

SparkseePython  
6.0.2

Generated by Doxygen 1.8.11

## Contents

<b>1</b>	<b>Namespace Index</b>	<b>1</b>
1.1	Packages . . . . .	1
<b>2</b>	<b>Hierarchical Index</b>	<b>1</b>
2.1	Class Hierarchy . . . . .	1
<b>3</b>	<b>Class Index</b>	<b>4</b>
3.1	Class List . . . . .	4
<b>4</b>	<b>Namespace Documentation</b>	<b>9</b>
4.1	sparksee Namespace Reference . . . . .	9
4.1.1	Detailed Description . . . . .	13
4.1.2	Function Documentation . . . . .	13
<b>5</b>	<b>Class Documentation</b>	<b>14</b>
5.1	sparksee.Attribute Class Reference . . . . .	14
5.1.1	Detailed Description . . . . .	15
5.1.2	Member Function Documentation . . . . .	15
5.2	sparksee.AttributeKind Class Reference . . . . .	17
5.2.1	Detailed Description . . . . .	17
5.2.2	Member Data Documentation . . . . .	17
5.3	sparksee.AttributeList Class Reference . . . . .	17
5.3.1	Detailed Description . . . . .	18
5.3.2	Constructor & Destructor Documentation . . . . .	18
5.3.3	Member Function Documentation . . . . .	18
5.4	sparksee.AttributeListIterator Class Reference . . . . .	19
5.4.1	Detailed Description . . . . .	19
5.4.2	Member Function Documentation . . . . .	19
5.5	sparksee.AttributeStatistics Class Reference . . . . .	20
5.5.1	Detailed Description . . . . .	20
5.5.2	Member Function Documentation . . . . .	21

---

5.6	<a href="#">sparksee.BooleanList Class Reference</a>	23
5.6.1	<a href="#">Detailed Description</a>	24
5.6.2	<a href="#">Constructor &amp; Destructor Documentation</a>	24
5.6.3	<a href="#">Member Function Documentation</a>	24
5.7	<a href="#">sparksee.BooleanListIterator Class Reference</a>	25
5.7.1	<a href="#">Detailed Description</a>	25
5.7.2	<a href="#">Member Function Documentation</a>	25
5.8	<a href="#">sparksee.CommunitiesSCD Class Reference</a>	26
5.8.1	<a href="#">Detailed Description</a>	27
5.8.2	<a href="#">Constructor &amp; Destructor Documentation</a>	28
5.8.3	<a href="#">Member Function Documentation</a>	28
5.9	<a href="#">sparksee.CommunityDetection Class Reference</a>	31
5.9.1	<a href="#">Detailed Description</a>	32
5.9.2	<a href="#">Member Function Documentation</a>	33
5.10	<a href="#">sparksee.Condition Class Reference</a>	34
5.10.1	<a href="#">Detailed Description</a>	35
5.10.2	<a href="#">Member Data Documentation</a>	35
5.11	<a href="#">sparksee.ConnectedComponents Class Reference</a>	36
5.11.1	<a href="#">Detailed Description</a>	37
5.11.2	<a href="#">Constructor &amp; Destructor Documentation</a>	37
5.11.3	<a href="#">Member Function Documentation</a>	38
5.12	<a href="#">sparksee.Connectivity Class Reference</a>	39
5.12.1	<a href="#">Detailed Description</a>	40
5.12.2	<a href="#">Member Function Documentation</a>	40
5.13	<a href="#">sparksee.Context Class Reference</a>	42
5.13.1	<a href="#">Detailed Description</a>	43
5.13.2	<a href="#">Constructor &amp; Destructor Documentation</a>	43
5.13.3	<a href="#">Member Function Documentation</a>	43
5.14	<a href="#">sparksee.CSVReader Class Reference</a>	45
5.14.1	<a href="#">Detailed Description</a>	47

---

5.14.2	Member Function Documentation	47
5.15	sparksee.CSVWriter Class Reference	50
5.15.1	Detailed Description	51
5.15.2	Member Function Documentation	52
5.16	sparksee.Database Class Reference	53
5.16.1	Detailed Description	54
5.16.2	Member Function Documentation	54
5.17	sparksee.DatabaseStatistics Class Reference	56
5.17.1	Detailed Description	56
5.17.2	Member Function Documentation	57
5.18	sparksee.DataType Class Reference	58
5.18.1	Detailed Description	58
5.18.2	Member Data Documentation	58
5.19	sparksee.DefaultExport Class Reference	59
5.19.1	Detailed Description	60
5.19.2	Member Function Documentation	60
5.20	sparksee.DisjointCommunities Class Reference	62
5.20.1	Detailed Description	62
5.20.2	Constructor & Destructor Documentation	63
5.20.3	Member Function Documentation	63
5.21	sparksee.DisjointCommunityDetection Class Reference	64
5.21.1	Detailed Description	66
5.21.2	Member Function Documentation	66
5.22	sparksee.EdgeData Class Reference	69
5.22.1	Detailed Description	69
5.22.2	Member Function Documentation	69
5.23	sparksee.EdgeExport Class Reference	70
5.23.1	Detailed Description	71
5.23.2	Member Function Documentation	71
5.24	sparksee.EdgesDirection Class Reference	73

---

5.24.1 Detailed Description . . . . .	73
5.25 sparksee.EdgeTypeExporter Class Reference . . . . .	74
5.25.1 Detailed Description . . . . .	75
5.25.2 Constructor & Destructor Documentation . . . . .	75
5.25.3 Member Function Documentation . . . . .	75
5.26 sparksee.EdgeTypeLoader Class Reference . . . . .	78
5.26.1 Detailed Description . . . . .	79
5.26.2 Constructor & Destructor Documentation . . . . .	80
5.26.3 Member Function Documentation . . . . .	80
5.27 sparksee.ExportManager Class Reference . . . . .	84
5.27.1 Detailed Description . . . . .	85
5.27.2 Member Function Documentation . . . . .	85
5.28 sparksee.ExportType Class Reference . . . . .	87
5.28.1 Detailed Description . . . . .	88
5.28.2 Member Data Documentation . . . . .	88
5.29 sparksee.Graph Class Reference . . . . .	88
5.29.1 Detailed Description . . . . .	91
5.29.2 Member Function Documentation . . . . .	92
5.30 sparksee.GraphExport Class Reference . . . . .	115
5.30.1 Detailed Description . . . . .	115
5.30.2 Member Function Documentation . . . . .	115
5.31 sparksee.Int32List Class Reference . . . . .	116
5.31.1 Detailed Description . . . . .	116
5.31.2 Constructor & Destructor Documentation . . . . .	116
5.31.3 Member Function Documentation . . . . .	116
5.32 sparksee.Int32ListIterator Class Reference . . . . .	117
5.32.1 Detailed Description . . . . .	117
5.32.2 Member Function Documentation . . . . .	118
5.33 sparksee.KeyValue Class Reference . . . . .	118
5.33.1 Member Function Documentation . . . . .	118

---

5.34	<a href="#">sparksee.KeyValues Class Reference</a>	118
5.34.1	<a href="#">Detailed Description</a>	119
5.34.2	<a href="#">Member Function Documentation</a>	119
5.35	<a href="#">sparksee.KOpt Class Reference</a>	120
5.35.1	<a href="#">Detailed Description</a>	121
5.35.2	<a href="#">Constructor &amp; Destructor Documentation</a>	121
5.35.3	<a href="#">Member Function Documentation</a>	122
5.36	<a href="#">sparksee.LogLevel Class Reference</a>	123
5.36.1	<a href="#">Detailed Description</a>	124
5.36.2	<a href="#">Member Data Documentation</a>	124
5.37	<a href="#">sparksee.MissingEndpoint Class Reference</a>	125
5.37.1	<a href="#">Detailed Description</a>	125
5.38	<a href="#">sparksee.NodeExport Class Reference</a>	125
5.38.1	<a href="#">Detailed Description</a>	126
5.38.2	<a href="#">Member Function Documentation</a>	127
5.39	<a href="#">sparksee.NodeShape Class Reference</a>	130
5.39.1	<a href="#">Detailed Description</a>	130
5.40	<a href="#">sparksee.NodeTypeExporter Class Reference</a>	130
5.40.1	<a href="#">Detailed Description</a>	131
5.40.2	<a href="#">Constructor &amp; Destructor Documentation</a>	131
5.40.3	<a href="#">Member Function Documentation</a>	132
5.41	<a href="#">sparksee.NodeTypeLoader Class Reference</a>	133
5.41.1	<a href="#">Detailed Description</a>	135
5.41.2	<a href="#">Constructor &amp; Destructor Documentation</a>	135
5.41.3	<a href="#">Member Function Documentation</a>	135
5.42	<a href="#">sparksee.Objects Class Reference</a>	138
5.42.1	<a href="#">Detailed Description</a>	139
5.42.2	<a href="#">Member Function Documentation</a>	139
5.43	<a href="#">sparksee.ObjectsIterator Class Reference</a>	145
5.43.1	<a href="#">Detailed Description</a>	145

5.43.2	Member Function Documentation	145
5.44	sparksee.ObjectType Class Reference	146
5.44.1	Detailed Description	147
5.45	sparksee.OIDList Class Reference	147
5.45.1	Detailed Description	147
5.45.2	Constructor & Destructor Documentation	147
5.45.3	Member Function Documentation	148
5.46	sparksee.OIDListIterator Class Reference	149
5.46.1	Detailed Description	149
5.46.2	Member Function Documentation	149
5.47	sparksee.Order Class Reference	150
5.47.1	Detailed Description	150
5.48	sparksee.PageRank Class Reference	150
5.48.1	Detailed Description	151
5.48.2	Constructor & Destructor Documentation	151
5.48.3	Member Function Documentation	151
5.49	sparksee.Platform Class Reference	154
5.49.1	Detailed Description	155
5.49.2	Member Function Documentation	155
5.50	sparksee.PlatformStatistics Class Reference	155
5.50.1	Detailed Description	155
5.50.2	Member Function Documentation	156
5.51	sparksee.Query Class Reference	157
5.51.1	Detailed Description	157
5.51.2	Member Function Documentation	157
5.52	sparksee.QueryContext Class Reference	158
5.52.1	Detailed Description	159
5.53	sparksee.QueryLanguage Class Reference	159
5.53.1	Detailed Description	159
5.54	sparksee.QueryStream Class Reference	159

---

5.54.1	Detailed Description	159
5.54.2	Member Function Documentation	160
5.55	sparksee.RandomWalk Class Reference	160
5.55.1	Detailed Description	162
5.55.2	Constructor & Destructor Documentation	162
5.55.3	Member Function Documentation	162
5.56	sparksee.ResultSet Class Reference	166
5.56.1	Detailed Description	167
5.56.2	Member Function Documentation	167
5.57	sparksee.ResultSetList Class Reference	169
5.57.1	Detailed Description	170
5.57.2	Constructor & Destructor Documentation	170
5.57.3	Member Function Documentation	170
5.58	sparksee.ResultSetListIterator Class Reference	171
5.58.1	Detailed Description	171
5.58.2	Member Function Documentation	171
5.59	sparksee.RowReader Class Reference	172
5.59.1	Detailed Description	172
5.59.2	Member Function Documentation	172
5.60	sparksee.RowWriter Class Reference	174
5.60.1	Detailed Description	174
5.60.2	Member Function Documentation	174
5.61	sparksee.ScriptParser Class Reference	175
5.61.1	Detailed Description	175
5.61.2	Member Function Documentation	175
5.62	sparksee.Session Class Reference	177
5.62.1	Detailed Description	177
5.62.2	Member Function Documentation	178
5.63	sparksee.ShortestPath Class Reference	179
5.63.1	Detailed Description	180

---



---

5.63.2	Member Function Documentation	180
5.64	sparksee.SinglePairShortestPath Class Reference	182
5.64.1	Detailed Description	183
5.64.2	Member Function Documentation	183
5.65	sparksee.SinglePairShortestPathBFS Class Reference	186
5.65.1	Detailed Description	187
5.65.2	Constructor & Destructor Documentation	187
5.65.3	Member Function Documentation	188
5.66	sparksee.SinglePairShortestPathDijkstra Class Reference	191
5.66.1	Detailed Description	192
5.66.2	Constructor & Destructor Documentation	193
5.66.3	Member Function Documentation	193
5.67	sparksee.SinglePairShortestPathDijkstraDynamicCost Class Reference	195
5.67.1	Detailed Description	196
5.67.2	Member Function Documentation	196
5.68	sparksee.Sparksee Class Reference	196
5.68.1	Detailed Description	197
5.68.2	Constructor & Destructor Documentation	198
5.68.3	Member Function Documentation	198
5.69	sparksee.SparkseeConfig Class Reference	204
5.69.1	Detailed Description	208
5.69.2	Constructor & Destructor Documentation	209
5.69.3	Member Function Documentation	210
5.70	sparksee.SparkseeProperties Class Reference	224
5.70.1	Detailed Description	225
5.70.2	Member Function Documentation	225
5.71	sparksee.StringList Class Reference	227
5.71.1	Detailed Description	227
5.71.2	Constructor & Destructor Documentation	227
5.71.3	Member Function Documentation	227

5.72	<a href="#">sparksee.StringListIterator Class Reference</a>	228
5.72.1	<a href="#">Detailed Description</a>	228
5.72.2	<a href="#">Member Function Documentation</a>	229
5.73	<a href="#">sparksee.StrongConnectivity Class Reference</a>	229
5.73.1	<a href="#">Detailed Description</a>	230
5.73.2	<a href="#">Member Function Documentation</a>	231
5.74	<a href="#">sparksee.StrongConnectivityGabow Class Reference</a>	233
5.74.1	<a href="#">Detailed Description</a>	234
5.74.2	<a href="#">Constructor &amp; Destructor Documentation</a>	234
5.74.3	<a href="#">Member Function Documentation</a>	235
5.75	<a href="#">sparksee.TextStream Class Reference</a>	237
5.75.1	<a href="#">Detailed Description</a>	237
5.75.2	<a href="#">Constructor &amp; Destructor Documentation</a>	237
5.75.3	<a href="#">Member Function Documentation</a>	238
5.76	<a href="#">sparksee.Traversal Class Reference</a>	239
5.76.1	<a href="#">Detailed Description</a>	240
5.76.2	<a href="#">Member Function Documentation</a>	240
5.77	<a href="#">sparksee.TraversalBFS Class Reference</a>	242
5.77.1	<a href="#">Detailed Description</a>	244
5.77.2	<a href="#">Constructor &amp; Destructor Documentation</a>	244
5.77.3	<a href="#">Member Function Documentation</a>	244
5.78	<a href="#">sparksee.TraversalDFS Class Reference</a>	246
5.78.1	<a href="#">Detailed Description</a>	248
5.78.2	<a href="#">Constructor &amp; Destructor Documentation</a>	248
5.78.3	<a href="#">Member Function Documentation</a>	248
5.79	<a href="#">sparksee.Type Class Reference</a>	251
5.79.1	<a href="#">Detailed Description</a>	251
5.79.2	<a href="#">Member Function Documentation</a>	252
5.80	<a href="#">sparksee.TypeExporter Class Reference</a>	253
5.80.1	<a href="#">Detailed Description</a>	254

5.80.2	Member Function Documentation	254
5.81	sparksee.TypeExporterEvent Class Reference	256
5.81.1	Detailed Description	256
5.81.2	Member Function Documentation	256
5.82	sparksee.TypeExporterListener Class Reference	257
5.82.1	Detailed Description	257
5.82.2	Member Function Documentation	257
5.83	sparksee.TypeList Class Reference	258
5.83.1	Detailed Description	258
5.83.2	Constructor & Destructor Documentation	258
5.83.3	Member Function Documentation	258
5.84	sparksee.TypeListIterator Class Reference	259
5.84.1	Detailed Description	259
5.84.2	Member Function Documentation	260
5.85	sparksee.TypeLoader Class Reference	260
5.85.1	Detailed Description	261
5.85.2	Member Function Documentation	261
5.86	sparksee.TypeLoaderEvent Class Reference	264
5.86.1	Detailed Description	265
5.86.2	Member Function Documentation	265
5.87	sparksee.TypeLoaderListener Class Reference	266
5.87.1	Member Function Documentation	266
5.88	sparksee.Value Class Reference	267
5.88.1	Detailed Description	268
5.88.2	Constructor & Destructor Documentation	269
5.88.3	Member Function Documentation	269
5.89	sparksee.ValueArray Class Reference	276
5.89.1	Detailed Description	277
5.89.2	Member Function Documentation	277
5.90	sparksee.ValueList Class Reference	285

5.90.1	Detailed Description	285
5.90.2	Constructor & Destructor Documentation	286
5.90.3	Member Function Documentation	286
5.91	sparksee.ValueListIterator Class Reference	287
5.91.1	Detailed Description	287
5.91.2	Member Function Documentation	287
5.92	sparksee.Values Class Reference	288
5.92.1	Detailed Description	288
5.92.2	Member Function Documentation	288
5.93	sparksee.ValuesIterator Class Reference	289
5.93.1	Detailed Description	290
5.93.2	Member Function Documentation	290
5.94	sparksee.WeakConnectivity Class Reference	291
5.94.1	Detailed Description	292
5.94.2	Member Function Documentation	292
5.95	sparksee.WeakConnectivityDFS Class Reference	294
5.95.1	Detailed Description	296
5.95.2	Constructor & Destructor Documentation	296
5.95.3	Member Function Documentation	297

## 1 Namespace Index

### 1.1 Packages

Here are the packages with brief descriptions (if available):

<b>sparksee</b>	<b>Embedded Graph Database</b>	<b>9</b>
-----------------	--------------------------------	----------

## 2 Hierarchical Index

### 2.1 Class Hierarchy

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

<code>sparksee.Attribute</code>	14
<code>sparksee.AttributeKind</code>	17
<code>sparksee.AttributeList</code>	17
<code>sparksee.AttributeListIterator</code>	19
<code>sparksee.AttributeStatistics</code>	20
<code>sparksee.BooleanList</code>	23
<code>sparksee.BooleanListIterator</code>	25
<code>sparksee.CommunityDetection</code>	31
<code>sparksee.DisjointCommunityDetection</code>	64
<code>sparksee.CommunitiesSCD</code>	26
<code>sparksee.Condition</code>	34
<code>sparksee.ConnectedComponents</code>	36
<code>sparksee.Connectivity</code>	39
<code>sparksee.StrongConnectivity</code>	229
<code>sparksee.StrongConnectivityGabow</code>	233
<code>sparksee.WeakConnectivity</code>	291
<code>sparksee.WeakConnectivityDFS</code>	294
<code>sparksee.Context</code>	42
<code>sparksee.Database</code>	53
<code>sparksee.DatabaseStatistics</code>	56
<code>sparksee.DataType</code>	58
<code>sparksee.DisjointCommunities</code>	62
<code>sparksee.EdgeData</code>	69
<code>sparksee.EdgeExport</code>	70
<code>sparksee.EdgesDirection</code>	73
<code>sparksee.ExportManager</code>	84
<code>sparksee.DefaultExport</code>	59
<code>sparksee.ExportType</code>	87
<code>sparksee.Graph</code>	88
<code>sparksee.GraphExport</code>	115
<code>sparksee.Int32List</code>	116
<code>sparksee.Int32ListIterator</code>	117

<code>sparksee.KeyValue</code>	118
<code>sparksee.KeyValues</code>	118
<code>sparksee.KOpt</code>	120
<code>sparksee.LogLevel</code>	123
<code>sparksee.MissingEndpoint</code>	125
<code>sparksee.NodeExport</code>	125
<code>sparksee.NodeShape</code>	130
<code>sparksee.Objects</code>	138
<code>sparksee.ObjectsIterator</code>	145
<code>sparksee.ObjectType</code>	146
<code>sparksee.OIDList</code>	147
<code>sparksee.OIDListIterator</code>	149
<code>sparksee.Order</code>	150
<code>sparksee.PageRank</code>	150
<code>sparksee.Platform</code>	154
<code>sparksee.PlatformStatistics</code>	155
<code>sparksee.Query</code>	157
<code>sparksee.QueryContext</code>	158
<code>sparksee.QueryLanguage</code>	159
<code>sparksee.QueryStream</code>	159
<code>sparksee.ResultSet</code>	166
<code>sparksee.ResultSetList</code>	169
<code>sparksee.ResultSetListIterator</code>	171
<code>sparksee.RowReader</code>	172
<code>sparksee.CSVReader</code>	45
<code>sparksee.RowWriter</code>	174
<code>sparksee.CSVWriter</code>	50
<code>sparksee.ScriptParser</code>	175
<code>sparksee.Session</code>	177
<code>sparksee.ShortestPath</code>	179
<code>sparksee.SinglePairShortestPath</code>	182
<code>sparksee.SinglePairShortestPathBFS</code>	186

sparksee.SinglePairShortestPathDijkstra	191
sparksee.SinglePairShortestPathDijkstraDynamicCost	195
sparksee.Sparksee	196
sparksee.SparkseeConfig	204
sparksee.SparkseeProperties	224
sparksee.StringList	227
sparksee.StringListIterator	228
sparksee.TextStream	237
sparksee.Traversal	239
sparksee.RandomWalk	160
sparksee.TraversalBFS	242
sparksee.TraversalDFS	246
sparksee.Type	251
sparksee.TypeExporter	253
sparksee.EdgeTypeExporter	74
sparksee.NodeTypeExporter	130
sparksee.TypeExporterEvent	256
sparksee.TypeExporterListener	257
sparksee.TypeList	258
sparksee.TypeListIterator	259
sparksee.TypeLoader	260
sparksee.EdgeTypeLoader	78
sparksee.NodeTypeLoader	133
sparksee.TypeLoaderEvent	264
sparksee.TypeLoaderListener	266
sparksee.Value	267
sparksee.ValueArray	276
sparksee.ValueList	285
sparksee.ValueListIterator	287
sparksee.Values	288
sparksee.ValuesIterator	289

## 3 Class Index

### 3.1 Class List

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

<a href="#">sparksee.Attribute</a> Attribute data class	14
<a href="#">sparksee.AttributeKind</a> Attribute kind enumeration	17
<a href="#">sparksee.AttributeList</a> Sparksee attribute identifier list	17
<a href="#">sparksee.AttributeListIterator</a> AttributeList iterator class	19
<a href="#">sparksee.AttributeStatistics</a> Attribute statistics class	20
<a href="#">sparksee.BooleanList</a> Boolean list	23
<a href="#">sparksee.BooleanListIterator</a> BooleanList iterator class	25
<a href="#">sparksee.CommunitiesSCD</a> CommunitiesSCD class	26
<a href="#">sparksee.CommunityDetection</a> CommunityDetection class	31
<a href="#">sparksee.Condition</a> Condition operators enumeration	34
<a href="#">sparksee.ConnectedComponents</a> ConnectedComponents class	36
<a href="#">sparksee.Connectivity</a> Connectivity class	39
<a href="#">sparksee.Context</a> Context class	42
<a href="#">sparksee.CSVReader</a> CSVReader interface	45
<a href="#">sparksee.CSVWriter</a> CSVWriter interface	50
<a href="#">sparksee.Database</a> Database class	53
<a href="#">sparksee.DatabaseStatistics</a> Database statistics	56
<a href="#">sparksee.DataType</a> Data type (domain) enumeration	58



<a href="#">sparksee.DefaultExport</a> Default implementation for <a href="#">ExportManager</a> class	59
<a href="#">sparksee.DisjointCommunities</a> <a href="#">DisjointCommunities</a> class	62
<a href="#">sparksee.DisjointCommunityDetection</a> <a href="#">DisjointCommunityDetection</a> class	64
<a href="#">sparksee.EdgeData</a> Edge data class	69
<a href="#">sparksee.EdgeExport</a> Stores edge exporting values	70
<a href="#">sparksee.EdgesDirection</a> Edges direction enumeration	73
<a href="#">sparksee.EdgeTypeExporter</a> <a href="#">EdgeTypeExporter</a> class	74
<a href="#">sparksee.EdgeTypeLoader</a> <a href="#">EdgeTypeLoader</a> class	78
<a href="#">sparksee.ExportManager</a> Defines how to export a graph to an external format	84
<a href="#">sparksee.ExportType</a> Export type	87
<a href="#">sparksee.Graph</a> <a href="#">Graph</a> class	88
<a href="#">sparksee.GraphExport</a> Stores the graph exporting values	115
<a href="#">sparksee.Int32List</a> <a href="#">Sparksee</a> 32-bit signed integer list	116
<a href="#">sparksee.Int32ListIterator</a> <a href="#">Int32List</a> iterator class	117
<a href="#">sparksee.KeyValue</a>	118
<a href="#">sparksee.KeyValues</a> Value set class	118
<a href="#">sparksee.KOpt</a> <a href="#">KOpt</a> class	120
<a href="#">sparksee.LogLevel</a> Log level enumeration	123
<a href="#">sparksee.MissingEndpoint</a> The policy to follow whenever and edge endpoint is missing during a loading	125
<a href="#">sparksee.NodeExport</a> Stores the node exporting values	125
<a href="#">sparksee.NodeShape</a> Node shape	130

<a href="#">sparksee.NodeTypeExporter</a> NodeTypeExporter class	130
<a href="#">sparksee.NodeTypeLoader</a> NodeTypeLoader class	133
<a href="#">sparksee.Objects</a> Object identifier set class	138
<a href="#">sparksee.ObjectsIterator</a> ObjectsIterator class	145
<a href="#">sparksee.ObjectType</a> Object type enumeration	146
<a href="#">sparksee.OIDList</a> Sparksee object identifier list	147
<a href="#">sparksee.OIDListIterator</a> OIDList iterator class	149
<a href="#">sparksee.Order</a> Order enumeration	150
<a href="#">sparksee.PageRank</a> PageRank class	150
<a href="#">sparksee.Platform</a> Platform class	154
<a href="#">sparksee.PlatformStatistics</a> Platform data and statistics	155
<a href="#">sparksee.Query</a> Query class	157
<a href="#">sparksee.QueryContext</a> Query context interface	158
<a href="#">sparksee.QueryLanguage</a> The supported query languages	159
<a href="#">sparksee.QueryStream</a> Query stream interface	159
<a href="#">sparksee.RandomWalk</a> RandomWalk class	160
<a href="#">sparksee.ResultSet</a> ResultSet class	166
<a href="#">sparksee.ResultSetList</a> ResultSet list	169
<a href="#">sparksee.ResultSetListIterator</a> ResultSetList iterator class	171
<a href="#">sparksee.RowReader</a> RowReader interface	172
<a href="#">sparksee.RowWriter</a> RowWriter interface	174

<a href="#">sparksee.ScriptParser</a> ScriptParser	175
<a href="#">sparksee.Session</a> Session class	177
<a href="#">sparksee.ShortestPath</a> ShortestPath class	179
<a href="#">sparksee.SinglePairShortestPath</a> SinglePairShortestPath class	182
<a href="#">sparksee.SinglePairShortestPathBFS</a> SinglePairShortestPathBFS class	186
<a href="#">sparksee.SinglePairShortestPathDijkstra</a> SinglePairShortestPathDijkstra class	191
<a href="#">sparksee.SinglePairShortestPathDijkstraDynamicCost</a> Defines how to calculate an edge weight	195
<a href="#">sparksee.Sparksee</a> Sparksee class	196
<a href="#">sparksee.SparkseeConfig</a> Sparksee configuration class	204
<a href="#">sparksee.SparkseeProperties</a> Sparksee properties file	224
<a href="#">sparksee.StringList</a> String list	227
<a href="#">sparksee.StringListIterator</a> StringList iterator class	228
<a href="#">sparksee.StrongConnectivity</a> StrongConnectivity class	229
<a href="#">sparksee.StrongConnectivityGabow</a> This class can be used to solve the problem of finding strongly connected components in a directed graph	233
<a href="#">sparksee.TextStream</a> TextStream class	237
<a href="#">sparksee.Traversal</a> Traversal class	239
<a href="#">sparksee.TraversalBFS</a> Breadth-First Search implementation of <a href="#">Traversal</a>	242
<a href="#">sparksee.TraversalDFS</a> Depth-First Search (DFS) implementation of <a href="#">Traversal</a>	246
<a href="#">sparksee.Type</a> Type data class	251
<a href="#">sparksee.TypeExporter</a> Base <a href="#">TypeExporter</a> class	253

<a href="#">sparksee.TypeExporterEvent</a>	Provides information about the progress of an <a href="#">TypeExproter</a> instance	256
<a href="#">sparksee.TypeExporterListener</a>	Interface to be implemented to receive <a href="#">TypeExporterEvent</a> events from a <a href="#">TypeExporter</a>	257
<a href="#">sparksee.TypeList</a>	Sparksee type identifier list	258
<a href="#">sparksee.TypeListIterator</a>	<a href="#">TypeList</a> iterator class	259
<a href="#">sparksee.TypeLoader</a>	Base <a href="#">TypeLoader</a> class	260
<a href="#">sparksee.TypeLoaderEvent</a>	Provides information about the progress of a <a href="#">TypeLoader</a> instance	264
<a href="#">sparksee.TypeLoaderListener</a>		266
<a href="#">sparksee.Value</a>	<a href="#">Value</a> class	267
<a href="#">sparksee.ValueArray</a>	<a href="#">ValueArray</a> class	276
<a href="#">sparksee.ValueList</a>	<a href="#">Value</a> list	285
<a href="#">sparksee.ValueListIterator</a>	<a href="#">ValueList</a> iterator class	287
<a href="#">sparksee.Values</a>	<a href="#">Value</a> set class	288
<a href="#">sparksee.ValuesIterator</a>	<a href="#">Values</a> iterator class	289
<a href="#">sparksee.WeakConnectivity</a>	<a href="#">WeakConnectivity</a> class	291
<a href="#">sparksee.WeakConnectivityDFS</a>	<a href="#">WeakConnectivityDFS</a> class	294

## 4 Namespace Documentation

### 4.1 sparksee Namespace Reference

Embedded [Graph Database](#).

