for the persistent pool in number of frames.

Returns

The maximum size for the persistent pool in number of frames.

References getPoolPersistentMaxSize.

Referenced by getPoolPersistentMaxSize.

getPoolPersistentMinSize

```
- (int) getPoolPersistentMinSize
```

Gets the minimum size for the persistent pool in number of frames.

Returns

The minimum size for the persistent pool in number of frames.

References getPoolPersistentMinSize.

Referenced by getPoolPersistentMinSize.

getPoolTemporaryMaxSize

```
- (int) getPoolTemporaryMaxSize
```

Gets the maximum size for the temporary pool in number of frames.

Returns

The maximum size for the temporary pool in number of frames.

References getPoolTemporaryMaxSize.

Referenced by getPoolTemporaryMaxSize.

getPoolTemporaryMinSize

```
- (int) getPoolTemporaryMinSize
```

Gets the minimum size for the temporary pool in number of frames.

Returns

The minimum size for the temporary pool in number of frames.

References getPoolTemporaryMinSize.

Referenced by getPoolTemporaryMinSize.

getRecoveryCacheMaxSize

```
- (int) getRecoveryCacheMaxSize
```

Gets the maximum size for the recovery log cache in extents.

Returns

The maximum size for the recovery log cache in extents.

References getRecoveryCacheMaxSize.

Referenced by getRecoveryCacheMaxSize.

getRecoveryCheckpointTime

```
- (long long) getRecoveryCheckpointTime
```

Gets the delay time (in microseconds) between automatic checkpoints.

Returns

The delay time (in microseconds) between automatic checkpoints.

References getRecoveryCheckpointTime.

Referenced by getRecoveryCheckpointTime.

getRecoveryEnabled

```
- (BOOL) getRecoveryEnabled
```

Gets whether the recovery is enabled or disabled.

Returns

TRUE if the recovery is enabled, FALSE otherwise.

References getRecoveryEnabled.

Referenced by getRecoveryEnabled.

getRecoveryLogFile

```
- (NSString *) getRecoveryLogFile
```

Gets the recovery log file.

Returns

The recovery log file.

References getRecoveryLogFile.

Referenced by getRecoveryLogFile.

getRollbackEnabled

```
- (BOOL) getRollbackEnabled
```

Gets whether the rollback is enabled or disabled.

Returns

TRUE if the rollback is enabled, FALSE otherwise.

References getRollbackEnabled.

Referenced by getRollbackEnabled.

getSparkseeConfigFile

```
- (NSString *) getSparkseeConfigFile
```

Gets the config file path.

Returns

Returns the configured path or an empty string

References getSparkseeConfigFile.

Referenced by getSparkseeConfigFile.

getTmpEnabled

```
- (BOOL) getTmpEnabled
```

Gets whether using temporary storage for computations is enabled.

Returns

True if enabled

References getTmpEnabled.

Referenced by getTmpEnabled.

getTmpFolder

```
- (NSString *) getTmpFolder
```

Gets the temporary folder used for temporary staging.

Returns

The temporary folder path

References getTmpFolder.

Referenced by getTmpFolder.

init

```
- (id) init
```

Creates a new instance.

Values are set with default values.

References init.

Referenced by init.

initWithPath:

```
- (id) initWithPath:
             (NSString *) path
```

Creates a new instance with a specific config file.

The config file will be loaded using the global properties class and the file may be automatically overwritten with the config changes. If the file doesn't exists, it will be created.

Parameters

path	[in] File path to the config file.

initWithPath:clientId:licenseId:

```
- (id) initWithPath:
             (NSString *) path
             clientId:(NSString *) clientId
             licenseId:(NSString *) licenseId
```

Creates a new instance with a specific config file and IDs.

The config file will be loaded using the global properties class and the file may be automatically overwritten with the config changes. If the config file already exists, the client and license ids should have the same values as the given arguments. If the file doesn't exists, it will be created.

Parameters

path	[in] File path to the config file.
clientId	[in] The client identifier.
licenseId	[in] The license identifier.

save

```
- (BOOL) save
```

Save the current configuration in the specified config file.

It will try to save the current configuration only if a config file was specified. This method will be automatically used when a Sparksee class downloads a new License.

Returns

Returns true if the config file is successfully written or false otherwise.

References save.

Referenced by save.

saveAll

```
- (BOOL) saveAll
```

Save all the current configuration settings in the specified config file.

It will try to save the current configuration only if a config file was specified. The saved config file WILL CONATAIN all the modified settings including the secret ENCRYPTION KEYS. You should usually just use the Save method instead.

Returns

Returns true if the config file is successfully written or false otherwise.

References saveAll.

Referenced by saveAll.

setAESEncryptionEnabled:ivInHex:

```
- (BOOL) setAESEncryptionEnabled:
             (NSString *) keyInHex
             ivInHex:(NSString *) ivInHex
```

Enables storage encryption using AES with the given key and iv.

The key and initialization vector will not be saved in the config file.

Parameters

keyInHex	[In] The AES encryption Key as an hexadecimal string (8, 16 or 32 bytes).
ivInHex	[In] The AES Initialization Vector as an hexadecimal string (16 bytes).

Returns

Returns true if the encryption had been enabled with the given arguments or false otherwise.

setCacheMaxSize:

```
- (void) setCacheMaxSize:
             (int) megaBytes
```

Sets the maximum size for the cache (all pools) in MB.

Parameters

megaBytes	[in] The maximum size for the cache (all pools) in MB. It must be non-negative.

setCacheStatisticsEnabled:

```
- (void) setCacheStatisticsEnabled:
             (BOOL) status
```

Enables or disables cache statistics.

Parameters

status	[in] If TRUE this enables cache statistics, if FALSE this disables cache statistics.

setCacheStatisticsFile:

```
- (void) setCacheStatisticsFile:
             (NSString *) filePath
```

Sets the cache statistics log file.

Useless if cache statistics are disabled.

Parameters

filePath	[in] The cache statistics log file.

setCacheStatisticsSnapshotTime:

```
- (void) setCacheStatisticsSnapshotTime:
             (long long) microSeconds
```

Sets the cache statistics snapshot time.

Useless if cache statistics are disabled.

Parameters

microSeconds	[in] The cache statistics snapshot time in microseconds.

setCallStackDump:

```
- (void) setCallStackDump:
             (BOOL) status
```

Sets whether the signals will be captured to dump the call stack or not.

Parameters

status	[in] If TRUE signals must be captured.

setChecksumEnabled:

```
- (void) setChecksumEnabled:
             (BOOL) status
```

Enables or disables the storage checksum usage.

Parameters

status	[in] If TRUE this enables the checksum, if FALSE then disables it.

setClientId:

```
- (void) setClientId:
             (NSString *) clientId
```

Set the client identifier.

If you don't have a client identifier, you may want to register at http://sparsity-technologies.com/↩#sparksee

Parameters

client↩Id	[in] The client identifier.

setExtentPages:

```
- (void) setExtentPages:
             (int) pages
```

Sets the number of pages per extent.

Parameters

pages	[in] The number of pages. It must be at least 1 page and the page size must be greater than or equal to 4KB.

setExtentSize:

```
- (void) setExtentSize:
             (int) kBytes
```

Sets the size of the extents in KB.

Parameters

kBytes	[in] The size of an extent in KB. An extent can have a size between 4KB and 64KB, and it must be a power of 2.

setHighAvailabilityCoordinators:

```
- (void) setHighAvailabilityCoordinators:
             (NSString *) ip
```

Sets the coordinators address and port list.

Parameters

ip	[in] The coordinators address and port list.

setHighAvailabilityEnabled:

```
- (void) setHighAvailabilityEnabled:
             (BOOL) status
```

Enables or disables high availability mode.

Parameters

status	[in] If TRUE this enables high availability mode, if FALSE this disables high availability mode.

setHighAvailabilityIP:

```
- (void) setHighAvailabilityIP:
             (NSString *) ip
```

Sets the IP address and port of the instance.

Parameters

ip	[in] The IP address and port of the instance.

setHighAvailabilityMasterHistory:

```
- (void) setHighAvailabilityMasterHistory:
             (long long) filePath
```

Sets the master's history log.

Parameters

filePath	[in] The master's history log.

setHighAvailabilitySynchronization:

```
- (void) setHighAvailabilitySynchronization:
             (long long) microSeconds
```

Sets the synchronization polling time.

Parameters

microSeconds	[in] The synchronization polling time.

setInMemAllocSize:

```
- (void) setInMemAllocSize:
             (long long) size
```

Sets the in-memory allocator size.

Parameters

size	The size to set the in-memory allocator to

setLicense:

```
- (int) setLicense:
            (NSString *) key
```

Sets the license key.

Parameters

key	[in] The license key. Returns -1 if the new key can't be used, 0 if it's the same license or 1 if the new license is applied.

setLicenseId:

```
- (void) setLicenseId:
            (NSString *) licenseId
```

Set the license identifier.

If you don't have a license identifier, you may want to register at http://sparsity-technologies.com/#sparksee

Parameters

licenseId	[in] The license identifier.

setLicensePreDownloadDays:

```
- (void) setLicensePreDownloadDays:
            (int) days
```

Start trying to automatically download a new license at the specified number of days before expiration.

Parameters

days	[in] Number of days before expiration or -1 if the license should never be downloaded.

setLogFile:

```
- (void) setLogFile:
            (NSString *) filePath
```

Sets the log file.

Parameters

filePath	[in] The log file.

setLogLevel:

```
- (void) setLogLevel:
             (enum STSLogLevel) level
```

Sets the log level.

Parameters

level	[in] The LogLevel.

setPluginsDir:

```
- (void) setPluginsDir:
             (NSString *) dirPath
```

Sets the Plugins directory.

Parameters

dirPath	[in] The plugins directory path.

setPoolFrameSize:

```
- (void) setPoolFrameSize:
             (int) extents
```

Sets the size of a pool frame in number of extents.

Parameters

extents	[in] The size of a pool frame in number of extents. It must be non-negative.

setPoolPartitions:

```
- (void) setPoolPartitions:
             (int) pools
```

Sets the number of pools in each PartitionedPool.

Parameters

pools	[in] The number of pools in each PartitionedPool. It must be non-negative.

setPoolPersistentMaxSize:

```
- (void) setPoolPersistentMaxSize:
             (int) frames
```

Sets the maximum size for the persistent pool in number of frames.

Parameters

frames	[in] The maximum size for the persistent pool in number of frames. It must be non-negative.

setPoolPersistentMinSize:

```
- (void) setPoolPersistentMinSize:
             (int) frames
```

Sets the minimum size for the persistent pool in number of frames.

Parameters

frames	[in] The minimum size for the persistent pool in number of frames. It must be non-negative.

setPoolTemporaryMaxSize:

```
- (void) setPoolTemporaryMaxSize:
             (int) frames
```

Sets the maximum size for the temporary pool in number of frames.

Parameters

frames	[in] The maximum size for the temporary pool in number of frames. It must be non-negative.

setPoolTemporaryMinSize:

```
- (void) setPoolTemporaryMinSize:
             (int) frames
```

Sets the minimum size for the temporary pool in number of frames.

Parameters

frames	[in] The minimum size for the temporary pool in number of frames. It must be non-negative.

setRecoveryCacheMaxSize:

```
- (void) setRecoveryCacheMaxSize:
             (int) extents
```

Sets the maximum size for the recovery log cache in extents.

Parameters

extents	[in] The maximum size for the recovery log cache in extents. A 0 sets the default value (extents up to 1MB).

setRecoveryCheckpointTime:

```
- (void) setRecoveryCheckpointTime:
             (long long) microSeconds
```

Sets the delay time (in microseconds) between automatic checkpoints.

Parameters

microSeconds	[in] The delay time (in microseconds) between automatic checkpoints. A 0 forces a checkpoint after each commited transaction.

setRecoveryEnabled:

```
- (void) setRecoveryEnabled:
             (BOOL) status
```

Enables or disables the recovery.

Parameters

status	[in] If TRUE this enables the recovery, if FALSE then disables it.

setRecoveryLogFile:

```
- (void) setRecoveryLogFile:
            (NSString *) filePath
```

Sets the recovery log file.

Parameters

filePath	[in] The recovery log file. Left it empty for the default log file (same as <database_file_name>.log)

setRollbackEnabled:

```
- (void) setRollbackEnabled:
            (BOOL) status
```

Enables or disables the rollback.

Parameters

status	[in] If TRUE this enables the rollback, if FALSE then disables it.

setSparkseeConfigFile:

```
- (void) setSparkseeConfigFile:
            (NSString *) path
```

Sets the config file path.

The file is not loaded in this operation, see the constructor options if you need to load the file. The config file may be automatically overwritten with the config changes. If the file doesn't exists, it will be created.

Parameters

path	[in] File path to the config file.

setTmpEnabled:

```
- (void) setTmpEnabled:
            (BOOL) enabled
```

Sets whether to use temporary storage for computations or not.

Parameters

enabled	True to use temporary storage for computation

setTmpFolder:

```
- (void) setTmpFolder:
              (NSString *) tmpFolder
```

Sets the temporary folder used for temporary staging.

Parameters

tmpFolder	The temporary folder to set

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

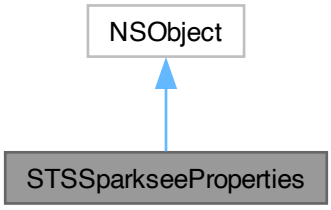
- Sparksee.h

3.60 STSSparkseeProperties Class Reference

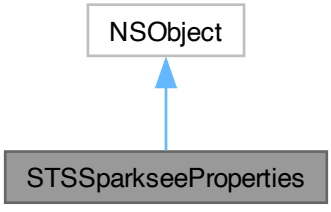
Sparksee properties file.

```
#include <Sparksee.h>
```

Inheritance diagram for STSSparkseeProperties:



Collaboration diagram for STSSparkseeProperties:



Class Methods

- (void) + load:

  Loads properties from the given file path.
- (NSString *) + get:def:

  Gets a property.
- (int) + getInteger:def:

  Gets a property as an integer.
- (BOOL) + getBoolean:def:

  Gets a property as a boolean.
- (long long) + getTimeUnit:def:

  Gets a property as a time unit.
- (void) + **clear**

  Remove all the properties.
- (void) + setHI:

  YOU SHOULD NOT USE THIS METHOD.

3.60.1 Detailed Description

Sparksee properties file.

This class is implemented as a singleton, so all public methods are static.

It allows for getting the property values stored in a properties file. A properties file is a file where there is one line per property. A property is defined by a key and a value as follows: key=value

By default, this loads properties from the file './sparksee.cfg'. The user may choose to load a different file by calling the method Load().

If the default properties file or the one loaded by the user do not exist, then this behaves as loading an empty properties file.

3.60.2 Method Documentation

get:def:

```
+ (NSString *) get:
             (NSString *) key
             def:(NSString *) def
```

Gets a property.

Parameters

key	[in] The name of the property to lookup.
def	[in] Default value to be returned in case there is no property with the name key.

Returns

The value of the property, or def if the key is not found.

getBoolean:def:

```
+ (BOOL) getBoolean:
             (NSString *) key
             def:(BOOL) def
```

Gets a property as a boolean.

Parameters

key	[in] The name of the property to lookup.
def	[in] Default value to be returned in case there is no property with the name key.

Returns

The property value, or def if the key is not found or in case of error.

getInteger:def:

```
+ (int) getInteger:
             (NSString *) key
             def:(int) def
```

Gets a property as an integer.

Parameters

key	[in] The name of the property to lookup.
def	[in] Default value to be returned in case there is no property with the name key.

Returns

The property value, or def if the key is not found or in case of error.

getTimeUnit:def:

```
+ (long long) getTimeUnit:
             (NSString *) key
             def:(long long) def
```

Gets a property as a time unit.

A time unit is a string representation of a time duration with a time unit such as '10s' or '3H'.

Valid format for the string representation: Blanks at the begining or at the end are ignored. No blanks are allowed between the time duration and the unit time.

Allowed time units: 'D' for days, 'H' for hours, 'M' for minutes, 'S' o 's' for seconds, 'm' for milliseconds and 'u' for microseconds.

There is a special case: If no time unit is given, seconds is the default. So, '10' means 10 seconds.

Parameters

key	[in] The name of the property to lookup.
def	[in] The default value (in microseconds) to be returned in case there is no property with the name key.

Returns

The time duration in microseconds, or def if the key is not found or in case of error.

load:

```
+ (void) load:
              (NSString *) path
```

Loads properties from the given file path.

Parameters

path	[in] File path to load properties from.

setHI:

```
+ (void) setHI:
              (NSString *) value
```

YOU SHOULD NOT USE THIS METHOD.

This method exists only for a specific service.

Parameters

value	null

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

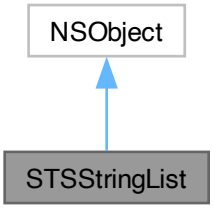
- Sparksee.h

3.61 STSStringList Class Reference

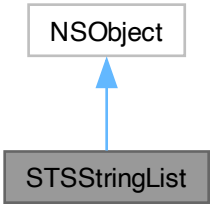
String list.

`#include <Sparksee.h>`

Inheritance diagram for STSStringList:



Collaboration diagram for STSStringList:



Instance Methods

- (int) - count

  Number of elements in the list.
- (id) - init

  Constructor.
- (void) - add:

  Adds a String at the end of the list.
- (void) - **clear**

  Clears the list.
- (id) - **initWithArray:**

  Creates a new StringList instance from the given array.
- (id) - **initWithNSEnumerator:**

  Creates a new StringList instance from the given NSEnumerator.
- (STSStringListIterator ∗) - **iterator**

  Gets a new StringListIterator.

3.61.1 Detailed Description

String list.

It stores a String (unicode) list.

Use StringListIterator to access all elements into this collection.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.61.2 Method Documentation

add:

```
- (void) add:
             (NSString *) str
```

Adds a String at the end of the list.

Parameters

str	[in] String.

count

```
- (int) count
```

Number of elements in the list.

Returns

Number of elements in the list.

References count.

Referenced by count.

init

```
- (id) init
```

Constructor.

This creates an empty list.

References init.

Referenced by init.

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

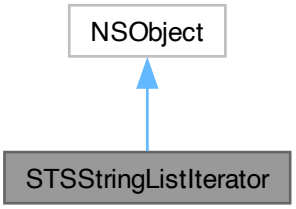
- Sparksee.h

3.62 STSStringListIterator Class Reference

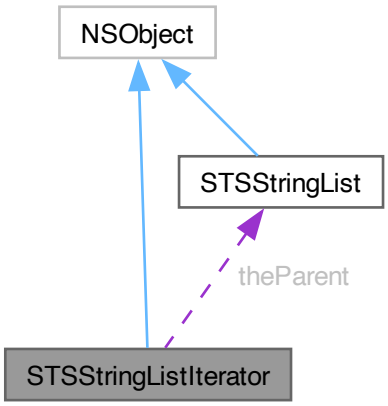
StringList iterator class.

`#include <Sparksee.h>`

Inheritance diagram for STSStringListIterator:



Collaboration diagram for STSStringListIterator:



Instance Methods

- (NSString ∗) - next

  Moves to the next element.
- (BOOL) - hasNext

  Gets if there are more elements.

3.62.1 Detailed Description

StringList iterator class.

Iterator to traverse all the strings into a StringList instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.62.2 Method Documentation

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements.