#### Classes

- class [Attribute](#)  
*Attribute* data class.
- class [AttributeKind](#)

- *Attribute* kind enumeration.
- class [AttributeList](#)
  - *Sparksee* attribute identifier list.
- class [AttributeListIterator](#)
  - *AttributeList* iterator class.
- class [AttributeStatistics](#)
  - *Attribute* statistics class.
- class [BooleanList](#)
  - *Boolean* list.
- class [BooleanListIterator](#)
  - *BooleanList* iterator class.
- class [CommunitiesSCD](#)
  - *CommunitiesSCD* class.
- class [CommunityDetection](#)
  - *CommunityDetection* class.
- class [Condition](#)
  - *Condition* operators enumeration.
- class [ConnectedComponents](#)
  - *ConnectedComponents* class.
- class [Connectivity](#)
  - *Connectivity* class.
- class [Context](#)
  - *Context* class.
- class [CSVReader](#)
  - *CSVReader* interface.
- class [CSVWriter](#)
  - *CSVWriter* interface.
- class [Database](#)
  - *Database* class.
- class [DatabaseStatistics](#)
  - *Database* statistics.
- class [DataType](#)
  - *Data type (domain)* enumeration.
- class [DefaultExport](#)
  - Default implementation for *ExportManager* class.
- class [DisjointCommunities](#)
  - *DisjointCommunities* class.
- class [DisjointCommunityDetection](#)
  - *DisjointCommunityDetection* class.
- class [EdgeData](#)
  - *Edge data* class.
- class [EdgeExport](#)
  - *Stores edge exporting values.*
- class [EdgesDirection](#)
  - *Edges direction* enumeration.
- class [EdgeTypeExporter](#)
  - *EdgeTypeExporter* class.
- class [EdgeTypeLoader](#)
  - *EdgeTypeLoader* class.
- class [ExportManager](#)
  - *Defines how to export a graph to an external format.*

- class [ExportType](#)  
*Export type.*
- class [Graph](#)  
*Graph class.*
- class [GraphExport](#)  
*Stores the graph exporting values.*
- class [Int32List](#)  
*Sparksee 32-bit signed integer list.*
- class [Int32ListIterator](#)  
*Int32List iterator class.*
- class [KeyValue](#)
- class [KeyValues](#)  
*Value set class.*
- class [KOpt](#)  
*KOpt class.*
- class [LogLevel](#)  
*Log level enumeration.*
- class [MissingEndpoint](#)  
*The policy to follow whenever and edge endpoint is missing during a loading.*
- class [NodeExport](#)  
*Stores the node exporting values.*
- class [NodeShape](#)  
*Node shape.*
- class [NodeTypeExporter](#)  
*NodeTypeExporter class.*
- class [NodeTypeLoader](#)  
*NodeTypeLoader class.*
- class [Objects](#)  
*Object identifier set class.*
- class [ObjectsIterator](#)  
*ObjectsIterator class.*
- class [ObjectType](#)  
*Object type enumeration.*
- class [OIDList](#)  
*Sparksee object identifier list.*
- class [OIDListIterator](#)  
*OIDList iterator class.*
- class [Order](#)  
*Order enumeration.*
- class [PageRank](#)  
*PageRank class.*
- class [Platform](#)  
*Platform class.*
- class [PlatformStatistics](#)  
*Platform data and statistics.*
- class [Query](#)  
*Query class.*
- class [QueryContext](#)  
*Query context interface.*
- class [QueryLanguage](#)  
*The supported query languages.*

- class [QueryStream](#)  
*Query stream interface.*
- class [RandomWalk](#)  
*RandomWalk class.*
- class [ResultSet](#)  
*ResultSet class.*
- class [ResultSetList](#)  
*ResultSet list.*
- class [ResultSetListIterator](#)  
*ResultSetList iterator class.*
- class [RowReader](#)  
*RowReader interface.*
- class [RowWriter](#)  
*RowWriter interface.*
- class [ScriptParser](#)  
*ScriptParser.*
- class [Session](#)  
*Session class.*
- class [ShortestPath](#)  
*ShortestPath class.*
- class [SinglePairShortestPath](#)  
*SinglePairShortestPath class.*
- class [SinglePairShortestPathBFS](#)  
*SinglePairShortestPathBFS class.*
- class [SinglePairShortestPathDijkstra](#)  
*SinglePairShortestPathDijkstra class.*
- class [SinglePairShortestPathDijkstraDynamicCost](#)  
*Defines how to calculate an edge weight.*
- class [Sparksee](#)  
*Sparksee class.*
- class [SparkseeConfig](#)  
*Sparksee configuration class.*
- class [SparkseeProperties](#)  
*Sparksee properties file.*
- class [StringList](#)  
*String list.*
- class [StringListIterator](#)  
*StringList iterator class.*
- class [StrongConnectivity](#)  
*StrongConnectivity class.*
- class [StrongConnectivityGabow](#)  
*This class can be used to solve the problem of finding strongly connected components in a directed graph.*
- class [TextStream](#)  
*TextStream class.*
- class [Traversal](#)  
*Traversal class.*
- class [TraversalBFS](#)  
*Breadth-First Search implementation of [Traversal](#).*
- class [TraversalDFS](#)  
*Depth-First Search (DFS) implementation of [Traversal](#).*
- class [Type](#)

- *Type data class.*
- class [TypeExporter](#)
  - *Base TypeExporter class.*
- class [TypeExporterEvent](#)
  - *Provides information about the progress of an TypeExproter instance.*
- class [TypeExporterListener](#)
  - *Interface to be implemented to receive TypeExporterEvent events from a TypeExporter.*
- class [TypeList](#)
  - *Sparksee type identifier list.*
- class [TypeListIterator](#)
  - *TypeList iterator class.*
- class [TypeLoader](#)
  - *Base TypeLoader class.*
- class [TypeLoaderEvent](#)
  - *Provides information about the progress of a TypeLoader instance.*
- class [TypeLoaderListener](#)
- class [Value](#)
  - *Value class.*
- class [ValueArray](#)
  - *ValueArray class.*
- class [ValueList](#)
  - *Value list.*
- class [ValueListIterator](#)
  - *ValueList iterator class.*
- class [Values](#)
  - *Value set class.*
- class [ValuesIterator](#)
  - *Values iterator class.*
- class [WeakConnectivity](#)
  - *WeakConnectivity class.*
- class [WeakConnectivityDFS](#)
  - *WeakConnectivityDFS class.*

## Functions

- def [load\\_nodes\\_csv](#) (self, graph, file\_name, node\_type, separator, header, columns, attr\_names, data\_types, attr\_kinds)
  - *Loads nodes from a CSV file.*
- def [load\\_edges\\_csv](#) (self, graph, file\_name, edge\_type, tail\_node\_type, head\_node\_type, tail, head, separator, directed, header, on\_missing\_tail, on\_missing\_head, columns, attr\_names, data\_types, attr\_kinds)
  - *Loads edges from a CSV file.*

### 4.1.1 Detailed Description

Embedded [Graph Database](#).

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>



## 4.1.2 Function Documentation

4.1.2.1 `def sparksee.load_edges_csv ( self, graph, file_name, edge_type, tail_node_type, head_node_type, tail, head, separator, directed, header, on_missing_tail, on_missing_head, columns, attr_names, data_types, attr_kinds )`

Loads edges from a CSV file.

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

### Parameters

<code>graph</code>	[in] The graph to load the edges into
<code>file_name</code>	[in] The name of the file
<code>edge_type</code>	null
<code>tail_node_type</code>	[in] The type of the tail nodes. If it does not exist and <code>onMissingTail</code> is set to "Create", it creates it. Otherwise, an exception is thrown
<code>head_node_type</code>	[in] The type of the head nodes. If it does not exist and <code>onMissingHead</code> is set to "Create", it creates it. Otherwise, an exception is thrown
<code>tail</code>	[in] The tail column index. Default: 0
<code>head</code>	[in] The head column index. Default: 1
<code>separator</code>	[in] The column separator. Default: ","
<code>directed</code>	[in] True if this edge is directed or not. False otherwise. Default: True
<code>header</code>	[in] True if the CSV contains a header. False otherwise. Default: True
<code>on_missing_tail</code>	[in] The policy to follow when a tail is missing. Default: "IsError"
<code>on_missing_head</code>	[in] The policy to follow when a head is missing. Default: "IsError"
<code>columns</code>	[in] The list of columns to load. <code>tail</code> and <code>head</code> columns must be in this list if this list is specified. If the list is empty, all columns are loaded. Default: empty
<code>attr_names</code>	[in] The attribute names. If this list is empty and <code>hasHeader</code> is set to true, the header values are used as attribute names. Default: empty
<code>data_types</code>	[in] The <code>dataTypes</code> 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
<code>attr_kinds</code>	[in] The <code>attributeKinds</code> of the attributes if these do not already exist. If this list is empty, the <code>attributeKind</code> of the created attributes are set to Basic. Default: empty

4.1.2.2 `def sparksee.load_nodes_csv ( self, graph, file_name, node_type, separator, header, columns, attr_names, data_types, attr_kinds )`

Loads nodes from a CSV file.

### Parameters

<code>graph</code>	[in] The graph to load the nodes into
<code>file_name</code>	[in] The name of the file
<code>node_type</code>	[in] The type of the node to load. If it does not exist, it creates it
<code>separator</code>	[in] The separator of the CSV file. Default: ","
<code>header</code>	[in] True if the CSV contains a header. False otherwise. Default: True
<code>columns</code>	[in] The list of columns to load. <code>tail</code> and <code>head</code> columns must be in this list if this list is specified. Default: all columns
<code>attr_names</code>	[in] The attribute names. If this list is empty and <code>hasHeader</code> is set to true, the header values are used as attribute names. Default: empty
<code>data_types</code>	[in] The <code>dataTypes</code> 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

## Parameters

<code>attr_kinds</code>	[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
-------------------------	---

## 5 Class Documentation

### 5.1 sparksee.Attribute Class Reference

[Attribute](#) data class.

#### Public Member Functions

- def [get\\_size](#) (self)  
*Gets the number of different values.*
- def [get\\_id](#) (self)  
*Gets the [Sparksee](#) attribute identifier.*
- def [get\\_type\\_id](#) (self)  
*Gets the [Sparksee](#) type identifier.*
- def [is\\_array\\_attribute](#) (self)  
*Check if it's an array attribute.*
- def [get\\_count](#) (self)  
*Gets the number of non-NULL values.*
- def [is\\_session\\_attribute](#) (self)  
*Check if it's a session attribute or a persistent one.*
- def [get\\_data\\_type](#) (self)  
*Gets the data type.*
- def [get\\_kind](#) (self)  
*Gets the attribute kind.*
- def [get\\_array\\_size](#) (self)  
*Gets the number of elements in the array.*
- def [get\\_name](#) (self)  
*Gets the unique attribute name.*

#### Static Public Attributes

- int [INVALID\\_ATTRIBUTE](#) = 0  
*Invalid attribute identifier constant.*

#### 5.1.1 Detailed Description

[Attribute](#) data class.

It contains information about an attribute.

#### Author

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

## 5.1.2 Member Function Documentation

### 5.1.2.1 `def sparksee.Attribute.get_array_size ( self )`

Gets the number of elements in the array.

#### Returns

The size of the array

### 5.1.2.2 `def sparksee.Attribute.get_count ( self )`

Gets the number of non-NULL values.

#### Returns

The number of non-NULL values.

### 5.1.2.3 `def sparksee.Attribute.get_data_type ( self )`

Gets the data type.

#### Returns

The [DataType](#).

### 5.1.2.4 `def sparksee.Attribute.get_id ( self )`

Gets the [Sparksee](#) attribute identifier.

#### Returns

The [Sparksee](#) attribute identifier.

### 5.1.2.5 `def sparksee.Attribute.get_kind ( self )`

Gets the attribute kind.

#### Returns

The [AttributeKind](#).

### 5.1.2.6 `def sparksee.Attribute.get_name ( self )`

Gets the unique attribute name.

#### Returns

The unique attribute name.

### 5.1.2.7 def sparksee.Attribute.get\_size ( self )

Gets the number of different values.

#### Returns

The number of different values.

### 5.1.2.8 def sparksee.Attribute.get\_type\_id ( self )

Gets the [Sparksee](#) type identifier.

#### Returns

The [Sparksee](#) type identifier.

### 5.1.2.9 def sparksee.Attribute.is\_array\_attribute ( self )

Check if it's an array attribute.

#### Returns

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

### 5.1.2.10 def sparksee.Attribute.is\_session\_attribute ( self )

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

#### Returns

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

## 5.2 sparksee.AttributeKind Class Reference

[Attribute](#) kind enumeration.

#### Static Public Attributes

- int [BASIC](#) = 0  
*Basic attribute (non indexed attribute).*
- int [INDEXED](#) = 1  
*Indexed attribute.*
- int [UNIQUE](#) = 2  
*Unique attribute (indexed + unique restriction).*

### 5.2.1 Detailed Description

[Attribute](#) kind enumeration.

It determines the indexing-capabilities of an attribute.

#### Author

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

### 5.2.2 Member Data Documentation

#### 5.2.2.1 `int sparksee.AttributeKind.UNIQUE = 2` [static]

Unique attribute (indexed + unique restriction).

Unique restriction sets two objects cannot have the same value for an attribute but NULL.

## 5.3 `sparksee.AttributeList` Class Reference

[Sparksee](#) attribute identifier list.

#### Public Member Functions

- def [add](#) (self, attr)  
*Adds a [Sparksee](#) attribute identifier at the end of the list.*
- def [clear](#) (self)  
*Clears the list.*
- def [\\_\\_iter\\_\\_](#) (self)  
*Gets a new [TypeListIterator](#).*
- def [iterator](#) (self)  
*Gets a new [AttributeListIterator](#).*
- def [\\_\\_init\\_\\_](#) (self)  
*Constructor.*
- def [count](#) (self)  
*Number of elements in the list.*

### 5.3.1 Detailed Description

[Sparksee](#) attribute identifier list.

It stores a [Sparksee](#) attribute identifier list.

Use [AttributeListIterator](#) to access all elements into this collection.

#### Author

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

### 5.3.2 Constructor & Destructor Documentation

#### 5.3.2.1 def sparksee.AttributeList.\_\_init\_\_( self )

Constructor.

This creates an empty list.

### 5.3.3 Member Function Documentation

#### 5.3.3.1 def sparksee.AttributeList.\_\_iter\_\_( self )

Gets a new [TypeListIterator](#).

Returns

[TypeListIterator](#) instance

#### 5.3.3.2 def sparksee.AttributeList.add( self, attr )

Adds a [Sparksee](#) attribute identifier at the end of the list.

Parameters

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
-------------	---

#### 5.3.3.3 def sparksee.AttributeList.count( self )

Number of elements in the list.

Returns

Number of elements in the list.

#### 5.3.3.4 def sparksee.AttributeList.iterator( self )

Gets a new [AttributeListIterator](#).

Returns

[AttributeListIterator](#) instance.

## 5.4 sparksee.AttributeListIterator Class Reference

[AttributeList](#) iterator class.

## Public Member Functions

- def `next` (self)  
*Moves to the next element.*
- def `has_next` (self)  
*Gets if there are more elements.*
- def `__next__` (self)  
*Used in `next()`*

### 5.4.1 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>

### 5.4.2 Member Function Documentation

#### 5.4.2.1 `def sparksee.AttributeListIterator.__next__( self )`

Used in `next()`

#### Returns

The next element

#### 5.4.2.2 `def sparksee.AttributeListIterator.has_next ( self )`

Gets if there are more elements.

#### Returns

TRUE if there are more elements, FALSE otherwise.

#### 5.4.2.3 `def sparksee.AttributeListIterator.next ( self )`

Moves to the next element.

#### Returns

The next element.

## 5.5 `sparksee.AttributeStatistics` Class Reference

`Attribute` statistics class.

### Public Member Functions

- def [get\\_min\\_length\\_string](#) (self)  
*Gets the minimum length.*
- def [get\\_mode\\_count](#) (self)  
*Gets the number of objects with a [Value](#) equal to the mode.*
- def [get\\_variance](#) (self)  
*Gets the variance.*
- def [get\\_mode](#) (self)  
*Gets the mode.*
- def [get\\_min](#) (self)  
*Gets the minimum existing value (BASIC statistics).*
- def [get\\_median](#) (self)  
*Gets the median.*
- def [get\\_total](#) (self)  
*Gets the number of objects with a non-NULL [Value](#) (BASIC statistic).*
- def [get\\_max\\_length\\_string](#) (self)  
*Gets the maximum length.*
- def [get\\_mean](#) (self)  
*Gets the mean or average.*
- def [get\\_null](#) (self)  
*Gets the number of objects NULL a [Value](#) (BASIC statistics).*
- def [get\\_distinct](#) (self)  
*Gets the number of distinct values (BASIC statistics).*
- def [get\\_avg\\_length\\_string](#) (self)  
*Gets the average length.*
- def [get\\_max](#) (self)  
*Gets the maximum existing value (BASIC statistics).*

#### 5.5.1 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>

#### 5.5.2 Member Function Documentation

##### 5.5.2.1 def sparksee.AttributeStatistics.get\_avg\_length\_string ( self )

Gets the average length.

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

#### Returns

The average length.



#### 5.5.2.2 `def sparksee.AttributeStatistics.get_distinct ( self )`

Gets the number of distinct values (BASIC statistics).

##### Returns

The number of distinct values.

#### 5.5.2.3 `def sparksee.AttributeStatistics.get_max ( self )`

Gets the maximum existing value (BASIC statistics).

##### Returns

The maximum existing value.

#### 5.5.2.4 `def sparksee.AttributeStatistics.get_max_length_string ( self )`

Gets the maximum length.

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

##### Returns

The maximum length.

#### 5.5.2.5 `def sparksee.AttributeStatistics.get_mean ( self )`

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.

#### 5.5.2.6 `def sparksee.AttributeStatistics.get_median ( self )`

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.

#### 5.5.2.7 def sparksee.AttributeStatistics.get\_min ( self )

Gets the minimum existing value (BASIC statistics).

##### Returns

The minimum existing value.

#### 5.5.2.8 def sparksee.AttributeStatistics.get\_min\_length\_string ( self )

Gets the minimum length.

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

##### Returns

The minimum length.

#### 5.5.2.9 def sparksee.AttributeStatistics.get\_mode ( self )

Gets the mode.

Mode: Most frequent [Value](#).

##### Returns

The mode.

#### 5.5.2.10 def sparksee.AttributeStatistics.get\_mode\_count ( self )

Gets the number of objects with a [Value](#) equal to the mode.

##### Returns

The number of objects with a [Value](#) equal to the mode.

#### 5.5.2.11 def sparksee.AttributeStatistics.get\_null ( self )

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

##### Returns

The number of objects NULL a [Value](#).

#### 5.5.2.12 def sparksee.AttributeStatistics.get\_total ( self )

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

##### Returns

The number of objects with a non-NULL [Value](#).

### 5.5.2.13 `def sparksee.AttributeStatistics.get_variance ( self )`

Gets the variance.

It is computed just for numerical attributes.

#### Returns

The variance.

## 5.6 `sparksee.BooleanList` Class Reference

Boolean list.

### Public Member Functions

- `def clear (self)`  
*Clears the list.*
- `def __init__ (self)`  
*Constructor.*
- `def __iter__ (self)`  
*Gets a new `TypeListIterator`.*
- `def iterator (self)`  
*Gets a new `BooleanListIterator`.*
- `def add (self, value)`  
*Adds a Boolean at the end of the list.*
- `def count (self)`  
*Number of elements in the list.*

### 5.6.1 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>

### 5.6.2 Constructor & Destructor Documentation

#### 5.6.2.1 `def sparksee.BooleanList.__init__ ( self )`

Constructor.

This creates an empty list.

### 5.6.3 Member Function Documentation

#### 5.6.3.1 def sparksee.BooleanList.\_\_iter\_\_( self )

Gets a new [TypeListIterator](#).

Returns

[TypeListIterator](#) instance

#### 5.6.3.2 def sparksee.BooleanList.add( self, value )

Adds a Boolean at the end of the list.

#### Parameters

<i>value</i>	[in] Boolean.
--------------	---------------

#### 5.6.3.3 def sparksee.BooleanList.count ( self )

Number of elements in the list.

#### Returns

Number of elements in the list.

#### 5.6.3.4 def sparksee.BooleanList.iterator ( self )

Gets a new [BooleanListIterator](#).

#### Returns

[BooleanListIterator](#) instance.

## 5.7 sparksee.BooleanListIterator Class Reference

[BooleanList](#) iterator class.

#### Public Member Functions

- def [next](#) (self)  
*Moves to the next element.*
- def [has\\_next](#) (self)  
*Gets if there are more elements.*
- def [\\_\\_next\\_\\_](#) (self)  
*Used in [next\(\)](#)*

#### 5.7.1 Detailed Description

[BooleanList](#) iterator class.

Iterator to traverse all the strings into a [BooleanList](#) instance.

#### Author

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

## 5.7.2 Member Function Documentation

## 5.7.2.1 def sparksee.BooleanListIterator.\_\_next\_\_( self )

Used in [next\(\)](#)

**Returns**

The next element

## 5.7.2.2 def sparksee.BooleanListIterator.has\_next( self )

Gets if there are more elements.

**Returns**

TRUE if there are more elements, FALSE otherwise.

## 5.7.2.3 def sparksee.BooleanListIterator.next( self )

Moves to the next element.

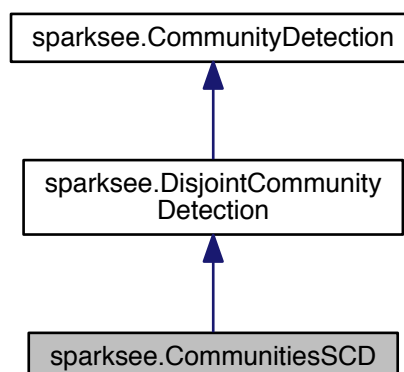
**Returns**

The next element.

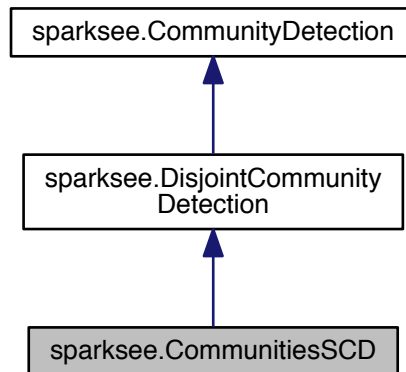
## 5.8 sparksee.CommunitiesSCD Class Reference

[CommunitiesSCD](#) class.

Inheritance diagram for sparksee.CommunitiesSCD:



Collaboration diagram for sparksee.CommunitiesSCD:



#### Public Member Functions

- def `__init__` (self, session)  
Creates a new instance of [CommunitiesSCD](#).
- def `exclude_nodes` (self, nodes)  
Set which nodes can't be used.
- def `add_all_edge_types` (self)  
Allows connectivity through all edge types of the graph.
- def `get_communities` (self)  
Returns the results generated by the execution of the algorithm.
- def `run` (self)  
Executes the algorithm.
- def `add_edge_type` (self, type)  
Allows connectivity through edges of the given type.
- def `exclude_edges` (self, edges)  
Set which edges can't be used.
- def `set_materialized_attribute` (self, attribute\_name)  
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.
- def `add_node_type` (self, type)  
Allows connectivity through nodes of the given type.
- def `include_nodes` (self, nodes)  
Set additional nodes that can be used.
- def `set_look_ahead` (self, lookahead)  
Sets the size of the lookahead iterations to look (5 by default).
- def `add_all_node_types` (self)  
Allows connectivity through all node types of the graph.
- def `include_edges` (self, edges)  
Set additional edges that can be used.
- def `close` (self)  
Closes the [CommunityDetection](#) instance.
- def `is_closed` (self)  
Gets if [CommunityDetection](#) has been closed or not.

### 5.8.1 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-Lluís 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>

### 5.8.2 Constructor & Destructor Documentation

#### 5.8.2.1 `def sparksee.CommunitiesSCD.__init__( self, 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

<code>session</code>	[in] <a href="#">Session</a> to get the graph from and calculate the communities
----------------------	--

### 5.8.3 Member Function Documentation

#### 5.8.3.1 `def sparksee.CommunitiesSCD.add_all_edge_types( self )`

Allows connectivity through all edge types of the graph.

The edges can be used in Any direction.

#### 5.8.3.2 `def sparksee.CommunitiesSCD.add_edge_type( self, type )`

Allows connectivity through edges of the given type.

The edges can be used in Any direction.



**Parameters**

<i>type</i>	[in] Edge type.
-------------	-----------------

5.8.3.3 `def sparksee.CommunitiesSCD.add_node_type ( self, type )`

Allows connectivity through nodes of the given type.

**Parameters**

<i>type</i>	null
-------------	------

5.8.3.4 `def sparksee.CommunityDetection.close ( self )` [inherited]

Closes the [CommunityDetection](#) instance.

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

5.8.3.5 `def sparksee.CommunitiesSCD.exclude_edges ( self, 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**

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

5.8.3.6 `def sparksee.CommunitiesSCD.exclude_nodes ( self, 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**

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

5.8.3.7 `def sparksee.CommunitiesSCD.get_communities ( self )`

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.

#### 5.8.3.8 def sparksee.CommunitiesSCD.include\_edges ( self, 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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

#### 5.8.3.9 def sparksee.CommunitiesSCD.include\_nodes ( self, 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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

#### 5.8.3.10 def sparksee.CommunityDetection.is\_closed ( self ) [inherited]

Gets if [CommunityDetection](#) has been closed or not.

##### See also

[close\(\)](#)

##### Returns

TRUE if the [CommunityDetection](#) instance has been closed, FALSE otherwise.

#### 5.8.3.11 def sparksee.CommunitiesSCD.set\_look\_ahead ( self, lookahead )

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

##### Parameters

<i>lookahead</i>	[in] Number of iterations. It must be positive or zero.
------------------	---

#### 5.8.3.12 def sparksee.CommunitiesSCD.set\_materialized\_attribute ( self, attribute\_name )

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.

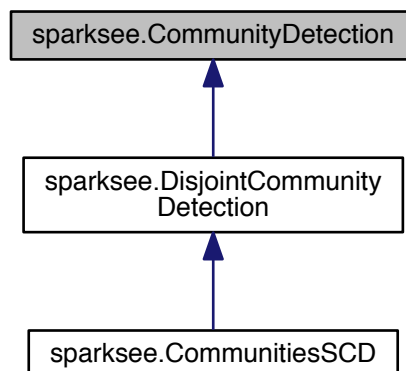
#### Parameters

<code>attribute_name</code>	[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.
-----------------------------	---

## 5.9 sparksee.CommunityDetection Class Reference

[CommunityDetection](#) class.

Inheritance diagram for `sparksee.CommunityDetection`:



#### Public Member Functions

- def [run](#) (self)  
*Runs the algorithm in order to find the connected components.*
- def [exclude\\_nodes](#) (self, nodes)  
*Set which nodes can't be used.*
- def [add\\_node\\_type](#) (self, type)  
*Allows connectivity through nodes of the given type.*
- def [exclude\\_edges](#) (self, edges)  
*Set which edges can't be used.*
- def [close](#) (self)  
*Closes the [CommunityDetection](#) instance.*
- def [include\\_nodes](#) (self, nodes)

- *Set additional nodes that can be used.*
- def `add_all_node_types` (self)  
*Allows connectivity through all node types of the graph.*
- def `include_edges` (self, edges)  
*Set additional edges that can be used.*
- def `is_closed` (self)  
*Gets if `CommunityDetection` has been closed or not.*

### 5.9.1 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>

### 5.9.2 Member Function Documentation

#### 5.9.2.1 def sparksee.CommunityDetection.add\_node\_type ( self, type )

Allows connectivity through nodes of the given type.

##### Parameters

<code>type</code>	null
-------------------	------

#### 5.9.2.2 def sparksee.CommunityDetection.close ( self )

Closes the `CommunityDetection` instance.

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

#### 5.9.2.3 def sparksee.CommunityDetection.exclude\_edges ( self, 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

<code>edges</code>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------------	---

#### 5.9.2.4 `def sparksee.CommunityDetection.exclude_nodes ( self, 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

<code>nodes</code>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------------	---

#### 5.9.2.5 `def sparksee.CommunityDetection.include_edges ( self, 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

<code>edges</code>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------------	---

#### 5.9.2.6 `def sparksee.CommunityDetection.include_nodes ( self, 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

<code>nodes</code>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------------	---

#### 5.9.2.7 `def sparksee.CommunityDetection.is_closed ( self )`

Gets if [CommunityDetection](#) has been closed or not.

##### See also

[close\(\)](#)

##### Returns

TRUE if the [CommunityDetection](#) instance has been closed, FALSE otherwise.

#### 5.9.2.8 `def sparksee.CommunityDetection.run ( self )`

Runs the algorithm in order to find the connected components.

This method can be called only once.

## 5.10 sparksee.Condition Class Reference

[Condition](#) operators enumeration.

### Static Public Attributes

- int [EQUAL](#) = 0  
*Equal condition (==).*
- int [GREATER\\_EQUAL](#) = 1  
*Greater or equal condition (>=).*
- int [GREATER\\_THAN](#) = 2  
*Greater than condition (>).*
- int [LESS\\_EQUAL](#) = 3  
*Less or equal condition (<=).*
- int [LESS\\_THAN](#) = 4  
*Less than condition (<).*
- int [NOT\\_EQUAL](#) = 5  
*Not equal condition (!=).*
- int [LIKE](#) = 6  
*Substring condition.*
- int [LIKE\\_NO\\_CASE](#) = 7  
*Substring (no case sensitive) condition.*
- int [BETWEEN](#) = 8  
*In range operator condition ([x,y]).*
- int [REG\\_EXP](#) = 9  
*Regular expression condition.*

### 5.10.1 Detailed Description

[Condition](#) operators enumeration.

It is mainly used in the attribute-based graph select operations.

#### Author

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

### 5.10.2 Member Data Documentation

#### 5.10.2.1 int sparksee.Condition.BETWEEN = 8 [static]

In range operator condition ([x,y]).

Null values cannot be used together with this condition.

#### 5.10.2.2 int sparksee.Condition.EQUAL = 0 [static]

Equal condition (==).

Null values can be used together with this condition to retrieve all objects having a null value for an attribute.

### 5.10.2.3 `int sparksee.Condition.GREATER_EQUAL = 1` [static]

Greater or equal condition ( $\geq$ ).

Null values cannot be used together with this condition.

### 5.10.2.4 `int sparksee.Condition.GREATER_THAN = 2` [static]

Greater than condition ( $>$ ).

Null values cannot be used together with this condition.

### 5.10.2.5 `int sparksee.Condition.LESS_EQUAL = 3` [static]

Less or equal condition ( $\leq$ ).

Null values cannot be used together with this condition.

### 5.10.2.6 `int sparksee.Condition.LESS_THAN = 4` [static]

Less than condition ( $<$ ).

Null values cannot be used together with this condition.

### 5.10.2.7 `int sparksee.Condition.LIKE = 6` [static]

Substring condition.

Null values cannot be used together with this condition.

This condition can just be used together with String values. It allows for searching substrings (case sensitive). Ex:

'AAABBBCCCD' Like 'BBB' returns TRUE

'AAABBBCCCD' Like 'bbb' returns FALSE

'AAABBBCCCD' Like 'E' returns FALSE

### 5.10.2.8 `int sparksee.Condition.LIKE_NO_CASE = 7` [static]

Substring (no case sensitive) condition.

Null values cannot be used together with this condition.

This condition can just be used together with String values. It allows for searching substrings (no case sensitive). Ex:

'AAABBBCCCD' LikeNoCase 'BBB' returns TRUE

'AAABBBCCCD' LikeNoCase 'bbb' returns TRUE

'AAABBBCCCD' LikeNoCase 'E' returns FALSE

#### 5.10.2.9 int sparksee.Condition.NOT\_EQUAL = 5 [static]

Not equal condition (!=).