Returns

TRUE if there are more elements, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (NSString *) next
```

Moves to the next element.

Returns

The next element.

References next.

Referenced by next.

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

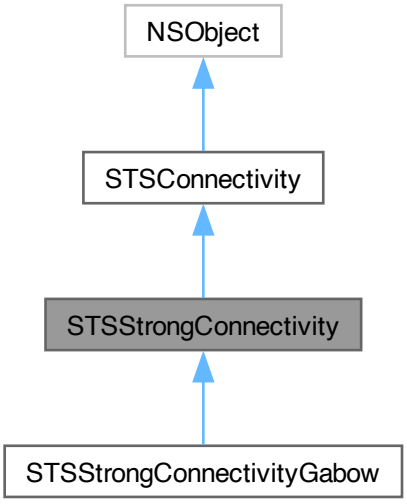
- Sparksee.h

3.63 STSStrongConnectivity Class Reference

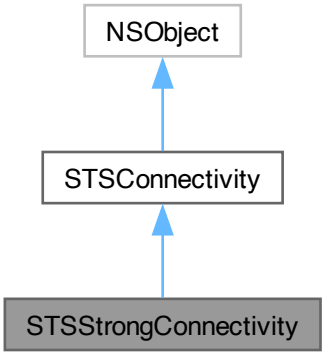
StrongConnectivity class.

`#include <Sparksee.h>`

Inheritance diagram for STSStrongConnectivity:



Collaboration diagram for STSStrongConnectivity:



Instance Methods

- (void) - addEdgeType:dir:

  Allows connectivity through edges of the given type.
- (void) - addAllEdgeTypes:

  Allows connectivity through all edge types of the graph.
- (void) - addNodeType:

  Allows connectivity through nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows connectivity through all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (STSConnectedComponents *) - getConnectedComponents

  Returns the results generated by the execution of the algorithm.
- (void) - run

  Runs the algorithm in order to find the connected components.
- (void) - setMaterializedAttribute:

  Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.
- (void) - close

  Closes the Connectivity instance.
- (BOOL) - **isClosed**

  Check if the Connectivity instance is closed.

3.63.1 Detailed Description

StrongConnectivity class.

Any class implementing this abstract class can be used to solve the problem of finding strongly connected components in a directed graph.

It consists in finding components where every pair (u,v) of nodes contained in it has a path from u to v using the specified direction for each edge type.

It is possible to set some restrictions after constructing a new instance of this class and before running it in order to limit the results.

After the execution, we can retrieve the results stored in an instance of the ConnectedComponents class using the GetConnectedComponents method.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.63.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows connectivity through all edge types of the graph.

Parameters

dir	[in] Edge direction.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
            dir:(enum STSEdgesDirection) dir
```

Allows connectivity through edges of the given type.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

addNodeType:

```
- (void) addNodeType:
             (int) t
```

Allows connectivity through nodes of the given type.

Parameters

t	null

close

```
- (void) close
```

Closes the Connectivity instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getConnectedComponents

```
- (STSConnectedComponents *) getConnectedComponents
```

Returns the results generated by the execution of the algorithm.

These results contain information related to the connected components found as the number of different components, the set of nodes contained in each component or many other data.

Returns

Returns an instance of the class ConnectedComponents which contain information related to the connected components found.

References getConnectedComponents.

Referenced by getConnectedComponents.

run

```
- (void) run
```

Runs the algorithm in order to find the connected components.

This method can be called only once.

Implemented in STSStrongConnectivityGabow, and STSWeakConnectivityDFS.

References run.

Referenced by run.

setMaterializedAttribute:

```
- (void) setMaterializedAttribute:
             (NSString *) attributeName
```

Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.

Whenever the user wants to retrieve the results, even when the graph has been closed and opened again, it is only necessary to create a new instance of the class ConnectedComponents indicating the graph and the name of the common attribute type which stores the results. This instance will have all the information related to the connected components found in the moment of the execution of the algorithm that stored this data.

It is possible to run the algorithm without specifying this parameter in order to avoid materializing the results of the execution.

Parameters

attributeName	[in] The name of the common attribute type for all node types in the graph which will store persistently the results generated by the execution of the algorithm.

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

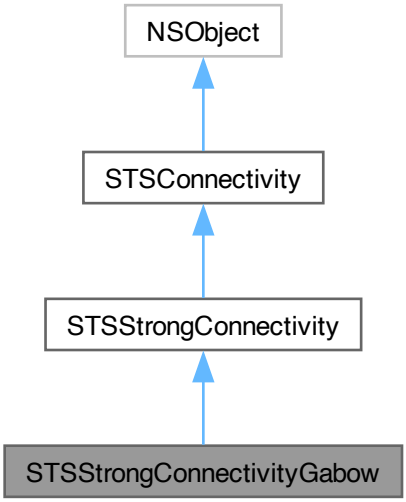
- Sparksee.h

3.64 STSStrongConnectivityGabow Class Reference

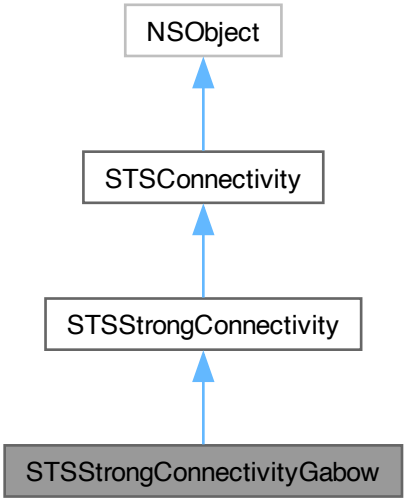
This class can be used to solve the problem of finding strongly connected components in a directed graph.

```
#include <Sparksee.h>
```

Inheritance diagram for STSStrongConnectivityGabow:



Collaboration diagram for STSStrongConnectivityGabow:



Instance Methods

- (id) - initWithSession:

  Creates a new instance of StrongConnectivityGabow.

- (void) - **run**

  Executes the algorithm.
- (void) - addEdgeType:dir:

  Allows connectivity through edges of the given type.
- (void) - addAllEdgeTypes:

  Allows connectivity through all edge types of the graph.
- (void) - addNodeType:

  Allows connectivity through nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows connectivity through all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (STSConnectedComponents ∗) - getConnectedComponents

  Returns the results generated by the execution of the algorithm.
- (void) - setMaterializedAttribute:

  Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.
- (void) - close

  Closes the Connectivity instance.
- (BOOL) - **isClosed**

  Check if the Connectivity instance is closed.

3.64.1 Detailed Description

This class can be used to solve the problem of finding strongly connected components in a directed graph.

It consists in finding components where every pair (u,v) of nodes contained in it has a path from u to v using the specified direction for each edge type. This implementation is based on the Gabow algorithm.

It is possible to set some restrictions after constructing a new instance of this class and before running it in order to limit the results.

After the execution, we can retrieve the results stored in an instance of the ConnectedComponents class using the GetConnectedComponents method.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.64.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows connectivity through all edge types of the graph.

Parameters

dir	[in] Edge direction.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
            dir:(enum STSEdgesDirection) dir
```

Allows connectivity through edges of the given type.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

addNodeType:

```
- (void) addNodeType:
             (int) t
```

Allows connectivity through nodes of the given type.

Parameters

t	null

close

```
- (void) close
```

Closes the Connectivity instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getConnectedComponents

```
- (STSConnectedComponents *) getConnectedComponents
```

Returns the results generated by the execution of the algorithm.

These results contain information related to the connected components found as the number of different components, the set of nodes contained in each component or many other data.

Returns

Returns an instance of the class ConnectedComponents which contain information related to the connected components found.

References getConnectedComponents.

Referenced by getConnectedComponents.

initWithSession:

```
- (id) initWithSession:
             (STSSession *) session
```

Creates a new instance of StrongConnectivityGabow.

After creating this instance is required to indicate the set of edge types and the set of node types which will be navigated through while traversing the graph in order to find the strong connected components.

Parameters

session	[in] Session to get the graph from and calculate the connectivity

setMaterializedAttribute:

```
- (void) setMaterializedAttribute:
             (NSString *) attributeName
```

Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.

Whenever the user wants to retrieve the results, even when the graph has been closed and opened again, it is only necessary to create a new instance of the class ConnectedComponents indicating the graph and the name of the common attribute type which stores the results. This instance will have all the information related to the connected components found in the moment of the execution of the algorithm that stored this data.

It is possible to run the algorithm without specifying this parameter in order to avoid materializing the results of the execution.

Parameters

attributeName	[in] The name of the common attribute type for all node types in the graph which will store persistently the results generated by the execution of the algorithm.

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

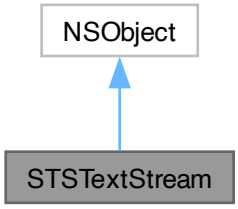
- Sparksee.h

3.65 STSTextStream Class Reference

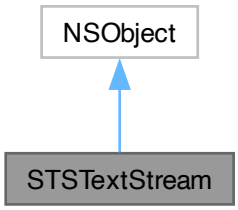
TextStream class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSTextStream:



Collaboration diagram for STSTextStream:



Instance Methods

- (id) - initWithAppend:

  Creates a new instance.
- (BOOL) - isNull

  Returns TRUE if the stream is not available.
- (void) - writeString:

  Write data to the stream.
- (NSString ∗) - readString:

  Read data from the stream.
- (void) - close

  Closes the TextStream instance.
- (BOOL) - **isClosed**

  Check if the TextStream instance is closed.

3.65.1 Detailed Description

TextStream class.

It allows for reading and writting Text attribute values.

It is very important to close the stream once no more reading or writting operations will be performed to ensure data is successfully stored.

Whereas string attributes are set and got using the Value class, text attributes are operated using a stream pattern.

Use of TextStream for writing: (i) Create a TextStream instance and (ii) set the stream for a text attribute of a node or edge instance with the graph SetAttributeText method. Once the set attribute text has been done, (iii) perform as many write operations as you need to the TextStream instance. Lastly, (iv) exeucte Close to flush and close the stream.

Use of TextStream for reading: (i) Get the stream of a text attribute of a node or edge instance with the Get↩ AttributeText graph method. Once you have the TextStream instance, (ii) you can execute Read operations to read from the stream. (iii) The end of the stream is reached when Read returns 0. Finally, (iv) execute Close to close stream resources.

Check out the 'Attributes and values' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.65.2 Method Documentation

close

```
- (void) close
```

Closes the TextStream instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

initWithAppend:

```
- (id) initWithAppend:
              (BOOL) append
```

Creates a new instance.

A TextStream only can be created by the user to write data.

Parameters

append	[in] If TRUE, the it is created in append mode to write from the end of the stream, otherwise it is created to write from the begining of the stream.

isNull

```
- (BOOL) isNull
```

Returns TRUE if the stream is not available.

It returns for reading or writing data.

Returns

FALSE if the stream is ready

References isNull.

Referenced by isNull.

readString:

```
- (NSString *) readString:
              (int) length
```

Read data from the stream.

Reads a certain amount of characters (<= length). If the result string is empty, there is no more data to be read from the stream.

Parameters

length	[in] Maximum length to be read. It must be > 0.

Returns

Returns the string readed.

writeString:

```
- (void) writeString:
             (NSString *) inStr
```

Write data to the stream.

Parameters

inStr	[in] The source string.

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

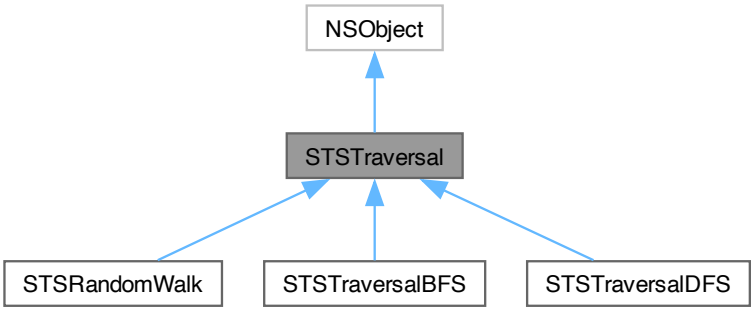
- Sparksee.h

3.66 STSTraversal Class Reference

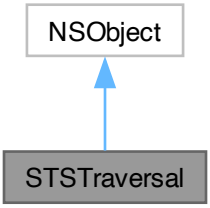
Traversal class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSTraversal:



Collaboration diagram for STSTraversal:



Instance Methods

- (void) - addEdgeType:dir:

  Allows for traversing edges of the given type.
- (void) - addAllEdgeTypes:

  Allows for traversing all edge types of the graph.
- (void) - addNodeType:

  Allows for traversing nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows for traversing all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (long long) - next

  Gets the next object of the traversal.
- (BOOL) - hasNext

  Gets if there are more objects to be traversed.
- (int) - getCurrentDepth

  Returns the depth of the current node.
- (void) - setMaximumHops:

  Sets the maximum hops restriction.
- (void) - close

  Closes the Traversal instance.
- (BOOL) - **isClosed**

  Check if the Traversal instance is closed.

3.66.1 Detailed Description

Traversal class.

Any class implementing this abstract class can be used to traverse a graph.

Once the instance has been created and the allowed node and edge types has been set, it can be used as an iterator, retrieving the next object identifier of the traversal until there are no more.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.66.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows for traversing all edge types of the graph.

Parameters

dir	[in] Edge direction.

Implemented in STSRandomWalk.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
             dir:(enum STSEdgesDirection) dir
```

Allows for traversing edges of the given type.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

Implemented in STSRandomWalk.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows for traversing nodes of the given type.

Parameters

type	The node type to add

Implemented in STSRandomWalk.

close

```
- (void) close
```

Closes the Traversal instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

Implemented in STSRandomWalk.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

Implemented in STSRandomWalk.

getCurrentDepth

```
- (int) getCurrentDepth
```

Returns the depth of the current node.

That is, it returns the depth of the node returned in the last call to Next().

Returns

The depth of the current node.

Implemented in STSRandomWalk, STSTraversalBFS, and STSTraversalDFS.

References getCurrentDepth.

Referenced by getCurrentDepth.

hasNext

```
- (BOOL) hasNext
```

Gets if there are more objects to be traversed.

Returns

TRUE if there are more objects, FALSE otherwise.

Implemented in STSRandomWalk, STSTraversalBFS, and STSTraversalDFS.

References hasNext.

Referenced by hasNext.

next

```
- (long long) next
```

Gets the next object of the traversal.

Returns

A node or edge identifier.

Implemented in STSRandomWalk, STSTraversalBFS, and STSTraversalDFS.

References next.

Referenced by next.

setMaximumHops:

```
- (void) setMaximumHops:
             (int) maxhops
```

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored.

Parameters

maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

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

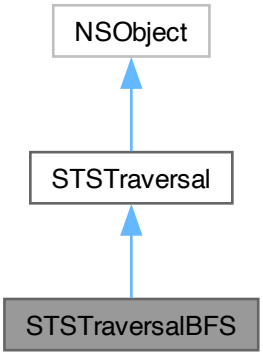
- Sparksee.h

3.67 STSTraversalBFS Class Reference

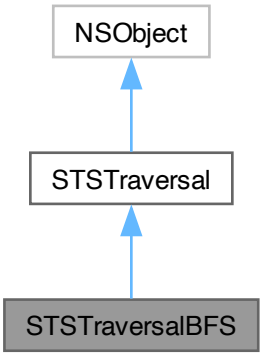
Breadth-First Search implementation of Traversal.

`#include <Sparksee.h>`

Inheritance diagram for STSTraversalBFS:



Collaboration diagram for STSTraversalBFS:



Instance Methods

- (long long) - next

  Gets the next object of the traversal.
- (BOOL) - hasNext

  Gets if there are more objects to be traversed.

- (int) - getCurrentDepth

  Returns the depth of the current node.
- (id) - initWithSession:node:

  Creates a new instance.
- (void) - addEdgeType:dir:

  Allows for traversing edges of the given type.
- (void) - addAllEdgeTypes:

  Allows for traversing all edge types of the graph.
- (void) - addNodeType:

  Allows for traversing nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows for traversing all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (void) - setMaximumHops:

  Sets the maximum hops restriction.
- (void) - close

  Closes the Traversal instance.
- (BOOL) - **isClosed**

  Check if the Traversal instance is closed.

3.67.1 Detailed Description

Breadth-First Search implementation of Traversal.

Starting from a source node, it visits all its neighbors at distance 1, then all its neighbors at distance 2, and so on.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.67.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows for traversing all edge types of the graph.

Parameters

dir	[in] Edge direction.

Implemented in STSRandomWalk.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
             dir:(enum STSEdgesDirection) dir
```

Allows for traversing edges of the given type.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

Implemented in STSRandomWalk.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows for traversing nodes of the given type.

Parameters

type	The node type to add

Implemented in STSRandomWalk.

close

```
- (void) close
```

Closes the Traversal instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

Implemented in STSRandomWalk.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

Implemented in STSRandomWalk.

getCurrentDepth

```
- (int) getCurrentDepth
```

Returns the depth of the current node.

That is, it returns the depth of the node returned in the last call to Next().

Returns

The depth of the current node.

Implements STSTraversal.

References getCurrentDepth.

Referenced by getCurrentDepth.

hasNext

```
- (BOOL) hasNext
```

Gets if there are more objects to be traversed.

Returns

TRUE if there are more objects, FALSE otherwise.

Implements STSTraversal.

References hasNext.

Referenced by hasNext.

initWithSession:node:

```
- (id) initWithSession:
             (STSSession *) session
             node:(long long) node
```

Creates a new instance.

Parameters

session	[in] Session to get the graph from and perform traversal.
node	[in] Node to start traversal from.

next

```
- (long long) next
```

Gets the next object of the traversal.

Returns

A node or edge identifier.

Implements STSTraversal.

References next.

Referenced by next.

setMaximumHops:

```
- (void) setMaximumHops:
              (int) maxhops
```

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored.

Parameters

maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

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

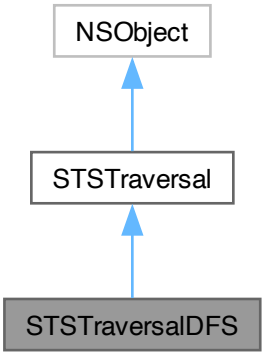
- Sparksee.h

3.68 STSTraversalDFS Class Reference

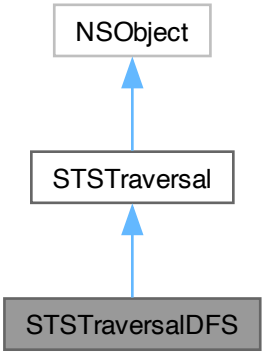
Depth-First Search (DFS) implementation of Traversal.

```
#include <Sparksee.h>
```

Inheritance diagram for STSTraversalDFS:



Collaboration diagram for STSTraversalDFS:



Instance Methods

- (long long) - next

  Gets the next object of the traversal.
- (BOOL) - hasNext

  Gets if there are more objects to be traversed.
- (int) - getCurrentDepth

  Returns the depth of the current node.
- (id) - initWithSession:node:

  Creates a new instance.
- (void) - addEdgeType:dir:

  Allows for traversing edges of the given type.
- (void) - addAllEdgeTypes:

  Allows for traversing all edge types of the graph.
- (void) - addNodeType:

  Allows for traversing nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows for traversing all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (void) - setMaximumHops:

  Sets the maximum hops restriction.
- (void) - close

  Closes the Traversal instance.
- (BOOL) - **isClosed**

  Check if the Traversal instance is closed.

3.68.1 Detailed Description

Depth-First Search (DFS) implementation of Traversal.

Starting from a source or root node, it visits as far as possible along each branch before backtracking.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.68.2 Method Documentation

addAllEdgeTypes:

```
- (void) addAllEdgeTypes:
             (enum STSEdgesDirection) dir
```

Allows for traversing all edge types of the graph.

Parameters

dir	[in] Edge direction.

Implemented in STSRandomWalk.

addEdgeType:dir:

```
- (void) addEdgeType:
             (int) type
             dir:(enum STSEdgesDirection) dir
```

Allows for traversing edges of the given type.

Parameters

type	[in] Edge type.
dir	[in] Edge direction.

Implemented in STSRandomWalk.

addNodeType:

```
- (void) addNodeType:
             (int) type
```

Allows for traversing nodes of the given type.

Parameters

type	The node type to add

Implemented in STSRandomWalk.

close

```
- (void) close
```

Closes the Traversal instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

Implemented in STSRandomWalk.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

Implemented in STSRandomWalk.

getCurrentDepth

```
- (int) getCurrentDepth
```

Returns the depth of the current node.

That is, it returns the depth of the node returned in the last call to Next().

Returns

The depth of the current node.

Implements STSTraversal.

References getCurrentDepth.

Referenced by getCurrentDepth.

hasNext

```
- (BOOL) hasNext
```

Gets if there are more objects to be traversed.

Returns

TRUE if there are more objects, FALSE otherwise.

Implements STSTraversal.

References hasNext.

Referenced by hasNext.

initWithSession:node:

```
- (id) initWithSession:
              (STSSession *) session
              node:(long long) node
```

Creates a new instance.

Parameters

session	[in] Session to get the graph from and perform traversal.
node	[in] Node to start traversal from.

next

```
- (long long) next
```

Gets the next object of the traversal.

Returns

A node or edge identifier.

Implements STSTraversal.

References next.

Referenced by next.

setMaximumHops:

```
- (void) setMaximumHops:
              (int) maxhops
```

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored.

Parameters

maxhops	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

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

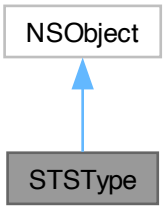
- Sparksee.h

3.69 STSType Class Reference

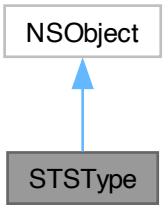
Type data class.

`#include <Sparksee.h>`

Inheritance diagram for STSType:



Collaboration diagram for STSType:



Instance Methods

- (int) - getId

  Gets the Sparksee type identifier.
- (enum STSObjectType) - getObjectType

  Gets the object type.
- (NSString ∗) - getName

  Gets the unique type name.
- (long long) - getNumObjects

  Gets the number of objects belonging to the type.
- (BOOL) - getIsDirected

  Gets if this is a directed edge type.
- (BOOL) - getIsRestricted

  Gets if this is a restricted edge type.

- (BOOL) - getAreNeighborsIndexed

  Gets if this is an edge type with neighbors index.
- (int) - getRestrictedFrom

  Gets the tail or source type identifier for restricted edge types.
- (int) - getRestrictedTo

  Gets the head or target type identifier for restricted edge types.

Class Methods

- (int) + **getInvalidType**

  Invalid type identifier.
- (int) + **getGlobalType**

  Global type identifier.
- (int) + **getNodesType**

  Identifier for all nodeType attributes.
- (int) + **getEdgesType**

  Identifier for all edgeType attributes.

3.69.1 Detailed Description

Type data class.

It contains information about a node or edge type.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.69.2 Method Documentation

getAreNeighborsIndexed

```
- (BOOL) getAreNeighborsIndexed
```

Gets if this is an edge type with neighbors index.

Returns

TRUE for edges types with neighbors index, FALSE otherwise.

References getAreNeighborsIndexed.

Referenced by getAreNeighborsIndexed.

getId

```
- (int) getId
```

Gets the Sparksee type identifier.

Returns

The Sparksee type identifier.

References getId.

Referenced by getId.

getIsDirected

```
- (BOOL) getIsDirected
```

Gets if this is a directed edge type.

Returns

TRUE for directed edge types, FALSE otherwise.

References getIsDirected.

Referenced by getIsDirected.

getIsRestricted

```
- (BOOL) getIsRestricted
```

Gets if this is a restricted edge type.

Returns

TRUE for restricted edge types, FALSE otherwise.

References getIsRestricted.

Referenced by getIsRestricted.

getName

```
- (NSString *) getName
```

Gets the unique type name.

Returns

The unique type name.

References getName.

Referenced by getName.

getNumObjects

```
- (long long) getNumObjects
```

Gets the number of objects belonging to the type.

Returns

The number of objects belonging to the type.

References getNumObjects.

Referenced by getNumObjects.

getObjectType

```
- (enum STSObjectType) getObjectType
```

Gets the object type.

Returns

The object type.

References getObjectType.

Referenced by getObjectType.

getRestrictedFrom

```
- (int) getRestrictedFrom
```

Gets the tail or source type identifier for restricted edge types.

Returns

For restricted edge types, the tail or source type identifier, the Type InvalidType otherwise.

References getRestrictedFrom.

Referenced by getRestrictedFrom.

getRestrictedTo

```
- (int) getRestrictedTo
```

Gets the head or target type identifier for restricted edge types.

Returns

For restricted edge types, the head or target type identifier, the Type InvalidType otherwise.

References getRestrictedTo.

Referenced by getRestrictedTo.

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

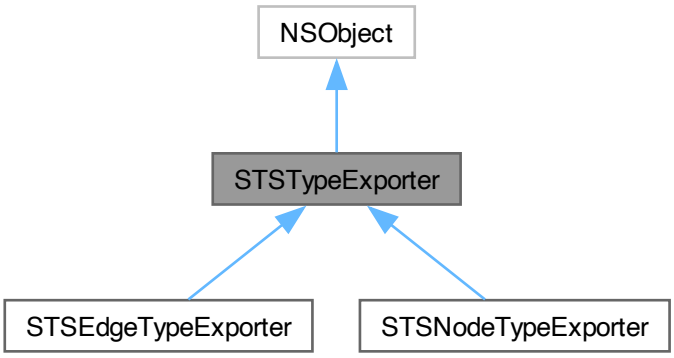
- Sparksee.h

3.70 STSTypeExporter Class Reference

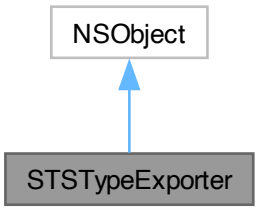
Base TypeExporter class.

`#include <Sparksee.h>`

Inheritance diagram for STSTypeExporter:



Collaboration diagram for STSTypeExporter:



Instance Methods

- (void) - registerListener:
  Registers a new listener.
- (void) - run
  Runs export process.
- (void) - setRowWriter:
  Sets the output data destination.
- (void) - setGraph:
  Sets the graph that will be exported.

- (void) - setType:

  Sets the type to be exported.
- (void) - setAttributes:

  Sets the list of Attributes.
- (void) - setFrequency:

  Sets the frequency of listener notification.
- (void) - setHeader:

  Sets the presence of a header row.

3.70.1 Detailed Description

Base TypeExporter class.

Base class to export a node or edge type from a graph using a RowWriter.

TypeExporterListener can be registered to receive information about the progress of the export process by means of TypeExporterEvent. The default frequency of notification to listeners is 100000.

By default no header row is created.

Check out the 'Data export' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.70.2 Method Documentation

registerListener:

```
- (void) registerListener:
            (STSTypeExporterListener *) tel
```

Registers a new listener.

Parameters

tel	[in] TypeExporterListener to be registered.

run

```
- (void) run
```

Runs export process.

WrongArgumentErrorIn case of bad settings ErrorFor other issues

Exceptions

System.ApplicationException	null
System.ArgumentException	null
System.IO.IOException	If bad things happen writting to the RowWriter.

Implemented in STSEdgeTypeExporter, and STSNodeTypeExporter.

References run.

Referenced by run.

setAttributes:

```
- (void) setAttributes:
             (STSAttributeList *) attrs
```

Sets the list of Attributes.

Parameters

attrs	[in] Attribute identifiers to be exported

setFrequency:

```
- (void) setFrequency:
             (int) freq
```

Sets the frequency of listener notification.

Parameters

freq	[in] Frequency in number of rows managed to notify progress to all listeners

setGraph:

```
- (void) setGraph:
             (STSGraph *) graph
```

Sets the graph that will be exported.

Parameters

graph	[in] Graph.

setHeader:

```
- (void) setHeader:
             (BOOL) header
```

Sets the presence of a header row.

Parameters

header	[in] If TRUE, a header row is dumped with the name of the attributes.

setRowWriter:

```
- (void) setRowWriter:
             (STSRowWriter *) rw
```

Sets the output data destination.

Parameters

rw	[in] Input RowWriter.

setType:

```
- (void) setType:
             (int) type
```

Sets the type to be exported.

Parameters

type	[in] Type identifier.

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

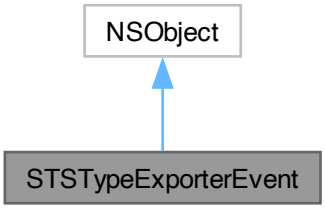
- Sparksee.h

3.71 STSTypeExporterEvent Class Reference

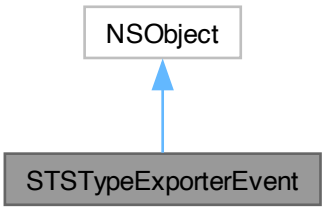
Provides information about the progress of an TypeExproter instance.

`#include <Sparksee.h>`

Inheritance diagram for STSTypeExporterEvent:



Collaboration diagram for STSTypeExporterEvent:



Instance Methods

- (int) - getTypeId

  Gets the type identifier.
- (long long) - getCount

  Gets the current number of objects exported.
- (long long) - getTotal

  Gets the total number of objects exported.
- (BOOL) - isLast

  Gets if this is the last event or not.

3.71.1 Detailed Description

Provides information about the progress of an TypeExproter instance.

Check out the 'Data export' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.71.2 Method Documentation

getCount

```
- (long long) getCount
```

Gets the current number of objects exported.

Returns

The current number of objects exported.

References getCount.

Referenced by getCount.

getTotal

```
- (long long) getTotal
```

Gets the total number of objects exported.

Returns

The total number of objects exported.

References getTotal.

Referenced by getTotal.

getTypeId

```
- (int) getTypeId
```

Gets the type identifier.

Returns

The type identifier.

References getTypeId.

Referenced by getTypeId.

isLast

```
- (BOOL) isLast
```

Gets if this is the last event or not.

Returns

TRUE if this is the last event, FALSE otherwise.

References isLast.

Referenced by isLast.

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

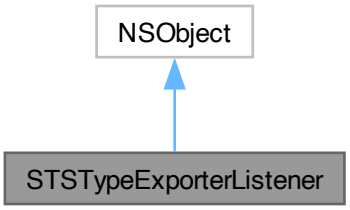
- Sparksee.h

3.72 STSTypeExporterListener Class Reference

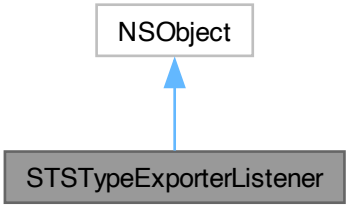
Interface to be implemented to receive TypeExporterEvent events from a TypeExporter.

```
#include <Sparksee.h>
```

Inheritance diagram for STSTypeExporterListener:



Collaboration diagram for STSTypeExporterListener:



Instance Methods

- (void) - notifyEvent:

  Method to be notified from a TypeExporter.

3.72.1 Detailed Description

Interface to be implemented to receive TypeExporterEvent events from a TypeExporter.

Check out the 'Data export' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.72.2 Method Documentation

notifyEvent:

```
- (void) notifyEvent:
             (STSTypeExporterEvent *) tee
```

Method to be notified from a TypeExporter.

Parameters

tee	[in] Notified event.

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

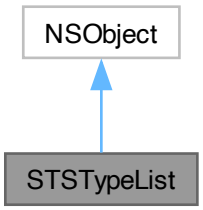
- Sparksee.h

3.73 STSTypeList Class Reference

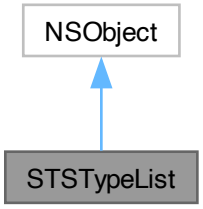
Sparksee type identifier list.

`#include <Sparksee.h>`

Inheritance diagram for STSTypeList:



Collaboration diagram for STSTypeList:



Instance Methods

- (int) - count

  Number of elements in the list.
- (id) - init

  Constructor.
- (void) - add:

  Adds a Sparksee type identifier at the end of the list.
- (void) - **clear**

  Clears the list.
- (id) - **initWithArray:**

  Creates a new TypeList instance from the given array.
- (id) - **initWithNSEnumerator:**

  Creates a new TypeList instance from the given NSEnumerator.
- (STSTypeListIterator ∗) - **iterator**

  Gets a new TypeListIterator.

3.73.1 Detailed Description

Sparksee type identifier list.

It stores a Sparksee node or edge type identifier list.

Use TypeListIterator to access all elements into this collection.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.73.2 Method Documentation

add:

```
- (void) add:
              (int) type
```

Adds a Sparksee type identifier at the end of the list.

Parameters

type	[in] Sparksee type identifier.

count

```
- (int) count
```

Number of elements in the list.

Returns

Number of elements in the list.

References count.

Referenced by count.

init

```
- (id) init
```

Constructor.

This creates an empty list.

References init.

Referenced by init.

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

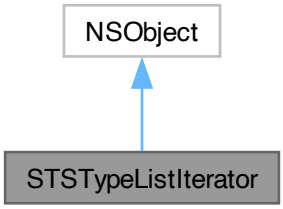
- Sparksee.h

3.74 STSTypeListIterator Class Reference

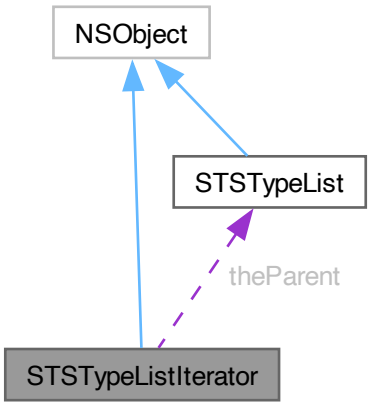
TypeList iterator class.

`#include <Sparksee.h>`

Inheritance diagram for STSTypeListIterator:



Collaboration diagram for STSTypeListIterator:



Instance Methods

- (int) - next

  Moves to the next element.
- (BOOL) - hasNext

  Gets if there are more elements.

3.74.1 Detailed Description

TypeList iterator class.

Iterator to traverse all the Sparksee node or edge type identifiers into a TypeList instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.74.2 Method Documentation

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements.

Returns

TRUE if there are more elements, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (int) next
```

Moves to the next element.

Returns

The next element.

References next.

Referenced by next.

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

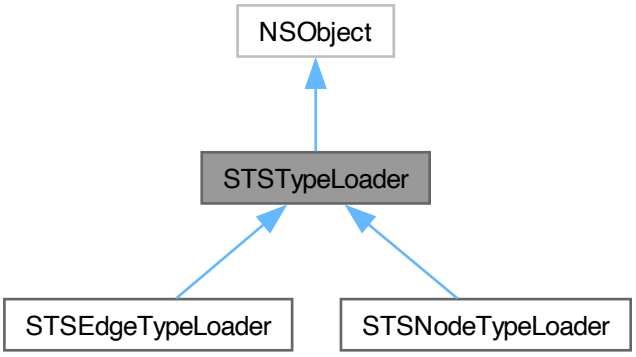
- Sparksee.h

3.75 STSTypeLoader Class Reference

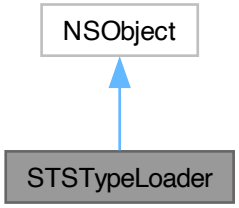
Base TypeLoader class.

`#include <Sparksee.h>`

Inheritance diagram for STSTypeLoader:



Collaboration diagram for STSTypeLoader:



Instance Methods

- (void) - setLogError:
  Sets a log error file.
- (void) - setLogOff
  Truns off all the error reporting.
- (void) - registerListener:
  Registers a new listener.
- (void) - run
  Run the loader.

- (void) - runTwoPhases

  Run the loader for two phases loading.
- (void) - runNPhases:

  Run the loader for N phases loading.
- (void) - setRowReader:

  Sets the input data source.
- (void) - setGraph:

  Sets the graph where the data will be loaded.
- (void) - setLocale:

  Sets the locale that will be used to read the data.
- (void) - setType:

  Sets the type to be loaded.
- (void) - setAttributes:

  Sets the list of Attributes.
- (void) - setAttributePositions:

  Sets the list of attribute positions.
- (void) - setTimestampFormat:

  Sets a specific timestamp format.
- (void) - setFrequency:

  Sets the frequency of listener notification.

3.75.1 Detailed Description

Base TypeLoader class.

Base class to load a node or edge type from a graph using a RowReader.

TypeLoaderListener can be registered to receive information about the progress of the load process by means of TypeLoaderEvent. The default frequency of notification to listeners is 100000.

Check out the 'Data import' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.75.2 Method Documentation

registerListener:

```
- (void) registerListener:
             (STSTypeLoaderListener *) tel
```

Registers a new listener.

Parameters

tel	TypeLoaderListener to be registered.

run

```
- (void) run
```

Run the loader.

ErrorFor other issues

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen using to the RowReader.

Implemented in STSEdgeTypeLoader, and STSNodeTypeLoader.

References run.

Referenced by run.

runNPhases:

```
- (void) runNPhases:
             (int) partitions
```

Run the loader for N phases loading.

Firstly load all objects (and create them if necessary) and secondly loads all the attributes. But in this case, attributes are loaded one by one. This way, if there are three attributes, then 4 traverses are necessary.

Working on this mode it is necessary to build a temporary file. ErrorFor other issues

Parameters

partitions	[in] Number of horizontal partitions to perform the load.

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen using to the RowReader.

Implemented in STSEdgeTypeLoader, and STSNodeTypeLoader.

runTwoPhases

```
- (void) runTwoPhases
```

Run the loader for two phases loading.

Firstly load all objects (and create them if necessary) and secondly loads all the attributes.

Working on this mode it is necessary to build a temporary file. ErrorFor other issues

Exceptions

System.ApplicationException	null
System.IO.IOException	If bad things happen using to the RowReader.

Implemented in STSEdgeTypeLoader, and STSNodeTypeLoader.

References runTwoPhases.

Referenced by runTwoPhases.

setAttributePositions:

```
- (void) setAttributePositions:
             (STSInt32List *) attrsPos
```

Sets the list of attribute positions.

Parameters

attrsPos	[in] Attribute positions (column index >=0).

setAttributes:

```
- (void) setAttributes:
             (STSAttributeList *) attrs
```

Sets the list of Attributes.

Parameters

attrs	[in] Attribute identifiers to be loaded

setFrequency:

```
- (void) setFrequency:
             (int) freq
```

Sets the frequency of listener notification.

Parameters

freq	[in] Frequency in number of rows managed to notify progress to all listeners

setGraph:

```
- (void) setGraph:
             (STSGraph *) graph
```

Sets the graph where the data will be loaded.

Parameters

graph	[in] Graph.

setLocale:

```
- (void) setLocale:
             (NSString *) localeStr
```

Sets the locale that will be used to read the data.

It should match the locale used in the rowreader.

Parameters

localeStr	[in] The locale string for the read data. See CSVReader.

setLogError:

```
- (void) setLogError:
             (NSString *) path
```

Sets a log error file.

By default errors are thrown as a exception and the load process ends. If a log file is set, errors are logged there and the load process does not stop.

Parameters

path	[in] The path to the error log file.

Exceptions

System.IO.IOException	If bad things happen opening the file.

setLogOff

```
- (void) setLogOff
```

Truns off all the error reporting.

The log file will not be created and no exceptions for invalid data will be thrown. If you just want to turn off the logs, but abort at the first error what you should do is not call this method and not set a logError file.

References setLogOff.

Referenced by setLogOff.

setRowReader:

```
- (void) setRowReader:
            (STSRowReader *) rr
```

Sets the input data source.

Parameters

rr	[in] Input RowReader.

setTimestampFormat:

```
- (void) setTimestampFormat:
            (NSString *) timestampFormat
```

Sets a specific timestamp format.

Parameters

timestampFormat	[in] A string with the timestamp format definition.

setType:

```
- (void) setType:
              (int) type
```

Sets the type to be loaded.

Parameters

type	[in] Type identifier.

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

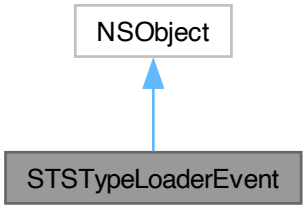
- Sparksee.h

3.76 STSTypeLoaderEvent Class Reference

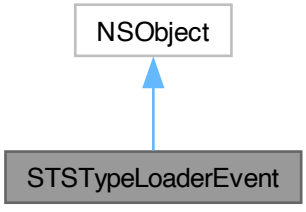
Provides information about the progress of a TypeLoader instance.

`#include <Sparksee.h>`

Inheritance diagram for STSTypeLoaderEvent:



Collaboration diagram for STSTypeLoaderEvent:



3.76.1 Detailed Description

Provides information about the progress of a TypeLoader instance.