Null values can be used together with this condition to retrieve all objects having a non-null value for an attribute.

#### 5.10.2.10 int sparksee.Condition.REG\_EXP = 9 [static]

Regular expression condition.

Null values cannot be used together with this condition.

This condition can just be used together with String values.

Regular expression format conforms most of the POSIX Extended Regular Expressions so it is case sensitive.

See the 'Regular expressions' section in the 'SPARKSEE User Manual' for details.

## 5.11 sparksee.ConnectedComponents Class Reference

[ConnectedComponents](#) class.

### Public Member Functions

- def `__init__` (self, s, materializedattribute)  
*Creates a new instance of [ConnectedComponents](#).*
- def `get_size` (self, id\_connected\_component)  
*Returns the number of nodes contained in the given connected component.*
- def `get_count` (self)  
*Returns the number of connected components found in the graph.*
- def `get_connected_component` (self, id\_node)  
*Returns the connected component where the given node belongs to.*
- def `get_nodes` (self, id\_connected\_component)  
*Returns the collection of nodes contained in the given connected component.*
- def `close` (self)  
*Closes the [ConnectedComponents](#) instance.*
- def `is_closed` (self)  
*Gets if [ConnectedComponents](#) has been closed or not.*

### 5.11.1 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>



### 5.11.2 Constructor & Destructor Documentation

#### 5.11.2.1 `def sparksee.ConnectedComponents.__init__( self, s, 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

<code>s</code>	[in] <a href="#">Session</a> to get the graph <a href="#">Graph</a> 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.
<code>materializedattribute</code>	[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.

### 5.11.3 Member Function Documentation

#### 5.11.3.1 `def sparksee.ConnectedComponents.close( self )`

Closes the [ConnectedComponents](#) instance.

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

#### 5.11.3.2 `def sparksee.ConnectedComponents.get_connected_component( self, id_node )`

Returns the connected component where the given node belongs to.

#### Parameters

<code>id_node</code>	[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.

#### 5.11.3.3 `def sparksee.ConnectedComponents.get_count( self )`

Returns the number of connected components found in the graph.

#### Returns

The number of connected components found in the graph.

#### 5.11.3.4 `def sparksee.ConnectedComponents.get_nodes( self, id_connected_component )`

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

## Parameters

<i>id_connected_component</i>	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.

## 5.11.3.5 def sparksee.ConnectedComponents.get\_size ( self, id\_connected\_component )

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

## Parameters

<i>id_connected_component</i>	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.

## 5.11.3.6 def sparksee.ConnectedComponents.is\_closed ( self )

Gets if [ConnectedComponents](#) has been closed or not.

## See also

[close\(\)](#)

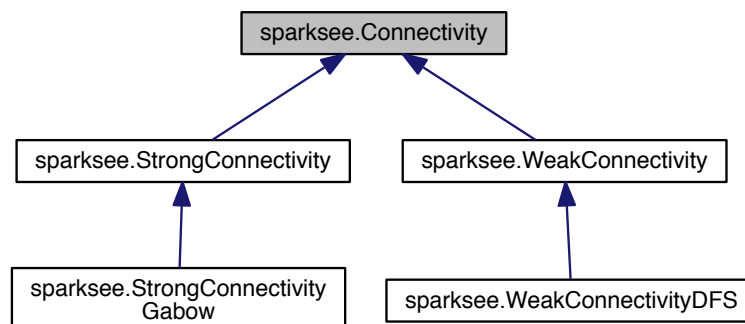
## Returns

TRUE if the [ConnectedComponents](#) instance has been closed, FALSE otherwise.

## 5.12 sparksee.Connectivity Class Reference

[Connectivity](#) class.

Inheritance diagram for sparksee.Connectivity:



## Public Member Functions

- def `run` (self)
  - Runs the algorithm in order to find the connected components.*
- def `exclude_nodes` (self, nodes)
  - Set which nodes can't be used.*
- def `get_connected_components` (self)
  - Returns the results generated by the execution of the algorithm.*
- def `exclude_edges` (self, edges)
  - Set which edges can't be used.*
- def `add_node_type` (self, t)
  - Allows connectivity through nodes of the given type.*
- def `set_materialized_attribute` (self, attribute\_name)
  - 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.*
- def `close` (self)
  - Closes the `Connectivity` instance.*
- def `add_all_node_types` (self)
  - Allows connectivity through all node types of the graph.*
- def `is_closed` (self)
  - Gets if `Connectivity` has been closed or not.*

### 5.12.1 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>

### 5.12.2 Member Function Documentation

#### 5.12.2.1 `def sparksee.Connectivity.add_node_type ( self, t )`

Allows connectivity through nodes of the given type.

#### Parameters

<code>t</code>	null
----------------	------

#### 5.12.2.2 `def sparksee.Connectivity.close ( self )`

Closes the `Connectivity` instance.

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

### 5.12.2.3 def sparksee.Connectivity.exclude\_edges ( self, 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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

### 5.12.2.4 def sparksee.Connectivity.exclude\_nodes ( self, 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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

### 5.12.2.5 def sparksee.Connectivity.get\_connected\_components ( self )

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.

### 5.12.2.6 def sparksee.Connectivity.is\_closed ( self )

Gets if [Connectivity](#) has been closed or not.

#### See also

[close\(\)](#)

#### Returns

TRUE if the [Connectivity](#) instance has been closed, FALSE otherwise.

### 5.12.2.7 def sparksee.Connectivity.run ( self )

Runs the algorithm in order to find the connected components.

This method can be called only once.

### 5.12.2.8 `def sparksee.Connectivity.set_materialized_attribute ( self, attribute_name )`

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

<code>attribute_name</code>	[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.
-----------------------------	---

## 5.13 `sparksee.Context` Class Reference

[Context](#) class.

#### Public Member Functions

- `def exclude\_nodes (self, nodes)`  
*Set which nodes can't be used.*
- `def \_\_init\_\_ (self, session, node)`  
*Creates a new instance.*
- `def exclude\_edges (self, edges)`  
*Set which edges can't be used.*
- `def compute (self)`  
*Gets the resulting collection of nodes.*
- `def add\_edge\_type (self, t, d)`  
*Allows for traversing edges of the given type.*
- `def compute (self, session, node, node_types, edge_types, dir, maxhops, include)`  
*Helper method to easily compute a context from a node.*
- `def set\_maximum\_hops (self, maxhops, include)`  
*Sets the maximum hops restriction.*
- `def add\_node\_type (self, t)`  
*Allows for traversing nodes of the given type.*
- `def close (self)`  
*Closes the [Context](#) instance.*
- `def add\_all\_edge\_types (self, d)`  
*Allows for traversing all edge types of the graph.*
- `def add\_all\_node\_types (self)`  
*Allows for traversing all node types of the graph.*
- `def is\_closed (self)`  
*Gets if [Context](#) has been closed or not.*

## 5.13.1 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>

## 5.13.2 Constructor &amp; Destructor Documentation

5.13.2.1 `def sparksee.Context.__init__( self, session, node )`

Creates a new instance.

## Parameters

<i>session</i>	[in] <a href="#">Session</a> to get the graph from and perform operation.
<i>node</i>	[in] Node to start traversal from.

## 5.13.3 Member Function Documentation

5.13.3.1 `def sparksee.Context.add_all_edge_types( self, d )`

Allows for traversing all edge types of the graph.

## Parameters

<i>d</i>	[in] Edge direction.
----------	----------------------

5.13.3.2 `def sparksee.Context.add_edge_type( self, t, d )`

Allows for traversing edges of the given type.

## Parameters

<i>t</i>	[in] Edge type.
<i>d</i>	[in] Edge direction.

5.13.3.3 `def sparksee.Context.add_node_type( self, t )`

Allows for traversing nodes of the given type.

## Parameters

<i>t</i>	null
----------	------

5.13.3.4 `def sparksee.Context.close ( self )`

Closes the [Context](#) instance.

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

5.13.3.5 `def sparksee.Context.compute ( self )`

Gets the resulting collection of nodes.

## Returns

The resulting collection of nodes.

Referenced by `sparksee.Context.compute()`.

5.13.3.6 `def sparksee.Context.compute ( self, session, node, node_types, edge_types, dir, maxhops, include )`

Helper method to easily compute a context from a node.

## Parameters

<i>session</i>	[in] <a href="#">Session</a> to get the graph from and perform operation.
<i>node</i>	[in] Node to start traversal from.
<i>node_types</i>	[in] Allowed node type list. NULL means all node types are allowed.
<i>edge_types</i>	[in] Allowed edge type list. NULL means all edge types are allowed.
<i>dir</i>	[in] Allowed direction for the allowed edge types.
<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
<i>include</i>	[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.

References `sparksee.Context.compute()`.

5.13.3.7 `def sparksee.Context.exclude_edges ( self, 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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

5.13.3.8 def sparksee.Context.exclude\_nodes ( *self*, *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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

5.13.3.9 def sparksee.Context.is\_closed ( *self* )

Gets if [Context](#) has been closed or not.

## See also

[close\(\)](#)

## Returns

TRUE if the [Context](#) instance has been closed, FALSE otherwise.

5.13.3.10 def sparksee.Context.set\_maximum\_hops ( *self*, *maxhops*, *include* )

Sets the maximum hops restriction.

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

## Parameters

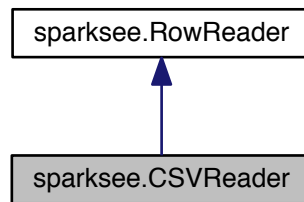
<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
<i>include</i>	[in] If TRUE, the resulting collection will include those nodes at distance less or equal than the given one, otherwise it will just include those nodes at distance 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.

## 5.14 sparksee.CSVReader Class Reference

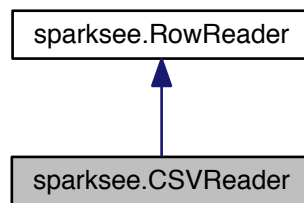
[CSVReader](#) interface.



Inheritance diagram for sparksee.CSVReader:



Collaboration diagram for sparksee.CSVReader:



### Public Member Functions

- def [set\\_separator](#) (self, sep)  
*Sets the character used to separate fields in the file.*
- def [set\\_num\\_lines](#) (self, num\_lines)  
*Used to limit the number of lines that will be read.*
- def [close](#) (self)  
*Closes the reader.*
- def [set\\_locale](#) (self, locale\_str)  
*Sets the locale that will be used to read the file.*
- def [read](#) (self, row)  
*Reads the next row as a string array.*
- def [set\\_multilines](#) (self, num\_extralines)  
*Allows the use of fields with more than one line.*
- def [open](#) (self, file\_path)  
*Opens the source file path.*
- def [reset](#) (self)  
*Moves the reader to the beginning.*
- def [set\\_start\\_line](#) (self, start\_line)  
*Sets the number of lines to be skipped from the beginning.*

- def `set_single_line` (self)  
*Only allows single line fields.*
- def `get_row` (self)  
*The row number for the current row.*
- def `__init__` (self)  
*Constructs `CSVReader`.*
- def `set_quotes` (self, quotes)  
*Sets the character used to quote fields.*

### 5.14.1 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>

### 5.14.2 Member Function Documentation

#### 5.14.2.1 def sparksee.CSVReader.close ( self )

Closes the reader.

**Exceptions**

<i>IOError</i>	If the close fails.
----------------	---------------------

**5.14.2.2 def sparksee.CSVReader.get\_row ( self )**

The row number for the current row.

**Returns**

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

**Exceptions**

<i>IOError</i>	If it fails.
----------------	--------------

**5.14.2.3 def sparksee.CSVReader.open ( self, file\_path )**

Opens the source file path.

File can be optionally compressed in GZIP format.

**Parameters**

<i>file_path</i>	[in] CSV file path.
------------------	---------------------

**Exceptions**

<i>IOError</i>	If bad things happen opening the file.
----------------	--

**5.14.2.4 def sparksee.CSVReader.read ( self, row )**

Reads the next row as a string array.

**Parameters**

<i>row</i>	[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**

<i>IOError</i>	If bad things happen during the read.
----------------	---------------------------------------

#### 5.14.2.5 def sparksee.CSVReader.reset ( self )

Moves the reader to the beginning.

Restarts the reader.

##### Returns

true if the reader can be restarted, false otherwise.

##### Exceptions

<i>IOError</i>	If bad things happen during the restart.
----------------	--

#### 5.14.2.6 def sparksee.CSVReader.set\_locale ( self, locale\_str )

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

##### Parameters

<i>locale_str</i>	[in] The locale string for the file encoding.
-------------------	---

#### 5.14.2.7 def sparksee.CSVReader.set\_multilines ( self, num\_extralines )

Allows the use of fields with more than one line.

##### Parameters

<i>num_extralines</i>	[in] Maximum number of extra lines for each column (0==unlimited, N==N+1 total rows).
-----------------------	---

#### 5.14.2.8 def sparksee.CSVReader.set\_num\_lines ( self, num\_lines )

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

##### Parameters

<i>num_lines</i>	[in] The maximum number of lines to read (0 == unlimited)
------------------	---

#### 5.14.2.9 def sparksee.CSVReader.set\_quotes ( self, quotes )

Sets the character used to quote fields.

##### Parameters

<i>quotes</i>	[in] Quote character.
---------------	-----------------------

**Exceptions**

<i>RuntimeError</i>	null
---------------------	------

5.14.2.10 `def sparksee.CSVReader.set_separator ( self, sep )`

Sets the character used to separate fields in the file.

**Parameters**

<i>sep</i>	[in] Separator character.
------------	---------------------------

**Exceptions**

<i>RuntimeError</i>	null
---------------------	------

5.14.2.11 `def sparksee.CSVReader.set_start_line ( self, start_line )`

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

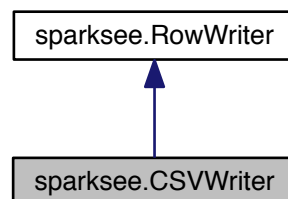
**Parameters**

<i>start_line</i>	[in] The line number to skip for start reading
-------------------	--

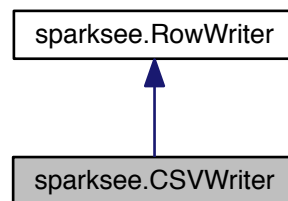
## 5.15 sparksee.CSVWriter Class Reference

[CSVWriter](#) interface.

Inheritance diagram for `sparksee.CSVWriter`:



Collaboration diagram for sparksee.CSVWriter:



### Public Member Functions

- def `__init__` (self)  
*Creates a new instance.*
- def `set_separator` (self, sep)  
*Sets the character used to separate fields in the file.*
- def `set_auto_quotes` (self, autoquotes)  
*Sets on/off the automatic quote mode.*
- def `open` (self, f)  
*Opens the output file path.*
- def `write` (self, row)  
*Writes the next row.*
- def `set_forced_quotes` (self, forcequotes)  
*Disables the automatic quote mode and forces to be quoted those positions set to TRUE in the given vector.*
- def `close` (self)  
*Closes the writer.*
- def `set_locale` (self, locale\_str)  
*Sets the locale that will be used to write the file.*
- def `set_quotes` (self, quotes)  
*Sets the character used to quote fields.*

#### 5.15.1 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>

## 5.15.2 Member Function Documentation

### 5.15.2.1 `def sparksee.CSVWriter.close ( self )`

Closes the writer.

#### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the close fails.

### 5.15.2.2 `def sparksee.CSVWriter.open ( self, f )`

Opens the output file path.

#### Parameters

<i>f</i>	[in] Output file path.
----------	------------------------

#### Exceptions

<i>IOError</i>	If bad things happen opening the file.
----------------	--

### 5.15.2.3 `def sparksee.CSVWriter.set_auto_quotes ( self, 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

<i>autoquotes</i>	[in] If TRUE it enables the automatic quote mode, if FALSE it disables it.
-------------------	--

### 5.15.2.4 `def sparksee.CSVWriter.set_forced_quotes ( self, forcequotes )`

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

#### Parameters

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

### 5.15.2.5 `def sparksee.CSVWriter.set_locale ( self, locale_str )`

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

## Parameters

<i>locale_str</i>	[in] The locale string for the file encoding.
-------------------	---

## 5.15.2.6 def sparksee.CSVWriter.set\_quotes ( self, quotes )

Sets the character used to quote fields.

## Parameters

<i>quotes</i>	[in] Quote character.
---------------	-----------------------

## Exceptions

<i>RuntimeError</i>	null
---------------------	------

## 5.15.2.7 def sparksee.CSVWriter.set\_separator ( self, sep )

Sets the character used to separate fields in the file.

## Parameters

<i>sep</i>	[in] Separator character.
------------	---------------------------

## Exceptions

<i>RuntimeError</i>	null
---------------------	------

## 5.15.2.8 def sparksee.CSVWriter.write ( self, row )

Writes the next row.

## Parameters

<i>row</i>	[in] Row of data.
------------	-------------------

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If bad things happen during the write.

## 5.16 sparksee.Database Class Reference

[Database](#) class.



### Public Member Functions

- def `dump_schema` (self, file\_path)  
*Dump the database schema to the given file using the [Sparksee](#) scripting format.*
- def `new_session` (self)  
*Creates a new [Session](#).*
- def `get_path` (self)  
*Gets the path of the [Database](#).*
- def `enable_rollback` (self)  
*Enables the rollback mechanism.*
- def `fix_current_cache_max_size` (self)  
*Sets the cache maximum size to the current cache size in use.*
- def `redo_precommitted` (self, tx\_id)  
*Redo a pending precommitted transaction recovered.*
- def `get_statistics` (self, stats)  
*Gets [Database](#) statistics.*
- def `set_cache_max_size` (self, mega\_bytes)  
*Sets the cache maximum size (in MB).*
- def `disable_rollback` (self)  
*Disables the rollback mechanism.*
- def `close` (self)  
*Closes the [Database](#) instance.*
- def `get_alias` (self)  
*Gets the alias of the [Database](#).*
- def `get_cache_max_size` (self)  
*Gets the cache maximum size (in MB).*
- def `is_closed` (self)  
*Gets if [Database](#) instance has been closed or not.*

#### 5.16.1 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 SPARKSEConfig.

#### Author

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

#### 5.16.2 Member Function Documentation

##### 5.16.2.1 `def sparksee.Database.close ( self )`

Closes the [Database](#) instance.

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

##### 5.16.2.2 `def sparksee.Database.dump_schema ( self, file_path )`

Dump the database schema to the given file using the [Sparksee](#) scripting format.

## Parameters

<code>file_path</code>	null
------------------------	------

## 5.16.2.3 def sparksee.Database.get\_alias ( self )

Gets the alias of the [Database](#).

## Returns

The alias of the [Database](#).

## 5.16.2.4 def sparksee.Database.get\_cache\_max\_size ( self )

Gets the cache maximum size (in MB).

## Returns

Returns the current cache max size.

## 5.16.2.5 def sparksee.Database.get\_path ( self )

Gets the path of the [Database](#).

## Returns

The path of the [Database](#).

## 5.16.2.6 def sparksee.Database.get\_statistics ( self, stats )

Gets [Database](#) statistics.

## Parameters

<code>stats</code>	[out] The <a href="#">DatabaseStatistics</a> instance.
--------------------	--

## 5.16.2.7 def sparksee.Database.is\_closed ( self )

Gets if [Database](#) instance has been closed or not.

## See also

[close\(\)](#)

## Returns

TRUE if the [Database](#) instance has been closed, FALSE otherwise.

#### 5.16.2.8 `def sparksee.Database.redo_precommitted ( self, tx_id )`

Redo a pending precommitted transaction recovered.

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

##### Parameters

<code>tx↔ _id</code>	null
--------------------------	------

#### 5.16.2.9 `def sparksee.Database.set_cache_max_size ( self, mega_bytes )`

Sets the cache maximum size (in MB).

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

##### Parameters

<code>mega_bytes</code>	[in] The new cache max size.
-------------------------	------------------------------

## 5.17 `sparksee.DatabaseStatistics` Class Reference

[Database](#) statistics.

### Public Member Functions

- `def get\_read (self)`  
*Gets total read data in KBytes.*
- `def get\_data (self)`  
*Gets database size in KBytes.*
- `def get\_temp (self)`  
*Gets temporary storage file size in KBytes.*
- `def get\_write (self)`  
*Gets total written data in KBytes.*
- `def get\_cache (self)`  
*Gets cache size in KBytes.*
- `def get\_sessions (self)`  
*Gets the number of sessions.*

#### 5.17.1 Detailed Description

[Database](#) statistics.

##### Author

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

## 5.17.2 Member Function Documentation

### 5.17.2.1 def sparksee.DatabaseStatistics.get\_cache ( self )

Gets cache size in KBytes.

#### Returns

Cache size in KBytes.

### 5.17.2.2 def sparksee.DatabaseStatistics.get\_data ( self )

Gets database size in KBytes.

#### Returns

[Database](#) size in KBytes.

### 5.17.2.3 def sparksee.DatabaseStatistics.get\_read ( self )

Gets total read data in KBytes.

#### Returns

Total read data in KBytes.

### 5.17.2.4 def sparksee.DatabaseStatistics.get\_sessions ( self )

Gets the number of sessions.

#### Returns

The number of sessions.

### 5.17.2.5 def sparksee.DatabaseStatistics.get\_temp ( self )

Gets temporary storage file size in KBytes.

#### Returns

Temporary storage file size in KBytes.

### 5.17.2.6 def sparksee.DatabaseStatistics.get\_write ( self )

Gets total written data in KBytes.

#### Returns

Total read written in KBytes.

## 5.18 sparksee.DataType Class Reference

Data type (domain) enumeration.

### Static Public Attributes

- int **BOOLEAN** = 0  
*Boolean data type.*
- int **INTEGER** = 1  
*32-bit signed integer data type.*
- int **LONG** = 2  
*64-bit signed integer data type.*
- int **DOUBLE** = 3  
*64-bit signed double data type.*
- int **TIMESTAMP** = 4  
*Distance from Epoch (UTC) time in milliseconds precision.*
- int **STRING** = 5  
*Unicode string data type.*
- int **TEXT** = 6  
*Large unicode character object (CLOB) data type.*
- int **OID** = 7  
*Object identifier (OID) data type.*

### 5.18.1 Detailed Description

Data type (domain) enumeration.

#### Author

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

### 5.18.2 Member Data Documentation

#### 5.18.2.1 int sparksee.DataType.STRING = 5 [static]

Unicode string data type.

2048 characters maximum length.

#### 5.18.2.2 int sparksee.DataType.TEXT = 6 [static]

Large unicode character object (CLOB) data type.

#### TextStream

#### 5.18.2.3 int sparksee.DataType.TIMESTAMP = 4 [static]

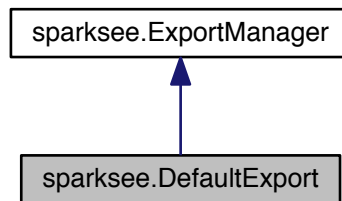
Distance from Epoch (UTC) time in milliseconds precision.

It just works properly with timestamps in the range ['1970-01-01 00:00:01' UTC, '2038-01-19 03:14:07' UTC].

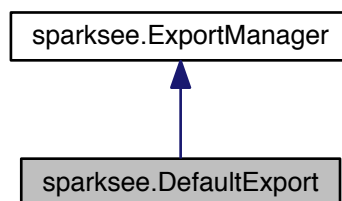
## 5.19 sparksee.DefaultExport Class Reference

Default implementation for [ExportManager](#) class.

Inheritance diagram for sparksee.DefaultExport:



Collaboration diagram for sparksee.DefaultExport:



### Public Member Functions

- def [prepare](#) (self, graph)  
*Default implementation of the [ExportManager](#) class method Prepare.*
- def [get\\_graph](#) (self, graph\_export)  
*Default implementation of the [ExportManager](#) class method GetGraph.*
- def [release](#) (self)  
*Default implementation of the [ExportManager](#) class method Release.*
- def [\\_\\_init\\_\\_](#) (self)  
*Creates a new instance.*
- def [get\\_node\\_type](#) (self, type, node\_export)  
*Default implementation of the [ExportManager](#) class method GetNodeType.*
- def [enable\\_type](#) (self, type)  
*Default implementation of the [ExportManager](#) class method EnableType.*
- def [get\\_node](#) (self, node, node\_export)  
*Default implementation of the [ExportManager](#) class method GetNode.*
- def [get\\_edge](#) (self, edge, edge\_export)  
*Default implementation of the [ExportManager](#) class method GetEdge.*
- def [get\\_edge\\_type](#) (self, type, edge\_export)  
*Default implementation of the [ExportManager](#) class method GetEdgeType.*

### 5.19.1 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>

### 5.19.2 Member Function Documentation

#### 5.19.2.1 `def sparksee.DefaultExport.enable_type ( self, type )`

Default implementation of the [ExportManager](#) class method `EnableType`.

This enables all node and edge types to be exported.

#### Parameters

<i>type</i>	[in] The type to enable.
-------------	--------------------------

#### Returns

TRUE.

#### 5.19.2.2 `def sparksee.DefaultExport.get_edge ( self, edge, edge_export )`

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

<i>edge</i>	[in] An edge.
<i>edge_export</i>	[out] The <a href="#">EdgeExport</a> that will store the information.

#### Returns

TRUE.

#### 5.19.2.3 `def sparksee.DefaultExport.get_edge_type ( self, type, edge_export )`

Default implementation of the [ExportManager](#) class method `GetEdgeType`.

This sets de default [EdgeExport](#) values.

## Parameters

<i>type</i>	[in] An edge type.
<i>edge_export</i>	[out] The <a href="#">EdgeExport</a> that will store the information.

## Returns

TRUE.

## 5.19.2.4 def sparksee.DefaultExport.get\_graph ( self, graph\_export )

Default implementation of the [ExportManager](#) class method GetGraph.

This sets the default [GraphExport](#) values and "Graph" as the label.

## Parameters

<i>graph_export</i>	[out] The <a href="#">GraphExport</a> that will store the information.
---------------------	--

## Returns

TRUE.

## 5.19.2.5 def sparksee.DefaultExport.get\_node ( self, node, node\_export )

Default implementation of the [ExportManager](#) class method GetNode.

This sets the default [NodeExport](#) values and sets the OID as the label.

## Parameters

<i>node</i>	[in] A node.
<i>node_export</i>	[out] The <a href="#">NodeExport</a> that will store the information.

## Returns

TRUE.

## 5.19.2.6 def sparksee.DefaultExport.get\_node\_type ( self, type, node\_export )

Default implementation of the [ExportManager](#) class method GetNodeType.

This sets de default [NodeExport](#) values.

## Parameters

<i>type</i>	[in] A node type.
<i>node_export</i>	[out] The <a href="#">NodeExport</a> that will store the information.



**Returns**

TRUE.

**5.19.2.7** `def sparksee.DefaultExport.prepare ( self, graph )`

Default implementation of the [ExportManager](#) class method Prepare.

**Parameters**

<code>graph</code>	<code>null</code>
--------------------	-------------------

**5.20 sparksee.DisjointCommunities Class Reference**

[DisjointCommunities](#) class.

**Public Member Functions**

- `def get\_nodes (self, id_community)`  
*Returns the collection of nodes contained in the given community.*
- `def get\_community (self, id_node)`  
*Returns the disjoint community where the given node belongs to.*
- `def get\_count (self)`  
*Returns the number of communities found in the graph.*
- `def \_\_init\_\_ (self, session, materializedattribute)`  
*Creates a new instance of [DisjointCommunities](#).*
- `def get\_size (self, id_community)`  
*Returns the number of nodes contained in the given community.*
- `def close (self)`  
*Closes the [DisjointCommunities](#) instance.*
- `def is\_closed (self)`  
*Gets if [DisjointCommunities](#) has been closed or not.*

**5.20.1 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>

## 5.20.2 Constructor &amp; Destructor Documentation

## 5.20.2.1 def sparksee.DisjointCommunities.\_\_init\_\_( self, session, materializedattribute )

Creates a new instance of [DisjointCommunities](#).

This constructor method can only be called when a previous execution of any implementation of the [DisjointCommunityDetection](#) 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

<i>session</i>	[in] <a href="#">Session</a> to get the graph <a href="#">Graph</a> 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.
<i>materializedattribute</i>	[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.

## 5.20.3 Member Function Documentation

## 5.20.3.1 def sparksee.DisjointCommunities.close ( self )

Closes the [DisjointCommunities](#) instance.

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

## 5.20.3.2 def sparksee.DisjointCommunities.get\_community ( self, id\_node )

Returns the disjoint community where the given node belongs to.

## Parameters

<i>id_node</i>	[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.

## 5.20.3.3 def sparksee.DisjointCommunities.get\_count ( self )

Returns the number of communities found in the graph.

## Returns

The number of communities found in the graph.

## 5.20.3.4 def sparksee.DisjointCommunities.get\_nodes ( self, id\_community )

Returns the collection of nodes contained in the given community.

**Parameters**

<i>id_community</i>	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.

5.20.3.5 `def sparksee.DisjointCommunities.get_size ( self, id_community )`

Returns the number of nodes contained in the given community.

**Parameters**

<i>id_community</i>	The community for which the number of nodes contained in it will be returned.
---------------------	---

**Returns**

The number of nodes contained in the given community.

5.20.3.6 `def sparksee.DisjointCommunities.is_closed ( self )`

Gets if [DisjointCommunities](#) has been closed or not.

**See also**

[close\(\)](#)

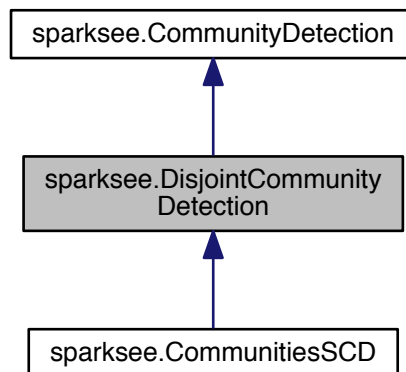
**Returns**

TRUE if the [DisjointCommunities](#) instance has been closed, FALSE otherwise.

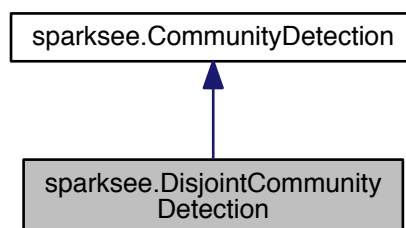
## 5.21 `sparksee.DisjointCommunityDetection` Class Reference

[DisjointCommunityDetection](#) class.

Inheritance diagram for sparksee.DisjointCommunityDetection:



Collaboration diagram for sparksee.DisjointCommunityDetection:



#### Public Member Functions

- def `run` (self)  
*Runs the algorithm in order to find the communities.*
- def `exclude_nodes` (self, nodes)  
*Set which nodes can't be used.*
- def `add_all_edge_types` (self)  
*Allows connectivity through all edge types of the graph.*
- def `get_communities` (self)  
*Returns the results generated by the execution of the algorithm.*
- def `add_edge_type` (self, type)  
*Allows connectivity through edges of the given type.*
- def `add_node_type` (self, type)  
*Allows connectivity through nodes of the given type.*

- def `exclude_edges` (self, edges)  
*Set which edges can't be used.*
- def `set_materialized_attribute` (self, attribute\_name)  
*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.*
- def `include_nodes` (self, nodes)  
*Set additional nodes that can be used.*
- def `add_all_node_types` (self)  
*Allows connectivity through all node types of the graph.*
- def `include_edges` (self, edges)  
*Set additional edges that can be used.*
- def `close` (self)  
*Closes the `CommunityDetection` instance.*
- def `is_closed` (self)  
*Gets if `CommunityDetection` has been closed or not.*

### 5.21.1 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>

### 5.21.2 Member Function Documentation

#### 5.21.2.1 `def sparksee.DisjointCommunityDetection.add_all_edge_types ( self )`

Allows connectivity through all edge types of the graph.

The edges can be used in Any direction.

#### 5.21.2.2 `def sparksee.DisjointCommunityDetection.add_edge_type ( self, type )`

Allows connectivity through edges of the given type.

The edges can be used in Any direction.

#### Parameters

<code>type</code>	[in] Edge type.
-------------------	-----------------

#### 5.21.2.3 `def sparksee.DisjointCommunityDetection.add_node_type ( self, type )`

Allows connectivity through nodes of the given type.

## Parameters

<i>type</i>	null
-------------	------

5.21.2.4 def sparksee.CommunityDetection.close ( *self* ) [inherited]

Closes the [CommunityDetection](#) instance.

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

5.21.2.5 def sparksee.DisjointCommunityDetection.exclude\_edges ( *self*, *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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

5.21.2.6 def sparksee.DisjointCommunityDetection.exclude\_nodes ( *self*, *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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

5.21.2.7 def sparksee.DisjointCommunityDetection.get\_communities ( *self* )

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.

5.21.2.8 def sparksee.DisjointCommunityDetection.include\_edges ( *self*, *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

<code>edges</code>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------------	---

5.21.2.9 `def sparksee.DisjointCommunityDetection.include_nodes ( self, 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

<code>nodes</code>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------------	---

5.21.2.10 `def sparksee.CommunityDetection.is_closed ( self )` [inherited]

Gets if [CommunityDetection](#) has been closed or not.

## See also

[close\(\)](#)

## Returns

TRUE if the [CommunityDetection](#) instance has been closed, FALSE otherwise.

5.21.2.11 `def sparksee.DisjointCommunityDetection.run ( self )`

Runs the algorithm in order to find the communities.

This method can be called only once.

5.21.2.12 `def sparksee.DisjointCommunityDetection.set_materialized_attribute ( self, attribute_name )`

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

<i>attribute_name</i>	[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.
-----------------------	---

## 5.22 sparksee.EdgeData Class Reference

Edge data class.

### Public Member Functions

- def [get\\_head](#) (self)  
*Gets the head of the edge.*
- def [get\\_edge](#) (self)  
*Gets the edge identifier.*
- def [get\\_tail](#) (self)  
*Gets the tail of the edge.*

#### 5.22.1 Detailed Description

Edge data class.

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

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

#### Author

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

#### 5.22.2 Member Function Documentation

##### 5.22.2.1 def sparksee.EdgeData.get\_edge ( self )

Gets the edge identifier.

#### Returns

The [Sparksee](#) edge identifier.

##### 5.22.2.2 def sparksee.EdgeData.get\_head ( self )

Gets the head of the edge.

#### Returns

The [Sparksee](#) edge identifier of the head of the edge.



### 5.22.2.3 `def sparksee.EdgeData.get_tail ( self )`

Gets the tail of the edge.

#### Returns

The [Sparksee](#) edge identifier of the tail of the edge.

## 5.23 `sparksee.EdgeExport` Class Reference

Stores edge exporting values.

#### Public Member Functions

- `def set\_as\_directed (self, directed)`  
*Sets if the edge should be managed as directed.*
- `def get\_labelcolor\_rgb (self)`  
*Gets the edge label color.*
- `def set\_label (self, label)`  
*Sets the edge label.*
- `def set\_width (self, width)`  
*Sets the edge width.*
- `def set\_color\_rgb (self, color)`  
*Sets the edge color.*
- `def set\_font\_size (self, size)`  
*Sets the edge label font size.*
- `def set\_labelcolor\_rgb (self, color)`  
*Sets the edge label color.*
- `def get\_color\_rgb (self)`  
*Gets the edge color.*
- `def get\_font\_size (self)`  
*Gets the edge label font size.*
- `def set\_defaults (self)`  
*Sets to default values.*
- `def \_\_init\_\_ (self)`  
*Creates a new instance.*
- `def get\_label (self)`  
*Gets the edge label.*
- `def get\_width (self)`  
*Gets the edge width.*
- `def as\_directed (self)`  
*Gets if the edge should be managed as directed.*

### 5.23.1 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 (0xD3D3D3, Light gray).

Label color: 0 (0x000000, Black).

Width: 5px.

Font size: 10.

#### Author

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

### 5.23.2 Member Function Documentation

#### 5.23.2.1 def sparksee.EdgeExport.as\_directed ( self )

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.

#### 5.23.2.2 def sparksee.EdgeExport.get\_color\_rgb ( self )

Gets the edge color.

#### Returns

The edge color.

#### 5.23.2.3 def sparksee.EdgeExport.get\_font\_size ( self )

Gets the edge label font size.

#### Returns

The edge label font size.

#### 5.23.2.4 `def sparksee.EdgeExport.get_label ( self )`

Gets the edge label.

##### Returns

The edge label.

#### 5.23.2.5 `def sparksee.EdgeExport.get_labelcolor_rgb ( self )`

Gets the edge label color.

##### Returns

The edge label color.

#### 5.23.2.6 `def sparksee.EdgeExport.get_width ( self )`

Gets the edge width.

##### Returns

The edge width.

#### 5.23.2.7 `def sparksee.EdgeExport.set_as_directed ( self, directed )`

Sets if the edge should be managed as directed.

##### Parameters

<i>directed</i>	[in] If TRUE, use as directed, otherwise use as undirected.
-----------------	---

#### 5.23.2.8 `def sparksee.EdgeExport.set_color_rgb ( self, color )`

Sets the edge color.

##### Parameters

<i>color</i>	[in] The edge color.
--------------	----------------------

#### 5.23.2.9 `def sparksee.EdgeExport.set_font_size ( self, size )`

Sets the edge label font size.

##### Parameters

<i>size</i>	[in] The edge label font size.
-------------	--------------------------------

5.23.2.10 `def sparksee.EdgeExport.set_label ( self, label )`

Sets the edge label.

Parameters

<i>label</i>	[in] The edge label.
--------------	----------------------

5.23.2.11 `def sparksee.EdgeExport.set_labelcolor_rgb ( self, color )`

Sets the edge label color.

Parameters

<i>color</i>	[in] The edge label color.
--------------	----------------------------

5.23.2.12 `def sparksee.EdgeExport.set_width ( self, width )`

Sets the edge width.

Parameters

<i>width</i>	[in] The edge width.
--------------	----------------------

## 5.24 sparksee.EdgesDirection Class Reference

Edges direction enumeration.

Static Public Attributes

- int **INGOING** = 0  
*In-going edges.*
- int **OUTGOING** = 1  
*Out-going edges.*
- int **ANY** = 2  
*In-going or out-going edges.*

### 5.24.1 Detailed Description

Edges direction enumeration.

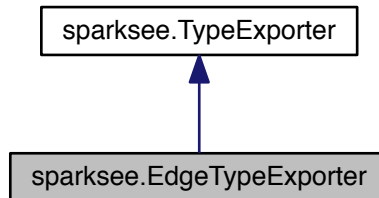
Author

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

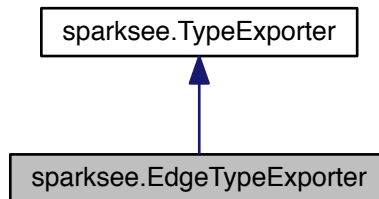
## 5.25 sparksee.EdgeTypeExporter Class Reference

[EdgeTypeExporter](#) class.

Inheritance diagram for sparksee.EdgeTypeExporter:



Collaboration diagram for sparksee.EdgeTypeExporter:



### Public Member Functions

- def [run](#) (self)  
See the [TypeExporter](#) class [Run](#) method.
- def [set\\_type](#) (self, type)  
Sets the type to be exported.
- def [set\\_head\\_position](#) (self, pos)  
Sets the position (index column) of the head attribute in the exported data.
- def [set\\_attributes](#) (self, attrs)  
Sets the list of Attributes.
- def [set\\_frequency](#) (self, freq)  
Sets the frequency of listener notification.
- def [set\\_head\\_attribute](#) (self, attr)  
Sets the attribute that will be used to get the value to be dumped for the head of the edge.
- def [set\\_tail\\_attribute](#) (self, attr)  
Sets the attribute that will be used to get the value to be dumped for the tail of the edge.

- def `set_tail_position` (self, pos)  
*Sets the position (index column) of the tail attribute in the exported data.*
- def `__init__` (self)  
*Creates a new instance.*
- def `set_header` (self, header)  
*Sets the presence of a header row.*
- def `set_row_writer` (self, rw)  
*Sets the output data destination.*
- def `set_graph` (self, graph)  
*Sets the graph that will be exported.*
- def `__init__` (self, row\_writer, graph, type, attrs, h\_pos, t\_pos, h\_attr, t\_attr)  
*Creates a new instance.*
- def `register` (self, tel)  
*Registers a new listener.*

### 5.25.1 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>

### 5.25.2 Constructor & Destructor Documentation

#### 5.25.2.1 def sparksee.EdgeTypeExporter.\_\_init\_\_( self, row\_writer, graph, type, attrs, h\_pos, t\_pos, h\_attr, t\_attr )

Creates a new instance.

#### Parameters

<code>row_writer</code>	[in] Output <code>RowWriter</code> .
<code>graph</code>	[in] <code>Graph</code> .
<code>type</code>	[in] <code>Type</code> identifier.
<code>attrs</code>	[in] <code>Attribute</code> identifiers to be exported.
<code>h_pos</code>	[in] The position (index column) for the head value.
<code>t_pos</code>	[in] The position (index column) for the tail value.
<code>h_attr</code>	[in] The attribute identifier to get the value to be dumped for the head.
<code>t_attr</code>	[in] The attribute identifier to get the value to be dumped for the tail.

References `sparksee.EdgeTypeExporter.__init__()`.

### 5.25.3 Member Function Documentation

### 5.25.3.1 `def sparksee.EdgeTypeExporter.register ( self, tel )`

Registers a new listener.

#### Parameters

<i>tel</i>	[in] <a href="#">TypeExporterListener</a> to be registered.
------------	---

### 5.25.3.2 `def sparksee.EdgeTypeExporter.run ( self )`

See the [TypeExporter](#) class Run method.

#### Exceptions

<i>RuntimeError</i>	null
<i>IOException</i>	null

### 5.25.3.3 `def sparksee.EdgeTypeExporter.set_attributes ( self, attrs )`

Sets the list of Attributes.

#### Parameters

<i>attrs</i>	[in] <a href="#">Attribute</a> identifiers to be exported
--------------	---

### 5.25.3.4 `def sparksee.EdgeTypeExporter.set_frequency ( self, freq )`

Sets the frequency of listener notification.

#### Parameters

<i>freq</i>	[in] Frequency in number of rows managed to notify progress to all listeners
-------------	--

### 5.25.3.5 `def sparksee.EdgeTypeExporter.set_graph ( self, graph )`

Sets the graph that will be exported.

#### Parameters

<i>graph</i>	[in] <a href="#">Graph</a> .
--------------	------------------------------

### 5.25.3.6 `def sparksee.EdgeTypeExporter.set_head_attribute ( self, attr )`

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

## Parameters

<i>attr</i>	[in] Head <a href="#">Attribute</a>
-------------	-------------------------------------

5.25.3.7 def sparksee.EdgeTypeExporter.set\_head\_position ( *self*, *pos* )

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

## Parameters

<i>pos</i>	[in] Head position
------------	--------------------

5.25.3.8 def sparksee.EdgeTypeExporter.set\_header ( *self*, *header* )

Sets the presence of a header row.

## Parameters

<i>header</i>	[in] If TRUE, a header row is dumped with the name of the attributes.
---------------	---

5.25.3.9 def sparksee.EdgeTypeExporter.set\_row\_writer ( *self*, *rw* )

Sets the output data destination.

## Parameters

<i>rw</i>	[in] Input <a href="#">RowWriter</a> .
-----------	--

5.25.3.10 def sparksee.EdgeTypeExporter.set\_tail\_attribute ( *self*, *attr* )

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

## Parameters

<i>attr</i>	[in] Tail <a href="#">Attribute</a>
-------------	-------------------------------------

5.25.3.11 def sparksee.EdgeTypeExporter.set\_tail\_position ( *self*, *pos* )

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

## Parameters

<i>pos</i>	[in] Tail position
------------	--------------------



### 5.25.3.12 `def sparksee.EdgeTypeExporter.set_type ( self, type )`

Sets the type to be exported.

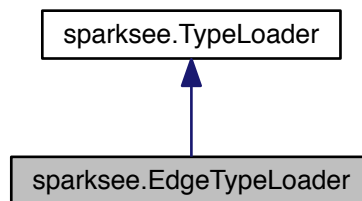
#### Parameters

<code>type</code>	[in] <a href="#">Type</a> identifier.
-------------------	---------------------------------------

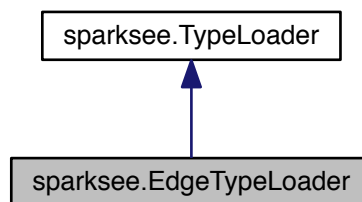
## 5.26 `sparksee.EdgeTypeLoader` Class Reference

[EdgeTypeLoader](#) class.

Inheritance diagram for `sparksee.EdgeTypeLoader`:



Collaboration diagram for `sparksee.EdgeTypeLoader`:



#### Public Member Functions

- `def run (self)`  
See the [TypeLoader](#) class `Run` method.
- `def set_log_error (self, path)`  
Sets a log error file.

- def `set_type` (self, type)  
*Sets the type to be loaded.*
- def `set_head_position` (self, pos)  
*Sets the position of the head attribute in the source data.*
- def `set_locale` (self, locale\_str)  
*Sets the locale that will be used to read the data.*
- def `set_timestamp_format` (self, timestamp\_format)  
*Sets a specific timestamp format.*
- def `set_attributes` (self, attrs)  
*Sets the list of Attributes.*
- def `__init__` (self)  
*Creates a new instance.*
- def `register` (self, tel)  
*Registers a new listener.*
- def `__init__` (self, row\_reader, graph, type, attrs, attrs\_pos, h\_pos, t\_pos, h\_attr, t\_attr, head\_mep, tail\_mep)  
*Creates a new instance.*
- def `set_frequency` (self, freq)  
*Sets the frequency of listener notification.*
- def `run_two_phases` (self)  
*See the `TypeLoader` class `RunTwoPhases` method.*
- def `set_head_attribute` (self, attr)  
*Sets the attribute that will be used to find the head of the edge.*
- def `set_tail_attribute` (self, attr)  
*Sets the attribute that will be used to find the tail of the edge.*
- def `set_tail_position` (self, pos)  
*Sets the position of the tail attribute in the source data.*
- def `set_row_reader` (self, rr)  
*Sets the input data source.*
- def `set_tail_mep` (self, mep)  
*Sets the flag to create missing tail nodes when loading nodes or edges.*
- def `set_attribute_positions` (self, attrs\_pos)  
*Sets the list of attribute positions.*
- def `set_graph` (self, graph)  
*Sets the graph where the data will be loaded.*
- def `run_n_phases` (self, partitions)  
*See the `TypeLoader` class `RunNPhases` method.*
- def `set_head_mep` (self, mep)  
*Sets the flag to create missing head nodes when loading nodes or edges.*
- def `set_log_off` (self)  
*Truns off all the error reporting.*

### 5.26.1 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>

## 5.26.2 Constructor & Destructor Documentation

5.26.2.1 `def sparksee.EdgeTypeLoader.__init__( self, row_reader, graph, type, attrs, attrs_pos, h_pos, t_pos, h_attr, t_attr, head_mep, tail_mep )`

Creates a new instance.

### Parameters

<i>row_reader</i>	[in] Input <a href="#">RowReader</a> .
<i>graph</i>	[in] <a href="#">Graph</a> .
<i>type</i>	[in] <a href="#">Type</a> identifier.
<i>attrs</i>	[in] <a href="#">Attribute</a> identifiers to be loaded.
<i>attrs_pos</i>	[in] <a href="#">Attribute</a> positions (column index $\geq 0$ ). to all listeners.
<i>h_pos</i>	[in] The position (index column) for the head value.
<i>t_pos</i>	[in] The position (index column) for the tail value.
<i>h_attr</i>	[in] The attribute identifier for the head.
<i>t_attr</i>	[in] The attribute identifier for the tail.
<i>head_mep</i>	[in] The <a href="#">MissingEndpoint</a> policy for the head nodes
<i>tail_mep</i>	[in] The <a href="#">MissingEndpoint</a> policy for the tail nodes

References `sparksee.EdgeTypeLoader.__init__()`.

## 5.26.3 Member Function Documentation

5.26.3.1 `def sparksee.EdgeTypeLoader.register( self, tel )`

Registers a new listener.

### Parameters

<i>tel</i>	<a href="#">TypeLoaderListener</a> to be registered.
------------	--

5.26.3.2 `def sparksee.EdgeTypeLoader.run( self )`

See the [TypeLoader](#) class Run method.

### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

5.26.3.3 `def sparksee.EdgeTypeLoader.run_n_phases( self, partitions )`

See the [TypeLoader](#) class RunNPhases method.

## Parameters

<i>partitions</i>	null
-------------------	------

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

## 5.26.3.4 def sparksee.EdgeTypeLoader.run\_two\_phases ( self )

See the [TypeLoader](#) class RunTwoPhases method.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

## 5.26.3.5 def sparksee.EdgeTypeLoader.set\_attribute\_positions ( self, attrs\_pos )

Sets the list of attribute positions.

## Parameters

<i>attrs_pos</i>	[in] <a href="#">Attribute</a> positions (column index >=0).
------------------	--

## 5.26.3.6 def sparksee.EdgeTypeLoader.set\_attributes ( self, attrs )

Sets the list of Attributes.

## Parameters

<i>attrs</i>	[in] <a href="#">Attribute</a> identifiers to be loaded
--------------	---

## 5.26.3.7 def sparksee.EdgeTypeLoader.set\_frequency ( self, freq )

Sets the frequency of listener notification.

## Parameters

<i>freq</i>	[in] Frequency in number of rows managed to notify progress to all listeners
-------------	--

## 5.26.3.8 def sparksee.EdgeTypeLoader.set\_graph ( self, graph )

Sets the graph where the data will be loaded.

## Parameters

<i>graph</i>	[in] <a href="#">Graph</a> .
--------------	------------------------------

5.26.3.9 `def sparksee.EdgeTypeLoader.set_head_attribute ( self, 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

<i>attr</i>	[in] Head <a href="#">Attribute</a>
-------------	-------------------------------------

5.26.3.10 `def sparksee.EdgeTypeLoader.set_head_mep ( self, mep )`

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

flag True if the nodes need to be created. False otherwise

## Parameters

<i>mep</i>	null
------------	------

5.26.3.11 `def sparksee.EdgeTypeLoader.set_head_position ( self, 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

<i>pos</i>	[in] Head position
------------	--------------------

5.26.3.12 `def sparksee.EdgeTypeLoader.set_locale ( self, locale_str )`

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

It should match the locale used in the rowreader.

## Parameters

<i>locale_str</i>	[in] The locale string for the read data. See <a href="#">CSVReader</a> .
-------------------	---

5.26.3.13 `def sparksee.EdgeTypeLoader.set_log_error ( self, 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

<i>path</i>	[in] The path to the error log file.
-------------	--------------------------------------

#### Exceptions

<i>IOException</i>	If bad things happen opening the file.
--------------------	--

#### 5.26.3.14 def sparksee.EdgeTypeLoader.set\_log\_off ( self )

Turns 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.

#### 5.26.3.15 def sparksee.EdgeTypeLoader.set\_row\_reader ( self, rr )

Sets the input data source.

#### Parameters

<i>rr</i>	[in] Input <a href="#">RowReader</a> .
-----------	--

#### 5.26.3.16 def sparksee.EdgeTypeLoader.set\_tail\_attribute ( self, 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

<i>attr</i>	[in] Tail <a href="#">Attribute</a>
-------------	-------------------------------------

#### 5.26.3.17 def sparksee.EdgeTypeLoader.set\_tail\_mep ( self, 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

<i>mep</i>	null
------------	------

5.26.3.18 `def sparksee.EdgeTypeLoader.set_tail_position ( self, 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

<code>pos</code>	[in] Tail position
------------------	--------------------

5.26.3.19 `def sparksee.EdgeTypeLoader.set_timestamp_format ( self, timestamp_format )`

Sets a specific timestamp format.

Parameters

<code>timestamp_format</code>	[in] A string with the timestamp format definition.
-------------------------------	---

5.26.3.20 `def sparksee.EdgeTypeLoader.set_type ( self, type )`

Sets the type to be loaded.

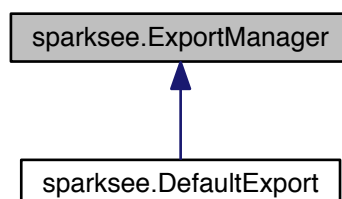
Parameters

<code>type</code>	[in] <a href="#">Type</a> identifier.
-------------------	---------------------------------------

## 5.27 sparksee.ExportManager Class Reference

Defines how to export a graph to an external format.

Inheritance diagram for `sparksee.ExportManager`:



## Public Member Functions

- def [get\\_edge](#) (self, edge, edge\_export)  
*Gets the edge export definition for the given edge.*
- def [release](#) (self)  
*Ends the export process.*
- def [get\\_graph](#) (self, graph\_export)  
*Gets the graph export definition.*
- def [get\\_node](#) (self, node, node\_export)  
*Gets the node export definition for the given node.*
- def [get\\_node\\_type](#) (self, type, node\_export)  
*Gets the default node export definition for the given node type.*
- def [enable\\_type](#) (self, type)  
*Gets whether a node or edge type must be exported or not.*
- def [prepare](#) (self, graph)  
*Prepares the graph for the export process.*
- def [get\\_edge\\_type](#) (self, type, edge\_export)  
*Gets the default node export definition for the given edge type.*

## 5.27.1 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 fortformats. Nowadays, available formats are defined in the [ExportType](#) enum.

## Author

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

## 5.27.2 Member Function Documentation

## 5.27.2.1 def sparksee.ExportManager.enable\_type ( self, type )

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

## Parameters

<i>type</i>	Node or edge type identifier.
-------------	-------------------------------

## Returns

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

## 5.27.2.2 def sparksee.ExportManager.get\_edge ( self, edge, edge\_export )

Gets the edge export definition for the given edge.



## Parameters

<i>edge</i>	Edge identifier.
<i>edge_export</i>	[out] The <a href="#">EdgeExport</a> which defines how to export given edge.

## Returns

TRUE if the given [EdgeExport](#) has been updated, otherwise FALSE will be returned and the default [EdgeExport](#) for the type the edge belongs to will be used.

```
5.27.2.3 def sparksee.ExportManager.get_edge_type ( self, type, edge_export )
```

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

<i>type</i>	[in] Edge type identifier.
<i>edge_export</i>	[out] The <a href="#">EdgeExport</a> which defines how to export the edges of the given type.

## Returns

TRUE.

```
5.27.2.4 def sparksee.ExportManager.get_graph ( self, graph_export )
```

Gets the graph export definition.

## Parameters

<i>graph_export</i>	[out] The <a href="#">GraphExport</a> which defines how to export the graph.
---------------------	--

## Returns

TRUE.

```
5.27.2.5 def sparksee.ExportManager.get_node ( self, node, node_export )
```

Gets the node export definition for the given node.

## Parameters

<i>node</i>	Node identifier.
<i>node_export</i>	[out] The <a href="#">NodeExport</a> which defines how to export given node.

**Returns**

TRUE if the given [NodeExport](#) has been updated, otherwise FALSE will be returned and the default [NodeExport](#) for the type the node belongs to will be used.

5.27.2.6 `def sparksee.ExportManager.get_node_type ( self, type, node_export )`

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**

<i>type</i>	[in] Node type identifier.
<i>node_export</i>	[out] The <a href="#">NodeExport</a> which defines how to export the nodes of the given type.

**Returns**

TRUE.

5.27.2.7 `def sparksee.ExportManager.prepare ( self, graph )`

Prepares the graph for the export process.

It is called once before the export process.

**Parameters**

<i>graph</i>	<a href="#">Graph</a> to be exported.
--------------	---------------------------------------

5.27.2.8 `def sparksee.ExportManager.release ( self )`

Ends the export process.

It is called once after the export process.

**5.28 sparksee.ExportType Class Reference**

Export type.

**Static Public Attributes**

- int [GRAPHVIZ](#) = 0  
*Export to Graphviz format.*
- int [GRAPHML](#) = 1  
*Export to GraphML format.*
- int [YGRAPHML](#) = 2  
*Export to YGRAPHML format.*

### 5.28.1 Detailed Description

Export type.

#### Author

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

### 5.28.2 Member Data Documentation

#### 5.28.2.1 `int sparksee.ExportType.GRAPHML = 1` [static]

Export to GraphML format.

GraphML home page: <http://graphml.graphdrawing.org/>

#### 5.28.2.2 `int sparksee.ExportType.GRAPHVIZ = 0` [static]

Export to Graphviz format.

Graphviz home page: <http://www.graphviz.org>

#### 5.28.2.3 `int sparksee.ExportType.YGRAPHML = 2` [static]

Export to YGRAPHML format.

It is an GraphML format extended with a set of yWorks ("http://www.yworks.com") extensions. Thus, it allows for the visualization of the exported graph with the public yEd visualization tool ("http://www.yworks.com/products/yed").

## 5.29 sparksee.Graph Class Reference

[Graph](#) class.