Check out the 'Data import' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

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

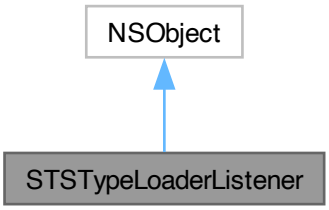
- Sparksee.h

3.77 STSTypeLoaderListener Class Reference

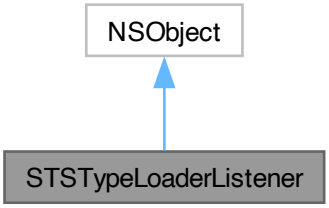
Interface to be implemented to receive TypeLoaderEvent events from a TypeLoader.

`#include <Sparksee.h>`

Inheritance diagram for STSTypeLoaderListener:



Collaboration diagram for STSTypeLoaderListener:



Instance Methods

- (void) - notifyEvent:

  Method to receive events from a Loader.

3.77.1 Detailed Description

Interface to be implemented to receive TypeLoaderEvent events from a TypeLoader.

Check out the 'Data import' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.77.2 Method Documentation

notifyEvent:

```
- (void) notifyEvent:
             (STSTypeLoaderEvent *) ev
```

Method to receive events from a Loader.

Parameters

ev	Loader.LoaderEvent with information from a running Loader.

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

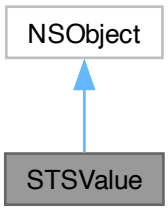
- Sparksee.h

3.78 STSValue Class Reference

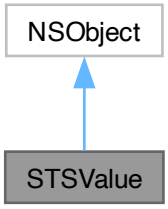
Value class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSValue:



Collaboration diagram for STSValue:



Instance Methods

- (id) - init

  Creates a new instance.
- (id) - initWithValue:

  Copy constructor.
- (BOOL) - isNull

  Gets if this is a NULL Value.
- (void) - **setNullVoid**

  Sets the Value to NULL.
- (enum STSDataType) - getDataType

  Gets the DataType.
- (BOOL) - getBoolean

  Gets Boolean Value.
- (int) - getInteger

  Gets Integer Value.
- (long long) - getLong

  Gets Long Value.
- (double) - getDouble

  Gets Double Value.
- (long long) - getTimestamp

  Gets Timestamp Value.
- (NSString ∗) - getString

  Gets String Value.
- (long long) - getOid

  Gets OID Value.
- (void) - setBooleanVoid:

  Sets the Value.
- (void) - setIntegerVoid:

  Sets the Value.
- (void) - setLongVoid:

  Sets the Value.
- (void) - setDoubleVoid:

  Sets the Value.
- (void) - setTimestampVoid:

Sets the Value.

- (void) - setTimestampVoidWithYear:month:day:hour:minutes:seconds:millisecs:

  Sets the Value.
- (void) - setStringVoid:

  Sets the Value.
- (void) - setOidVoid:

  Sets the OID Value.
- (void) - setVoid:

  Sets the Value.
- (int) - compare:

  Compares with the given Value.
- (BOOL) - equals:

  Compares with the given Value.
- (NSString ∗) - **stringValue**

  Returns the receiver's value as a human-readable string.
- (BOOL) - **isEqual:**

  Check if both Value instances are equal.
- (NSUInteger) - **hash**

  Get the hash value of this Value.
- (STSValue ∗) - setNull

  Sets the value to NULL.
- (STSValue ∗) - setBoolean:

  Sets the value to the given boolean.
- (STSValue ∗) - setInteger:

  Sets the value to the given integer number.
- (STSValue ∗) - setLong:

  Sets the value to the given long long number.
- (STSValue ∗) - setDouble:

  Sets the value to the given double number.
- (STSValue ∗) - setTimestamp:

  Sets the value to the given timestamp.
- (STSValue ∗) - setTimestampWithYear:month:day:hour:minutes:seconds:milliseconds:

  Sets the value to the given timestamp.
- (STSValue ∗) - setString:

  Sets the value to the given string.
- (STSValue ∗) - setOid:

  Sets the value to the given OID.
- (STSValue ∗) - set:

  Sets the value to the given Value.
- (NSDate ∗) - getTimestampAsNSDate

  Gets Timestamp Value as a NSDate.
- (STSValue ∗) - setTimestampWithNSDate:

  Sets the value to the given NSDate timestamp.

Class Methods

- (int) + **getMaxLengthString**

  Maximum number of characters allowed for a String.

3.78.1 Detailed Description

Value class.

It is a container which stores a value and its data type (domain). A Value can be NULL.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.78.2 Method Documentation

compare:

```
- (int) compare:
              (STSValue *) value
```

Compares with the given Value.

It does not work if the given Value objects does not have the same DataType.

Parameters

value	Given value to compare to.

Returns

0 if this Value is equal to the given one; a value less than 0 if this Value is less than the given one; and a value greater than 0 if this Value is greater than the given one.

equals:

```
- (BOOL) equals:
              (STSValue *) value
```

Compares with the given Value.

It does not work if the given Value objects does not have the same DataType.

Parameters

value	Given value to compare to.

Returns

TRUE if this Value is equal to the given one; FALSE otherwise.

getBoolean

```
- (BOOL) getBoolean
```

Gets Boolean Value.

This must be a non-NULL Boolean Value.

Returns

The Boolean Value.

References getBoolean.

Referenced by getBoolean.

getDataType

```
- (enum STSDataType) getDataType
```

Gets the DataType.

Value cannot be NULL.

Returns

The DataType.

References getDataType.

Referenced by getDataType.

getDouble

```
- (double) getDouble
```

Gets Double Value.

This must be a non-NULL Double Value.

Returns

The Double Value.

References getDouble.

Referenced by getDouble.

getInteger

```
- (int) getInteger
```

Gets Integer Value.

This must be a non-NULL Integer Value.

Returns

The Integer Value.

References getInteger.

Referenced by getInteger.

getLong

```
- (long long) getLong
```

Gets Long Value.

This must be a non-NULL Long Value.

Returns

The Long Value.

References getLong.

Referenced by getLong.

getOid

```
- (long long) getOid
```

Gets OID Value.

This must be an non-NULL OID Value.

Returns

The OID Value.

References getOid.

Referenced by getOid.

getString

```
- (NSString *) getString
```

Gets String Value.

This must be a non-NULL String Value.

Returns

The String Value.

References getString.

Referenced by getString.

getTimestamp

```
- (long long) getTimestamp
```

Gets Timestamp Value.

This must be a non-NULL Timestamp Value.

Returns

The Timestamp Value.

References getTimestamp.

Referenced by getTimestamp.

getTimestampAsNSDate

```
- (NSDate *) getTimestampAsNSDate
```

Gets Timestamp Value as a NSDate.

This must be a non-NULL Timestamp Value.

Returns

The Timestamp Value.

References getTimestampAsNSDate.

Referenced by getTimestampAsNSDate.

init

```
- (id) init
```

Creates a new instance.

It creates a NULL Value.

References init.

Referenced by init.

initWithValue:

```
- (id) initWithValue:
              (STSValue *) value
```

Copy constructor.

Parameters

value	[in] Value to be copied.

isNull

```
- (BOOL) isNull
```

Gets if this is a NULL Value.

Returns

TRUE if this is a NULL Value, FALSE otherwise.

References isNull.

Referenced by isNull.

set:

```
- (STSValue *) set:
              (STSValue *) value
```

Sets the value to the given Value.

Parameters

value	[in] The new value.

Returns

Returns this Value.

setBoolean:

```
- (STSValue *) setBoolean:
              (BOOL) value
```

Sets the value to the given boolean.

Parameters

value	[in] The new value.

Returns

Returns this Value.

setBooleanVoid:

```
- (void) setBooleanVoid:
              (BOOL) value
```

Sets the Value.

Parameters

value	[in] New Boolean value.

setDouble:

```
- (STSValue *) setDouble:
              (double) value
```

Sets the value to the given double number.

Parameters

value	[in] The new value.

Returns

Returns this Value.

setDoubleVoid:

```
- (void) setDoubleVoid:
              (double) value
```

Sets the Value.

Parameters

value	[in] New Double value.

setInteger:

```
- (STSValue *) setInteger:
              (int) value
```

Sets the value to the given integer number.

Parameters

value	[in] The new value.

Returns

Returns this Value.

setIntegerVoid:

```
- (void) setIntegerVoid:
              (int) value
```

Sets the Value.

Parameters

value	[in] New Integer value.

setLong:

```
- (STSValue *) setLong:
              (long long) value
```

Sets the value to the given long long number.

Parameters

value	[in] The new value.

Returns

Returns this Value.

setLongVoid:

```
- (void) setLongVoid:
             (long long) value
```

Sets the Value.

Parameters

value	[in] New Long value.

setNull

```
- (STSValue *) setNull
```

Sets the value to NULL.

Returns

Returns this Value.

References setNull.

Referenced by setNull.

setOid:

```
- (STSValue *) setOid:
             (long long) value
```

Sets the value to the given OID.

Parameters

value	[in] The new value.

Returns

Returns this Value.

setOidVoid:

```
- (void) setOidVoid:
             (long long) value
```

Sets the OID Value.

Parameters

value	[in] New OID value.

setString:

```
- (STSValue *) setString:
             (NSString *) value
```

Sets the value to the given string.

Parameters

value	[in] The new value.

Returns

Returns this Value.

setStringVoid:

```
- (void) setStringVoid:
             (NSString *) value
```

Sets the Value.

Parameters

value	[in] New String value.

setTimestamp:

```
- (STSValue *) setTimestamp:
             (long long) value
```

Sets the value to the given timestamp.

Parameters

value	[in] The new value.

Returns

Returns this Value.

setTimestampVoid:

```
- (void) setTimestampVoid:
             (long long) value
```

Sets the Value.

Parameters

value	[in] New Timestamp value.

setTimestampVoidWithYear:month:day:hour:minutes:seconds:millisecs:

```
- (void) setTimestampVoidWithYear:
             (int) year
             month:(int) month
             day:(int) day
             hour:(int) hour
             minutes:(int) minutes
             seconds:(int) seconds
             millisecs:(int) millisecs
```

Sets the Value.

Parameters

year	[in] The year (>=1970).
month	[in] The month ([1..12]).
day	[in] The of the month ([1..31]).
hour	[in] The hour ([0..23]).
minutes	[in] The minutes ([0..59]).
seconds	[in] The seconds ([0..59]).
millisecs	[in] The milliseconds ([0..999]).

setTimestampWithNSDate:

```
- (STSValue *) setTimestampWithNSDate:
              (NSDate *) date
```

Sets the value to the given NSDate timestamp.

Parameters

value	[in] The new value.

Returns

Returns this Value.

setTimestampWithYear:month:day:hour:minutes:seconds:milliseconds:

```
- (STSValue *) setTimestampWithYear:
              (int) year
             month:(int) month
             day:(int) day
             hour:(int) hour
             minutes:(int) minutes
             seconds:(int) seconds
             milliseconds:(int) millisecs
```

Sets the value to the given timestamp.

Parameters

year	[in] The year (>=1970).
month	[in] The month ([1..12]).
day	[in] The of the month ([1..31]).
hour	[in] The hour ([0..23]).
minutes	[in] The minutes ([0..59]).
seconds	[in] The seconds ([0..59]).
millisecs	[in] The milliseconds ([0..999]).

Returns

Returns this Value.

setVoid:

```
- (void) setVoid:
              (STSValue *) value
```

Sets the Value.

Parameters

value	[in] New value.

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

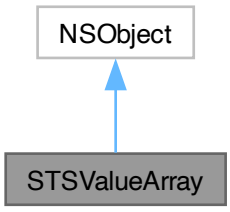
- Sparksee.h

3.79 STSValueArray Class Reference

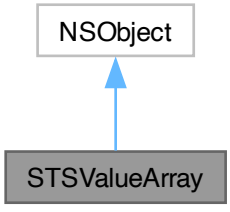
ValueArray class.

`#include <Sparksee.h>`

Inheritance diagram for STSValueArray:



Collaboration diagram for STSValueArray:



Instance Methods

- (void) - get:value:

  Get a Value from the array.
- (void) - setAt:value:

  Set a Value to a specific array position.
- (void) - set:

  Set a Value to the whole array.
- (void) - makeNull

  Sets the attribute array to Null.
- (int) - size

  Returns the array size.
- (void) - setDouble:

  Set all the values of this double array.
- (void) - setDoubleRange:values:

  Set a subset of the values of this double array.
- (void) - setInteger:

  Set all the values of this int array.
- (void) - setIntegerRange:values:

  Set a subset of the values of this int array.
- (void) - setLong:

  Set all the values of this long array.
- (void) - setLongRange:values:

  Set a subset of the values of this long array.
- (void) - setBoolean:

  Set all the values of this bool array.
- (void) - setBooleanRange:values:

  Set a subset of the values of this bool array.
- (void) - setTimestamp:

  Set all the values of this timestamp array.
- (void) - setTimestampRange:values:

  Set a subset of the values of this timestamp array.
- (void) - setOid:

  Set all the values of this oid array.
- (void) - setOidRange:values:

  Set a subset of the values of this oid array.
- (NSArray ∗) - getDouble

  Get all the Values from this double array.
- (NSArray ∗) - getDoubleRange:size:

  Get a subset of the Values from this double array.
- (NSArray ∗) - getInteger

  Get all the values from this int array.
- (NSArray ∗) - getIntegerRange:size:

  Get a subset of the values from this int array.
- (NSArray ∗) - getLong

  Get all the values from this long array.
- (NSArray ∗) - getBoolean

  Get all the values from this bool array.
- (NSArray ∗) - getTimestamp

  Get all the values from this timestamp array.
- (NSArray ∗) - getOid

Get all the values from this oid array.

- (void) - close

  Closes the ValueArray instance.

- (BOOL) - **isClosed**

  Check if the ValueArray instance is closed.

3.79.1 Detailed Description

ValueArray class.

It allows for getting and setting ValueArray attribute values.

It is very important to close the ValueArray once no more get or set operations will be performed.

Creation of a new ValueArray: (i) Set all the ValueArray elements using Graph::SetAttributeArray (ii) perform as many get/set operations as you need to the ValueArray instance. Lastly, (iii) Close the ValueArray

Use of an existing ValueArray: (i) Get a ValueArray instance using Graph::GetAttributeArray (ii) perform as many get/set operations as you need to the ValueArray instance. Lastly, (iii) Close the ValueArray

Check out the 'Attributes and values' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.79.2 Method Documentation

close

```
- (void) close
```

Closes the ValueArray instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

get:value:

```
- (void) get:
             (int) index
             value:(STSValue *) value
```

Get a Value from the array.

AppErrorIf the ValueArray is not available

Parameters

index	[in] Position of the element to get [0..N-1]
value	[out] Value to get the array element

Exceptions

System.ArgumentException	If the index is out of range or the Value not valid
System.ApplicationException	null

getBoolean

```
- (NSArray *) getBoolean
```

Get all the values from this bool array.

Exceptions

AppError	If the ValueArray is not available
UnsupportedOperationError	If array DataType is not bool

References getBoolean.

Referenced by getBoolean.

getDouble

```
- (NSArray *) getDouble
```

Get all the Values from this double array.

Returns

Returns all the array values

References getDouble.

Referenced by getDouble.

getDoubleRange:size:

```
- (NSArray *) getDoubleRange:
              (int) index
             size:(int) size
```

Get a subset of the Values from this double array.

Parameters

index	[in] Position of the first element to get [0..N-1]
- size	[in] Number of elements to get [1..N]

Returns

Returns the array values in the selected range

getInteger

```
- (NSArray *) getInteger
```

Get all the values from this int array.

Exceptions

AppError	If the ValueArray is not available
UnsupportedOperationError	If array DataType is not int

References getInteger.

Referenced by getInteger.

getIntegerRange:size:

```
- (NSArray *) getIntegerRange:
              (int) index
             size:(int) size
```

Get a subset of the values from this int array.

Parameters

index	[in] Position of the first element to get [0..N-1]
- size	[in] Number of elements to get [1..N]

Exceptions

WrongArgumentError	If the index or size is out of range
UnsupportedOperationError	If array DataType is not int
AppError	If the ValueArray is not available

getLong

```
- (NSArray *) getLong
```

Get all the values from this long array.

Exceptions

AppError	If the ValueArray is not available
UnsupportedOperationError	If array DataType is not long

References getLong.

Referenced by getLong.

getOid

```
- (NSArray *) getOid
```

Get all the values from this oid array.

Exceptions

AppError	If the ValueArray is not available
UnsupportedOperationError	If array DataType is not oid

References getOid.

Referenced by getOid.

getTimestamp

```
- (NSArray *) getTimestamp
```

Get all the values from this timestamp array.

Exceptions

AppError	If the ValueArray is not available

UnsupportedOperationError	If array DataType is not timestamp

References getTimestamp.

Referenced by getTimestamp.

makeNull

```
- (void) makeNull
```

Sets the attribute array to Null.

The ValueArray can not be used after this call.

Exceptions

System.ApplicationException	null

References makeNull.

Referenced by makeNull.

set:

```
- (void) set:
             (STSValue *) value
```

Set a Value to the whole array.

AppErrorIf the ValueArray is not available

Parameters

value	[in] Value to set in all the array elements

Exceptions

System.ArgumentException	If the Value is not valid
System.ApplicationException	null

setAt:value:

```
- (void) setAt:
              (int) index
             value:(STSValue *) value
```

Set a Value to a specific array position.

AppErrorIf the ValueArray is not available

Parameters

index	[in] Position of the element to set [0..N-1]
value	[in] Value to set in the array element

Exceptions

System.ArgumentException	If the index is out of range or the Value not valid
System.ApplicationException	null

setBoolean:

```
- (void) setBoolean:
              (NSArray *) values
```

Set all the values of this bool array.

AppErrorIf the ValueArray is not available

Parameters

values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ApplicationException	If array DataType is not bool

setBooleanRange:values:

```
- (void) setBooleanRange:
              (int) index
             values:(NSArray *) values
```

Set a subset of the values of this bool array.

UnsupportedOperationErrorIf array DataType is not bool AppErrorIf the ValueArray is not available

Parameters

index	[in] Position of the first element to set [0..N-1]
values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ArgumentException	If the index or size is out of range
System.ApplicationException	null

setDouble:

```
- (void) setDouble:
             (NSArray *) values
```

Set all the values of this double array.

AppErrorIf the ValueArray is not available

Parameters

values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ApplicationException	If array DataType is not Double

setDoubleRange:values:

```
- (void) setDoubleRange:
             (int) index
             values:(NSArray *) values
```

Set a subset of the values of this double array.

UnsupportedOperationErrorIf array DataType is not Double AppErrorIf the ValueArray is not available

Parameters

index	[in] Position of the first element to set [0..N-1]
values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ArgumentException	If the index or size is out of range
System.ApplicationException	null

setInteger:

```
- (void) setInteger:
             (NSArray *) values
```

Set all the values of this int array.

AppErrorIf the ValueArray is not available

Parameters

values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ApplicationException	If array DataType is not int

setIntegerRange:values:

```
- (void) setIntegerRange:
             (int) index
             values:(NSArray *) values
```

Set a subset of the values of this int array.

UnsupportedOperationErrorIf array DataType is not int AppErrorIf the ValueArray is not available

Parameters

index	[in] Position of the first element to set [0..N-1]
values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ArgumentException	If the index or size is out of range
System.ApplicationException	null

setLong:

```
- (void) setLong:
             (NSArray *) values
```

Set all the values of this long array.

AppErrorIf the ValueArray is not available

Parameters

values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ApplicationException	If array DataType is not long

setLongRange:values:

```
- (void) setLongRange:
             (int) index
             values:(NSArray *) values
```

Set a subset of the values of this long array.

UnsupportedOperationErrorIf array DataType is not long AppErrorIf the ValueArray is not available

Parameters

index	[in] Position of the first element to set [0..N-1]
values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ArgumentException	If the index or size is out of range
System.ApplicationException	null

setOid:

```
- (void) setOid:
             (NSArray *) values
```

Set all the values of this oid array.

AppErrorIf the ValueArray is not available

Parameters

values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ApplicationException	If array DataType is not oid

setOidRange:values:

```
- (void) setOidRange:
             (int) index
             values:(NSArray *) values
```

Set a subset of the values of this oid array.

UnsupportedOperationErrorIf array DataType is not oid AppErrorIf the ValueArray is not available

Parameters

index	[in] Position of the first element to set [0..N-1]
values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ArgumentException	If the index or size is out of range
System.ApplicationException	null

setTimestamp:

```
- (void) setTimestamp:
             (NSArray *) values
```

Set all the values of this timestamp array.