### Public Member Functions

- def [find\\_or\\_create\\_edge](#) (self, etype, tail, head)  
*Gets any of the edges of the specified type between two given nodes (tail and head).*
- def [find\\_edge\\_types](#) (self)  
*Gets all existing [Sparksee](#) edge type identifiers.*
- def [get\\_attribute\\_interval\\_count](#) (self, attr, lower, include\_lower, higher, include\_higher)  
*Gets how many objects have a value into the given range for the given attribute.*
- def [neighbors](#) (self, objs, etype, dir)  
*Selects all neighbor nodes from or to each of the node OID in the given collection and for the given edge type.*
- def [backup](#) (self, file)  
*Dumps all the data to a backup file.*
- def [remove\\_attribute](#) (self, attr)  
*Removes the given attribute.*
- def [remove\\_type](#) (self, type)  
*Removes the given type.*

- def [degree](#) (self, oid, etype, dir)  
*Gets the number of edges from or to the given node OID and for the given edge type.*
- def [find\\_types](#) (self)  
*Gets all existing [Sparksee](#) node and edge type identifiers.*
- def [new\\_array\\_attribute](#) (self, type, name, dt, size)  
*Creates a new array attribute.*
- def [find\\_type](#) (self, name)  
*Gets the [Sparksee](#) type identifier for the given type name.*
- def [top\\_k](#) (self, attribute, operation, lower, order, k)  
*Gets a [KeyValues](#) iterator as a result of the TopK operation.*
- def [get\\_edge\\_peer](#) (self, edge, node)  
*Gets the other end for the given edge.*
- def [set\\_attribute\\_text](#) (self, oid, attr, tstream)  
*Sets the writable [TextStream](#) for the given text attribute and OID.*
- def [set\\_attribute](#) (self, attr, value)  
*Sets the value of the given attribute for all the objects of the types the attribute applies to.*
- def [index\\_neighbors](#) (self, edge\_type, neighbors)  
*Creates or destroys the neighbors index of an edge type.*
- def [get\\_object\\_type](#) (self, oid)  
*Gets the [Sparksee](#) node or edge type identifier for the given OID.*
- def [index\\_attribute](#) (self, attr, kind)  
*Updates the index of the given attribute.*
- def [set\\_array\\_attribute](#) (self, attr, value)  
*Sets all the values of the array of the given array attribute for all the objects of the types the attribute applies to.*
- def [top\\_k](#) (self, attribute, order, k)  
*Gets a [KeyValues](#) iterator as a result of the TopK operation.*
- def [get\\_attribute](#) (self, oid, attr, value)  
*Gets the [Value](#) for the given attribute and OID.*
- def [get\\_attribute\\_statistics](#) (self, attr, basic)  
*Gets statistics from the given attribute.*
- def [select](#) (self, attr, cond, lower, higher, restriction)  
*Selects all OIDs satisfying the given condition for the given attribute.*
- def [count\\_nodes](#) (self)  
*Gets the number of nodes.*
- def [set\\_attribute\\_default\\_value](#) (self, attr, value)  
*Sets a default value for an attribute.*
- def [find\\_object](#) (self, attr, value)  
*Finds one object having the given [Value](#) for the given attribute.*
- def [neighbors](#) (self, oid, etype, dir)  
*Selects all neighbor nodes from or to the given node OID and for the given edge type.*
- def [new\\_session\\_attribute](#) (self, type, dt, kind)  
*Creates a new [Session](#) attribute.*
- def [encrypted\\_backup](#) (self, file, key\_in\_hex, iv\_in\_hex)  
*Dumps all the data to a backup file.*
- def [drop](#) (self, oid)  
*Drops the given OID.*
- def [set\\_array\\_attribute\\_void](#) (self, oid, attr, value)  
*Sets all the elements of the [ValueArray](#) for the given array attribute and OID.*
- def [edges](#) (self, etype, tail, head)  
*Gets all the edges of the given type between two given nodes (tail and head).*
- def [set\\_attribute](#) (self, oid, attr, value)

- Sets the [Value](#) for the given attribute and OID.*
- def [find\\_edge](#) (self, etype, tail, head)
  - Gets any of the edges of the given type between two given nodes (tail and head).*
- def [tails](#) (self, [edges](#))
  - Gets all the tails from the given edges collection.*
- def [select](#) (self, attr, cond, lower, higher)
  - Selects all OIDs satisfying the given condition for the given attribute.*
- def [get\\_values](#) (self, attr)
  - Gets the [Value](#) collection for the given attribute.*
- def [export](#) (self, file, type, em)
  - Exports the [Graph](#).*
- def [new\\_restricted\\_edge\\_type](#) (self, name, tail, head, [neighbors](#))
  - Creates a new restricted edge type.*
- def [get\\_attribute\\_text](#) (self, oid, attr)
  - Gets the read-only [TextStream](#) for the given text attribute and OID.*
- def [explode](#) (self, objs, etype, dir)
  - Selects all edges from or to each of the node OID in the given collection and for the given edge type.*
- def [new\\_edge\\_type](#) (self, name, directed, [neighbors](#))
  - Creates a new edge type.*
- def [top\\_k](#) (self, attribute, operation, lower, higher, order, k)
  - Gets a [KeyValues](#) iterator as a result of the TopK operation.*
- def [dump\\_data](#) (self, file)
  - Dumps logical data to a file.*
- def [set\\_array\\_attribute](#) (self, oid, attr, value)
  - Sets all the elements of the [ValueArray](#) for the given array attribute and OID and returns it.*
- def [top\\_k](#) (self, attribute, operation, lower, higher, order, k, restriction)
  - Gets a [KeyValues](#) iterator as a result of the TopK operation.*
- def [find\\_attribute](#) (self, type, name)
  - Gets the [Sparksee](#) attribute identifier for the given type identifier and attribute name.*
- def [new\\_node](#) (self, type)
  - Creates a new node instance.*
- def [find\\_node\\_types](#) (self)
  - Gets all existing [Sparksee](#) node type identifiers.*
- def [new\\_attribute](#) (self, type, name, dt, kind)
  - Creates a new attribute.*
- def [top\\_k](#) (self, attribute, operation, lower, order, k, restriction)
  - Gets a [KeyValues](#) iterator as a result of the TopK operation.*
- def [new\\_session\\_array\\_attribute](#) (self, type, dt, size)
  - Creates a new [Session](#) array attribute.*
- def [new\\_edge](#) (self, type, tail\_attr, tail\_v, head\_attr, head\_v)
  - Creates a new edge instance.*
- def [get\\_attribute](#) (self, attr)
  - Gets information about the given attribute.*
- def [new\\_node\\_type](#) (self, name)
  - Creates a new node type.*
- def [dump\\_storage](#) (self, file)
  - Dumps internal storage data to a file.*
- def [select](#) (self, type)
  - Selects all OIDs belonging to the given type.*
- def [select](#) (self, attr, cond, value, restriction)
  - Selects all OIDs satisfying the given condition for the given attribute.*

- def [select](#) (self, attr, cond, value)  
*Selects all OIDs satisfying the given condition for the given attribute.*
- def [rename\\_attribute](#) (self, attr, new\_name)  
*Renames an attribute.*
- def [top\\_k](#) (self, attribute, order, k, restriction)  
*Gets a [KeyValues](#) iterator as a result of the TopK operation.*
- def [count\\_edges](#) (self)  
*Gets the number of edges.*
- def [rename\\_type](#) (self, type, new\_name)  
*Renames a type.*
- def [tails\\_and\\_heads](#) (self, edges, tails, heads)  
*Gets all the tails and heads from the given edges collection.*
- def [new\\_session\\_attribute](#) (self, type, dt, kind, default\_value)  
*Creates a new [Session](#) attribute with a default value.*
- def [heads](#) (self, edges)  
*Gets all the heads from the given edges collection.*
- def [new\\_attribute](#) (self, type, name, dt, kind, default\_value)  
*Creates a new attribute with a default value.*
- def [rename\\_type](#) (self, old\_name, new\_name)  
*Renames a type.*
- def [explode](#) (self, oid, etype, dir)  
*Selects all edges from or to the given node OID and for the given edge type.*
- def [get\\_attributes](#) (self, oid)  
*Gets all [Sparksee](#) attribute identifiers with a non-NULL value for the given [Sparksee](#) OID.*
- def [find\\_attributes](#) (self, type)  
*Gets all existing [Sparksee](#) attribute identifiers for the given type identifier.*
- def [get\\_array\\_attribute](#) (self, oid, attr)  
*Gets the [ValueArray](#) for the given array attribute and OID or NULL if it does not exist.*
- def [get\\_attribute](#) (self, oid, attr)  
*Gets the [Value](#) for the given attribute and OID.*
- def [find\\_or\\_create\\_object](#) (self, attr, value)  
*Finds one object having the given [Value](#) for the attribute or it creates one does not exist any.*
- def [get\\_edge\\_data](#) (self, edge)  
*Gets information about an edge.*
- def [drop](#) (self, objs)  
*Drops all the OIDs from the given collection.*
- def [new\\_edge](#) (self, type, tail, head)  
*Creates a new edge instance.*
- def [get\\_type](#) (self, type)  
*Gets information about the given type.*

### 5.29.1 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>

## 5.29.2 Member Function Documentation

### 5.29.2.1 `def sparksee.Graph.backup ( self, file )`

Dumps all the data to a backup file.

See the [Sparksee](#) class Restore methods.

#### Parameters

<i>file</i>	[in] Output backup file path.
-------------	-------------------------------

#### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file cannot be created.

### 5.29.2.2 `def sparksee.Graph.count_edges ( self )`

Gets the number of edges.

#### Returns

The number of edges.

### 5.29.2.3 `def sparksee.Graph.count_nodes ( self )`

Gets the number of nodes.

#### Returns

The number of nodes.

### 5.29.2.4 `def sparksee.Graph.degree ( self, oid, etype, dir )`

Gets the number of edges from or to the given node OID and for the given edge type.

#### Parameters

<i>oid</i>	[in] <a href="#">Sparksee</a> node OID.
<i>etype</i>	[in] <a href="#">Sparksee</a> edge type identifier.
<i>dir</i>	[in] Direction.

#### Returns

The number of edges.

## 5.29.2.5 def sparksee.Graph.drop ( self, oid )

Drops the given OID.

It also removes its edges as well as its attribute values.

## Parameters

<i>oid</i>	[in] Sparksee OID to be removed.
------------	----------------------------------

Referenced by sparksee.Graph.drop().

## 5.29.2.6 def sparksee.Graph.drop ( self, 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

<i>objs</i>	[in] Objects collection with the OIDs to be removed.
-------------	--

References sparksee.Graph.drop().

## 5.29.2.7 def sparksee.Graph.dump\_data ( self, file )

Dumps logical data to a file.

## Parameters

<i>file</i>	[in] Output file path.
-------------	------------------------

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file cannot be created.

## 5.29.2.8 def sparksee.Graph.dump\_storage ( self, file )

Dumps internal storage data to a file.

## Parameters

<i>file</i>	[in] Output file path.
-------------	------------------------

## Exceptions

<i>RuntimeError</i>	null
---------------------	------



## Exceptions

<i>IOError</i>	If the given file cannot be created.
----------------	--------------------------------------

5.29.2.9 `def sparksee.Graph.edges ( self, etype, tail, head )`

Gets all the edges of the given type between two given nodes (tail and head).

## Parameters

<i>etype</i>	[in] <a href="#">Sparksee</a> edge type identifier.
<i>tail</i>	[in] Tail node identifier.
<i>head</i>	[in] Head node identifier.

## Returns

[Objects](#) instance.

5.29.2.10 `def sparksee.Graph.encrypted_backup ( self, file, key_in_hex, iv_in_hex )`

Dumps all the data to a backup file.

See the [Sparksee](#) class `RestoreEncryptedBackup` methods.

## Parameters

<i>file</i>	[in] Output backup file path.
<i>key_in_hex</i>	[In] The AES encryption Key as a hexadecimal string (8, 16 or 32 bytes).
<i>iv_in_hex</i>	[In] The AES Initialization Vector as a hexadecimal string (16 bytes).

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file cannot be created.

5.29.2.11 `def sparksee.Graph.explode ( self, objs, etype, dir )`

Selects all edges from or to each of the node OID in the given collection and for the given edge type.

## Parameters

<i>objs</i>	[in] <a href="#">Sparksee</a> node OID collection.
<i>etype</i>	[in] <a href="#">Sparksee</a> edge type identifier.
<i>dir</i>	[in] Direction.

**Returns**

[Objects](#) instance.

Referenced by `sparksee.Graph.explode()`.

5.29.2.12 `def sparksee.Graph.explode ( self, oid, etype, dir )`

Selects all edges from or to the given node OID and for the given edge type.

**Parameters**

<i>oid</i>	[in] <a href="#">Sparksee</a> node OID.
<i>etype</i>	[in] <a href="#">Sparksee</a> edge type identifier.
<i>dir</i>	[in] Direction.

**Returns**

[Objects](#) instance.

References `sparksee.Graph.explode()`.

5.29.2.13 `def sparksee.Graph.export ( self, file, type, em )`

Exports the [Graph](#).

**Parameters**

<i>file</i>	[in] Output file.
<i>type</i>	[in] Export type.
<i>em</i>	[in] Defines how to do the export for each graph object.

**Exceptions**

<i>IOError</i>	null
----------------	------

5.29.2.14 `def sparksee.Graph.find_attribute ( self, type, name )`

Gets the [Sparksee](#) attribute identifier for the given type identifier and attribute name.

**Parameters**

<i>type</i>	[in] <a href="#">Sparksee</a> type identifier.
<i>name</i>	[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.

5.29.2.15 `def sparksee.Graph.find_attributes ( self, type )`

Gets all existing [Sparksee](#) attribute identifiers for the given type identifier.

Parameters

<i>type</i>	[in] <a href="#">Sparksee</a> type identifier.
-------------	--

Returns

[Sparksee](#) attribute identifier list.

5.29.2.16 `def sparksee.Graph.find_edge ( self, etype, tail, 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

<i>etype</i>	[in] <a href="#">Sparksee</a> edge type identifier.
<i>tail</i>	[in] Tail node identifier.
<i>head</i>	[in] Head node identifier.

Returns

Any of the edges or the [Objects](#) InvalidOID.

5.29.2.17 `def sparksee.Graph.find_edge_types ( self )`

Gets all existing [Sparksee](#) edge type identifiers.

Returns

[Sparksee](#) edge type identifier list.

5.29.2.18 `def sparksee.Graph.find_node_types ( self )`

Gets all existing [Sparksee](#) node type identifiers.

Returns

[Sparksee](#) node type identifier list.

5.29.2.19 `def sparksee.Graph.find_object ( self, attr, 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

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>value</i>	[in] <a href="#">Value</a> .

## Returns

[Sparksee](#) OID or the [Objects](#) InvalidOID.

5.29.2.20 `def sparksee.Graph.find_or_create_edge ( self, etype, tail, 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

<i>etype</i>	[in] <a href="#">Sparksee</a> edge type identifier.
<i>tail</i>	[in] Tail node identifier.
<i>head</i>	[in] Head node identifier.

## Returns

Any of the edges or the [Objects](#) InvalidOID.

5.29.2.21 `def sparksee.Graph.find_or_create_object ( self, attr, 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

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>value</i>	[in] <a href="#">Value</a> .

## Returns

[Sparksee](#) OID or the [Objects](#) InvalidOID.

5.29.2.22 `def sparksee.Graph.find_type ( self, name )`

Gets the [Sparksee](#) type identifier for the given type name.

## Parameters

<i>name</i>	[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.

5.29.2.23 `def sparksee.Graph.find_types ( self )`

Gets all existing [Sparksee](#) node and edge type identifiers.

## Returns

[Sparksee](#) node and edge type identifier list.

5.29.2.24 `def sparksee.Graph.get_array_attribute ( self, oid, attr )`

Gets the [ValueArray](#) for the given array attribute and OID or NULL if it does not exist.

## Parameters

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
<i>attr</i>	[in] <a href="#">Sparksee</a> array attribute identifier.

## Returns

A [ValueArray](#). This returns a [ValueArray](#), a NULL or throws an exception.

5.29.2.25 `def sparksee.Graph.get_attribute ( self, oid, attr, value )`

Gets the [Value](#) for the given attribute and OID.

## Parameters

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>value</i>	[in out] <a href="#">Value</a> for the given attribute and for the given OID.

Referenced by `sparksee.Graph.get_attribute()`.

5.29.2.26 `def sparksee.Graph.get_attribute ( self, attr )`

Gets information about the given attribute.

## Parameters

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
-------------	---

## Returns

The [Attribute](#) for the given [Sparksee](#) attribute identifier.

References `sparksee.Graph.get_attribute()`.

5.29.2.27 `def sparksee.Graph.get_attribute ( self, oid, 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

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.

## Returns

A new [Value](#) instance having the attribute value for the given OID.

References `sparksee.Graph.get_attribute()`.

5.29.2.28 `def sparksee.Graph.get_attribute_interval_count ( self, attr, lower, include_lower, higher, include_higher )`

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

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>lower</i>	[in] Lower bound <a href="#">Value</a> of the range.
<i>include_lower</i>	[in] If TRUE, include lower bound <a href="#">Value</a> of the range.
<i>higher</i>	[in] Higher bound <a href="#">Value</a> of the range.
<i>include_higher</i>	[in] If TRUE, include higher bound <a href="#">Value</a> of the range.

**Returns**

Number of objects having a value into the given range.

5.29.2.29 `def sparksee.Graph.get_attribute_statistics ( self, attr, basic )`

Gets statistics from the given attribute.

The statistics can only be obtained from Indexed or Unique attributes.

**Parameters**

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>basic</i>	[in] If FALSE all statistics are computed, if TRUE just those statistics marked as basic will be computed (see description of the <a href="#">AttributeStatistics</a> class). Of course, computing just basic statistics will be faster than computing all of them.

**Returns**

An [AttributeStatistics](#) instance.

5.29.2.30 `def sparksee.Graph.get_attribute_text ( self, oid, attr )`

Gets the read-only [TextStream](#) for the given text attribute and OID.

**Parameters**

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.

**Returns**

A [TextStream](#). This returns a [TextStream](#) to read.

5.29.2.31 `def sparksee.Graph.get_attributes ( self, oid )`

Gets all [Sparksee](#) attribute identifiers with a non-NULL value for the given [Sparksee](#) OID.

**Parameters**

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
------------	------------------------------------

**Returns**

[Sparksee](#) attribute identifier list.

5.29.2.32 `def sparksee.Graph.get_edge_data ( self, edge )`

Gets information about an edge.

## Parameters

<i>edge</i>	[in] <a href="#">Sparksee</a> edge identifier.
-------------	--

## Returns

An [EdgeData](#) instance.

```
5.29.2.33 def sparksee.Graph.get_edge_peer ( self, edge, node )
```

Gets the other end for the given edge.

## Parameters

<i>edge</i>	[in] <a href="#">Sparksee</a> edge identifier.
<i>node</i>	[in] <a href="#">Sparksee</a> node identifier. It must be one of the ends of the edge.

## Returns

The other end of the edge.

```
5.29.2.34 def sparksee.Graph.get_object_type ( self, oid )
```

Gets the [Sparksee](#) node or edge type identifier for the given OID.

## Parameters

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
------------	------------------------------------

## Returns

[Sparksee](#) node or edge type identifier.

```
5.29.2.35 def sparksee.Graph.get_type ( self, type )
```

Gets information about the given type.

## Parameters

<i>type</i>	[in] <a href="#">Sparksee</a> type identifier.
-------------	--

## Returns

The [Type](#) for the given [Sparksee](#) type identifier.

```
5.29.2.36 def sparksee.Graph.get_values ( self, attr )
```

Gets the [Value](#) collection for the given attribute.



## Parameters

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
-------------	---

## Returns

Returns a [Values](#) object.

5.29.2.37 `def sparksee.Graph.heads ( self, edges )`

Gets all the heads from the given edges collection.

## Parameters

<i>edges</i>	[in] <a href="#">Sparksee</a> edge identifier collection.
--------------	---

## Returns

The heads collection.

5.29.2.38 `def sparksee.Graph.index_attribute ( self, attr, 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

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>kind</i>	[in] <a href="#">Attribute</a> kind.

5.29.2.39 `def sparksee.Graph.index_neighbors ( self, edge_type, neighbors )`

Creates or destroys the neighbors index of an edge type.

## Parameters

<i>edge_type</i>	[in] <a href="#">Sparksee</a> Edge type identifier.
<i>neighbors</i>	[in] If TRUE, it indexes the neighbor nodes, otherwise it removes the index.

5.29.2.40 `def sparksee.Graph.neighbors ( self, objs, etype, dir )`

Selects all neighbor nodes from or to each of the node OID in the given collection and for the given edge type.

## Parameters

<i>objs</i>	[in] <a href="#">Sparksee</a> node OID collection.
-------------	--

## Parameters

<i>etype</i>	[in] <a href="#">Sparksee</a> edge type identifier.
<i>dir</i>	[in] Direction.

## Returns

[Objects](#) instance.

Referenced by `sparksee.Graph.neighbors()`.

5.29.2.41 `def sparksee.Graph.neighbors ( self, oid, etype, dir )`

Selects all neighbor nodes from or to the given node OID and for the given edge type.

## Parameters

<i>oid</i>	[in] <a href="#">Sparksee</a> node OID.
<i>etype</i>	[in] <a href="#">Sparksee</a> edge type identifier.
<i>dir</i>	[in] Direction.

## Returns

[Objects](#) instance.

References `sparksee.Graph.neighbors()`.

5.29.2.42 `def sparksee.Graph.new_array_attribute ( self, type, name, dt, size )`

Creates a new array attribute.

## Parameters

<i>type</i>	[in] <a href="#">Sparksee</a> node or edge type identifier.
<i>name</i>	[in] Unique name for the new attribute.
<i>dt</i>	[in] Base Data type for the new array attribute elements.
<i>size</i>	[in] The array size.

## Returns

Unique [Sparksee](#) attribute identifier.

5.29.2.43 `def sparksee.Graph.new_attribute ( self, type, name, dt, kind )`

Creates a new attribute.

## Parameters

<i>type</i>	[in] <a href="#">Sparksee</a> node or edge type identifier.
<i>name</i>	[in] Unique name for the new attribute.
<i>dt</i>	[in] Data type for the new attribute.
<i>kind</i>	[in] <a href="#">Attribute</a> kind.

## Returns

Unique [Sparksee](#) attribute identifier.

Referenced by `sparksee.Graph.new_attribute()`.

```
5.29.2.44 def sparksee.Graph.new_attribute ( self, type, name, dt, kind, default_value )
```

Creates a new attribute with a default value.

## Parameters

<i>type</i>	[in] <a href="#">Sparksee</a> node or edge type identifier.
<i>name</i>	[in] Unique name for the new attribute.
<i>dt</i>	[in] Data type for the new attribute.
<i>kind</i>	[in] <a href="#">Attribute</a> kind.
<i>default_value</i>	[in] The default value to use in each new node/edge.

## Returns

Unique [Sparksee](#) attribute identifier.

References `sparksee.Graph.new_attribute()`.

```
5.29.2.45 def sparksee.Graph.new_edge ( self, type, tail_attr, tail_v, head_attr, head_v )
```

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

<i>type</i>	[in] <a href="#">Sparksee</a> type identifier.
<i>tail_attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>tail_v</i>	[in] <a href="#">Value</a> .
<i>head_attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>head_v</i>	[in] <a href="#">Value</a> .

**Returns**

Unique OID of the new edge instance.

Referenced by `sparksee.Graph.new_edge()`.

5.29.2.46 `def sparksee.Graph.new_edge ( self, type, tail, head )`

Creates a new edge instance.

**Parameters**

<i>type</i>	[in] <a href="#">Sparksee</a> type identifier.
<i>tail</i>	[in] Source <a href="#">Sparksee</a> OID.
<i>head</i>	[in] Target <a href="#">Sparksee</a> OID.

**Returns**

Unique OID of the new edge instance.

References `sparksee.Graph.new_edge()`.

5.29.2.47 `def sparksee.Graph.new_edge_type ( self, name, directed, neighbors )`

Creates a new edge type.

**Parameters**

<i>name</i>	[in] Unique name for the new edge type.
<i>directed</i>	[in] If TRUE, this creates a directed edge type, otherwise this creates a undirected edge type.
<i>neighbors</i>	[in] If TRUE, this indexes neighbor nodes, otherwise not.

**Returns**

Unique [Sparksee](#) type identifier.

5.29.2.48 `def sparksee.Graph.new_node ( self, type )`

Creates a new node instance.

**Parameters**

<i>type</i>	[in] <a href="#">Sparksee</a> type identifier.
-------------	--

**Returns**

Unique OID of the new node instance.

5.29.2.49 `def sparksee.Graph.new_node_type ( self, name )`

Creates a new node type.

#### Parameters

<i>name</i>	[in] Unique name for the new node type.
-------------	---

#### Returns

Unique [Sparksee](#) type identifier.

5.29.2.50 `def sparksee.Graph.new_restricted_edge_type ( self, name, tail, head, neighbors )`

Creates a new restricted edge type.

#### Parameters

<i>name</i>	[in] Unique name for the new edge type.
<i>tail</i>	[in] Tail <a href="#">Sparksee</a> node type identifier.
<i>head</i>	[in] Head <a href="#">Sparksee</a> node type identifier.
<i>neighbors</i>	[in] If TRUE, this indexes neighbor nodes, otherwise not.

#### Returns

Unique [Sparksee](#) type identifier.

5.29.2.51 `def sparksee.Graph.new_session_array_attribute ( self, type, dt, 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

<i>type</i>	[in] <a href="#">Sparksee</a> node or edge type identifier.
<i>dt</i>	[in] Base Data type for the new array attribute elements.
<i>size</i>	[in] The array size.

#### Returns

Unique [Sparksee](#) attribute identifier.

5.29.2.52 `def sparksee.Graph.new_session_attribute ( self, type, dt, 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

<i>type</i>	[in] <a href="#">Sparksee</a> node or edge type identifier.
<i>dt</i>	[in] Data type for the new attribute.
<i>kind</i>	[in] <a href="#">Attribute</a> kind.

#### Returns

Unique [Sparksee](#) attribute identifier.

Referenced by `sparksee.Graph.new_session_attribute()`.

```
5.29.2.53 def sparksee.Graph.new_session_attribute ( self, type, dt, kind, default_value )
```

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

<i>type</i>	[in] <a href="#">Sparksee</a> node or edge type identifier.
<i>dt</i>	[in] Data type for the new attribute.
<i>kind</i>	[in] <a href="#">Attribute</a> kind.
<i>default_value</i>	[in] The default value to use in each new node/edge.

#### Returns

Unique [Sparksee](#) attribute identifier.

References `sparksee.Graph.new_session_attribute()`.

```
5.29.2.54 def sparksee.Graph.remove_attribute ( self, attr )
```

Removes the given attribute.

#### Parameters

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
-------------	---

```
5.29.2.55 def sparksee.Graph.remove_type ( self, 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

<i>type</i>	[in] <a href="#">Sparksee</a> type identifier.
-------------	--

5.29.2.56 `def sparksee.Graph.rename_attribute ( self, attr, new_name )`

Renames an attribute.

The new name must be available.

#### Parameters

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>new_name</i>	[in] The new name for the attribute.

5.29.2.57 `def sparksee.Graph.rename_type ( self, type, new_name )`

Renames a type.

The new name must be available.

#### Parameters

<i>type</i>	[in] The type to be renamed.
<i>new_name</i>	[in] The new name for the type.

Referenced by `sparksee.Graph.rename_type()`.

5.29.2.58 `def sparksee.Graph.rename_type ( self, old_name, new_name )`

Renames a type.

The new name must be available.

#### Parameters

<i>old_name</i>	[in] The current name of the type to be renamed.
<i>new_name</i>	[in] The new name for the type.

References `sparksee.Graph.rename_type()`.

5.29.2.59 `def sparksee.Graph.select ( self, attr, cond, lower, higher, 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

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>cond</i>	[in] <a href="#">Condition</a> to be satisfied. It must be the <a href="#">Between Condition</a> .
<i>lower</i>	[in] Lower-bound <a href="#">Value</a> to be satisfied.
<i>higher</i>	[in] Higher-bound <a href="#">Value</a> to be satisfied.
<i>restriction</i>	[in] <a href="#">Objects</a> to limit the select in this set of objects.

## Returns

[Objects](#) instance.

Referenced by `sparksee.Graph.select()`.

5.29.2.60 `def sparksee.Graph.select ( self, attr, cond, lower, 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

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>cond</i>	[in] <a href="#">Condition</a> to be satisfied. It must be the <a href="#">Between Condition</a> .
<i>lower</i>	[in] Lower-bound <a href="#">Value</a> to be satisfied.
<i>higher</i>	[in] Higher-bound <a href="#">Value</a> to be satisfied.

## Returns

[Objects](#) instance.

References `sparksee.Graph.select()`.

5.29.2.61 `def sparksee.Graph.select ( self, type )`

Selects all OIDs belonging to the given type.

## Parameters

<i>type</i>	[in] <a href="#">Sparksee</a> type identifier.
-------------	--

## Returns

[Objects](#) instance.

References `sparksee.Graph.select()`.

5.29.2.62 `def sparksee.Graph.select ( self, attr, cond, value, restriction )`

Selects all OIDs satisfying the given condition for the given attribute.



## Parameters

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>cond</i>	[in] <a href="#">Condition</a> to be satisfied.
<i>value</i>	[in] <a href="#">Value</a> to be satisfied.
<i>restriction</i>	[in] <a href="#">Objects</a> to limit the select in this set of objects.

## Returns

[Objects](#) instance.

References `sparksee.Graph.select()`.

```
5.29.2.63 def sparksee.Graph.select ( self, attr, cond, value )
```

Selects all OIDs satisfying the given condition for the given attribute.

## Parameters

<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>cond</i>	[in] <a href="#">Condition</a> to be satisfied.
<i>value</i>	[in] <a href="#">Value</a> to be satisfied.

## Returns

[Objects](#) instance.

References `sparksee.Graph.select()`.

```
5.29.2.64 def sparksee.Graph.set_array_attribute ( self, attr, value )
```

Sets all the values of the array of the given array attribute for all the objects of the types the attribute applies to.

## Parameters

<i>attr</i>	[in] <a href="#">Sparksee</a> array attribute identifier.
<i>value</i>	[in] <a href="#">Value</a> for all the array elements of the arrays.

Referenced by `sparksee.Graph.set_array_attribute()`.

```
5.29.2.65 def sparksee.Graph.set_array_attribute ( self, oid, attr, 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

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
<i>attr</i>	[in] <a href="#">Sparksee</a> array attribute identifier.
<i>value</i>	[in] <a href="#">Value</a> for all the array elements of the given attribute and OID.

## Returns

A [ValueArray](#). This returns a [ValueArray](#), a NULL or throws an exception.

References `sparksee.Graph.set_array_attribute()`.

```
5.29.2.66 def sparksee.Graph.set_array_attribute_void ( self, oid, attr, 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

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
<i>attr</i>	[in] <a href="#">Sparksee</a> array attribute identifier.
<i>value</i>	[in] <a href="#">Value</a> for all the array elements of the given attribute and OID.

```
5.29.2.67 def sparksee.Graph.set_attribute ( self, attr, value )
```

Sets the value of the given attribute for all the objects of the types the attribute applies to.

Does not work for TEXT attribute types

txThe Transaction this operation belongs to attributeThe attribute to initialize

## Parameters

<i>attr</i>	null
<i>value</i>	The value to initialize the attribute to

Referenced by `sparksee.Graph.set_attribute()`.

```
5.29.2.68 def sparksee.Graph.set_attribute ( self, oid, attr, value )
```

Sets the [Value](#) for the given attribute and OID.

## Parameters

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>value</i>	[in] <a href="#">Value</a> for the given attribute and for the given OID.

References `sparksee.Graph.set_attribute()`.

5.29.2.69 `def sparksee.Graph.set_attribute_default_value ( self, attr, 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

<i>attr</i>	[in] The attribute.
<i>value</i>	[in] The default value to use for this attribute.

5.29.2.70 `def sparksee.Graph.set_attribute_text ( self, oid, attr, tstream )`

Sets the writable [TextStream](#) for the given text attribute and OID.

#### Parameters

<i>oid</i>	[in] <a href="#">Sparksee</a> OID.
<i>attr</i>	[in] <a href="#">Sparksee</a> attribute identifier.
<i>tstream</i>	[in] New Text value. This corresponds to a <a href="#">TextStream</a> to write.

5.29.2.71 `def sparksee.Graph.tails ( self, edges )`

Gets all the tails from the given edges collection.

#### Parameters

<i>edges</i>	[in] <a href="#">Sparksee</a> edge identifier collection.
--------------	---

#### Returns

The tails collection.

5.29.2.72 `def sparksee.Graph.tails_and_heads ( self, edges, tails, heads )`

Gets all the tails and heads from the given edges collection.

#### Parameters

<i>edges</i>	[in] <a href="#">Sparksee</a> edge identifier collection.
<i>tails</i>	[in out] If not NULL, all the tails will be stored here.
<i>heads</i>	[in out] If not NULL, all the heads will be stored here.

5.29.2.73 `def sparksee.Graph.top_k( self, attribute, operation, lower, order, k )`

Gets a [KeyValues](#) iterator as a result of the TopK operation.

#### Parameters

<i>attribute</i>	The attribute to perform the TopK on
<i>operation</i>	The operation to perform in the top k
<i>lower</i>	The lower bound <a href="#">Value</a> to be satisfied
<i>order</i>	The ordering semantics of the top-k
<i>k</i>	The size of the TopK

#### Returns

A [KeyValues](#) instance

Referenced by `sparksee.Graph.top_k()`.

5.29.2.74 `def sparksee.Graph.top_k( self, attribute, order, k )`

Gets a [KeyValues](#) iterator as a result of the TopK operation.

#### Parameters

<i>attribute</i>	The attribute to perform the TopK on
<i>order</i>	The ordering semantics of the top-k
<i>k</i>	The size of the TopK

#### Returns

A [KeyValues](#) instance

References `sparksee.Graph.top_k()`.

5.29.2.75 `def sparksee.Graph.top_k( self, attribute, operation, lower, higher, order, k )`

Gets a [KeyValues](#) iterator as a result of the TopK operation.

#### Parameters

<i>attribute</i>	The attribute to perform the TopK on
<i>operation</i>	The operation to perform in the top k
<i>lower</i>	The lower bound <a href="#">Value</a> to be satisfied
<i>higher</i>	The upper bound <a href="#">Value</a> to be satisfied
<i>order</i>	The ordering semantics of the top-k
<i>k</i>	The size of the TopK

**Returns**

A [KeyValues](#) instance

References `sparksee.Graph.top_k()`.

```
5.29.2.76 def sparksee.Graph.top_k( self, attribute, operation, lower, higher, order, k, restriction )
```

Gets a [KeyValues](#) iterator as a result of the TopK operation.

`restrictonObjects` to limit the topk to this set of objects

**Parameters**

<i>attribute</i>	The attribute to perform the TopK on
<i>operation</i>	The operation to perform in the top k
<i>lower</i>	The lower bound <a href="#">Value</a> to be satisfied
<i>higher</i>	The upper bound <a href="#">Value</a> to be satisfied
<i>order</i>	The ordering semantincs of the top-k
<i>k</i>	The size of the TopK
<i>restriction</i>	null

**Returns**

A [KeyValues](#) instance

References `sparksee.Graph.top_k()`.

```
5.29.2.77 def sparksee.Graph.top_k( self, attribute, operation, lower, order, k, restriction )
```

Gets a [KeyValues](#) iterator as a result of the TopK operation.

`restrictonObjects` to limit the topk to this set of objects

**Parameters**

<i>attribute</i>	The attribute to perform the TopK on
<i>operation</i>	The operation to perform in the top k
<i>lower</i>	The lower bound <a href="#">Value</a> to be satisfied
<i>order</i>	The ordering semantincs of the top-k
<i>k</i>	The size of the TopK
<i>restriction</i>	null

**Returns**

A [KeyValues](#) instance

References `sparksee.Graph.top_k()`.

5.29.2.78 `def sparksee.Graph.top_k( self, attribute, order, k, restriction )`

Gets a [KeyValues](#) iterator as a result of the TopK operation.

restrictionObjects to limit the topk to this set of objects

#### Parameters

<i>attribute</i>	The attribute to perform the TopK on
<i>order</i>	The ordering semantics of the top-k
<i>k</i>	The size of the TopK
<i>restriction</i>	null

#### Returns

A [KeyValues](#) instance

References `sparksee.Graph.top_k()`.

## 5.30 sparksee.GraphExport Class Reference

Stores the graph exporting values.

#### Public Member Functions

- `def set_defaults( self )`  
*Sets to default values.*
- `def set_label( self, label )`  
*Sets the graph label.*
- `def get_label( self )`  
*Gets the graph label.*
- `def __init__( self )`  
*Creates a new [GraphExport](#) instance.*

#### 5.30.1 Detailed Description

Stores the graph exporting values.

#### Author

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

#### 5.30.2 Member Function Documentation

5.30.2.1 `def sparksee.GraphExport.get_label( self )`

Gets the graph label.

#### Returns

The graph label.

5.30.2.2 `def sparksee.GraphExport.set_label( self, label )`

Sets the graph label.

## Parameters

<i>label</i>	[in] The graph label.
--------------	-----------------------

### 5.31 sparksee.Int32List Class Reference

[Sparksee](#) 32-bit signed integer list.

#### Public Member Functions

- def [clear](#) (self)  
*Clears the list.*
- def [\\_\\_init\\_\\_](#) (self)  
*Constructor.*
- def [\\_\\_iter\\_\\_](#) (self)  
*Gets a new [TypeListIterator](#).*
- def [iterator](#) (self)  
*Gets a new [Int32ListIterator](#).*
- def [add](#) (self, value)  
*Adds an 32-bit signed integer at the end of the list.*
- def [count](#) (self)  
*Number of elements in the list.*

#### 5.31.1 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>

#### 5.31.2 Constructor & Destructor Documentation

##### 5.31.2.1 def sparksee.Int32List.\_\_init\_\_ ( self )

Constructor.

This creates an empty list.

#### 5.31.3 Member Function Documentation

##### 5.31.3.1 def sparksee.Int32List.\_\_iter\_\_ ( self )

Gets a new [TypeListIterator](#).

#### Returns

[TypeListIterator](#) instance

##### 5.31.3.2 def sparksee.Int32List.add ( self, value )

Adds an 32-bit signed integer at the end of the list.

## Parameters

<i>value</i>	[in] The integer.
--------------	-------------------

5.31.3.3 def sparksee.Int32List.count ( *self* )

Number of elements in the list.

## Returns

Number of elements in the list.

5.31.3.4 def sparksee.Int32List.iterator ( *self* )

Gets a new [Int32ListIterator](#).

## Returns

[Int32ListIterator](#) instance.

## 5.32 sparksee.Int32ListIterator Class Reference

[Int32List](#) iterator class.

## Public Member Functions

- def [next](#) (self)  
*Moves to the next element.*
- def [has\\_next](#) (self)  
*Gets if there are more elements.*
- def [\\_\\_next\\_\\_](#) (self)  
*Used in [next\(\)](#)*

## 5.32.1 Detailed Description

[Int32List](#) iterator class.

Iterator to traverse all the integer into a [Int32List](#) instance.

## Author

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



### 5.32.2 Member Function Documentation

#### 5.32.2.1 `def sparksee.Int32ListIterator.__next__( self )`

Used in [next\(\)](#)

##### Returns

The next element

#### 5.32.2.2 `def sparksee.Int32ListIterator.has_next( self )`

Gets if there are more elements.

##### Returns

TRUE if there are more elements, FALSE otherwise.

#### 5.32.2.3 `def sparksee.Int32ListIterator.next( self )`

Moves to the next element.

##### Returns

The next element.

## 5.33 `sparksee.KeyValue` Class Reference

### Public Member Functions

- `def to_string( self )`

*Returns a String representation of the `KeyValue`, used in `unicode` and `str`*

### 5.33.1 Member Function Documentation

#### 5.33.1.1 `def sparksee.KeyValue.to_string( self )`

Returns a String representation of the [KeyValue](#), used in `unicode` and `str`

## 5.34 `sparksee.KeyValues` Class Reference

[Value](#) set class.

### Public Member Functions

- def `next` (self)  
*Gets the next [KeyValue](#) pair.*
- def `next` (self, kv)  
*Gets the next [KeyValue](#) pair.*
- def `has_next` (self)  
*Checks if the [KeyValues](#) has more [KeyValue](#) pairs.*
- def `close` (self)  
*Closes the [KeyValues](#) instance.*
- def `__next__` (self)  
*Used in `next()`*
- def `is_closed` (self)  
*Gets if [KeyValues](#) instance has been closed or not.*

#### 5.34.1 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>

#### 5.34.2 Member Function Documentation

##### 5.34.2.1 `def sparksee.KeyValues.__next__ ( self )`

Used in `next()`

#### Returns

The next element

##### 5.34.2.2 `def sparksee.KeyValues.close ( self )`

Closes the [KeyValues](#) instance.

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

##### 5.34.2.3 `def sparksee.KeyValues.has_next ( self )`

Checks if the [KeyValues](#) has more [KeyValue](#) pairs.

#### Returns

Returns true if there are more [KeyValue](#) pairs

#### 5.34.2.4 `def sparksee.KeyValues.is_closed ( self )`

Gets if `KeyValues` instance has been closed or not.

See also

[close\(\)](#)

Returns

TRUE if the `Values` instance has been closed, FALSE otherwise.

#### 5.34.2.5 `def sparksee.KeyValues.next ( self )`

Gets the next `KeyValue` pair.

Returns

Returns the next `KeyValue` pair

Referenced by `sparksee.KeyValues.next()`.

#### 5.34.2.6 `def sparksee.KeyValues.next ( self, kv )`

Gets the next `KeyValue` pair.

Parameters

<code>kv</code>	Returns the next <code>KeyValue</code> pair
-----------------	---

References `sparksee.KeyValues.next()`.

### 5.35 `sparksee.KOpt` Class Reference

`KOpt` class.

Public Member Functions

- `def __init__ (self, session, tour)`  
*Creates a new instance.*
- `def set_current_tour (self, tour)`  
*Sets current tour as a list of nodes.*
- `def get_current_cost (self)`  
*Returns tour cost.*
- `def add_all_edge_types (self, dir)`  
*Allows for traversing all edge types of the graph.*
- `def get_current_tour (self)`

- Returns *tour* as a list of nodes.
- def `set_edge_weight_attribute_type` (self, attr)  
Sets the attribute to use as edge weight.
- def `run_two_opt` (self)  
Runs 2-Opt local search.
- def `add_edge_type` (self, type, dir)  
Allows for traversing edges of the given type.
- def `run_three_opt` (self)  
Runs 3-Opt local search.
- def `set_max_iterations` (self, max\_iterations)  
Sets maximum number of iterations.
- def `set_time_limit` (self, max\_time)  
Limits execution time.
- def `add_node_type` (self, type)  
Allows for traversing nodes of the given type.
- def `__init__` (self, session)  
Creates a new instance.
- def `add_all_node_types` (self)  
Allows for traversing all node types of the graph.

### 5.35.1 Detailed Description

KOpt class.

Implements the 2-Opt and 3-Opt algorithms

#### Author

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

### 5.35.2 Constructor & Destructor Documentation

#### 5.35.2.1 def sparksee.KOpt.\_\_init\_\_( self, session, tour )

Creates a new instance.

#### Parameters

<i>session</i>	[in] <a href="#">Session</a> to get the graph from and perform algorithm.
<i>tour</i>	[in] Initial tour that needs to be improved.

Referenced by `sparksee.KOpt.__init__()`.

#### 5.35.2.2 def sparksee.KOpt.\_\_init\_\_( self, session )

Creates a new instance.

## Parameters

<i>session</i>	[in] <a href="#">Session</a> to get the graph from and perform algorithm.
----------------	---

References `sparksee.KOpt.__init__()`.

## 5.35.3 Member Function Documentation

5.35.3.1 `def sparksee.KOpt.add_all_edge_types ( self, dir )`

Allows for traversing all edge types of the graph.

## Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

5.35.3.2 `def sparksee.KOpt.add_edge_type ( self, type, dir )`

Allows for traversing edges of the given type.

If the edge type was already added, the existing direction is overwritten

## Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

## Exceptions

<i>RuntimeError</i>	null
---------------------	------

5.35.3.3 `def sparksee.KOpt.add_node_type ( self, type )`

Allows for traversing nodes of the given type.

`sparksee::gdb::Error`

## Parameters

<i>type</i>	[in] Node type.
-------------	-----------------

## Exceptions

<i>RuntimeError</i>	null
---------------------	------

#### 5.35.3.4 def sparksee.KOpt.set\_current\_tour ( self, tour )

Sets current tour as a list of nodes.

##### Parameters

<i>tour</i>	[in] Initial tour that needs to be improved.
-------------	--

#### 5.35.3.5 def sparksee.KOpt.set\_edge\_weight\_attribute\_type ( self, 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.

sparksee::gdb::Error

##### Parameters

<i>attr</i>	[in] The attribute type to use as a weight. Default: InvalidAttribute
-------------	---

##### Exceptions

<i>RuntimeError</i>	null
---------------------	------

#### 5.35.3.6 def sparksee.KOpt.set\_max\_iterations ( self, max\_iterations )

Sets maximum number of iterations.

By default the algorithm will run until no improvement can be made in the current tour.

##### Parameters

<i>max_iterations</i>	[in] Maximum number of iterations
-----------------------	-----------------------------------

#### 5.35.3.7 def sparksee.KOpt.set\_time\_limit ( self, max\_time )

Limits execution time.

##### Parameters

<i>max_time</i>	[in] Time limit in milliseconds
-----------------	---------------------------------

## 5.36 sparksee.LogLevel Class Reference

Log level enumeration.

## Static Public Attributes

- int **OFF** = 0  
*Disable log.*
- int **SEVERE** = 1  
*Severe log level.*
- int **WARNING** = 2  
*Warning log level.*
- int **INFO** = 3  
*Info log level.*
- int **CONFIG** = 4  
*Config log level.*
- int **FINE** = 5  
*Fine log level.*
- int **DEBUG** = 6  
*Debug log level.*

### 5.36.1 Detailed Description

Log level enumeration.

Log level priority order is as follows, from minimum to maximum log information: Off (log is disabled), Severe, Warning, Info, Config, Fine, Debug.

#### Author

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

### 5.36.2 Member Data Documentation

#### 5.36.2.1 int sparksee.LogLevel.CONFIG = 4 [static]

Config log level.

Errors, warnings, information messages and configuration details of the different components are shown.

#### 5.36.2.2 int sparksee.LogLevel.DEBUG = 6 [static]

Debug log level.

This is for [Sparksee](#) development purposes and just works with debug versions of the library.

#### 5.36.2.3 int sparksee.LogLevel.FINE = 5 [static]

Fine log level.

This is the higher and finest public log level, everything is dumped to the log.

#### 5.36.2.4 int sparksee.LogLevel.INFO = 3 [static]

Info log level.

Errors, warnings and information messages are shown.

#### 5.36.2.5 int sparksee.LogLevel.SEVERE = 1 [static]

Severe log level.

This is the lower log level, just errors are shown.

#### 5.36.2.6 int sparksee.LogLevel.WARNING = 2 [static]

Warning log level.

Errors and warnings are shown.

## 5.37 sparksee.MissingEndpoint Class Reference

The policy to follow whenever and edge endpoint is missing during a loading.

### Static Public Attributes

- int [IS\\_ERROR](#) = 0  
*Throw an error.*
- int [CREATE](#) = 1  
*Create the endpoint.*
- int [IGNORE](#) = 2  
*Ignore the edge.*

#### 5.37.1 Detailed Description

The policy to follow whenever and edge endpoint is missing during a loading.

## 5.38 sparksee.NodeExport Class Reference

Stores the node exporting values.



## Public Member Functions

- def `get_labelcolor_rgb` (self)  
*Gets the node label color.*
- def `set_label` (self, label)  
*Sets the node label.*
- def `set_width` (self, width)  
*Gets the node width.*
- def `set_color_rgb` (self, color)  
*Sets the node color.*
- def `set_font_size` (self, size)  
*Sets the node label font size.*
- def `set_height` (self, height)  
*Sets the node height.*
- def `get_shape` (self)  
*Gets the node shape.*
- def `set_labelcolor_rgb` (self, color)  
*Sets the node label color.*
- def `is_fit` (self)  
*Gets whether the node size is fitted to the label or not.*
- def `get_color_rgb` (self)  
*Gets the node color.*
- def `set_fit` (self, fit)  
*Sets the node fit property.*
- def `__init__` (self)  
*Creates a new instance.*
- def `get_font_size` (self)  
*Gets the node label font size.*
- def `set_defaults` (self)  
*Sets to default values.*
- def `get_label` (self)  
*Gets the node label.*
- def `get_height` (self)  
*Gets the node height.*
- def `get_width` (self)  
*Gets the node width.*
- def `set_shape` (self, shape)  
*Sets the node shape.*

### 5.38.1 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 (0x000000, Black).

Height: 25px.

Width: 25px.

Fit: TRUE.

Font size: 10.

#### Author

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

### 5.38.2 Member Function Documentation

#### 5.38.2.1 def sparksee.NodeExport.get\_color\_rgb ( self )

Gets the node color.

#### Returns

The node color.

#### 5.38.2.2 def sparksee.NodeExport.get\_font\_size ( self )

Gets the node label font size.

#### Returns

The node label font size.

#### 5.38.2.3 def sparksee.NodeExport.get\_height ( self )

Gets the node height.

#### Returns

The node height in pixels.

#### 5.38.2.4 def sparksee.NodeExport.get\_label ( self )

Gets the node label.

#### Returns

The node label.

#### 5.38.2.5 `def sparksee.NodeExport.get_labelcolor_rgb ( self )`

Gets the node label color.

##### Returns

The node label color.

#### 5.38.2.6 `def sparksee.NodeExport.get_shape ( self )`

Gets the node shape.

##### Returns

The node shape.

#### 5.38.2.7 `def sparksee.NodeExport.get_width ( self )`

Gets the node width.

##### Returns

The node width in pixels.

#### 5.38.2.8 `def sparksee.NodeExport.is_fit ( self )`

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'.

#### 5.38.2.9 `def sparksee.NodeExport.set_color_rgb ( self, color )`

Sets the node color.

##### Parameters

<i>color</i>	The node color.
--------------	-----------------

#### 5.38.2.10 `def sparksee.NodeExport.set_fit ( self, fit )`

Sets the node fit property.

##### Parameters

<i>fit</i>	[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'.
------------	--

5.38.2.11 `def sparksee.NodeExport.set_font_size ( self, size )`

Sets the node label font size.

Parameters

<i>size</i>	[in] The node label font size.
-------------	--------------------------------

5.38.2.12 `def sparksee.NodeExport.set_height ( self, height )`

Sets the node height.

Parameters

<i>height</i>	[in] The node height in pixels.
---------------	---------------------------------

5.38.2.13 `def sparksee.NodeExport.set_label ( self, label )`

Sets the node label.

Parameters

<i>label</i>	[in] The node label.
--------------	----------------------

5.38.2.14 `def sparksee.NodeExport.set_labelcolor_rgb ( self, color )`

Sets the node label color.

Parameters

<i>color</i>	[in] The node label color.
--------------	----------------------------

5.38.2.15 `def sparksee.NodeExport.set_shape ( self, shape )`

Sets the node shape.

Parameters

<i>shape</i>	[in] The node shape.
--------------	----------------------

5.38.2.16 `def sparksee.NodeExport.set_width ( self, width )`

Gets the node width.

Parameters

<i>width</i>	The node width in pixels.
--------------	---------------------------

## 5.39 sparksee.NodeShape Class Reference

Node shape.

### Static Public Attributes

- int `BOX` = 0  
*Box shape.*
- int `ROUND` = 1  
*Round shape.*

### 5.39.1 Detailed Description

Node shape.

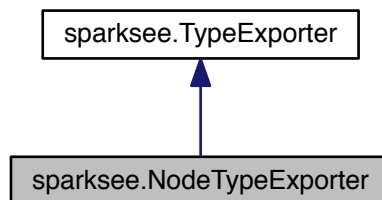
### Author

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

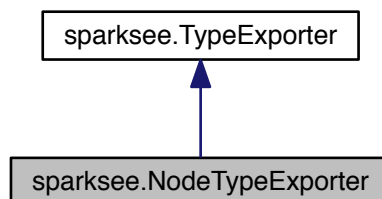
## 5.40 sparksee.NodeTypeExporter Class Reference

`NodeTypeExporter` class.

Inheritance diagram for `sparksee.NodeTypeExporter`:



Collaboration diagram for `sparksee.NodeTypeExporter`:



## Public Member Functions

- def `__init__` (self)  
*Creates a new instance.*
- def `run` (self)  
*See the [TypeExporter](#) class `Run` method.*
- def `__init__` (self, row\_writer, graph, type, attrs)  
*Creates a new instance.*
- def `set_type` (self, type)  
*Sets the type to be exported.*
- def `set_header` (self, header)  
*Sets the presence of a header row.*
- def `set_row_writer` (self, rw)  
*Sets the output data destination.*
- def `set_graph` (self, graph)  
*Sets the graph that will be exported.*
- def `register` (self, tel)  
*Registers a new listener.*
- def `set_attributes` (self, attrs)  
*Sets the list of Attributes.*
- def `set_frequency` (self, freq)  
*Sets the frequency of listener notification.*

## 5.40.1 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>

## 5.40.2 Constructor &amp; Destructor Documentation

## 5.40.2.1 def sparksee.NodeTypeExporter.\_\_init\_\_( self, row\_writer, graph, type, attrs )

Creates a new instance.

## Parameters

<i>row_writer</i>	[in] Output <a href="#">RowWriter</a> .
<i>graph</i>	[in] <a href="#">Graph</a> .
<i>type</i>	[in] <a href="#">Type</a> identifier.
<i>attrs</i>	[in] <a href="#">Attribute</a> identifiers to be exported.

References [sparksee.NodeTypeExporter.\\_\\_init\\_\\_\(\)](#).

### 5.40.3 Member Function Documentation

#### 5.40.3.1 `def sparksee.NodeTypeExporter.register ( self, tel )`

Registers a new listener.

##### Parameters

<i>tel</i>	[in] <a href="#">TypeExporterListener</a> to be registered.
------------	---

#### 5.40.3.2 `def sparksee.NodeTypeExporter.run ( self )`

See the [TypeExporter](#) class Run method.

##### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

#### 5.40.3.3 `def sparksee.NodeTypeExporter.set_attributes ( self, attrs )`

Sets the list of Attributes.

##### Parameters

<i>attrs</i>	[in] <a href="#">Attribute</a> identifiers to be exported
--------------	---

#### 5.40.3.4 `def sparksee.NodeTypeExporter.set_frequency ( self, freq )`

Sets the frequency of listener notification.

##### Parameters

<i>freq</i>	[in] Frequency in number of rows managed to notify progress to all listeners
-------------	--

#### 5.40.3.5 `def sparksee.NodeTypeExporter.set_graph ( self, graph )`

Sets the graph that will be exported.

##### Parameters

<i>graph</i>	[in] <a href="#">Graph</a> .
--------------	------------------------------

#### 5.40.3.6 `def sparksee.NodeTypeExporter.set_header ( self, header )`

Sets the presence of a header row.

## Parameters

<i>header</i>	[in] If TRUE, a header row is dumped with the name of the attributes.
---------------	---

5.40.3.7 `def sparksee.NodeTypeExporter.set_row_writer ( self, rw )`

Sets the output data destination.

## Parameters

<i>rw</i>	[in] Input <a href="#">RowWriter</a> .
-----------	--

5.40.3.8 `def sparksee.NodeTypeExporter.set_type ( self, type )`

Sets the type to be exported.

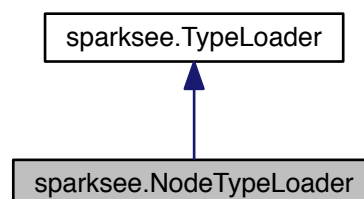
## Parameters

<i>type</i>	[in] <a href="#">Type</a> identifier.
-------------	---------------------------------------

## 5.41 sparksee.NodeTypeLoader Class Reference

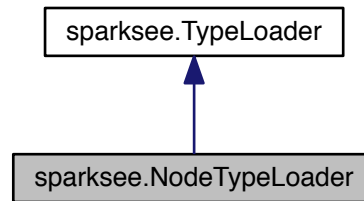
[NodeTypeLoader](#) class.

Inheritance diagram for `sparksee.NodeTypeLoader`:





Collaboration diagram for sparksee.NodeTypeLoader:



### Public Member Functions

- def `run` (self)  
See the `TypeLoader` class `Run` method.
- def `set_log_error` (self, path)  
Sets a log error file.
- def `set_type` (self, type)  
Sets the type to be loaded.
- def `set_locale` (self, locale\_str)  
Sets the locale that will be used to read the data.
- def `set_timestamp_format` (self, timestamp\_format)  
Sets a specific timestamp format.
- def `set_attributes` (self, attrs)  
Sets the list of Attributes.
- def `__init__` (self)  
Creates a new instance.
- def `register` (self, tel)  
Registers a new listener.
- def `set_frequency` (self, freq)  
Sets the frequency of listener notification.
- def `__init__` (self, row\_reader, graph, type, attrs, attrs\_pos)  
Creates a new instance.
- def `run_two_phases` (self)  
See the `TypeLoader` class `RunTwoPhases` method.
- def `set_row_reader` (self, rr)  
Sets the input data source.
- def `set_attribute_positions` (self, attrs\_pos)  
Sets the list of attribute positions.
- def `set_graph` (self, graph)  
Sets the graph where the data will be loaded.
- def `run_n_phases` (self, partitions)  
See the `TypeLoader` class `RunNPhases` method.
- def `set_log_off` (self)  
Turns off all the error reporting.

## 5.41.1 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>

## 5.41.2 Constructor &amp; Destructor Documentation

## 5.41.2.1 def sparksee.NodeTypeLoader.\_\_init\_\_( self, row\_reader, graph, type, attrs, attrs\_pos )

Creates a new instance.

## Parameters

<i>row_reader</i>	[in] Input <a href="#">RowReader</a> .
<i>graph</i>	[in] <a href="#">Graph</a> .
<i>type</i>	[in] <a href="#">Type</a> identifier.
<i>attrs</i>	[in] <a href="#">Attribute</a> identifiers to be loaded.
<i>attrs_pos</i>	[in] <a href="#">Attribute</a> positions (column index $\geq 0$ ).

References [sparksee.NodeTypeLoader.\\_\\_init\\_\\_\(\)](#).

## 5.41.3 Member Function Documentation

## 5.41.3.1 def sparksee.NodeTypeLoader.register( self, tel )

Registers a new listener.

## Parameters

<i>tel</i>	<a href="#">TypeLoaderListener</a> to be registered.
------------	--

## 5.41.3.2 def sparksee.NodeTypeLoader.run( self )

See the [TypeLoader](#) class Run method.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

5.41.3.3 `def sparksee.NodeTypeLoader.run_n_phases ( self, partitions )`

See the [TypeLoader](#) class `RunNPhases` method.

Parameters

<i>partitions</i>	null
-------------------	------

Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

5.41.3.4 `def sparksee.NodeTypeLoader.run_two_phases ( self )`

See the [TypeLoader](#) class `RunTwoPhases` method.

Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

5.41.3.5 `def sparksee.NodeTypeLoader.set_attribute_positions ( self, attrs_pos )`

Sets the list of attribute positions.

Parameters

<i>attrs_pos</i>	[in] <a href="#">Attribute</a> positions (column index $\geq 0$ ).
------------------	--

5.41.3.6 `def sparksee.NodeTypeLoader.set_attributes ( self, attrs )`

Sets the list of Attributes.

Parameters

<i>attrs</i>	[in] <a href="#">Attribute</a> identifiers to be loaded
--------------	---

5.41.3.7 `def sparksee.NodeTypeLoader.set_frequency ( self, freq )`

Sets the frequency of listener notification.

Parameters

<i>freq</i>	[in] Frequency in number of rows managed to notify progress to all listeners
-------------	--

#### 5.41.3.8 def sparksee.NodeTypeLoader.set\_graph ( self, graph )

Sets the graph where the data will be loaded.

##### Parameters

<i>graph</i>	[in] <a href="#">Graph</a> .
--------------	------------------------------

#### 5.41.3.9 def sparksee.NodeTypeLoader.set\_locale ( self, locale\_str )

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

It should match the locale used in the rowreader.

##### Parameters

<i>locale_str</i>	[in] The locale string for the read data. See <a href="#">CSVReader</a> .
-------------------	---

#### 5.41.3.10 def sparksee.NodeTypeLoader.set\_log\_error ( self, 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

<i>path</i>	[in] The path to the error log file.
-------------	--------------------------------------

##### Exceptions

<i>IOError</i>	If bad things happen opening the file.
----------------	--

#### 5.41.3.11 def sparksee.NodeTypeLoader.set\_log\_off ( self )

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.

#### 5.41.3.12 def sparksee.NodeTypeLoader.set\_row\_reader ( self, rr )

Sets the input data source.

##### Parameters

<i>rr</i>	[in] Input <a href="#">RowReader</a> .
-----------	--

#### 5.41.3.13 `def sparksee.NodeTypeLoader.set_timestamp_format ( self, timestamp_format )`

Sets a specific timestamp format.

##### Parameters

<code>timestamp_format</code>	[in] A string with the timestamp format definition.
-------------------------------	---

#### 5.41.3.14 `def sparksee.NodeTypeLoader.set_type ( self, type )`

Sets the type to be loaded.

##### Parameters

<code>type</code>	[in] <a href="#">Type</a> identifier.
-------------------	---------------------------------------

## 5.42 `sparksee.Objects` Class Reference

Object identifier set class.

### Public Member Functions

- `def copy` (self, objs)  
*Performs the copy operation.*
- `def count` (self)  
*Gets the number of elements into the collection.*
- `def equals` (self, objs)  
*Checks if the given [Objects](#) contains the same information.*
- `def any` (self)  
*Gets an element from the collection.*
- `def contains` (self, objs)  
*Check if this objects contains the other one.*
- `def combine_intersection` (self, objs1, objs2)  
*Creates a new [Objects](#) instance which is the intersection of the two given.*
- `def __iter__` (self)  
*Gets a new [ObjectsIterator](#).*
- `def iterator` (self)  
*Gets an [ObjectsIterator](#).*
- `def union` (self, objs)  
*Performs the union operation.*
- `def combine_union` (self, objs1, objs2)  
*Creates a new [Objects](#) instance which is the union of the two given.*
- `def intersection` (self, objs)  
*Performs the intersection operation.*
- `def clear` (self)  
*Clears the collection removing all its elements.*
- `def iterator_from_index` (self, index)

- Gets an [ObjectsIterator](#) skipping index elements.*
- def [exists](#) (self, e)  
*Gets if the given element exists into the collection.*
- def [copy](#) (self)  
*Creates a new [Objects](#) instance as a copy of the given one.*
- def [difference](#) (self, objs)  
*Performs the difference operation.*
- def [combine\\_difference](#) (self, objs1, objs2)  
*Creates a new [Objects](#) instance which is the difference of the two given.*
- def [sample](#) (self, exclude, samples)  
*Creates a new [Objects](#) instance which is a sample of the calling one.*
- def [close](#) (self)  
*Closes the [Objects](#) instance.*
- def [add](#) (self, e)  
*Adds an element into the collection.*
- def [remove](#) (self, e)  
*Removes an element from the collection.*
- def [iterator\\_from\\_element](#) (self, e)  
*Gets an [ObjectsIterator](#) starting from the given element.*
- def [is\\_closed](#) (self)  
*Gets if [Objects](#) instance has been closed or not.*

#### Static Public Attributes

- int [INVALID\\_OID](#) = 0  
*Invalid OID constant.*

#### 5.42.1 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>

#### 5.42.2 Member Function Documentation

##### 5.42.2.1 def sparksee.Objects.\_\_iter\_\_( self )

Gets a new [ObjectsIterator](#).

#### Returns

[ObjectsIterator](#) instance

##### 5.42.2.2 def sparksee.Objects.add( self, e )

Adds an element into the collection.

**Parameters**

<i>e</i>	[in] Element to be added.
----------	---------------------------

**Returns**

TRUE if the element is added, FALSE if the element was already into the collection.

**5.42.2.3 def sparksee.Objects.any ( self )**

Gets an element from the collection.

**Returns**

Any element from the collection.

**Exceptions**

<i>RuntimeError</i>	null
<i>RuntimeError</i>	whether the collection is empty.

**5.42.2.4 def sparksee.Objects.close ( self )**

Closes the [Objects](#) instance.

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

**5.42.2.5 def sparksee.Objects.combine\_difference ( self, objs1, objs2 )**

Creates a new [Objects](#) instance which is the difference of the two given.

Two given [Objects](#) belong to the same [Session](#).

**Parameters**

<i>objs1</i>	[in] <a href="#">Objects</a> instance.
<i>objs2</i>	[in] <a href="#">Objects</a> instance.

**Returns**

New [Objects](#) instance.

**5.42.2.6 def sparksee.Objects.combine\_intersection ( self, objs1, objs2 )**

Creates a new [Objects](#) instance which is the intersection of the two given.

Two given [Objects](#) belong to the same [Session](#).

## Parameters

<i>objs1</i>	[in] <a href="#">Objects</a> instance.
<i>objs2</i>	[in] <a href="#">Objects</a> instance.

## Returns

New [Objects](#) instance.

5.42.2.7 `def sparksee.Objects.combine_union ( self, objs1, objs2 )`

Creates a new [Objects](#) instance which is the union of the two given.

Two given [Objects](#) belong to the same [Session](#).

## Parameters

<i>objs1</i>	[in] <a href="#">Objects</a> instance.
<i>objs2</i>	[in] <a href="#">Objects</a> instance.

## Returns

New [Objects](#) instance.

5.42.2.8 `def sparksee.Objects.contains ( self, objs )`

Check if this objects contains the other one.

## Parameters

<i>objs</i>	<a href="#">Objects</a> collection.
-------------	-------------------------------------

## Returns

True if it contains the given object.

5.42.2.9 `def sparksee.Objects.copy ( self, objs )`