AppErrorIf the ValueArray is not available

Parameters

values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ApplicationException	If array DataType is not timestamp

setTimestampRange:values:

```
- (void) setTimestampRange:
              (int) index
              values:(NSArray *) values
```

Set a subset of the values of this timestamp array.

UnsupportedOperationErrorIf array DataType is not timestamp AppErrorIf the ValueArray is not available

Parameters

index	[in] Position of the first element to set [0..N-1]
values	[in] All the values to set

Exceptions

System.ApplicationException	null
System.ArgumentException	If the index or size is out of range
System.ApplicationException	null

size

```
- (int) size
```

Returns the array size.

Returns

The number of elements in the array.

References size.

Referenced by size.

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

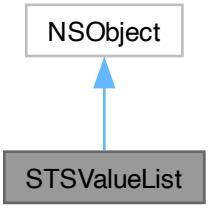
- Sparksee.h

3.80 STSValueList Class Reference

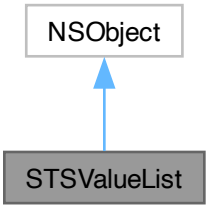
Value list.

`#include <Sparksee.h>`

Inheritance diagram for STSValueList:



Collaboration diagram for STSValueList:



Instance Methods

- (int) - count
  Number of elements in the list.
- (id) - init
  Constructor.
- (void) - **clear**
  Clears the list.
- (void) - add:
  Adds a value to the end of the list.
- (STSValue ∗) - get:
  Returns the Value at the specified position in the list.
- (id) - **initWithArray:**
  Creates a new ValueList instance from the given array.
- (id) - **initWithNSEnumerator:**
  Creates a new ValueList instance from the given NSEnumerator.
- (STSValueListIterator ∗) - **iterator**
  Gets a new ValueListIterator.

3.80.1 Detailed Description

Value list.

It stores a Value list.

Use ValueListIterator to access all elements into this collection.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.80.2 Method Documentation

add:

```
- (void) add:
              (STSValue *) value
```

Adds a value to the end of the list.

Parameters

value	[in] The value to add

count

```
- (int) count
```

Number of elements in the list.

Returns

Number of elements in the list.

References count.

Referenced by count.

get:

```
- (STSValue *) get:
                (int) index
```

Returns the Value at the specified position in the list.

Parameters

index	[in] Index of the element to return, starting at 0.

init

```
- (id) init
```

Constructor.

This creates an empty list.

References init.

Referenced by init.

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

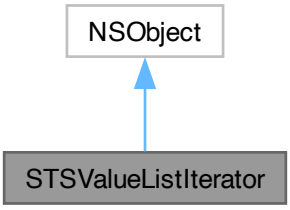
- Sparksee.h

3.81 STSValueListIterator Class Reference

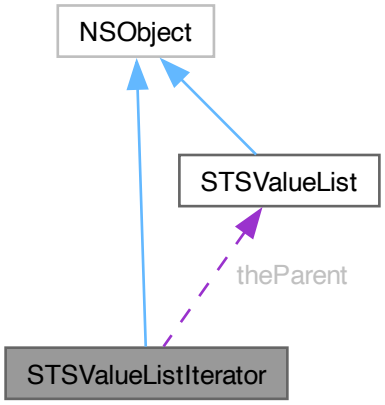
ValueList iterator class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSValueListIterator:



Collaboration diagram for STSValueListIterator:



Instance Methods

- (STSValue ∗) - next
  Moves to the next element.
- (BOOL) - hasNext
  Gets if there are more elements.

3.81.1 Detailed Description

ValueList iterator class.

Iterator to traverse all the values into a ValueList instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.81.2 Method Documentation

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements.

Returns

TRUE if there are more elements, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (STSValue *) next
```

Moves to the next element.

Returns

The next element.

References next.

Referenced by next.

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

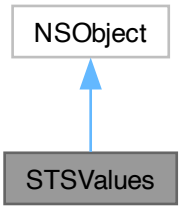
- Sparksee.h

3.82 STSValues Class Reference

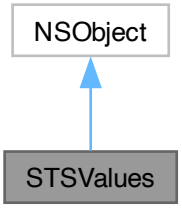
Value set class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSValues:



Collaboration diagram for STSValues:



Instance Methods

- (long long) - count

  Gets the number of elements into the collection.
- (STSValuesIterator *) - iterator:

  Gets a ValuesIterator.
- (void) - close

  Closes the Values instance.
- (BOOL) - **isClosed**

  Check if the Values instance is closed.

3.82.1 Detailed Description

Value set class.

This is a set of Value instances, that is there is no duplicated elements.

Use a ValuesIterator to traverse all the elements into the set.

When the Values instance is closed, it closes all existing and non-closed ValuesIterator instances too.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.82.2 Method Documentation

close

```
- (void) close
```

Closes the Values instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

count

```
- (long long) count
```

Gets the number of elements into the collection.

Returns

The number of elements into the collection.

References count.

Referenced by count.

iterator:

```
- (STSValuesIterator *) iterator:
              (enum STSOrder) order
```

Gets a ValuesIterator.

Parameters

order	[in] Ascending or descending order.

Returns

ValuesIterator instance.

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

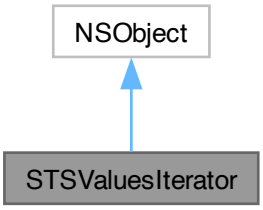
- Sparksee.h

3.83 STSValuesIterator Class Reference

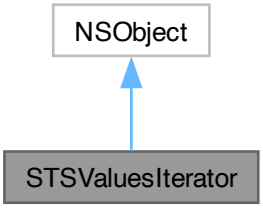
Values iterator class.

`#include <Sparksee.h>`

Inheritance diagram for STSValuesIterator:



Collaboration diagram for STSValuesIterator:



Instance Methods

- (BOOL) - hasNext

  Gets if there are more elements to traverse.
- (STSValue ∗) - next

  Gets the next element to traverse.
- (void) - close

  Closes the ValuesIterator instance.
- (BOOL) - **isClosed**

  Check if the ValuesIterator instance is closed.

3.83.1 Detailed Description

Values iterator class.

It allows for traversing all the elements into a Values instance.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.83.2 Method Documentation

close

```
- (void) close
```

Closes the ValuesIterator instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

hasNext

```
- (BOOL) hasNext
```

Gets if there are more elements to traverse.

Returns

TRUE if there are more elements to traverse, FALSE otherwise.

References hasNext.

Referenced by hasNext.

next

```
- (STSValue *) next
```

Gets the next element to traverse.

Returns

The next element.

References next.

Referenced by next.

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

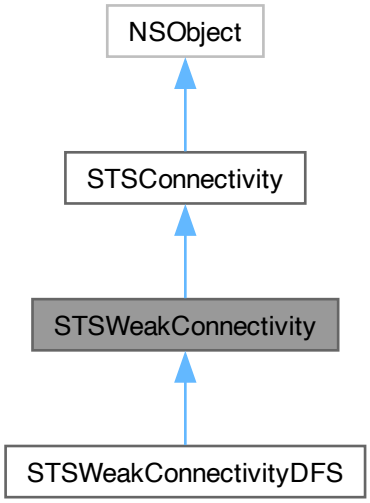
- Sparksee.h

3.84 STSWeakConnectivity Class Reference

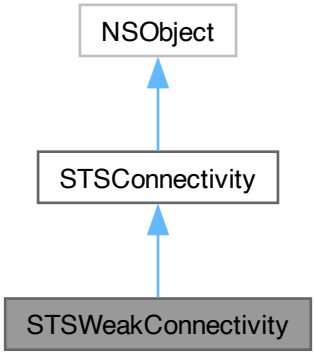
WeakConnectivity class.

```
#include <Sparksee.h>
```

Inheritance diagram for STSWeakConnectivity:



Collaboration diagram for STSWeakConnectivity:



Instance Methods

- (void) - addEdgeType:

  Allows connectivity through edges of the given type.
- (void) - addAllEdgeTypes

  Allows connectivity through all edge types of the graph.
- (void) - addNodeType:

  Allows connectivity through nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows connectivity through all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (STSConnectedComponents ∗) - getConnectedComponents

  Returns the results generated by the execution of the algorithm.
- (void) - run

  Runs the algorithm in order to find the connected components.
- (void) - setMaterializedAttribute:

  Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.
- (void) - close

  Closes the Connectivity instance.
- (BOOL) - **isClosed**

  Check if the Connectivity instance is closed.

3.84.1 Detailed Description

WeakConnectivity class.

Any class implementing this abstract class can be used to solve the problem of finding weakly connected components in an undirected graph or in a directed graph which will be considered as an undirected one.

It consists in finding components where every pair (u,v) of nodes contained in it has a path from u to v and from v to u.

It is possible to set some restrictions after constructing a new instance of this class and before running it in order to limit the results.

After the execution, we can retrieve the results stored in an instance of the ConnectedComponents class using the getConnectedComponents method.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.84.2 Method Documentation

addAllEdgeTypes

```
- (void) addAllEdgeTypes
```

Allows connectivity through all edge types of the graph.

In a weak connectivity the edges can be used in Any direction.

References addAllEdgeTypes.

Referenced by addAllEdgeTypes.

addEdgeType:

```
- (void) addEdgeType:
             (int) type
```

Allows connectivity through edges of the given type.

In a weak connectivity the edges can be used in Any direction.

Parameters

type	[in] Edge type.

addNodeType:

```
- (void) addNodeType:
             (int) t
```

Allows connectivity through nodes of the given type.

Parameters

t	null

close

```
- (void) close
```

Closes the Connectivity instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getConnectedComponents

```
- (STSConnectedComponents *) getConnectedComponents
```

Returns the results generated by the execution of the algorithm.

These results contain information related to the connected components found as the number of different components, the set of nodes contained in each component or many other data.

Returns

Returns an instance of the class ConnectedComponents which contain information related to the connected components found.

References getConnectedComponents.

Referenced by getConnectedComponents.

run

```
- (void) run
```

Runs the algorithm in order to find the connected components.

This method can be called only once.

Implemented in STSStrongConnectivityGabow, and STSWeakConnectivityDFS.

References run.

Referenced by run.

setMaterializedAttribute:

```
- (void) setMaterializedAttribute:
            (NSString *) attributeName
```

Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.

Whenever the user wants to retrieve the results, even when the graph has been closed and opened again, it is only necessary to create a new instance of the class ConnectedComponents indicating the graph and the name of the common attribute type which stores the results. This instance will have all the information related to the connected components found in the moment of the execution of the algorithm that stored this data.

It is possible to run the algorithm without specifying this parameter in order to avoid materializing the results of the execution.

Parameters

attributeName	[in] The name of the common attribute type for all node types in the graph which will store persistently the results generated by the execution of the algorithm.

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

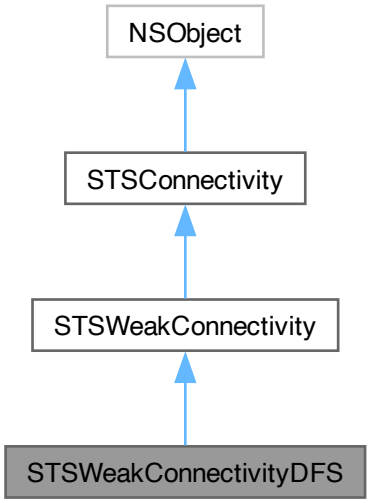
- Sparksee.h

3.85 STSWeakConnectivityDFS Class Reference

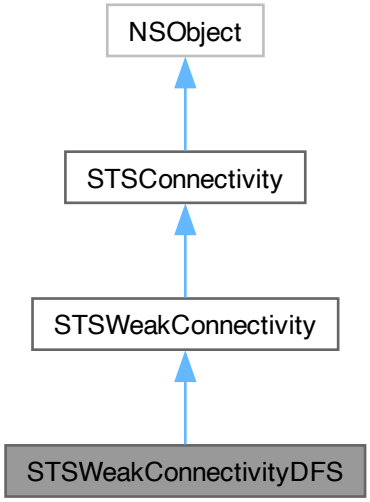
WeakConnectivityDFS class.

`#include <Sparksee.h>`

Inheritance diagram for STSWeakConnectivityDFS:



Collaboration diagram for STSWeakConnectivityDFS:



Instance Methods

- (id) - initWithSession:

  Creates a new instance of WeakConnectivityDFS.
- (void) - **run**

  Executes the algorithm.
- (void) - addEdgeType:

  Allows connectivity through edges of the given type.
- (void) - addAllEdgeTypes

  Allows connectivity through all edge types of the graph.
- (void) - addNodeType:

  Allows connectivity through nodes of the given type.
- (void) - **addAllNodeTypes**

  Allows connectivity through all node types of the graph.
- (void) - excludeNodes:

  Set which nodes can't be used.
- (void) - excludeEdges:

  Set which edges can't be used.
- (STSConnectedComponents ∗) - getConnectedComponents

  Returns the results generated by the execution of the algorithm.
- (void) - setMaterializedAttribute:

  Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.
- (void) - close

  Closes the Connectivity instance.
- (BOOL) - **isClosed**

  Check if the Connectivity instance is closed.

3.85.1 Detailed Description

WeakConnectivityDFS class.

This class can be used to solve the problem of finding weakly connected components in an undirected graph or in a directed graph which will be considered as an undirected one.

It consists in finding components where every pair (u,v) of nodes contained in it has a path from u to v and from v to u. This implementation is based on the Depth-First Search (DFS) algorithm.

It is possible to set some restrictions after constructing a new instance of this class and before running it in order to limit the results.

After the execution, we can retrieve the results stored in an instance of the ConnectedComponents class using the getConnectedComponents method.

Check out the 'Algorithms' section in the SPARKSEE User Manual for more details on this.

Author

Sparsity Technologies http://www.sparsity-technologies.com

3.85.2 Method Documentation

addAllEdgeTypes

```
- (void) addAllEdgeTypes
```

Allows connectivity through all edge types of the graph.

In a weak connectivity the edges can be used in Any direction.

References addAllEdgeTypes.

Referenced by addAllEdgeTypes.

addEdgeType:

```
- (void) addEdgeType:
             (int) type
```

Allows connectivity through edges of the given type.

In a weak connectivity the edges can be used in Any direction.

Parameters

type	[in] Edge type.

addNodeType:

```
- (void) addNodeType:
             (int) t
```

Allows connectivity through nodes of the given type.

Parameters

t	null

close

```
- (void) close
```

Closes the Connectivity instance.

It must be called to ensure the integrity of all data.

References close.

Referenced by close.

excludeEdges:

```
- (void) excludeEdges:
             (STSObjects *) edges
```

Set which edges can't be used.

This will replace any previously specified set of excluded edges. Should only be used to exclude the usage of specific edges from allowed edge types because it's less efficient than not allowing an edge type.

Parameters

edges	[in] A set of edge identifiers that must be kept intact until the destruction of the class.

excludeNodes:

```
- (void) excludeNodes:
             (STSObjects *) nodes
```

Set which nodes can't be used.

This will replace any previously specified set of excluded nodes. Should only be used to exclude the usage of specific nodes from allowed node types because it's less efficient than not allowing a node type.

Parameters

nodes	[in] A set of node identifiers that must be kept intact until the destruction of the class.

getConnectedComponents

```
- (STSConnectedComponents *) getConnectedComponents
```

Returns the results generated by the execution of the algorithm.

These results contain information related to the connected components found as the number of different components, the set of nodes contained in each component or many other data.

Returns

Returns an instance of the class ConnectedComponents which contain information related to the connected components found.

References getConnectedComponents.

Referenced by getConnectedComponents.

initWithSession:

```
- (id) initWithSession:
             (STSSession *) session
```

Creates a new instance of WeakConnectivityDFS.

After creating this instance is required to indicate the set of edge types and the set of node types which will be navigated through while traversing the graph in order to find the weak connected components.

Parameters

session	[in] Session to get the graph from and calculate the connectivity

setMaterializedAttribute:

```
- (void) setMaterializedAttribute:
             (NSString *) attributeName
```

Creates a new common attribute type for all node types in the graph in order to store, persistently, the results related to the connected components found while executing this algorithm.

Whenever the user wants to retrieve the results, even when the graph has been closed and opened again, it is only necessary to create a new instance of the class ConnectedComponents indicating the graph and the name of the common attribute type which stores the results. This instance will have all the information related to the connected components found in the moment of the execution of the algorithm that stored this data.

It is possible to run the algorithm without specifying this parameter in order to avoid materializing the results of the execution.

Parameters

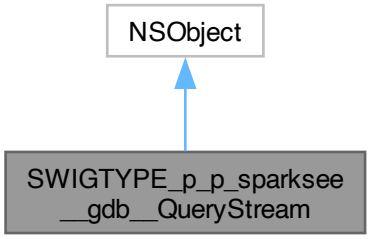
attributeName	[in] The name of the common attribute type for all node types in the graph which will store persistently the results generated by the execution of the algorithm.

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

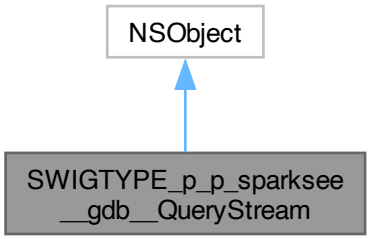
- Sparksee.h

3.86 SWIGTYPE_p_p_sparksee__gdb__QueryStream Class Reference

Inheritance diagram for SWIGTYPE_p_p_sparksee__gdb__QueryStream:



Collaboration diagram for SWIGTYPE_p_p_sparksee__gdb__QueryStream:

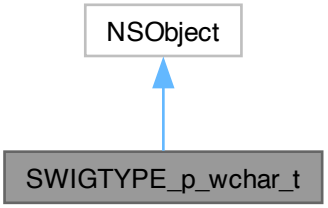


The documentation for this class was generated from the following file:

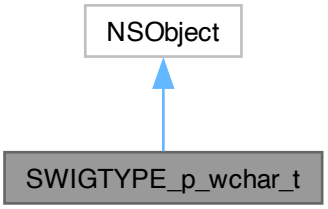
- Sparksee.h

3.87 SWIGTYPE_p_wchar_t Class Reference

Inheritance diagram for SWIGTYPE_p_wchar_t:



Collaboration diagram for SWIGTYPE_p_wchar_t:



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

- Sparksee.h

Index

add:
- STSAttributeList, 14
- STSBooleanList, 24
- STSInt32List, 163
- STSObjects, 194
- STSOidList, 204
- STSStringList, 319
- STSTypeList, 364
- STSValueList, 403

addAllEdgeTypes
- STSCommunitiesSCD, 29
- STSDisjointCommunityDetection, 90
- STSWeakConnectivity, 412
- STSWeakConnectivityDFS, 417

addAllEdgeTypes:
- STSContext, 45
- STSPageRank, 209
- STSRandomWalk, 227
- STSShortestPath, 254
- STSSinglePairShortestPath, 259
- STSSinglePairShortestPathBFS, 264
- STSSinglePairShortestPathDijkstra, 270
- STSStrongConnectivity, 324
- STSStrongConnectivityGabow, 329
- STSTraversal, 337
- STSTraversalBFS, 341
- STSTraversalDFS, 347

addChecksums:
- STSSparksee, 278

addEdgeType:
- STSCommunitiesSCD, 29
- STSDisjointCommunityDetection, 90
- STSWeakConnectivity, 412
- STSWeakConnectivityDFS, 417

addEdgeType:d:
- STSContext, 45

addEdgeType:dir:
- STSPageRank, 209
- STSRandomWalk, 227
- STSShortestPath, 254
- STSSinglePairShortestPath, 259
- STSSinglePairShortestPathBFS, 264
- STSSinglePairShortestPathDijkstra, 270
- STSStrongConnectivity, 324
- STSStrongConnectivityGabow, 329
- STSTraversal, 337
- STSTraversalBFS, 341
- STSTraversalDFS, 347

addNodeType:
- STSCommunitiesSCD, 29
- STSCommunityDetection, 34
- STSConnectivity, 41
- STSContext, 46
- STSDisjointCommunityDetection, 90
- STSPageRank, 210
- STSRandomWalk, 227
- STSShortestPath, 255
- STSSinglePairShortestPath, 259
- STSSinglePairShortestPathBFS, 264
- STSSinglePairShortestPathDijkstra, 270
- STSStrongConnectivity, 324
- STSStrongConnectivityGabow, 329
- STSTraversal, 337
- STSTraversalBFS, 342
- STSTraversalDFS, 347
- STSWeakConnectivity, 412
- STSWeakConnectivityDFS, 417

addQueryStreamManager:
- STSDatabase, 63

addWeightedEdgeType:dir:attr:
- STSSinglePairShortestPathDijkstra, 270

any
- STSObjects, 194

asDirected
- STSEdgeExport, 98

backup:
- STSGraph, 128

calculateEdgeCost:sourceCost:sourceLevel:targetNode:edge:edgeWeight
- STSSinglePairShortestPathDijkstraDynamicCost, 275

checkOnlyExistence
- STSSinglePairShortestPathBFS, 265

clone
- STSObjects, 194

cloneWithObjects:
- STSObjects, 194

close
- STSCommunitiesSCD, 29
- STSCommunityDetection, 34
- STSConnectedComponents, 38
- STSConnectivity, 41
- STSContext, 46
- STSCSVReader, 53
- STSCSVWriter, 59
- STSDatabase, 63
- STSDisjointCommunities, 86
- STSDisjointCommunityDetection, 91
- STSExportManager, 119
- STSKeyValues, 168
- STSKOpt, 171
- STSObjects, 195
- STSObjectsIterator, 202
- STSPageRank, 210
- STSRandomWalk, 227
- STSRowReader, 242
- STSRowWriter, 245
- STSSession, 251
- STSShortestPath, 255
- STSSinglePairShortestPath, 259

STSSinglePairShortestPathBFS, 265
STSSinglePairShortestPathDijkstra, 271
STSStrongConnectivity, 324
STSStrongConnectivityGabow, 329
STSTextStream, 333
STSTraversal, 337
STSTraversalBFS, 342
STSTraversalDFS, 348
STSValueArray, 391
STSValues, 407
STSValuesIterator, 409
STSWeakConnectivity, 413
STSWeakConnectivityDFS, 418
combineDifference:objs2:
STSObjects, 195
combineIntersection:objs2:
STSObjects, 195
combineUnion:objs2:
STSObjects, 196
compare:
STSValue, 378
compute
STSContext, 46
computeWithArguments:node:nodeTypes:edgeTypes:dir:maxhops:include:
STSContext, 46
contains:
STSObjects, 196
copyDefaults:
STSEdgeExport, 98
STSNodeExport, 173
count
STSAttributeList, 14
STSBooleanList, 24
STSInt32List, 163
STSObjects, 197
STSOidList, 204
STSResultSetList, 238
STSStringList, 319
STSTypeList, 364
STSValueList, 403
STSValues, 407
countEdges
STSGraph, 128
countNodes
STSGraph, 128
create:alias:
STSSparksee, 278
createArrayAttribute:name:dt:size:
STSGraph, 129
createAttribute:name:dt:kind:
STSGraph, 129
createAttributeWithDefault:name:dt:kind:defaultValue:
STSGraph, 130
createEdge:tail:head:
STSGraph, 130
createEdgeType:directed:neighbors:
STSGraph, 130
createEdgeWithAttributes:tailAttr:tailV:headAttr:headV:
STSGraph, 131
createNode:
STSGraph, 131
createNodeType:
STSGraph, 132
createObjects
STSSession, 251
createQuery:
STSSession, 251
createRestrictedEdgeType:tail:head:neighbors:
STSGraph, 132
createSession
STSDatabase, 64
createSessionArrayAttribute:dt:size:
STSGraph, 132
createSessionAttribute:dt:kind:
STSGraph, 133
createSessionAttributeWithDefault:dt:kind:defaultValue:
STSGraph, 133
createWithConfig:alias:config:
STSSparksee, 279

decrypt:
STSSparksee, 279
degree:etype:dir:
STSGraph, 134
difference:
STSObjects, 197
downloadLicense
STSSparkseeConfig, 292
drop:
STSGraph, 134
dropWithObjects:
STSGraph, 135
dumpData:
STSGraph, 135
dumpSchema:
STSDatabase, 64
dumpStorage:
STSGraph, 135

edges:tail:head:
STSGraph, 136
enableType:
STSDefaultExport, 72
STSExportManager, 119
enableTypeSwigExplicitSTSDefaultExport:
STSDefaultExport, 72
encrypt:
STSSparksee, 280
encryptedBackup:keyInHex:ivInHex:
STSGraph, 136
equals:
STSObjects, 197
STSValue, 378
excludeEdges:
STSCommunitiesSCD, 29
STSCommunityDetection, 34
STSConnectivity, 42

STSContext, 47
STSDisjointCommunityDetection, 91
STSRandomWalk, 228
STSShortestPath, 255
STSSinglePairShortestPath, 259
STSSinglePairShortestPathBFS, 265
STSSinglePairShortestPathDijkstra, 271
STSStrongConnectivity, 324
STSStrongConnectivityGabow, 329
STSTraversal, 337
STSTraversalBFS, 342
STSTraversalDFS, 348
STSWeakConnectivity, 413
STSWeakConnectivityDFS, 418
excludeNodes:
STSCommunitiesSCD, 30
STSCommunityDetection, 35
STSConnectivity, 42
STSContext, 47
STSDisjointCommunityDetection, 91
STSRandomWalk, 228
STSShortestPath, 256
STSSinglePairShortestPath, 260
STSSinglePairShortestPathBFS, 265
STSSinglePairShortestPathDijkstra, 271
STSStrongConnectivity, 325
STSStrongConnectivityGabow, 330
STSTraversal, 338
STSTraversalBFS, 343
STSTraversalDFS, 348
STSWeakConnectivity, 413
STSWeakConnectivityDFS, 418
execute:reiterable:
STSQuery, 219
exists:
STSObjects, 198
explode:etype:dir:
STSGraph, 137
explodeWithObjects:etype:dir:
STSGraph, 137
exportAll
STSDefaultExport, 72
STSExportManager, 119
exportAllSwigExplicitSTSDefaultExport
STSDefaultExport, 73
exportAllSwigExplicitSTSExportManager
STSDefaultExport, 73
STSExportManager, 120
exportGraph:type:em:
STSGraph, 137

fetch:
STSQueryStream, 222
findAttribute:name:
STSGraph, 138
findAttributes:
STSGraph, 138
findEdge:tail:head:
STSGraph, 139
findEdgeTypes
STSGraph, 139
findNodeTypes
STSGraph, 139
findObject:value:
STSGraph, 140
findOrCreateEdge:tail:head:
STSGraph, 140
findOrCreateObject:value:
STSGraph, 141
findType:
STSGraph, 141
findTypes
STSGraph, 141

generateSchemaScript:db:
STSScriptParser, 247
get:
STSResultSetList, 238
STSValueList, 403
get:def:
STSSparkseeProperties, 315
get:value:
STSValueArray, 391
getAESIV
STSSparkseeConfig, 292
getAESKey
STSSparkseeConfig, 293
getAlias
STSDatabase, 64
getAreNeighborsIndexed
STSType, 352
getArrayAttribute:attr:
STSGraph, 142
getArraySize
STSAttribute, 9
getAttribute:
STSGraph, 142
getAttributeIntervalCount:lower:includeLower:higher:includeHigher:
STSGraph, 142
getAttributeInValue:attr:value:
STSGraph, 143
getAttributes:
STSGraph, 143
getAttributeStatistics:basic:
STSGraph, 144
getAttributeText:attr:
STSGraph, 144
getAttributeWithOid:attr:
STSGraph, 145
getAvailableMem
STSPlatformStatistics, 216
getAvgLengthString
STSAttributeStatistics, 18
getBoolean
STSValue, 378
STSValueArray, 392
getBoolean:def:
STSSparkseeProperties, 315

getCache
STSDatabaseStatistics, 67
getCacheMaxSize
STSDatabase, 64
STSSparkseeConfig, 293
getCacheStatisticsEnabled
STSSparkseeConfig, 293
getCacheStatisticsFile
STSSparkseeConfig, 293
getCacheStatisticsSnapshotTime
STSSparkseeConfig, 294
getCallStackDump
STSSparkseeConfig, 294
getChecksumEnabled
STSSparkseeConfig, 294
getClientId
STSSparkseeConfig, 295
getColorRed:green:blue:alpha:
STSEdgeExport, 98
STSGraphExport, 160
STSNodeExport, 173
getColorRGB
STSEdgeExport, 99
STSGraphExport, 160
STSNodeExport, 174
getColumn:
STSResultSet, 234
getColumnDataType:
STSResultSet, 234
getColumnIndex:
STSResultSet, 234
getColumnName:
STSResultSet, 234
getColumnWithValue:value:
STSResultSet, 235
getCommunities
STSCommunitiesSCD, 30
STSDisjointCommunityDetection, 92
getCommunity:
STSDisjointCommunities, 86
getConnectedComponent:
STSConnectedComponents, 38
getConnectedComponents
STSConnectivity, 42
STSStrongConnectivity, 325
STSStrongConnectivityGabow, 330
STSWeakConnectivity, 414
STSWeakConnectivityDFS, 419
getCost
STSSinglePairShortestPath, 260
STSSinglePairShortestPathBFS, 266
STSSinglePairShortestPathDijkstra, 271
getCount
STSAttribute, 9
STSConnectedComponents, 38
STSDisjointCommunities, 86
STSTypeExporterEvent, 360
getCurrentDepth
STSRandomWalk, 228
STSTraversal, 338
STSTraversalBFS, 343
STSTraversalDFS, 349
getCurrentValue:
STSAttributeListIterator, 16
getData
STSDatabaseStatistics, 67
getDataType
STSAttribute, 10
STSValue, 379
getDefaultValue
STSAttribute, 10
getDistinct
STSAttributeStatistics, 18
getDouble
STSValue, 379
STSValueArray, 392
getDoubleRange:size:
STSValueArray, 392
getDownloadStatus
STSSparkseeConfig, 295
getEdge
STSEdgeData, 95
getEdge:edgeExport:
STSDefaultExport, 73
STSExportManager, 120
getEdgeData:
STSGraph, 145
getEdgePeer:node:
STSGraph, 145
getEdgeSwigExplicitSTSDefaultExport:edgeExport:
STSDefaultExport, 74
getEdgeType:edgeExport:
STSDefaultExport, 74
STSExportManager, 120
getEdgeTypeSwigExplicitSTSDefaultExport:edgeExport:
STSDefaultExport, 75
getEncryptionEnabled
STSSparkseeConfig, 295
getExtentPages
STSSparkseeConfig, 295
getExtentSize
STSSparkseeConfig, 296
getFontSize
STSEdgeExport, 99
STSNodeExport, 174
getGraph
STSSession, 251
getGraph:
STSDefaultExport, 75
STSExportManager, 121
getGraphSwigExplicitSTSDefaultExport:
STSDefaultExport, 76
getHead
STSEdgeData, 95
getHeight
STSNodeExport, 174

getHighAvailabilityCoordinators
STSSparkseeConfig, 296
getHighAvailabilityEnabled
STSSparkseeConfig, 296
getHighAvailabilityIP
STSSparkseeConfig, 296
getHighAvailabilityMasterHistory
STSSparkseeConfig, 297
getHighAvailabilitySynchronization
STSSparkseeConfig, 297
getId
STSAttribute, 10
STSType, 352
getImage
STSNodeExport, 175
getInMemAllocSize
STSSparkseeConfig, 297
getInMemoryPoolCapacity
STSSession, 252
STSSparksee, 280
getInteger
STSValue, 379
STSValueArray, 393
getInteger:def:
STSSparkseeProperties, 316
getIntegerRange:size:
STSValueArray, 393
getIsDirected
STSType, 353
getIsRestricted
STSType, 353
getJSON:
STSResultSet, 235
getKey
STSKeyValue, 167
getKind
STSAttribute, 10
getLabel
STSEdgeExport, 99
STSGraphExport, 160
STSNodeExport, 175
getLabelColorRed:green:blue:alpha:
STSEdgeExport, 99
STSNodeExport, 175
getLabelColorRGB
STSEdgeExport, 100
STSNodeExport, 176
getLicense
STSSparkseeConfig, 297
getLicenseId
STSSparkseeConfig, 298
getLicensePreDownloadDays
STSSparkseeConfig, 298
getLogFile
STSSparkseeConfig, 298
getLogLevel
STSSparkseeConfig, 298
getLong
STSValue, 380
STSValueArray, 394
getMax
STSAttributeStatistics, 19
getMaxLengthString
STSAttributeStatistics, 19
getMean
STSAttributeStatistics, 19
getMedian
STSAttributeStatistics, 20
getMin
STSAttributeStatistics, 20
getMinLengthString
STSAttributeStatistics, 20
getMode
STSAttributeStatistics, 21
getModeCount
STSAttributeStatistics, 21
getName
STSAttribute, 11
STSType, 353
getNode:nodeExport:
STSDefaultExport, 76
STSExportManager, 121
getNodes:
STSConnectedComponents, 38
STSDisjointCommunities, 87
getNodeSwigExplicitSTSDefaultExport:nodeExport:
STSDefaultExport, 77
getNodeType:nodeExport:
STSDefaultExport, 77
STSExportManager, 122
getNodeTypeSwigExplicitSTSDefaultExport:nodeExport:
STSDefaultExport, 78
getNull
STSAttributeStatistics, 21
getNumColumns
STSResultSet, 235
getNumCPUs
STSPlatformStatistics, 216
getNumObjects
STSType, 353
getObjectType
STSType, 354
getObjectType:
STSGraph, 146
getOid
STSValue, 380
STSValueArray, 394
getPath
STSDatabase, 65
getPathAsEdges
STSSinglePairShortestPath, 260
STSSinglePairShortestPathBFS, 266
STSSinglePairShortestPathDijkstra, 272
getPathAsNodes
STSSinglePairShortestPath, 261
STSSinglePairShortestPathBFS, 266

STSSinglePairShortestPathDijkstra, 272
getPluginsDir
STSSparkseeConfig, 299
getPoolFrameSize
STSSparkseeConfig, 299
getPoolPartitions
STSSparkseeConfig, 299
getPoolPersistentMaxSize
STSSparkseeConfig, 299
getPoolPersistentMinSize
STSSparkseeConfig, 300
getPoolTemporaryMaxSize
STSSparkseeConfig, 300
getPoolTemporaryMinSize
STSSparkseeConfig, 300
getQueryStream:session:
STSQueryStreamManager, 224
getRead
STSDatabaseStatistics, 68
getRealTime
STSPlatformStatistics, 217
getRecoveryCacheMaxSize
STSSparkseeConfig, 300
getRecoveryCheckpointTime
STSSparkseeConfig, 301
getRecoveryEnabled
STSSparkseeConfig, 301
getRecoveryLogFile
STSSparkseeConfig, 301
getRestrictedFrom
STSType, 354
getRestrictedTo
STSType, 354
getRollbackEnabled
STSSparkseeConfig, 301
getRow
STSCSVReader, 53
STSRowReader, 242
getSession
STSGraph, 146
getSessions
STSDatabaseStatistics, 68
getShape
STSNodeExport, 176
getSize
STSAttribute, 11
getSize:
STSConnectedComponents, 39
STSDisjointCommunities, 87
getSparkseeConfigFile
STSSparkseeConfig, 302
getStatistics:
STSDatabase, 65
STSPlatform, 215
getString
STSValue, 380
getSystemTime
STSPlatformStatistics, 217
getTail
STSEdgeData, 95
getTemp
STSDatabaseStatistics, 68
getTimestamp
STSValue, 381
STSValueArray, 394
getTimestampAsNSDate
STSValue, 381
getTimeUnit:def:
STSSparkseeProperties, 316
getTitle
STSEdgeExport, 100
STSNodeExport, 176
getTmpEnabled
STSSparkseeConfig, 302
getTmpFolder
STSSparkseeConfig, 302
getTotal
STSAttributeStatistics, 22
STSTypeExporterEvent, 360
getTotalMem
STSPlatformStatistics, 217
getType:
STSGraph, 146
getTypeId
STSAttribute, 11
STSTypeExporterEvent, 360
getUserTime
STSPlatformStatistics, 217
getValue
STSKeyValue, 167
getValues:
STSGraph, 147
getVariance
STSAttributeStatistics, 22
getWidth
STSEdgeExport, 100
STSNodeExport, 177
getWrite
STSDatabaseStatistics, 68

hasNext
STSAttributeListIterator, 16
STSBooleanListIterator, 26
STSInt32ListIterator, 165
STSKeyValues, 168
STSObjectsIterator, 202
STSOidListIterator, 207
STSRandomWalk, 229
STSResultSetListIterator, 240
STSStringListIterator, 321
STSTraversal, 338
STSTraversalBFS, 343
STSTraversalDFS, 349
STSTypeListIterator, 366
STSValueListIterator, 405
STSValuesIterator, 409
heads:

STSGraph, 147

includeEdges:
STSCommunitiesSCD, 30
STSCommunityDetection, 35
STSDisjointCommunityDetection, 92
includeNodes:
STSCommunitiesSCD, 31
STSCommunityDetection, 35
STSDisjointCommunityDetection, 92
indexAttribute:kind:
STSGraph, 147
indexNeighbors:neighbors:
STSGraph, 148
init
STSAttributeList, 14
STSBooleanList, 24
STSInt32List, 163
STSOidList, 204
STSResultSetList, 238
STSSparkseeConfig, 302
STSStringList, 319
STSTypeList, 364
STSValue, 381
STSValueList, 404
initWithAppend:
STSTextStream, 333
initWithConfig:
STSSparksee, 280
initWithNumInvalidOIDs:
STSOidList, 205
initWithPath:
STSSparkseeConfig, 303
initWithPath:clientId:licenseId:
STSSparkseeConfig, 303
initWithRowReader:graph:type:attrs:attrsPos:
STSNodeTypeLoader, 187
initWithRowWriter:graph:type:attrs:
STSNodeTypeExporter, 183
initWithRowWriter:graph:type:attrs:hPos:tPos:hAttr:tAttr:
STSEdgeTypeExporter, 105
initWithS:materializedattribute:
STSConnectedComponents, 39
initWithSession:
STSCommunitiesSCD, 31
STSPageRank, 210
STSStrongConnectivityGabow, 330
STSWeakConnectivityDFS, 419
initWithSession:materializedattribute:
STSDisjointCommunities, 87
initWithSession:node:
STSContext, 48
STSRandomWalk, 229
STSTraversalBFS, 344
STSTraversalDFS, 349
initWithSession:src:dst:
STSSinglePairShortestPathBFS, 267
STSSinglePairShortestPathDijkstra, 272
initWithValue:
STSValue, 382
intersection:
STSObjects, 198
isArrayAttribute
STSAttribute, 11
isFit
STSNodeExport, 177
isLast
STSTypeExporterEvent, 360
isNull
STSTextStream, 334
STSValue, 382
isSessionAttribute
STSAttribute, 12
iterator
STSObjects, 198
iterator:
STSValues, 407
iteratorFromElement:
STSObjects, 199
iteratorFromIndex:
STSObjects, 199

load:
STSSparkseeProperties, 317
loadEdges:fileName:edgeType:tailNodeType:headNodeType:tail:head:sep
STSCSVLoader, 49
loadNodes:fileName:nodeType:separator:header:columns:attrNames:data
STSCSVLoader, 50

makeNull
STSValueArray, 395

neighbors:etype:dir:
STSGraph, 148
neighborsWithObjects:etype:dir:
STSGraph, 148
next
STSAttributeListIterator, 16
STSBooleanListIterator, 26
STSInt32ListIterator, 165
STSKeyValues, 169
STSObjectsIterator, 202
STSOidListIterator, 207
STSRandomWalk, 229
STSResultSet, 236
STSResultSetListIterator, 240
STSStringListIterator, 321
STSTraversal, 339
STSTraversalBFS, 344
STSTraversalDFS, 350
STSTypeListIterator, 366
STSValueListIterator, 405
STSValuesIterator, 409
nextEdge
STSDefaultExport, 78
STSExportManager, 122
nextEdgeSwigExplicitSTSDefaultExport
STSDefaultExport, 78

nextEdgeSwigExplicitSTSExportManager
STSDefaultExport, 79
STSExportManager, 122
nextKeyValue:
STSKeyValues, 169
nextNode
STSDefaultExport, 79
STSExportManager, 123
nextNodeSwigExplicitSTSDefaultExport
STSDefaultExport, 79
nextNodeSwigExplicitSTSExportManager
STSDefaultExport, 80
STSExportManager, 123
notifyEvent:
STSTypeExporterListener, 362
STSTypeLoaderListener, 375

open:
STSCSVReader, 54
STSCSVWriter, 59
open:readOnly:
STSSparksee, 281
openWithConfig:readOnly:config:
STSSparksee, 281

parse:execute:localeStr:
STSScriptParser, 248
preCommit
STSSession, 252
prepare:
STSDefaultExport, 80
STSExportManager, 123
STSQueryStream, 222
prepareSwigExplicitSTSDefaultExport:
STSDefaultExport, 80

read:
STSCSVReader, 54
STSRowReader, 242
readString:
STSTextStream, 334
rebuildDatabase:config:minimizeDisk:
STSSparksee, 282
redoPrecommitted:
STSDatabase, 65
registerListener:
STSEdgeTypeExporter, 106
STSEdgeTypeLoader, 111
STSNodeTypeExporter, 183
STSNodeTypeLoader, 187
STSTypeExporter, 356
STSTypeLoader, 368
releaseQueryStream:
STSQueryStreamManager, 224
remove:
STSObjects, 199
removeAttribute:
STSGraph, 149
removeChecksums:
STSSparksee, 283
removeType:
STSGraph, 149
renameAttribute:newName:
STSGraph, 149
renameType:newName:
STSGraph, 150
renameTypeWithName:newName:
STSGraph, 150
reset
STSCSVReader, 55
STSRowReader, 243
reset:
STSRandomWalk, 230
resizeInMemoryPool:
STSSparksee, 283
restore:backupFile:
STSSparksee, 284
restoreEncryptedBackup:backupFile:keyInHex:ivInHex:
STSSparksee, 284
restoreEncryptedBackupWithConfig:backupFile:config:keyInHex:ivInHex:
STSSparksee, 285
restoreWithConfig:backupFile:config:
STSSparksee, 285
run
STSCommunityDetection, 36
STSConnectivity, 42
STSDisjointCommunityDetection, 93
STSEdgeTypeExporter, 106
STSEdgeTypeLoader, 111
STSNodeTypeExporter, 183
STSNodeTypeLoader, 187
STSPageRank, 211
STSShortestPath, 256
STSSinglePairShortestPath, 261
STSStrongConnectivity, 325
STSTypeExporter, 356
STSTypeLoader, 368
STSWeakConnectivity, 414
runNPhases:
STSEdgeTypeLoader, 112
STSNodeTypeLoader, 188
STSTypeLoader, 369
runTwoPhases
STSEdgeTypeLoader, 112
STSNodeTypeLoader, 188
STSTypeLoader, 369

sample:samples:
STSObjects, 200
save
STSSparkseeConfig, 303
saveAll
STSSparkseeConfig, 304
selectWithAttrValue:cond:value:
STSGraph, 150
selectWithAttrValueRestriction:cond:value:restriction:
STSGraph, 151
selectWithAttrValues:cond:lower:higher:

STSGraph, 151
selectWithAttrValuesRestriction:cond:lower:higher:restriction:
STSGraph, 152
selectWithType:
STSGraph, 152
set:
STSValue, 382
STSValueArray, 395
set:oid:
STSOidList, 205
setAESEncryptionEnabled:ivInHex:
STSSparkseeConfig, 304
setArrayAttribute:attr:value:
STSGraph, 153
setArrayAttributeVoid:attr:value:
STSGraph, 153
setAsDirected:
STSEdgeExport, 101
setAt:value:
STSValueArray, 395
setAttribute:attr:value:
STSGraph, 153
setAttributeDefaultValue:value:
STSGraph, 154
setAttributePositions:
STSEdgeTypeLoader, 113
STSNodeTypeLoader, 189
STSTypeLoader, 370
setAttributes:
STSEdgeTypeExporter, 106
STSEdgeTypeLoader, 113
STSNodeTypeExporter, 183
STSNodeTypeLoader, 189
STSTypeExporter, 357
STSTypeLoader, 370
setAttributeText:attr:tstream:
STSGraph, 154
setAutoQuotes:
STSCSVWriter, 59
setBoolean:
STSValue, 382
STSValueArray, 396
setBooleanRange:values:
STSValueArray, 396
setBooleanVoid:
STSValue, 383
setCacheMaxSize:
STSDatabase, 66
STSSparkseeConfig, 304
setCacheStatisticsEnabled:
STSSparkseeConfig, 305
setCacheStatisticsFile:
STSSparkseeConfig, 305
setCacheStatisticsSnapshotTime:
STSSparkseeConfig, 305
setCallStackDump:
STSSparkseeConfig, 305
setChecksumEnabled:
STSSparkseeConfig, 306
setClientId:
STSSparkseeConfig, 306
setColorRed:green:blue:alpha:
STSEdgeExport, 101
STSGraphExport, 160
STSNodeExport, 177
setColorRGB:
STSEdgeExport, 101
STSGraphExport, 161
STSNodeExport, 178
setDamping:
STSPageRank, 211
setDefaultWeight:
STSPageRank, 211
STSRandomWalk, 230
setDouble:
STSValue, 383
STSValueArray, 397
setDoubleRange:values:
STSValueArray, 397
setDoubleVoid:
STSValue, 383
setDynamic:value:
STSQuery, 219
setDynamicEdgeCostCallback:
STSSinglePairShortestPathDijkstra, 273
setEdgeWeightAttributeType:
STSPageRank, 212
STSRandomWalk, 230
setErrorLog:
STSScriptParser, 248
setExtentPages:
STSSparkseeConfig, 306
setExtentSize:
STSSparkseeConfig, 307
setFit:
STSNodeExport, 178
setFontSize:
STSEdgeExport, 102
STSNodeExport, 178
setForcedQuotes:
STSCSVWriter, 60
setFrequency:
STSEdgeTypeExporter, 107
STSEdgeTypeLoader, 113
STSNodeTypeExporter, 184
STSNodeTypeLoader, 189
STSTypeExporter, 357
STSTypeLoader, 370
setGraph:
STSEdgeTypeExporter, 107
STSEdgeTypeLoader, 113
STSNodeTypeExporter, 184
STSNodeTypeLoader, 189
STSTypeExporter, 357
STSTypeLoader, 371
setHeadAttribute:

STSEdgeTypeExporter, 107
STSEdgeTypeLoader, 114
setHeader:
STSEdgeTypeExporter, 107
STSNodeTypeExporter, 184
STSTypeExporter, 358
setHeadMEP:
STSEdgeTypeLoader, 114
setHeadPosition:
STSEdgeTypeExporter, 108
STSEdgeTypeLoader, 114
setHeight:
STSNodeExport, 179
setHI:
STSSparkseeProperties, 317
setHighAvailabilityCoordinators:
STSSparkseeConfig, 307
setHighAvailabilityEnabled:
STSSparkseeConfig, 307
setHighAvailabilityIP:
STSSparkseeConfig, 307
setHighAvailabilityMasterHistory:
STSSparkseeConfig, 308
setHighAvailabilitySynchronization:
STSSparkseeConfig, 308
setImage:
STSNodeExport, 179
setInitialPageRankValue:
STSPageRank, 212
setInMemAllocSize:
STSSparkseeConfig, 308
setInOutParameter:
STSRandomWalk, 231
setInteger:
STSValue, 384
STSValueArray, 398
setIntegerRange:values:
STSValueArray, 398
setIntegerVoid:
STSValue, 384
setLabel:
STSEdgeExport, 102
STSGraphExport, 161
STSNodeExport, 179
setLabelColorRed:green:blue:alpha:
STSEdgeExport, 102
STSNodeExport, 179
setLabelColorRGB:
STSEdgeExport, 103
STSNodeExport, 180
setLicense:
STSSparkseeConfig, 308
setLicenseId:
STSSparkseeConfig, 309
setLicensePreDownloadDays:
STSSparkseeConfig, 309
setLocale:
STSCSVReader, 55
STSCSVWriter, 60
STSEdgeTypeLoader, 115
STSNodeTypeLoader, 190
STSTypeLoader, 371
setLogError:
STSEdgeTypeLoader, 115
STSNodeTypeLoader, 190
STSTypeLoader, 371
setLogFile:
STSSparkseeConfig, 309
setLogLevel:
STSSparkseeConfig, 310
setLogOff
STSEdgeTypeLoader, 115
STSNodeTypeLoader, 190
STSTypeLoader, 372
setLong:
STSValue, 384
STSValueArray, 398
setLongRange:values:
STSValueArray, 399
setLongVoid:
STSValue, 385
setLookAhead:
STSCommunitiesSCD, 31
setMaterializedAttribute:
STSCommunitiesSCD, 32
STSConnectivity, 43
STSDisjointCommunityDetection, 93
STSStrongConnectivity, 326
STSStrongConnectivityGabow, 331
STSWeakConnectivity, 414
STSWeakConnectivityDFS, 419
setMaxArrayElements:
STSDefaultExport, 81
setMaximumHops:
STSRandomWalk, 231
STSShortestPath, 256
STSSinglePairShortestPath, 261
STSSinglePairShortestPathBFS, 267
STSSinglePairShortestPathDijkstra, 273
STSTraversal, 339
STSTraversalBFS, 344
STSTraversalDFS, 350
setMaximumHops:include:
STSContext, 48
setMaxTextCharacters:
STSDefaultExport, 81
setMultilines:
STSCSVReader, 55
setNodeTypeImage:image:
STSDefaultExport, 81
setNodeTypeImageAttribute:imageAttr:
STSDefaultExport, 82
setNodeTypeShape:shape:
STSDefaultExport, 82
setNull
STSValue, 385

setNumIterations:
STSPageRank, 212
setNumLines:
STSCSVReader, 56
setOid:
STSValue, 385
STSValueArray, 399
setOidRange:values:
STSValueArray, 400
setOidVoid:
STSValue, 385
setOutputAttributeType:
STSPageRank, 213
setOutputLog:
STSScriptParser, 248
setPluginsDir:
STSSparkseeConfig, 310
setPoolFrameSize:
STSSparkseeConfig, 310
setPoolPartitions:
STSSparkseeConfig, 310
setPoolPersistentMaxSize:
STSSparkseeConfig, 311
setPoolPersistentMinSize:
STSSparkseeConfig, 311
setPoolTemporaryMaxSize:
STSSparkseeConfig, 311
setPoolTemporaryMinSize:
STSSparkseeConfig, 311
setQuotes:
STSCSVReader, 56
STSCSVWriter, 60
setRecoveryCacheMaxSize:
STSSparkseeConfig, 312
setRecoveryCheckpointTime:
STSSparkseeConfig, 312
setRecoveryEnabled:
STSSparkseeConfig, 312
setRecoveryLogFile:
STSSparkseeConfig, 312
setReturnParameter:
STSRandomWalk, 231
setRollbackEnabled:
STSSparkseeConfig, 313
setRowReader:
STSEdgeTypeLoader, 115
STSNodeTypeLoader, 190
STSTypeLoader, 372
setRowWriter:
STSEdgeTypeExporter, 108
STSNodeTypeExporter, 184
STSTypeExporter, 358
setSeed:
STSRandomWalk, 232
setSeparator:
STSCSVReader, 56
STSCSVWriter, 61
setShape:
STSNodeExport, 180
setSparkseeConfigFile:
STSSparkseeConfig, 313
setStartingNode:
STSPageRank, 213
setStartLine:
STSCSVReader, 57
setStream:handler:
STSQuery, 220
setString:
STSValue, 386
setStringVoid:
STSValue, 386
setTailAttribute:
STSEdgeTypeExporter, 108
STSEdgeTypeLoader, 116
setTailMEP:
STSEdgeTypeLoader, 116
setTailPosition:
STSEdgeTypeExporter, 108
STSEdgeTypeLoader, 116
setTimestamp:
STSValue, 386
STSValueArray, 400
setTimestampFormat:
STSEdgeTypeLoader, 117
STSNodeTypeLoader, 191
STSTypeLoader, 372
setTimestampRange:values:
STSValueArray, 401
setTimestampVoid:
STSValue, 387
setTimestampVoidWithYear:month:day:hour:minutes:seconds:millisecs:
STSValue, 387
setTimestampWithNSDate:
STSValue, 387
setTimestampWithYear:month:day:hour:minutes:seconds:milliseconds:
STSValue, 388
setTitle:
STSEdgeExport, 103
STSNodeExport, 180
setTitleWithAttributes:
STSDefaultExport, 82
setTmpEnabled:
STSSparkseeConfig, 313
setTmpFolder:
STSSparkseeConfig, 313
setTolerance:
STSPageRank, 213
setType:
STSEdgeTypeExporter, 109
STSEdgeTypeLoader, 117
STSNodeTypeExporter, 185
STSNodeTypeLoader, 191
STSTypeExporter, 358
STSTypeLoader, 372
setTypeColor:red:green:blue:alpha:
STSDefaultExport, 83

setTypeColorRGB:color:
STSDefaultExport, 83
setTypeLabelAttribute:labelAttr:
STSDefaultExport, 83
setUnweightedEdgeCost:
STSSinglePairShortestPathDijkstra, 273
setVoid:
STSValue, 388
setWidth:
STSEdgeExport, 103
STSNodeExport, 181
size
STSValueArray, 401
start:
STSQueryStream, 222
STSAttribute, 8
getArraySize, 9
getCount, 9
getDataType, 10
getDefaultValue, 10
getId, 10
getKind, 10
getName, 11
getSize, 11
getTypeId, 11
isArrayAttribute, 11
isSessionAttribute, 12
STSAttributeList, 12
add:, 14
count, 14
init, 14
STSAttributeListIterator, 15
getCurrentValue:, 16
hasNext, 16
next, 16
STSAttributeStatistics, 17
getAvgLengthString, 18
getDistinct, 18
getMax, 19
getMaxLengthString, 19
getMean, 19
getMedian, 20
getMin, 20
getMinLengthString, 20
getMode, 21
getModeCount, 21
getNull, 21
getTotal, 22
getVariance, 22
STSBooleanList, 23
add:, 24
count, 24
init, 24
STSBooleanListIterator, 25
hasNext, 26
next, 26
STSCommunitiesSCD, 27
addAllEdgeTypes, 29
addEdgeType:, 29
addNodeType:, 29
close, 29
excludeEdges:, 29
excludeNodes:, 30
getCommunities, 30
includeEdges:, 30
includeNodes:, 31
initWithSession:, 31
setLookAhead:, 31
setMaterializedAttribute:, 32
STSCommunityDetection, 32
addNodeType:, 34
close, 34
excludeEdges:, 34
excludeNodes:, 35
includeEdges:, 35
includeNodes:, 35
run, 36
STSConnectedComponents, 36
close, 38
getConnectedComponent:, 38
getCount, 38
getNodes:, 38
getSize:, 39
initWithS:materializedattribute:, 39
STSConnectivity, 40
addNodeType:, 41
close, 41
excludeEdges:, 42
excludeNodes:, 42
getConnectedComponents, 42
run, 42
setMaterializedAttribute:, 43
STSContext, 43
addAllEdgeTypes:, 45
addEdgeType:d:, 45
addNodeType:, 46
close, 46
compute, 46
computeWithArguments:node:nodeTypes:edgeTypes:dir:maxhops:inc
46
excludeEdges:, 47
excludeNodes:, 47
initWithSession:node:, 48
setMaximumHops:include:, 48
STSCSVLoader, 49
loadEdges:fileName:edgeType:tailNodeType:headNodeType:tail:head
49
loadNodes:fileName:nodeType:separator:header:columns:attrNames
50
STSCSVReader, 51
close, 53
getRow, 53
open:, 54
read:, 54
reset, 55
setLocale:, 55

setMultilines:, 55
setNumLines:, 56
setQuotes:, 56
setSeparator:, 56
setStartLine:, 57
STSCSVWriter, 57
close, 59
open:, 59
setAutoQuotes:, 59
setForcedQuotes:, 60
setLocale:, 60
setQuotes:, 60
setSeparator:, 61
write:, 61
STSDatabase, 62
addQueryStreamManager:, 63
close, 63
createSession, 64
dumpSchema:, 64
getAlias, 64
getCacheMaxSize, 64
getPath, 65
getStatistics:, 65
redoPrecommitted:, 65
setCacheMaxSize:, 66
STSDatabaseStatistics, 66
getCache, 67
getData, 67
getRead, 68
getSessions, 68
getTemp, 68
getWrite, 68
STSDefaultExport, 69
enableType:, 72
enableTypeSwigExplicitSTSDefaultExport:, 72
exportAll, 72
exportAllSwigExplicitSTSDefaultExport, 73
exportAllSwigExplicitSTSExportManager, 73
getEdge:edgeExport:, 73
getEdgeSwigExplicitSTSDefaultExport:edgeExport:, 74
getEdgeType:edgeExport:, 74
getEdgeTypeSwigExplicitSTSDefaultExport:edgeExport:, 75
getGraph:, 75
getGraphSwigExplicitSTSDefaultExport:, 76
getNode:nodeExport:, 76
getNodeSwigExplicitSTSDefaultExport:nodeExport:, 77
getNodeType:nodeExport:, 77
getNodeTypeSwigExplicitSTSDefaultExport:nodeExport:, 78
nextEdge, 78
nextEdgeSwigExplicitSTSDefaultExport, 78
nextEdgeSwigExplicitSTSExportManager, 79
nextNode, 79
nextNodeSwigExplicitSTSDefaultExport, 79
nextNodeSwigExplicitSTSExportManager, 80
prepare:, 80
prepareSwigExplicitSTSDefaultExport:, 80
setMaxArrayElements:, 81
setMaxTextCharacters:, 81
setNodeTypeImage:image:, 81
setNodeTypeImageAttribute:imageAttr:, 82
setNodeTypeShape:shape:, 82
setTitleWithAttributes:, 82
setTypeColor:red:green:blue:alpha:, 83
setTypeColorRGB:color:, 83
setTypeLabelAttribute:labelAttr:, 83
subgraphEdges:, 84
subgraphNodes:, 84
STSDisjointCommunities, 85
close, 86
getCommunity:, 86
getCount, 86
getNodes:, 87
getSize:, 87
initWithSession:materializedattribute:, 87
STSDisjointCommunityDetection, 88
addAllEdgeTypes, 90
addEdgeType:, 90
addNodeType:, 90
close, 91
excludeEdges:, 91
excludeNodes:, 91
getCommunities, 92
includeEdges:, 92
includeNodes:, 92
run, 93
setMaterializedAttribute:, 93
STSEdgeData, 94
getEdge, 95
getHead, 95
getTail, 95
STSEdgeExport, 96
asDirected, 98
copyDefaults:, 98
getColorRed:green:blue:alpha:, 98
getColorRGB, 99
getFontSize, 99
getLabel, 99
getLabelColorRed:green:blue:alpha:, 99
getLabelColorRGB, 100
getTitle, 100