Performs the copy operation.

This updates the [Objects](#) calling instance and copies the given [Objects](#) instance.

## Parameters

<i>objs</i>	[in] <a href="#">Objects</a> instance.
-------------	--



**Returns**

Number of elements into the collection once the operation has been executed.

Referenced by `sparksee.Objects.copy()`.

**5.42.2.10** `def sparksee.Objects.copy ( self )`

Creates a new [Objects](#) instance as a copy of the given one.

**Returns**

The new [Objects](#) instance.

References `sparksee.Objects.copy()`.

**5.42.2.11** `def sparksee.Objects.count ( self )`

Gets the number of elements into the collection.

**Returns**

The number of elements into the collection.

**5.42.2.12** `def sparksee.Objects.difference ( self, objs )`

Performs the difference operation.

This updates the [Objects](#) calling instance removing those existing elements at the given [Objects](#) instance.

**Parameters**

<i>objs</i>	[in] <a href="#">Objects</a> instance.
-------------	--

**Returns**

Number of elements into the collection once the operation has been executed.

**5.42.2.13** `def sparksee.Objects.equals ( self, objs )`

Checks if the given [Objects](#) contains the same information.

**Parameters**

<i>objs</i>	[in] <a href="#">Objects</a> instance.
-------------	--

**Returns**

True if the objects are equal or false otherwise.

5.42.2.14 `def sparksee.Objects.exists ( self, e )`

Gets if the given element exists into the collection.

Parameters

<i>e</i>	[in] Element.
----------	---------------

Returns

TRUE if the element exists into the collection, FALSE otherwise.

5.42.2.15 `def sparksee.Objects.intersection ( self, objs )`

Performs the intersection operation.

Updates the [Objects](#) calling instance setting those existing elements at both two collections and removing all others.

Parameters

<i>objs</i>	[in] <a href="#">Objects</a> instance.
-------------	--

Returns

Number of elements into the collection once the operation has been executed.

5.42.2.16 `def sparksee.Objects.is_closed ( self )`

Gets if [Objects](#) instance has been closed or not.

See also

[close\(\)](#)

Returns

TRUE if the [Objects](#) instance has been closed, FALSE otherwise.

5.42.2.17 `def sparksee.Objects.iterator ( self )`

Gets an [ObjectsIterator](#).

Returns

[ObjectsIterator](#) instance.

5.42.2.18 `def sparksee.Objects.iterator_from_element ( self, 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

<i>e</i>	[in] The first element to traverse in the resulting <a href="#">ObjectsIterator</a> instance.
----------	---

## Returns

[ObjectsIterator](#) instance.

5.42.2.19 `def sparksee.Objects.iterator_from_index ( self, index )`

Gets an [ObjectsIterator](#) skipping index elements.

[Objects](#) collection has no order, so this method is implementation-dependent.

## Parameters

<i>index</i>	[in] The number of elements to skip from the beginning. It must be in the range [0..Size).
--------------	--

## Returns

[ObjectsIterator](#) instance.

5.42.2.20 `def sparksee.Objects.remove ( self, e )`

Removes an element from the collection.

## Parameters

<i>e</i>	[in] Element to be removed.
----------	-----------------------------

## Returns

TRUE if the element is removed, FALSE if the element was not into the collection.

5.42.2.21 `def sparksee.Objects.sample ( self, exclude, samples )`

Creates a new [Objects](#) instance which is a sample of the calling one.

## Parameters

<i>exclude</i>	[in] If not NULL, elements into this collection will be excluded from the resulting one.
<i>samples</i>	[in] Number of elements into the resulting collection.

## Returns

Sample collection.

5.42.2.22 `def sparksee.Objects.union ( self, objs )`

Performs the union operation.

This adds all existing elements of the given [Objects](#) instance to the [Objects](#) calling instance

#### Parameters

<code>objs</code>	[in] <a href="#">Objects</a> instance.
-------------------	--

#### Returns

Number of elements into the collection once the operation has been executed.

## 5.43 sparksee.ObjectsIterator Class Reference

[ObjectsIterator](#) class.

#### Public Member Functions

- `def next (self)`  
*Gets the next element to traverse.*
- `def has_next (self)`  
*Gets if there are more elements to traverse.*
- `def close (self)`  
*Closes the [ObjectsIterator](#) instance.*
- `def __next__ (self)`  
*Used in `next()`*
- `def is_closed (self)`  
*Gets if [ObjectsIterator](#) instance has been closed or not.*

#### 5.43.1 Detailed Description

[ObjectsIterator](#) class.

Iterator to traverse all the object identifiers from an [Objects](#) instance.

#### Author

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

#### 5.43.2 Member Function Documentation

5.43.2.1 `def sparksee.ObjectsIterator.__next__ ( self )`

Used in `next()`

#### Returns

The next element

#### 5.43.2.2 `def sparksee.ObjectsIterator.close ( self )`

Closes the [ObjectsIterator](#) instance.

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

#### 5.43.2.3 `def sparksee.ObjectsIterator.has_next ( self )`

Gets if there are more elements to traverse.

##### Returns

TRUE if there are more elements to traverse, FALSE otherwise.

#### 5.43.2.4 `def sparksee.ObjectsIterator.is_closed ( self )`

Gets if [ObjectsIterator](#) instance has been closed or not.

##### See also

[close\(\)](#)

##### Returns

TRUE if the [ObjectsIterator](#) instance has been closed, FALSE otherwise.

#### 5.43.2.5 `def sparksee.ObjectsIterator.next ( self )`

Gets the next element to traverse.

##### Returns

The next element.

## 5.44 `sparksee.ObjectType` Class Reference

Object type enumeration.

##### Static Public Attributes

- int [NODE](#) = 0  
*Node object type.*
- int [EDGE](#) = 1  
*Edge object type.*

### 5.44.1 Detailed Description

Object type enumeration.

#### Author

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

## 5.45 sparksee.OIDList Class Reference

[Sparksee](#) object identifier list.

### Public Member Functions

- def [clear](#) (self)  
*Clears the list.*
- def [\\_\\_init\\_\\_](#) (self, num\_invalid\_oids)  
*Constructor.*
- def [\\_\\_init\\_\\_](#) (self)  
*Constructor.*
- def [\\_\\_iter\\_\\_](#) (self)  
*Gets a new [TypeListIterator](#).*
- def [set](#) (self, pos, oid)  
*Sets a [Sparksee](#) object identifier at the specified position of the list.*
- def [iterator](#) (self)  
*Gets a new [OIDListIterator](#).*
- def [add](#) (self, attr)  
*Adds a [Sparksee](#) object identifier at the end of the list.*
- def [count](#) (self)  
*Number of elements in the list.*

### 5.45.1 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>

### 5.45.2 Constructor & Destructor Documentation

#### 5.45.2.1 def sparksee.OIDList.\_\_init\_\_ ( self, num\_invalid\_oids )

Constructor.

This creates a list with N invalid oids.

**Parameters**

<code>num_invalid_oids</code>	[in] The number of invalid oids added to the list.
-------------------------------	--

Referenced by `sparksee.OIDList.__init__()`.

**5.45.2.2 def sparksee.OIDList.\_\_init\_\_ ( self )**

Constructor.

This creates an empty list.

References `sparksee.OIDList.__init__()`.

**5.45.3 Member Function Documentation****5.45.3.1 def sparksee.OIDList.\_\_iter\_\_ ( self )**

Gets a new [TypeListIterator](#).

**Returns**

[TypeListIterator](#) instance

**5.45.3.2 def sparksee.OIDList.add ( self, attr )**

Adds a [Sparksee](#) object identifier at the end of the list.

**Parameters**

<code>attr</code>	[in] <a href="#">Sparksee</a> object identifier.
-------------------	--

**5.45.3.3 def sparksee.OIDList.count ( self )**

Number of elements in the list.

**Returns**

Number of elements in the list.

**5.45.3.4 def sparksee.OIDList.iterator ( self )**

Gets a new [OIDListIterator](#).

**Returns**

[OIDListIterator](#) instance.

**5.45.3.5 def sparksee.OIDList.set ( self, pos, oid )**

Sets a [Sparksee](#) object identifier at the specified position of the list.

## Parameters

<i>pos</i>	[in] List position [0..Count()-1].
<i>oid</i>	[in] <a href="#">Sparksee</a> object identifier.

## 5.46 sparksee.OIDListIterator Class Reference

[OIDList](#) iterator class.

## Public Member Functions

- def [next](#) (self)  
*Moves to the next element.*
- def [has\\_next](#) (self)  
*Gets if there are more elements.*
- def [\\_\\_next\\_\\_](#) (self)  
*Used in [next\(\)](#)*

## 5.46.1 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>

## 5.46.2 Member Function Documentation

## 5.46.2.1 def sparksee.OIDListIterator.\_\_next\_\_( self )

Used in [next\(\)](#)

## Returns

The next element

## 5.46.2.2 def sparksee.OIDListIterator.has\_next( self )

Gets if there are more elements.

## Returns

TRUE if there are more elements, FALSE otherwise.



### 5.46.2.3 `def sparksee.OIDListIterator.next ( self )`

Moves to the next element.

#### Returns

The next element.

## 5.47 `sparksee.Order` Class Reference

`Order` enumeration.

### Static Public Attributes

- int `ASCENDENT` = 0  
*From lower to higher.*
- int `DESCENDENT` = 1  
*From higher to lower.*

### 5.47.1 Detailed Description

`Order` enumeration.

#### Author

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

## 5.48 `sparksee.PageRank` Class Reference

`PageRank` class.

### Public Member Functions

- `def set_damping (self, damping)`  
*Sets the damping value for the `PageRank`.*
- `def set_default_weight (self, weight)`  
*Sets the default weight for those cases when a given edge does not have a weight attribute set.*
- `def run (self)`  
*Runs the algorithm.*
- `def add_all_edge_types (self, dir)`  
*Allows for traversing all edge types of the graph.*
- `def set_output_attribute_type (self, attr)`  
*Sets the output attribute type.*
- `def set_starting_node (self, start_node)`  
*Sets the starting node of the page rank to compute the Personalized `PageRank` variant.*
- `def set_initial_page_rank_value (self, start_value)`  
*Sets the initial `PageRank` value.*
- `def set_edge_weight_attribute_type (self, attr)`

- Sets the attribute to use as edge weight.*
- def `add_edge_type` (self, type, dir)  
*Allows for traversing edges of the given type.*
- def `set_num_iterations` (self, num\_iterations)  
*Sets the number of iterations to run the [PageRank](#) for.*
- def `set_tolerance` (self, tolerance)  
*Sets the tolerance threshold to continue computing the [PageRank](#) after each iteration.*
- def `add_node_type` (self, type)  
*Allows for traversing nodes of the given type.*
- def `__init__` (self, session)  
*Builds the [PageRank](#).*
- def `add_all_node_types` (self)  
*Allows for traversing all node types of the graph.*

### 5.48.1 Detailed Description

[PageRank](#) class.

Implements the [PageRank](#) algorithm

#### Author

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

### 5.48.2 Constructor & Destructor Documentation

#### 5.48.2.1 def sparksee.PageRank.\_\_init\_\_ ( self, session )

Builds the [PageRank](#).

#### Parameters

<code>session</code>	[in] The session to use
----------------------	-------------------------

### 5.48.3 Member Function Documentation

#### 5.48.3.1 def sparksee.PageRank.add\_all\_edge\_types ( self, 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

<code>dir</code>	[in] Edge direction.
------------------	----------------------

### 5.48.3.2 `def sparksee.PageRank.add_edge_type ( self, type, 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

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

#### Exceptions

<i>RuntimeError</i>	null
---------------------	------

### 5.48.3.3 `def sparksee.PageRank.add_node_type ( self, type )`

Allows for traversing nodes of the given type.

#### Parameters

<i>type</i>	null
-------------	------

#### Exceptions

<i>RuntimeError</i>	null
---------------------	------

### 5.48.3.4 `def sparksee.PageRank.run ( self )`

Runs the algorithm.

sparksee::gdb::Error

#### Exceptions

<i>RuntimeError</i>	null
---------------------	------

### 5.48.3.5 `def sparksee.PageRank.set_damping ( self, damping )`

Sets the damping value for the [PageRank](#).

#### Parameters

<i>damping</i>	[in] The damping value. Default: 0.85
----------------	---------------------------------------

## 5.48.3.6 def sparksee.PageRank.set\_default\_weight ( self, weight )

Sets the default weight for those cases when a given edge does not have a weight attribute set.

Default: 0.0

Parameters

<i>weight</i>	[in] The default weight
---------------	-------------------------

## 5.48.3.7 def sparksee.PageRank.set\_edge\_weight\_attribute\_type ( self, 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

<i>attr</i>	[in] The attribute type to use as a weight. Default: InvalidAttribute
-------------	---

Exceptions

<i>RuntimeError</i>	null
---------------------	------

## 5.48.3.8 def sparksee.PageRank.set\_initial\_page\_rank\_value ( self, start\_value )

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

<i>start_value</i>	[in] The initial value to set. Default: 0.0
--------------------	---

## 5.48.3.9 def sparksee.PageRank.set\_num\_iterations ( self, num\_iterations )

Sets the number of iterations to run the [PageRank](#) for.

Parameters

<i>num_iterations</i>	[in] The number of iterations to set. Default: 20
-----------------------	---

#### 5.48.3.10 `def sparksee.PageRank.set_output_attribute_type ( self, 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

<code>attr</code>	[in] The attribute to store the result. Default: <code>InvalidAttribute</code>
-------------------	--

##### Exceptions

<code>RuntimeError</code>	null
---------------------------	------

#### 5.48.3.11 `def sparksee.PageRank.set_starting_node ( self, start_node )`

Sets the starting node of the page rank to compute the Personalized [PageRank](#) variant.

`sparksee::gdb::Error`

##### Parameters

<code>start_node</code>	null
-------------------------	------

##### Exceptions

<code>RuntimeError</code>	null
---------------------------	------

#### 5.48.3.12 `def sparksee.PageRank.set_tolerance ( self, tolerance )`

Sets the tolerance threshold 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

<code>tolerance</code>	[in] The tolerance to use normalized between 0 and 1. Default: <code>0.000001</code>
------------------------	--

## 5.49 `sparksee.Platform` Class Reference

[Platform](#) class.

### Public Member Functions

- `def get_statistics (self, stats)`  
*Gets platform data and statistics.*

### 5.49.1 Detailed Description

[Platform](#) class.

#### Author

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

### 5.49.2 Member Function Documentation

#### 5.49.2.1 def sparksee.Platform.get\_statistics ( self, stats )

Gets platform data and statistics.

#### Parameters

<code>stats</code>	[in out] This updates the given <a href="#">PlatformStatistics</a> .
--------------------	--

## 5.50 sparksee.PlatformStatistics Class Reference

[Platform](#) data and statistics.

#### Public Member Functions

- def [\\_\\_init\\_\\_](#) (self)  
*Creates a new instance setting all values to 0.*
- def [get\\_total\\_mem](#) (self)  
*Gets physical memory size in Bytes.*
- def [get\\_available\\_mem](#) (self)  
*Gets avialable (free) memory size in Bytes.*
- def [get\\_real\\_time](#) (self)  
*Gets time in microseconds (since epoch).*
- def [get\\_system\\_time](#) (self)  
*Gets CPU system time.*
- def [get\\_user\\_time](#) (self)  
*Gets CPU user time.*
- def [get\\_num\\_c\\_p\\_us](#) (self)  
*Gets the number of CPUs.*

### 5.50.1 Detailed Description

[Platform](#) data and statistics.

#### Author

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

## 5.50.2 Member Function Documentation

### 5.50.2.1 `def sparksee.PlatformStatistics.get_available_mem ( self )`

Gets available (free) memory size in Bytes.

#### Returns

Available (free) memory size in Bytes.

### 5.50.2.2 `def sparksee.PlatformStatistics.get_num_c_p_us ( self )`

Gets the number of CPUs.

#### Returns

The number of CPUs.

### 5.50.2.3 `def sparksee.PlatformStatistics.get_real_time ( self )`

Gets time in microseconds (since epoch).

#### Returns

Time in microseconds (since epoch).

### 5.50.2.4 `def sparksee.PlatformStatistics.get_system_time ( self )`

Gets CPU system time.

#### Returns

CPU system time.

### 5.50.2.5 `def sparksee.PlatformStatistics.get_total_mem ( self )`

Gets physical memory size in Bytes.

#### Returns

Physical memory size in Bytes.

### 5.50.2.6 `def sparksee.PlatformStatistics.get_user_time ( self )`

Gets CPU user time.

#### Returns

CPU user time.

## 5.51 sparksee.Query Class Reference

[Query](#) class.

### Public Member Functions

- def [set\\_stream](#) (self, stream, handler)  
*Sets a query stream handler.*
- def [set\\_dynamic](#) (self, name, value)  
*Sets the value for a dynamic parameter.*
- def [execute](#) (self, stmt, reiterable)  
*Executes the given statement.*
- def [close](#) (self)  
*Closes the [Query](#) instance.*
- def [is\\_closed](#) (self)  
*Gets if the [Query](#) instance has been closed or not.*

### 5.51.1 Detailed Description

[Query](#) class.

#### Author

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

### 5.51.2 Member Function Documentation

#### 5.51.2.1 def sparksee.Query.close ( self )

Closes the [Query](#) instance.

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

#### 5.51.2.2 def sparksee.Query.execute ( self, stmt, reiterable )

Executes the given statement.

#### Parameters

<i>stmt</i>	[in] <a href="#">Query</a> statement.
<i>reiterable</i>	[in] Whether we want the resultset to be reiterable or not

#### Returns

A [ResultSet](#) instance with the contents of the result of the query.



### 5.51.2.3 def sparksee.Query.is\_closed ( self )

Gets if the [Query](#) instance has been closed or not.

See also

[close\(\)](#)

Returns

TRUE if the [Query](#) instance has been closed, FALSE otherwise.

### 5.51.2.4 def sparksee.Query.set\_dynamic ( self, name, value )

Sets the value for a dynamic paramater.

Parameters

<i>name</i>	[in] Parameter name
<i>value</i>	[in] Parameter value

### 5.51.2.5 def sparksee.Query.set\_stream ( self, stream, handler )

Sets a query stream handler.

[Query](#) streams handlers are created and destroyed by the caller.

Parameters

<i>stream</i>	[in] The stream name
<i>handler</i>	[in] <a href="#">Query</a> stream handler

Returns

The previous handler, or NULL if it does not exists

## 5.52 sparksee.QueryContext Class Reference

[Query](#) context interface.

Public Member Functions

- def [new\\_query](#) (self)  
*Creates a new [Query](#).*
- def [\\_\\_init\\_\\_](#) (self)  
*Default constructor.*

### 5.52.1 Detailed Description

[Query](#) context interface.

A [QueryContext](#) contains and manages the resources required to run a [Query](#). A [Session](#) is one example of a [QueryContext](#) connected to a [Sparksee](#) database. The applications can implement their own contexts to run queries out of [Sparksee](#).

#### Author

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

## 5.53 sparksee.QueryLanguage Class Reference

The supported query languages.

#### Static Public Attributes

- int [SPARKSEE\\_ALGEBRA](#) = 0  
*The internal [Sparksee](#) Algebra.*
- int [SPARKSEE\\_CYPHER](#) = 1  
*The [Sparksee](#) Cypher Language inspired by the OpenCypher [Query](#) Language.*

### 5.53.1 Detailed Description

The supported query languages.

## 5.54 sparksee.QueryStream Class Reference

[Query](#) stream interface.

#### Public Member Functions

- def [fetch](#) (self, list)  
*Gets the next row and moves the iterator forward.*
- def [prepare](#) (self, list)  
*Prepares the stream before it is started.*
- def [start](#) (self, list)  
*Starts the stream.*

### 5.54.1 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>

### 5.54.2 Member Function Documentation

#### 5.54.2.1 `def sparksee.QueryStream.fetch ( self, 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

<i>list</i>	[out] Storage for the new rows
-------------	--------------------------------

##### Returns

TRUE if there is a row or end of sequence, FALSE on error

#### 5.54.2.2 `def sparksee.QueryStream.prepare ( self, list )`

Prepares the stream before it is started.

##### Parameters

<i>list</i>	[in] Optional list of arguments
-------------	---------------------------------

##### Returns

FALSE on error

#### 5.54.2.3 `def sparksee.QueryStream.start ( self, list )`

Starts the stream.

##### Parameters

<i>list</i>	[in] Optional list of input ResultSets
-------------	--

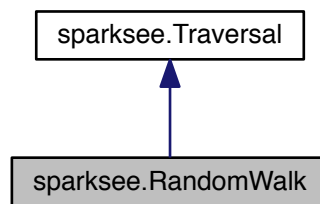
##### Returns

FALSE on error

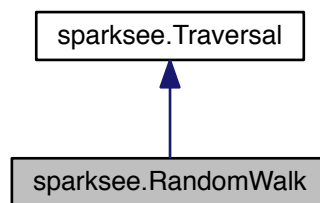
## 5.55 `sparksee.RandomWalk` Class Reference

[RandomWalk](#) class.

Inheritance diagram for sparksee.RandomWalk:



Collaboration diagram for sparksee.RandomWalk:



### Public Member Functions

- def `exclude_nodes` (self, nodes)  
*Set which nodes can't be used.*
- def `set_default_weight` (self, weight)  
*Sets the default weight for those cases when a given edge does not have a weight attribute set.*
- def `add_all_edge_types` (self, dir)  
*Allows for traversing all edge types of the graph.*
- def `exclude_edges` (self, edges)  
*Set which edges can't be used.*
- def `set_seed` (self, seed)  
*Sets the seed of the random walk.*
- def `set_edge_weight_attribute_type` (self, attr)  
*Sets the attribute to use as edge weight.*
- def `set_in_out_parameter` (self, val)  
*Sets the In-Out parameter of the `RandomWalk`.*
- def `add_edge_type` (self, type, dir)  
*Allows for traversing edges of the given type.*
- def `__init__` (self, session, node)  
*Builds the `RandomWalk`.*

- def `add_node_type` (self, type)
  - Allows for traversing nodes of the given type.*
- def `set_return_parameter` (self, val)
  - Sets the return parameter of the `RandomWalk`.*
- def `next` (self)
  - Gets the next object of the traversal.*
- def `has_next` (self)
  - Gets if there are more objects to be traversed.*
- def `get_current_depth` (self)
  - Returns the depth of the current node.*
- def `reset` (self, start\_node)
  - Sets the starting node of the `RandomWalk`.*
- def `add_all_node_types` (self)
  - Allows for traversing all node types of the graph.*
- def `set_maximum_hops` (self, maxhops)
  - Sets the maximum hops restriction.*
- def `__iter__` (self)
  - Gets a new `TraversalIterator`.*
- def `__next__` (self)
  - Used in `next()`*
- def `close` (self)
  - Closes the `Traversal` instance.*
- def `is_closed` (self)
  - Gets if `Traversal` has been closed or not.*

### 5.55.1 Detailed Description

`RandomWalk` class.

Implements the `RandomWalk` algorithm

#### Author

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

### 5.55.2 Constructor & Destructor Documentation

#### 5.55.2.1 `def sparksee.RandomWalk.__init__( self, session, node )`

Builds the `RandomWalk`.

#### Parameters

<code>session</code>	[in] The session to use
<code>node</code>	[in] The starting node of the traversal

### 5.55.3 Member Function Documentation

5.55.3.1 `def sparksee.Traversal.__iter__( self )` [inherited]

Gets a new TraversalIterator.

#### Returns

TraversalIterator instance

5.55.3.2 `def sparksee.Traversal.__next__( self )` [inherited]

Used in [next\(\)](#)

#### Returns

The next element

5.55.3.3 `def sparksee.RandomWalk.add_all_edge_types( self, dir )`

Allows for traversing all edge types of the graph.

#### Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

5.55.3.4 `def sparksee.RandomWalk.add_edge_type( self, type, dir )`

Allows for traversing edges of the given type.

If the edge type was already added, the existing direction is overwritten

#### Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

5.55.3.5 `def sparksee.RandomWalk.add_node_type( self, type )`

Allows for traversing nodes of the given type.

#### Parameters

<i>type</i>	The node type to add
-------------	----------------------

5.55.3.6 `def sparksee.Traversal.close( self )` [inherited]

Closes the [Traversal](#) instance.

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

#### 5.55.3.7 `def sparksee.RandomWalk.exclude_edges ( self, 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

<code>edges</code>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------------	---

#### 5.55.3.8 `def sparksee.RandomWalk.exclude_nodes ( self, 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

<code>nodes</code>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------------	---

#### 5.55.3.9 `def sparksee.RandomWalk.get_current_depth ( self )`

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.

#### 5.55.3.10 `def sparksee.RandomWalk.has_next ( self )`

Gets if there are more objects to be traversed.

##### Returns

TRUE if there are more objects, FALSE otherwise.

#### 5.55.3.11 `def sparksee.Traversal.is_closed ( self )` `[inherited]`

Gets if [Traversal](#) has been closed or not.

##### See also

[close\(\)](#)

##### Returns

TRUE if the [Traversal](#) instance has been closed, FALSE otherwise.

5.55.3.12 `def sparksee.RandomWalk.next ( self )`

Gets the next object of the traversal.

#### Returns

A node or edge identifier.

5.55.3.13 `def sparksee.RandomWalk.reset ( self, start_node )`

Sets the starting node of the [RandomWalk](#).

This method resets the [RandomWalk](#).

sparksee::gdb::Error

#### Parameters

<i>start_node</i>	null
-------------------	------

5.55.3.14 `def sparksee.RandomWalk.set_default_weight ( self, weight )`

Sets the default weight for those cases when a given edge does not have a weight attribute set.

Default: 0.0

#### Parameters

<i>weight</i>	[in] The default weight
---------------	-------------------------

5.55.3.15 `def sparksee.RandomWalk.set_edge_weight_attribute_type ( self, 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

<i>attr</i>	[in] The attribute type to use as a weight. Default: InvalidAttribute
-------------	---

5.55.3.16 `def sparksee.RandomWalk.set_in_out_parameter ( self, val )`

Sets the In-Out parameter of the [RandomWalk](#).

#### Parameters

<i>val</i>	The In-Out parameter to set. Default: 1.0
------------	---



### 5.55.3.17 def sparksee.RandomWalk.set\_maximum\_hops ( self, maxhops )

Sets the maximum hops restriction.

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

#### Parameters

<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
----------------	---

### 5.55.3.18 def sparksee.RandomWalk.set\_return\_parameter ( self, val )

Sets the return parameter of the [RandomWalk](#).

#### Parameters

<i>val</i>	The return parameter to set. Default: 1.0
------------	---

### 5.55.3.19 def sparksee.RandomWalk.set\_seed ( self, seed )

Sets the seed of the random walk.

#### Parameters

<i>seed</i>	The seed to generate the random numbers that drive the random walk
-------------	--

## 5.56 sparksee.ResultSet Class Reference

[ResultSet](#) class.

#### Public Member Functions

- def [rewind](#) (self)  
*Positions the cursor before the first row.*
- def [get\\_column\\_data\\_type](#) (self, index)  
*Gets the datatype for the given column.*
- def [get\\_column](#) (self, index)  
*Gets the value for the given column.*
- def [get\\_num\\_columns](#) (self)  
*Gets the number of columns.*
- def [get\\_column\\_name](#) (self, index)  
*Gets the name for the given column.*
- def [next](#) (self)  
*Fetches the next row.*
- def [get\\_j\\_s\\_o\\_n](#) (self, rows)

- Returns rows in JSON format.*
- def `get_column` (self, index, value)  
*Gets the value for the given column.*
  - def `get_column_index` (self, name)  
*Gets the column index for the given column name.*
  - def `is_closed` (self)  
*Gets if the `ResultSet` instance has been closed or not.*

### 5.56.1 Detailed Description

`ResultSet` class.

#### Author

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

### 5.56.2 Member Function Documentation

#### 5.56.2.1 def sparksee.ResultSet.get\_column ( self, index )

Gets the value for the given column.

`QueryException` if a database access error occurs.

#### Parameters

<i>index</i>	[in] Column index.
--------------	--------------------

#### Returns

The `Value` of the given column.

Referenced by `sparksee.ResultSet.get_column()`.

#### 5.56.2.2 def sparksee.ResultSet.get\_column ( self, index, value )

Gets the value for the given column.

`QueryException` if a database access error occurs.

#### Parameters

<i>index</i>	[in] Column index.
<i>value</i>	[in out] <code>Value</code> .

References `sparksee.ResultSet.get_column()`.

5.56.2.3 `def sparksee.ResultSet.get_column_data_type ( self, index )`

Gets the datatype for the given column.

Parameters

<i>index</i>	[in] Column index.
--------------	--------------------

Returns

[DataType](#) for the given column.

5.56.2.4 `def sparksee.ResultSet.get_column_index ( self, name )`

Gets the column index for the given column name.

Parameters

<i>name</i>	[in] Column name.
-------------	-------------------

Returns

Column index.

5.56.2.5 `def sparksee.ResultSet.get_column_name ( self, index )`

Gets the name for the given column.

Parameters

<i>index</i>	[in] Column index.
--------------	--------------------

Returns

Column name.

5.56.2.6 `def sparksee.ResultSet.get_j_s_o_n ( self, rows )`