getWidth, 100
setAsDirected:, 101
setColorRed:green:blue:alpha:, 101
setColorRGB:, 101
setFontSize:, 102
setLabel:, 102
setLabelColorRed:green:blue:alpha:, 102
setLabelColorRGB:, 103
setTitle:, 103
setWidth:, 103
STSEdgeTypeExporter, 104

initWithRowWriter:graph:type:attrs:hPos:tPos:hAttr:tAttr:, 105
registerListener:, 106
run, 106
setAttributes:, 106
setFrequency:, 107
setGraph:, 107
setHeadAttribute:, 107
setHeader:, 107
setHeadPosition:, 108
setRowWriter:, 108
setTailAttribute:, 108
setTailPosition:, 108
setType:, 109
STSEdgeTypeLoader, 109
registerListener:, 111
run, 111
runNPhases:, 112
runTwoPhases, 112
setAttributePositions:, 113
setAttributes:, 113
setFrequency:, 113
setGraph:, 113
setHeadAttribute:, 114
setHeadMEP:, 114
setHeadPosition:, 114
setLocale:, 115
setLogError:, 115
setLogOff, 115
setRowReader:, 115
setTailAttribute:, 116
setTailMEP:, 116
setTailPosition:, 116
setTimestampFormat:, 117
setType:, 117
STSExportManager, 117
close, 119
enableType:, 119
exportAll, 119
exportAllSwigExplicitSTSExportManager, 120
getEdge:edgeExport:, 120
getEdgeType:edgeExport:, 120
getGraph:, 121
getNode:nodeExport:, 121
getNodeType:nodeExport:, 122
nextEdge, 122
nextEdgeSwigExplicitSTSExportManager, 122
nextNode, 123
nextNodeSwigExplicitSTSExportManager, 123
prepare:, 123
STSGraph, 124
backup:, 128
countEdges, 128
countNodes, 128
createArrayAttribute:name:dt:size:, 129
createAttribute:name:dt:kind:, 129
createAttributeWithDefault:name:dt:kind:defaultValue:, 130
createEdge:tail:head:, 130
createEdgeType:directed:neighbors:, 130
createEdgeWithAttributes:tailAttr:tailV:headAttr:headV:, 131
createNode:, 131
createNodeType:, 132
createRestrictedEdgeType:tail:head:neighbors:, 132
createSessionArrayAttribute:dt:size:, 132
createSessionAttribute:dt:kind:, 133
createSessionAttributeWithDefault:dt:kind:defaultValue:, 133
degree:etype:dir:, 134
drop:, 134
dropWithObjects:, 135
dumpData:, 135
dumpStorage:, 135
edges:tail:head:, 136
encryptedBackup:keyInHex:ivInHex:, 136
explode:etype:dir:, 137
explodeWithObjects:etype:dir:, 137
exportGraph:type:em:, 137
findAttribute:name:, 138
findAttributes:, 138
findEdge:tail:head:, 139
findEdgeTypes, 139
findNodeTypes, 139
findObject:value:, 140
findOrCreateEdge:tail:head:, 140
findOrCreateObject:value:, 141
findType:, 141
findTypes, 141
getArrayAttribute:attr:, 142
getAttribute:, 142
getAttributeIntervalCount:lower:includeLower:higher:includeHigher:, 142
getAttributeInValue:attr:value:, 143
getAttributes:, 143
getAttributeStatistics:basic:, 144
getAttributeText:attr:, 144
getAttributeWithOid:attr:, 145
getEdgeData:, 145
getEdgePeer:node:, 145
getObjectType:, 146
getSession, 146
getType:, 146
getValues:, 147
heads:, 147
indexAttribute:kind:, 147
indexNeighbors:neighbors:, 148
neighbors:etype:dir:, 148
neighborsWithObjects:etype:dir:, 148
removeAttribute:, 149
removeType:, 149
renameAttribute:newName:, 149
renameType:newName:, 150
renameTypeWithName:newName:, 150
selectWithAttrValue:cond:value:, 150

selectWithAttrValueRestriction:cond:value:restriction:, 151
selectWithAttrValues:cond:lower:higher:, 151
selectWithAttrValuesRestriction:cond:lower:higher:restriction:, 152
selectWithType:, 152
setArrayAttribute:attr:value:, 153
setArrayAttributeVoid:attr:value:, 153
setAttribute:attr:value:, 153
setAttributeDefaultValue:value:, 154
setAttributeText:attr:tstream:, 154
tails:, 154
tailsAndHeads:tails:heads:, 155
topkWithAttr:order:k:, 155
topkWithAttrRestriction:order:k:restriction:, 155
topkWithAttrValue:operation:lower:order:k:, 156
topkWithAttrValueRestriction:operation:lower:order:k:restriction:, 156
topkWithAttrValues:operation:lower:higher:order:k:, 157
topkWithAttrValuesRestriction:operation:lower:higher:order:k:restriction:, 157
STSGraphExport, 158
getColorRed:green:blue:alpha:, 160
getColorRGB, 160
getLabel, 160
setColorRed:green:blue:alpha:, 160
setColorRGB:, 161
setLabel:, 161
STSInt32List, 162
add:, 163
count, 163
init, 163
STSInt32ListIterator, 164
hasNext, 165
next, 165
STSKeyValue, 166
getKey, 167
getValue, 167
STSKeyValues, 167
close, 168
hasNext, 168
next, 169
nextKeyValue:, 169
STSKOpt, 170
close, 171
STSNodeExport, 171
copyDefaults:, 173
getColorRed:green:blue:alpha:, 173
getColorRGB, 174
getFontSize, 174
getHeight, 174
getImage, 175
getLabel, 175
getLabelColorRed:green:blue:alpha:, 175
getLabelColorRGB, 176
getShape, 176
getTitle, 176
getWidth, 177
isFit, 177
setColorRed:green:blue:alpha:, 177
setColorRGB:, 178
setFit:, 178
setFontSize:, 178
setHeight:, 179
setImage:, 179
setLabel:, 179
setLabelColorRed:green:blue:alpha:, 179
setLabelColorRGB:, 180
setShape:, 180
setTitle:, 180
setWidth:, 181
STSNodeTypeExporter, 181
initWithRowWriter:graph:type:attrs:, 183
registerListener:, 183
run, 183
setAttributes:, 183
setFrequency:, 184
setGraph:, 184
setHeader:, 184
setRowWriter:, 184
setType:, 185
STSNodeTypeLoader, 185
initWithRowReader:graph:type:attrs:attrsPos:, 187
registerListener:, 187
run, 187
runNPhases:, 188
runTwoPhases, 188
setAttributePositions:, 189
setAttributes:, 189
setFrequency:, 189
setGraph:, 189
setLocale:, 190
setLogError:, 190
setLogOff, 190
setRowReader:, 190
setTimestampFormat:, 191
setType:, 191
STSObjects, 191
add:, 194
any, 194
clone, 194
cloneWithObjects:, 194
close, 195
combineDifference:objs2:, 195
combineIntersection:objs2:, 195
combineUnion:objs2:, 196
contains:, 196
count, 197
difference:, 197
equals:, 197
exists:, 198
intersection:, 198
iterator, 198
iteratorFromElement:, 199
iteratorFromIndex:, 199

remove:, 199
sample:samples:, 200
union:, 200
STSObjectsIterator, 201
close, 202
hasNext, 202
next, 202
STSOidList, 203
add:, 204
count, 204
init, 204
initWithNumInvalidOIDs:, 205
set:oid:, 205
STSOidListIterator, 206
hasNext, 207
next, 207
STSPageRank, 208
addAllEdgeTypes:, 209
addEdgeType:dir:, 209
addNodeType:, 210
close, 210
initWithSession:, 210
run, 211
setDamping:, 211
setDefaultWeight:, 211
setEdgeWeightAttributeType:, 212
setInitialPageRankValue:, 212
setNumIterations:, 212
setOutputAttributeType:, 213
setStartingNode:, 213
setTolerance:, 213
STSPlatform, 214
getStatistics:, 215
STSPlatformStatistics, 215
getAvailableMem, 216
getNumCPUs, 216
getRealTime, 217
getSystemTime, 217
getTotalMem, 217
getUserTime, 217
STSQuery, 218
execute:reiterable:, 219
setDynamic:value:, 219
setStream:handler:, 220
STSQueryStream, 220
fetch:, 222
prepare:, 222
start:, 222
STSQueryStreamManager, 223
getQueryStream:session:, 224
releaseQueryStream:, 224
STSRandomWalk, 225
addAllEdgeTypes:, 227
addEdgeType:dir:, 227
addNodeType:, 227
close, 227
excludeEdges:, 228
excludeNodes:, 228
getCurrentDepth, 228
hasNext, 229
initWithSession:node:, 229
next, 229
reset:, 230
setDefaultWeight:, 230
setEdgeWeightAttributeType:, 230
setInOutParameter:, 231
setMaximumHops:, 231
setReturnParameter:, 231
setSeed:, 232
STSResultSet, 232
getColumn:, 234
getColumnDataType:, 234
getColumnIndex:, 234
getColumnName:, 234
getColumnWithValue:value:, 235
getJSON:, 235
getNumColumns, 235
next, 236
STSResultSetList, 237
count, 238
get:, 238
init, 238
STSResultSetListIterator, 239
hasNext, 240
next, 240
STSRowReader, 241
close, 242
getRow, 242
read:, 242
reset, 243
STSRowWriter, 244
close, 245
write:, 245
STSScriptParser, 246
generateSchemaScript:db:, 247
parse:execute:localeStr:, 248
setErrorLog:, 248
setOutputLog:, 248
STSSession, 249
close, 251
createObjects, 251
createQuery:, 251
getGraph, 251
getInMemoryPoolCapacity, 252
preCommit, 252
STSShortestPath, 253
addAllEdgeTypes:, 254
addEdgeType:dir:, 254
addNodeType:, 255
close, 255
excludeEdges:, 255
excludeNodes:, 256
run, 256
setMaximumHops:, 256
STSSinglePairShortestPath, 257
addAllEdgeTypes:, 259

addEdgeType:dir:, 259
addNodeType:, 259
close, 259
excludeEdges:, 259
excludeNodes:, 260
getCost, 260
getPathAsEdges, 260
getPathAsNodes, 261
run, 261
setMaximumHops:, 261
STSSinglePairShortestPathBFS, 262
addAllEdgeTypes:, 264
addEdgeType:dir:, 264
addNodeType:, 264
checkOnlyExistence, 265
close, 265
excludeEdges:, 265
excludeNodes:, 265
getCost, 266
getPathAsEdges, 266
getPathAsNodes, 266
initWithSession:src:dst:, 267
setMaximumHops:, 267
STSSinglePairShortestPathDijkstra, 268
addAllEdgeTypes:, 270
addEdgeType:dir:, 270
addNodeType:, 270
addWeightedEdgeType:dir:attr:, 270
close, 271
excludeEdges:, 271
excludeNodes:, 271
getCost, 271
getPathAsEdges, 272
getPathAsNodes, 272
initWithSession:src:dst:, 272
setDynamicEdgeCostCallback:, 273
setMaximumHops:, 273
setUnweightedEdgeCost:, 273
STSSinglePairShortestPathDijkstraDynamicCost, 274
calculateEdgeCost:sourceCost:sourceLevel:targetNode:edgeType:, 275
STSSparksee, 276
addChecksums:, 278
create:alias:, 278
createWithConfig:alias:config:, 279
decrypt:, 279
encrypt:, 280
getInMemoryPoolCapacity, 280
initWithConfig:, 280
open:readOnly:, 281
openWithConfig:readOnly:config:, 281
rebuildDatabase:config:minimizeDisk:, 282
removeChecksums:, 283
resizeInMemoryPool:, 283
restore:backupFile:, 284
restoreEncryptedBackup:backupFile:keyInHex:ivInHex:, 284
restoreEncryptedBackupWithConfig:backupFile:config:keyInHex:ivInH 285
restoreWithConfig:backupFile:config:, 285
verifyChecksums:, 286
STSSparkseeConfig, 287
downloadLicense, 292
getAESIV, 292
getAESKey, 293
getCacheMaxSize, 293
getCacheStatisticsEnabled, 293
getCacheStatisticsFile, 293
getCacheStatisticsSnapshotTime, 294
getCallStackDump, 294
getChecksumEnabled, 294
getClientId, 295
getDownloadStatus, 295
getEncryptionEnabled, 295
getExtentPages, 295
getExtentSize, 296
getHighAvailabilityCoordinators, 296
getHighAvailabilityEnabled, 296
getHighAvailabilityIP, 296
getHighAvailabilityMasterHistory, 297
getHighAvailabilitySynchronization, 297
getInMemAllocSize, 297
getLicense, 297
getLicenseId, 298
getLicensePreDownloadDays, 298
getLogFile, 298
getLogLevel, 298
getPluginsDir, 299
getPoolFrameSize, 299
getPoolPartitions, 299
getPoolPersistentMaxSize, 299
getPoolPersistentMinSize, 300
getPoolTemporaryMaxSize, 300
getPoolTemporaryMinSize, 300
getRecoveryCacheMaxSize, 300
getRecoveryCheckpointTime, 301
getRecoveryEnabled, 301
getRecoveryLogFile, 301
getRollbackEnabled, 301
getSparkseeConfigFile, 302
getTmpEnabled, 302
getTmpFolder, 302
init, 302
initWithPath:, 303
initWithPath:clientId:licenseId:, 303
save, 303
saveAll, 304
setAESEncryptionEnabled:ivInHex:, 304
setCacheMaxSize:, 304
setCacheStatisticsEnabled:, 305
setCacheStatisticsFile:, 305
setCacheStatisticsSnapshotTime:, 305
setCallStackDump:, 305
setChecksumEnabled:, 306
setClientId:, 306

setExtentPages:, 306
setExtentSize:, 307
setHighAvailabilityCoordinators:, 307
setHighAvailabilityEnabled:, 307
setHighAvailabilityIP:, 307
setHighAvailabilityMasterHistory:, 308
setHighAvailabilitySynchronization:, 308
setInMemAllocSize:, 308
setLicense:, 308
setLicenseId:, 309
setLicensePreDownloadDays:, 309
setLogFile:, 309
setLogLevel:, 310
setPluginsDir:, 310
setPoolFrameSize:, 310
setPoolPartitions:, 310
setPoolPersistentMaxSize:, 311
setPoolPersistentMinSize:, 311
setPoolTemporaryMaxSize:, 311
setPoolTemporaryMinSize:, 311
setRecoveryCacheMaxSize:, 312
setRecoveryCheckpointTime:, 312
setRecoveryEnabled:, 312
setRecoveryLogFile:, 312
setRollbackEnabled:, 313
setSparkseeConfigFile:, 313
setTmpEnabled:, 313
setTmpFolder:, 313
STSSparkseeProperties, 314
get:def:, 315
getBoolean:def:, 315
getInteger:def:, 316
getTimeUnit:def:, 316
load:, 317
setHI:, 317
STSStringList, 318
add:, 319
count, 319
init, 319
STSStringListIterator, 320
hasNext, 321
next, 321
STSStrongConnectivity, 322
addAllEdgeTypes:, 324
addEdgeType:dir:, 324
addNodeType:, 324
close, 324
excludeEdges:, 324
excludeNodes:, 325
getConnectedComponents, 325
run, 325
setMaterializedAttribute:, 326
STSStrongConnectivityGabow, 326
addAllEdgeTypes:, 329
addEdgeType:dir:, 329
addNodeType:, 329
close, 329
excludeEdges:, 329
excludeNodes:, 330
getConnectedComponents, 330
initWithSession:, 330
setMaterializedAttribute:, 331
STSTextStream, 331
close, 333
initWithAppend:, 333
isNull, 334
readString:, 334
writeString:, 334
STSTraversal, 335
addAllEdgeTypes:, 337
addEdgeType:dir:, 337
addNodeType:, 337
close, 337
excludeEdges:, 337
excludeNodes:, 338
getCurrentDepth, 338
hasNext, 338
next, 339
setMaximumHops:, 339
STSTraversalBFS, 340
addAllEdgeTypes:, 341
addEdgeType:dir:, 341
addNodeType:, 342
close, 342
excludeEdges:, 342
excludeNodes:, 343
getCurrentDepth, 343
hasNext, 343
initWithSession:node:, 344
next, 344
setMaximumHops:, 344
STSTraversalDFS, 345
addAllEdgeTypes:, 347
addEdgeType:dir:, 347
addNodeType:, 347
close, 348
excludeEdges:, 348
excludeNodes:, 348
getCurrentDepth, 349
hasNext, 349
initWithSession:node:, 349
next, 350
setMaximumHops:, 350
STSType, 351
getAreNeighborsIndexed, 352
getId, 352
getIsDirected, 353
getIsRestricted, 353
getName, 353
getNumObjects, 353
getObjectType, 354
getRestrictedFrom, 354
getRestrictedTo, 354
STSTypeExporter, 355
registerListener:, 356
run, 356

setAttributes:, 357
setFrequency:, 357
setGraph:, 357
setHeader:, 358
setRowWriter:, 358
setType:, 358
STSTypeExporterEvent, 359
getCount, 360
getTotal, 360
getTypeId, 360
isLast, 360
STSTypeExporterListener, 361
notifyEvent:, 362
STSTypeList, 362
add:, 364
count, 364
init, 364
STSTypeListIterator, 365
hasNext, 366
next, 366
STSTypeLoader, 367
registerListener:, 368
run, 368
runNPhases:, 369
runTwoPhases, 369
setAttributePositions:, 370
setAttributes:, 370
setFrequency:, 370
setGraph:, 371
setLocale:, 371
setLogError:, 371
setLogOff, 372
setRowReader:, 372
setTimestampFormat:, 372
setType:, 372
STSTypeLoaderEvent, 373
STSTypeLoaderListener, 374
notifyEvent:, 375
STSValue, 375
compare:, 378
equals:, 378
getBoolean, 378
getDataType, 379
getDouble, 379
getInteger, 379
getLong, 380
getOid, 380
getString, 380
getTimestamp, 381
getTimestampAsNSDate, 381
init, 381
initWithValue:, 382
isNull, 382
set:, 382
setBoolean:, 382
setBooleanVoid:, 383
setDouble:, 383
setDoubleVoid:, 383
setInteger:, 384
setIntegerVoid:, 384
setLong:, 384
setLongVoid:, 385
setNull, 385
setOid:, 385
setOidVoid:, 385
setString:, 386
setStringVoid:, 386
setTimestamp:, 386
setTimestampVoid:, 387
setTimestampVoidWithYear:month:day:hour:minutes:seconds:millise
387
setTimestampWithNSDate:, 387
setTimestampWithYear:month:day:hour:minutes:seconds:millisecond
388
setVoid:, 388
STSValueArray, 389
close, 391
get:value:, 391
getBoolean, 392
getDouble, 392
getDoubleRange:size:, 392
getInteger, 393
getIntegerRange:size:, 393
getLong, 394
getOid, 394
getTimestamp, 394
makeNull, 395
set:, 395
setAt:value:, 395
setBoolean:, 396
setBooleanRange:values:, 396
setDouble:, 397
setDoubleRange:values:, 397
setInteger:, 398
setIntegerRange:values:, 398
setLong:, 398
setLongRange:values:, 399
setOid:, 399
setOidRange:values:, 400
setTimestamp:, 400
setTimestampRange:values:, 401
size, 401
STSValueList, 402
add:, 403
count, 403
get:, 403
init, 404
STSValueListIterator, 404
hasNext, 405
next, 405
STSValues, 406
close, 407
count, 407
iterator:, 407
STSValuesIterator, 408
close, 409

hasNext, 409
next, 409
STSWeakConnectivity, 410
addAllEdgeTypes, 412
addEdgeType:, 412
addNodeType:, 412
close, 413
excludeEdges:, 413
excludeNodes:, 413
getConnectedComponents, 414
run, 414
setMaterializedAttribute:, 414
STSWeakConnectivityDFS, 415
addAllEdgeTypes, 417
addEdgeType:, 417
addNodeType:, 417
close, 418
excludeEdges:, 418
excludeNodes:, 418
getConnectedComponents, 419
initWithSession:, 419
setMaterializedAttribute:, 419
subgraphEdges:
STSDefaultExport, 84
subgraphNodes:
STSDefaultExport, 84
SWIGTYPE_p_p_sparksee__gdb__QueryStream, 420
SWIGTYPE_p_wchar_t, 421

tails:
STSGraph, 154
tailsAndHeads:tails:heads:
STSGraph, 155
topkWithAttr:order:k:
STSGraph, 155
topkWithAttrRestriction:order:k:restriction:
STSGraph, 155
topkWithAttrValue:operation:lower:order:k:
STSGraph, 156
topkWithAttrValueRestriction:operation:lower:order:k:restriction:
STSGraph, 156
topkWithAttrValues:operation:lower:higher:order:k:
STSGraph, 157
topkWithAttrValuesRestriction:operation:lower:higher:order:k:restriction:
STSGraph, 157

union:
STSObjects, 200

verifyChecksums:
STSSparksee, 286

write:
STSCSVWriter, 61
STSRowWriter, 245
writeString:
STSTextStream, 334