Returns rows in JSON format.

Rows are returned from the current position.

Parameters

<i>rows</i>	[in] Maximum number of rows
-------------	-----------------------------

**Returns**

JSON representation of the next <rows> rows in the resultset

**5.56.2.7 def sparksee.ResultSet.get\_num\_columns ( self )**

Gets the number of columns.

Columns are in the range [0...COLUMNS).

**Returns**

The number of columns.

**5.56.2.8 def sparksee.ResultSet.is\_closed ( self )**

Gets if the [ResultSet](#) instance has been closed or not.

**See also**

`#close()`

**Returns**

TRUE if the [ResultSet](#) instance has been closed, FALSE otherwise.

**5.56.2.9 def sparksee.ResultSet.next ( self )**

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.

## 5.57 sparksee.ResultSetList Class Reference

[ResultSet](#) list.

**Public Member Functions**

- def [clear](#) (self)  
*Clears the list.*
- def [get](#) (self, index)  
*Returns the [ResultSet](#) at the specified position in the list.*
- def [\\_\\_iter\\_\\_](#) (self)  
*Gets a new [ResultSetListIterator](#).*
- def [iterator](#) (self)  
*Gets a new [ResultSetListIterator](#).*
- def [\\_\\_init\\_\\_](#) (self)  
*Constructor.*
- def [count](#) (self)  
*Number of elements in the list.*

### 5.57.1 Detailed Description

[ResultSet](#) list.

It stores a [ResultSet](#) list.

#### Author

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

### 5.57.2 Constructor & Destructor Documentation

#### 5.57.2.1 `def sparksee.ResultSetList.__init__( self )`

Constructor.

This creates an empty list.

### 5.57.3 Member Function Documentation

#### 5.57.3.1 `def sparksee.ResultSetList.__iter__( self )`

Gets a new [ResultSetListIterator](#).

#### Returns

[ResultSetListIterator](#) instance

#### 5.57.3.2 `def sparksee.ResultSetList.count( self )`

Number of elements in the list.

#### Returns

Number of elements in the list.

#### 5.57.3.3 `def sparksee.ResultSetList.get( self, index )`

Returns the [ResultSet](#) at the specified position in the list.

#### Parameters

<i>index</i>	[in] Index of the element to return, starting at 0.
--------------	---

#### 5.57.3.4 `def sparksee.ResultSetList.iterator( self )`

Gets a new [ResultSetListIterator](#).

**Returns**

[ResultSetListIterator](#) instance.

**5.58 sparksee.ResultSetListIterator Class Reference**

[ResultSetList](#) iterator class.

**Public Member Functions**

- def [next](#) (self)  
*Moves to the next element.*
- def [has\\_next](#) (self)  
*Gets if there are more elements.*
- def [\\_\\_next\\_\\_](#) (self)  
*Used in [next\(\)](#)*

**5.58.1 Detailed Description**

[ResultSetList](#) iterator class.

Iterator to traverse all the values into a [ResultSetList](#) instance.

**Author**

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

**5.58.2 Member Function Documentation****5.58.2.1 def sparksee.ResultSetListIterator.\_\_next\_\_ ( self )**

Used in [next\(\)](#)

**Returns**

The next element

**5.58.2.2 def sparksee.ResultSetListIterator.has\_next ( self )**

Gets if there are more elements.

**Returns**

TRUE if there are more elements, FALSE otherwise.

### 5.58.2.3 `def sparksee.ResultSetListIterator.next ( self )`

Moves to the next element.

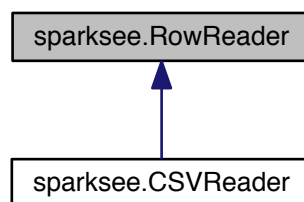
Returns

The next element.

## 5.59 `sparksee.RowReader` Class Reference

`RowReader` interface.

Inheritance diagram for `sparksee.RowReader`:



### Public Member Functions

- `def close (self)`  
*Closes the reader.*
- `def reset (self)`  
*Moves the reader to the beginning.*
- `def read (self, row)`  
*Reads the next row as a string array.*
- `def get_row (self)`  
*The row number for the current row.*

### 5.59.1 Detailed Description

`RowReader` interface.

Common interface for those readers which get the data as a 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>

### 5.59.2 Member Function Documentation

#### 5.59.2.1 `def sparksee.RowReader.close ( self )`

Closes the reader.

**Exceptions**

<i>IOError</i>	If the close fails.
----------------	---------------------

**5.59.2.2 def sparksee.RowReader.get\_row ( self )**

The row number for the current row.

**Returns**

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

**Exceptions**

<i>IOError</i>	If it fails.
----------------	--------------

**5.59.2.3 def sparksee.RowReader.read ( self, row )**

Reads the next row as a string array.

**Parameters**

<i>row</i>	[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**

<i>IOError</i>	If bad things happen during the read.
----------------	---------------------------------------

**5.59.2.4 def sparksee.RowReader.reset ( self )**

Moves the reader to the beginning.

Restarts the reader.

**Returns**

true if the reader can be restarted, false otherwise.

**Exceptions**

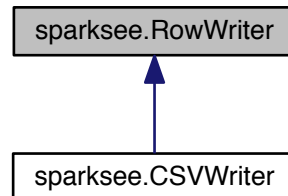
<i>IOError</i>	If bad things happen during the restart.
----------------	--



## 5.60 sparksee.RowWriter Class Reference

[RowWriter](#) interface.

Inheritance diagram for sparksee.RowWriter:



### Public Member Functions

- def `close` (self)  
*Closes the writer.*
- def `write` (self, row)  
*Writes the next row.*

### 5.60.1 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>

### 5.60.2 Member Function Documentation

#### 5.60.2.1 def sparksee.RowWriter.close ( self )

Closes the writer.

#### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the close fails.

## 5.60.2.2 def sparksee.RowWriter.write ( self, row )

Writes the next row.

## Parameters

<i>row</i>	[in] Row of data.
------------	-------------------

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If bad things happen during the write.

## 5.61 sparksee.ScriptParser Class Reference

[ScriptParser](#).

## Public Member Functions

- def [\\_\\_init\\_\\_](#) (self)  
*Constructor.*
- def [set\\_error\\_log](#) (self, path)  
*Sets the error log.*
- def [generate\\_schema\\_script](#) (self, path, db)  
*Writes an script with the schema definition for the given database.*
- def [parse](#) (self, path, execute, locale\_str)  
*Parses the given input file.*
- def [set\\_output\\_log](#) (self, path)  
*Sets the output log.*

## 5.61.1 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>

## 5.61.2 Member Function Documentation

## 5.61.2.1 def sparksee.ScriptParser.generate\_schema\_script ( self, path, db )

Writes an script with the schema definition for the given database.

## Parameters

<i>path</i>	[in] Filename of the script to be written.
<i>db</i>	[in] <a href="#">Database</a> .

## Exceptions

<i>IOError</i>	If bad things happen opening or writing the file.
----------------	---

5.61.2.2 `def sparksee.ScriptParser.parse ( self, path, execute, locale_str )`

Parses the given input file.

## Parameters

<i>path</i>	[in] Input file path.
<i>execute</i>	[in] If TRUE the script is executed, if FALSE it is just parsed.
<i>locale_str</i>	[in] The locale string for reading the input file. See <a href="#">CSVReader</a> .

## Returns

TRUE if ok, FALSE in case of error.

## Exceptions

<i>IOError</i>	If bad things happen opening the file.
----------------	--

5.61.2.3 `def sparksee.ScriptParser.set_error_log ( self, path )`

Sets the error log.

If not set, error log corresponds to standard error output.

## Parameters

<i>path</i>	[in] Path of the error log.
-------------	-----------------------------

## Exceptions

<i>IOError</i>	If bad things happen opening the file.
----------------	--

5.61.2.4 `def sparksee.ScriptParser.set_output_log ( self, path )`

Sets the output log.

If not set, output log corresponds to standard output.

## Parameters

<i>path</i>	[in] Path of the output log.
-------------	------------------------------

## Exceptions

<i>IOError</i>	If bad things happen opening the file.
----------------	--

## 5.62 sparksee.Session Class Reference

[Session](#) class.

## Public Member Functions

- def [rollback](#) (self)  
*Rollbacks a transaction.*
- def [pre\\_commit](#) (self)  
*PreCommits a transaction.*
- def [begin](#) (self)  
*Begins a transaction.*
- def [begin\\_update](#) (self)  
*Begins an update transaction.*
- def [new\\_query](#) (self, lang)  
*Creates a new [Query](#).*
- def [commit](#) (self)  
*Commits a transaction.*
- def [get\\_graph](#) (self)  
*Gets the [Graph](#) instance.*
- def [close](#) (self)  
*Closes the [Session](#) instance.*
- def [new\\_objects](#) (self)  
*Creates a new [Objects](#) instance.*
- def [get\\_in\\_memory\\_pool\\_capacity](#) (self)  
*Gets the capacity of the in-memory pool.*
- def [is\\_closed](#) (self)  
*Gets if [Session](#) instance has been closed or not.*

## 5.62.1 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>

**5.62.2 Member Function Documentation****5.62.2.1 `def sparksee.Session.close ( self )`**

Closes the [Session](#) instance.

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

**5.62.2.2 `def sparksee.Session.get_graph ( self )`**

Gets the [Graph](#) instance.

**Returns**

The [Graph](#) instance.

**5.62.2.3 `def sparksee.Session.get_in_memory_pool_capacity ( self )`**

Gets the capacity of the in-memory pool.

**Returns**

Returns the capacity of the in-memory pool

**5.62.2.4 `def sparksee.Session.is_closed ( self )`**

Gets if [Session](#) instance has been closed or not.

**See also**

[close\(\)](#)

**Returns**

TRUE if the [Session](#) instance has been closed, FALSE otherwise.

**5.62.2.5 `def sparksee.Session.new_objects ( self )`**

Creates a new [Objects](#) instance.

**Returns**

The new [Objects](#) instance.

**5.62.2.6 `def sparksee.Session.new_query ( self, lang )`**

Creates a new [Query](#).

## Parameters

<i>lang</i>	The query language to create the query for
-------------	--

## 5.62.2.7 def sparksee.Session.pre\_commit ( self )

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).

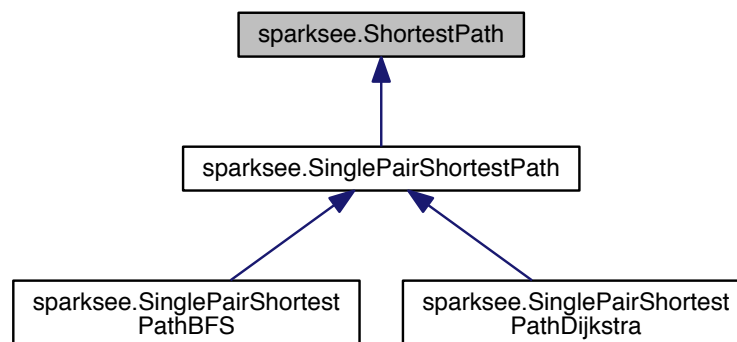
## Returns

The transaction id. This has to be used in RedoPrecommitted in case of a recovery process.

## 5.63 sparksee.ShortestPath Class Reference

[ShortestPath](#) class.

Inheritance diagram for sparksee.ShortestPath:



## Public Member Functions

- def [add\\_edge\\_type](#) (self, type, dir)  
*Allows for traversing edges of the given type.*
- def [run](#) (self)  
*Runs the algorithm.*
- def [exclude\\_nodes](#) (self, nodes)  
*Set which nodes can't be used.*
- def [add\\_all\\_edge\\_types](#) (self, dir)  
*Allows for traversing all edge types of the graph.*
- def [add\\_node\\_type](#) (self, type)  
*Allows for traversing nodes of the given type.*

- def `exclude_edges` (self, edges)  
*Set which edges can't be used.*
- def `close` (self)  
*Closes the `ShortestPath` instance.*
- def `set_maximum_hops` (self, maxhops)  
*Sets the maximum hops restriction.*
- def `add_all_node_types` (self)  
*Allows for traversing all node types of the graph.*
- def `is_closed` (self)  
*Gets if `ShortestPath` has been closed or not.*

### 5.63.1 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>

### 5.63.2 Member Function Documentation

#### 5.63.2.1 `def sparksee.ShortestPath.add_all_edge_types ( self, dir )`

Allows for traversing all edge types of the graph.

##### Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

#### 5.63.2.2 `def sparksee.ShortestPath.add_edge_type ( self, type, dir )`

Allows for traversing edges of the given type.

##### Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

#### 5.63.2.3 `def sparksee.ShortestPath.add_node_type ( self, type )`

Allows for traversing nodes of the given type.

## Parameters

<i>type</i>	null
-------------	------

5.63.2.4 def sparksee.ShortestPath.close ( *self* )

Closes the [ShortestPath](#) instance.

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

5.63.2.5 def sparksee.ShortestPath.exclude\_edges ( *self*, *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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

5.63.2.6 def sparksee.ShortestPath.exclude\_nodes ( *self*, *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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

5.63.2.7 def sparksee.ShortestPath.is\_closed ( *self* )

Gets if [ShortestPath](#) has been closed or not.

See also

[close\(\)](#)

## Returns

TRUE if the [ShortestPath](#) instance has been closed, FALSE otherwise.

5.63.2.8 def sparksee.ShortestPath.run ( *self* )

Runs the algorithm.

This method can only be called once.

5.63.2.9 def sparksee.ShortestPath.set\_maximum\_hops ( *self*, *maxhops* )

Sets the maximum hops restriction.

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



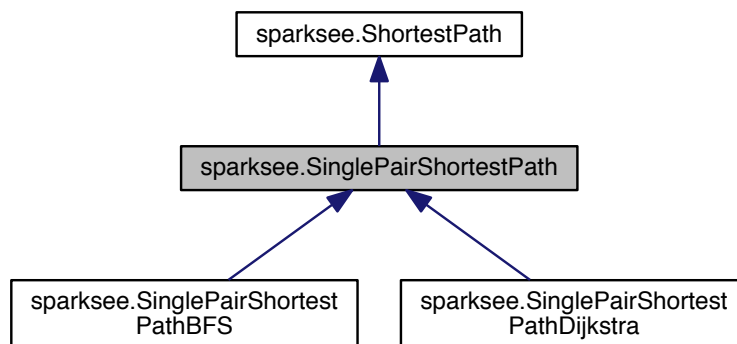
## Parameters

<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
----------------	---

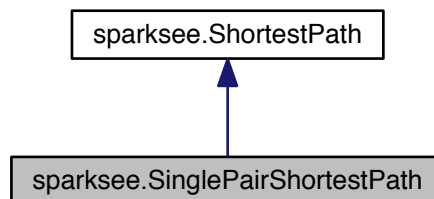
## 5.64 sparksee.SinglePairShortestPath Class Reference

[SinglePairShortestPath](#) class.

Inheritance diagram for sparksee.SinglePairShortestPath:



Collaboration diagram for sparksee.SinglePairShortestPath:



## Public Member Functions

- def `run` (self)  
*Runs the algorithm.*
- def `exclude_nodes` (self, nodes)  
*Set which nodes can't be used.*

- def `add_all_edge_types` (self, dir)  
*Allows for traversing all edge types of the graph.*
- def `exclude_edges` (self, edges)  
*Set which edges can't be used.*
- def `add_edge_type` (self, type, dir)  
*Allows for traversing edges of the given type.*
- def `get_path_as_nodes` (self)  
*Gets the shortest path between the source node and the destination node as an ordered set of nodes.*
- def `get_path_as_edges` (self)  
*Gets the shortest path between the source node and the destination node as an ordered set of edges.*
- def `add_node_type` (self, type)  
*Allows for traversing nodes of the given type.*
- def `exists` (self)  
*Returns TRUE If a path exists or FALSE otherwise.*
- def `get_cost` (self)  
*Gets the cost of the shortest path.*
- def `set_maximum_hops` (self, maxhops)  
*Sets the maximum hops restriction.*
- def `add_all_node_types` (self)  
*Allows for traversing all node types of the graph.*
- def `close` (self)  
*Closes the `ShortestPath` instance.*
- def `is_closed` (self)  
*Gets if `ShortestPath` has been closed or not.*

#### 5.64.1 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>

#### 5.64.2 Member Function Documentation

##### 5.64.2.1 `def sparksee.SinglePairShortestPath.add_all_edge_types ( self, dir )`

Allows for traversing all edge types of the graph.

#### Parameters

<code>dir</code>	[in] Edge direction.
------------------	----------------------

5.64.2.2 `def sparksee.SinglePairShortestPath.add_edge_type ( self, type, dir )`

Allows for traversing edges of the given type.

Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

5.64.2.3 `def sparksee.SinglePairShortestPath.add_node_type ( self, type )`

Allows for traversing nodes of the given type.

Parameters

<i>type</i>	null
-------------	------

5.64.2.4 `def sparksee.ShortestPath.close ( self )` [inherited]

Closes the [ShortestPath](#) instance.

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

5.64.2.5 `def sparksee.SinglePairShortestPath.exclude_edges ( self, 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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

5.64.2.6 `def sparksee.SinglePairShortestPath.exclude_nodes ( self, 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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

5.64.2.7 `def sparksee.SinglePairShortestPath.get_cost ( self )`

Gets the cost of the shortest path.

The cost for unweighted algorithms is the number of hops of the shortest path. For weighted algorithms, the cost is the sum of the costs of the edges of the shortest path.

**Returns**

The cost of the shortest path.

**5.64.2.8** `def sparksee.SinglePairShortestPath.get_path_as_edges ( self )`

Gets the shortest path between the source node and the destination node as an ordered set of edges.

**Returns**

Ordered set of edge identifiers.

**5.64.2.9** `def sparksee.SinglePairShortestPath.get_path_as_nodes ( self )`

Gets the shortest path between the source node and the destination node as an ordered set of nodes.

**Returns**

Ordered set of node identifiers.

**5.64.2.10** `def sparksee.ShortestPath.is_closed ( self )` [inherited]

Gets if [ShortestPath](#) has been closed or not.

**See also**

[close\(\)](#)

**Returns**

TRUE if the [ShortestPath](#) instance has been closed, FALSE otherwise.

**5.64.2.11** `def sparksee.SinglePairShortestPath.run ( self )`

Runs the algorithm.

This method can only be called once.

**5.64.2.12** `def sparksee.SinglePairShortestPath.set_maximum_hops ( self, maxhops )`

Sets the maximum hops restriction.

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

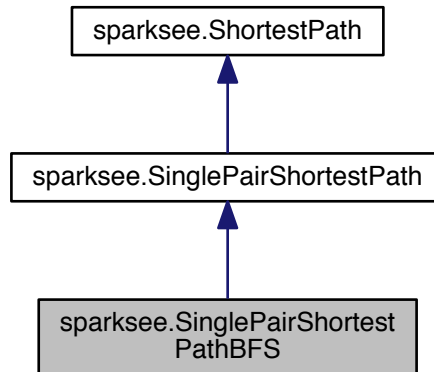
## Parameters

<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
----------------	---

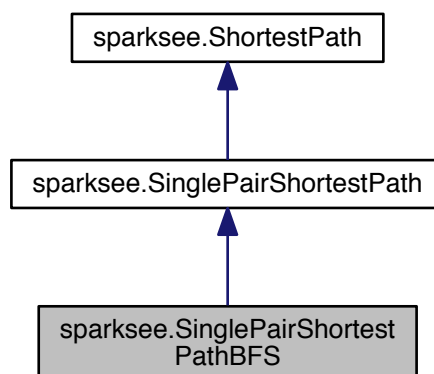
## 5.65 sparksee.SinglePairShortestPathBFS Class Reference

[SinglePairShortestPathBFS](#) class.

Inheritance diagram for sparksee.SinglePairShortestPathBFS:



Collaboration diagram for sparksee.SinglePairShortestPathBFS:



## Public Member Functions

- def `exclude_nodes` (self, nodes)  
*Set which nodes can't be used.*
- def `get_cost` (self)  
*Gets the cost of the shortest path.*
- def `run` (self)  
*Executes the algorithm.*
- def `add_all_edge_types` (self, dir)  
*Allows for traversing all edge types of the graph.*
- def `exclude_edges` (self, edges)  
*Set which edges can't be used.*
- def `__init__` (self, session, src, dst)  
*Creates a new instance.*
- def `add_edge_type` (self, type, dir)  
*Allows for traversing edges of the given type.*
- def `get_path_as_nodes` (self)  
*Gets the shortest path between the source node and the destination node as an ordered set of nodes.*
- def `add_node_type` (self, type)  
*Allows for traversing nodes of the given type.*
- def `exists` (self)  
*Returns TRUE if a path exists or FALSE otherwise.*
- def `check_only_existence` (self)  
*Set that only the path existence must be calculated and not the path itself.*
- def `get_path_as_edges` (self)  
*Gets the shortest path between the source node and the destination node as an ordered set of edges.*
- def `set_maximum_hops` (self, maxhops)  
*Sets the maximum hops restriction.*
- def `add_all_node_types` (self)  
*Allows for traversing all node types of the graph.*
- def `close` (self)  
*Closes the `ShortestPath` instance.*
- def `is_closed` (self)  
*Gets if `ShortestPath` has been closed or not.*

## 5.65.1 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>

## 5.65.2 Constructor &amp; Destructor Documentation

## 5.65.2.1 def sparksee.SinglePairShortestPathBFS.\_\_init\_\_( self, session, src, dst )

Creates a new instance.

## Parameters

<i>session</i>	[in] <a href="#">Session</a> to get the graph from and perform traversal.
<i>src</i>	[in] Source node.
<i>dst</i>	[dst] Destination node.

## 5.65.3 Member Function Documentation

5.65.3.1 `def sparksee.SinglePairShortestPathBFS.add_all_edge_types ( self, dir )`

Allows for traversing all edge types of the graph.

## Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

5.65.3.2 `def sparksee.SinglePairShortestPathBFS.add_edge_type ( self, type, dir )`

Allows for traversing edges of the given type.

## Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

5.65.3.3 `def sparksee.SinglePairShortestPathBFS.add_node_type ( self, type )`

Allows for traversing nodes of the given type.

## Parameters

<i>type</i>	null
-------------	------

5.65.3.4 `def sparksee.SinglePairShortestPathBFS.check_only_existence ( self )`

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.

5.65.3.5 `def sparksee.ShortestPath.close ( self )` `[inherited]`

Closes the [ShortestPath](#) instance.

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

5.65.3.6 `def sparksee.SinglePairShortestPathBFS.exclude_edges ( self, 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

<code>edges</code>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------------	---

5.65.3.7 `def sparksee.SinglePairShortestPathBFS.exclude_nodes ( self, 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

<code>nodes</code>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------------	---

5.65.3.8 `def sparksee.SinglePairShortestPathBFS.get_cost ( self )`

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.

5.65.3.9 `def sparksee.SinglePairShortestPathBFS.get_path_as_edges ( self )`

Gets the shortest path between the source node and the destination node as an ordered set of edges.

## Returns

Ordered set of edge identifiers.

5.65.3.10 `def sparksee.SinglePairShortestPathBFS.get_path_as_nodes ( self )`

Gets the shortest path between the source node and the destination node as an ordered set of nodes.

## Returns

Ordered set of node identifiers.

5.65.3.11 `def sparksee.ShortestPath.is_closed ( self )` [inherited]

Gets if [ShortestPath](#) has been closed or not.

## See also

[close\(\)](#)

## Returns

TRUE if the [ShortestPath](#) instance has been closed, FALSE otherwise.

5.65.3.12 `def sparksee.SinglePairShortestPathBFS.set_maximum_hops ( self, maxhops )`

Sets the maximum hops restriction.

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

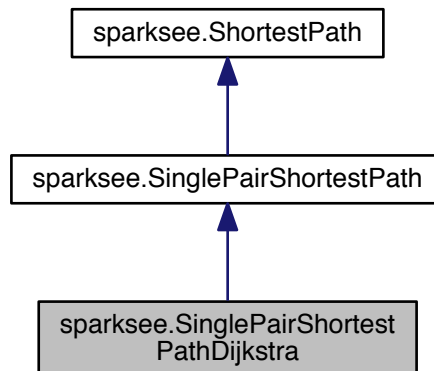
## Parameters

<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
----------------	---

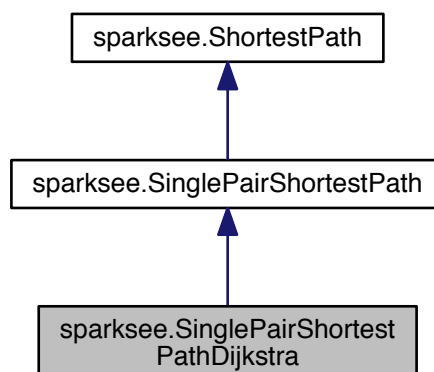
## 5.66 sparksee.SinglePairShortestPathDijkstra Class Reference

[SinglePairShortestPathDijkstra](#) class.

Inheritance diagram for sparksee.SinglePairShortestPathDijkstra:



Collaboration diagram for sparksee.SinglePairShortestPathDijkstra:



### Public Member Functions

- def `__init__` (self, session, src, dst)  
*Creates a new instance.*
- def `exclude_nodes` (self, nodes)  
*Set which nodes can't be used.*
- def `get_cost` (self)  
*Gets the cost of the shortest path.*
- def `run` (self)  
*Executes the algorithm.*
- def `add_all_edge_types` (self, dir)  
*Allows for traversing all edge types of the graph.*
- def `exclude_edges` (self, edges)  
*Set which edges can't be used.*
- def `add_edge_type` (self, type, dir)  
*Allows for traversing edges of the given type.*
- def `set_unweighted_edge_cost` (self, weight)  
*Sets the weight assigned to the unweighted edges.*
- def `get_path_as_nodes` (self)  
*Gets the shortest path between the source node and the destination node as an ordered set of nodes.*
- def `set_dynamic_edge_cost_callback` (self, dyn\_cost\_calculator)  
*Set a class callback to dynamically calculate the cost of the edges.*
- def `add_node_type` (self, type)  
*Allows for traversing nodes of the given type.*
- def `exists` (self)  
*Returns TRUE if a path exists or FALSE otherwise.*
- def `get_path_as_edges` (self)  
*Gets the shortest path between the source node and the destination node as an ordered set of edges.*
- def `add_weighted_edge_type` (self, type, dir, attr)  
*Allows for traversing edges of the given type using the given attribute as the weight.*
- def `set_maximum_hops` (self, maxhops)  
*Sets the maximum hops restriction.*
- def `add_all_node_types` (self)  
*Allows for traversing all node types of the graph.*
- def `close` (self)  
*Closes the `ShortestPath` instance.*
- def `is_closed` (self)  
*Gets if `ShortestPath` has been closed or not.*

#### 5.66.1 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>

## 5.66.2 Constructor &amp; Destructor Documentation

## 5.66.2.1 def sparksee.SinglePairShortestPathDijkstra.\_\_init\_\_( self, session, src, dst )

Creates a new instance.

## Parameters

<i>session</i>	[in] <a href="#">Session</a> to get the graph from and perform traversal.
<i>src</i>	[in] Source node.
<i>dst</i>	[dst] Destination node.

## 5.66.3 Member Function Documentation

## 5.66.3.1 def sparksee.SinglePairShortestPathDijkstra.add\_all\_edge\_types( self, dir )

Allows for traversing all edge types of the graph.

## Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

## 5.66.3.2 def sparksee.SinglePairShortestPathDijkstra.add\_edge\_type( self, type, dir )

Allows for traversing edges of the given type.

## Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

## 5.66.3.3 def sparksee.SinglePairShortestPathDijkstra.add\_node\_type( self, type )

Allows for traversing nodes of the given type.

## Parameters

<i>type</i>	null
-------------	------

## 5.66.3.4 def sparksee.SinglePairShortestPathDijkstra.add\_weighted\_edge\_type( self, type, dir, attr )

Allows for traversing edges of the given type using the given attribute as the weight.

## Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.
<i>attr</i>	[in] <a href="#">Attribute</a> to be used as the weight. It must be a global attribute or an attribute of the given edge type.

#### 5.66.3.5 `def sparksee.ShortestPath.close ( self ) [inherited]`

Closes the [ShortestPath](#) instance.

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

#### 5.66.3.6 `def sparksee.SinglePairShortestPathDijkstra.exclude_edges ( self, 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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

#### 5.66.3.7 `def sparksee.SinglePairShortestPathDijkstra.exclude_nodes ( self, 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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

#### 5.66.3.8 `def sparksee.SinglePairShortestPathDijkstra.get_cost ( self )`

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.

#### 5.66.3.9 `def sparksee.SinglePairShortestPathDijkstra.get_path_as_edges ( self )`

Gets the shortest path between the source node and the destination node as an ordered set of edges.

##### Returns

Ordered set of edge identifiers.

#### 5.66.3.10 `def sparksee.SinglePairShortestPathDijkstra.get_path_as_nodes ( self )`

Gets the shortest path between the source node and the destination node as an ordered set of nodes.

##### Returns

Ordered set of node identifiers.

5.66.3.11 `def sparksee.ShortestPath.is_closed ( self ) [inherited]`

Gets if [ShortestPath](#) has been closed or not.

See also

[close\(\)](#)

Returns

TRUE if the [ShortestPath](#) instance has been closed, FALSE otherwise.

5.66.3.12 `def sparksee.SinglePairShortestPathDijkstra.set_dynamic_edge_cost_callback ( self, dyn_cost_calculator )`

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

<i>dyn_cost_calculator</i>	[in] Class callback to calculate the edge costs
----------------------------	---

5.66.3.13 `def sparksee.SinglePairShortestPathDijkstra.set_maximum_hops ( self, maxhops )`

Sets the maximum hops restriction.

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

Parameters

<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
----------------	---

5.66.3.14 `def sparksee.SinglePairShortestPathDijkstra.set_unweighted_edge_cost ( self, 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

<i>weight</i>	[in] The weight value for unweighted edges.
---------------	---

## 5.67 sparksee.SinglePairShortestPathDijkstraDynamicCost Class Reference

Defines how to calculate an edge weight.

## Public Member Functions

- def `calculate_edge_cost` (self, source\_node, source\_cost, source\_level, target\_node, edge, edge\_weight\_attr)

*Dynamic calculation of an edge weight.*

### 5.67.1 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>

### 5.67.2 Member Function Documentation

- 5.67.2.1 `def sparksee.SinglePairShortestPathDijkstraDynamicCost.calculate_edge_cost ( self, source_node, source_cost, source_level, target_node, edge, edge_weight_attr )`

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 overridden in a derived user class to implement a dynamic edge weight, that may need the sourceCost and sourceLevels to calculate the weight.

#### Parameters

<code>source_node</code>	[in] The current node
<code>source_cost</code>	[in] Current total cost up to the source node
<code>source_level</code>	[in] Current path size up to the source node
<code>target_node</code>	[in] The next node
<code>edge</code>	[in] The edge to the next node that this method must weight
<code>edge_weight_attr</code>	[in] The attribute that stores the edge weight or InvalidAttribute

#### Returns

Returns the current weight of the edge ( $\geq 0$ ) or  $< 0$  if the edge can not be used.

## 5.68 sparksee.Sparksee Class Reference

[Sparksee](#) class.

### Public Member Functions

- def [restore\\_encrypted\\_backup](#) (self, path, backup\_file, config, key\_in\_hex, iv\_in\_hex)  
*Restores a [Database](#) from an encrypted backup file.*
- def [resize\\_in\\_memory\\_pool](#) (self, new\_capacity)  
*Resizes the in memory pool allocator.*
- def [create](#) (self, path, alias, config)  
*Creates a new [Database](#) instance.*
- def [verify\\_checksums](#) (self, path)  
*Verifies the Checksum integrity of the given image file and it's recovery log file.*
- def [remove\\_checksums](#) (self, path)  
*Converts a database WITH checksums into a database WITHOUT checksums.*
- def [restore](#) (self, path, backup\_file)  
*Restores a [Database](#) from a backup file.*
- def [add\\_checksums](#) (self, path)  
*Converts a database WITHOUT checksums into a database WITH checksums.*
- def [restore\\_encrypted\\_backup](#) (self, path, backup\_file, key\_in\_hex, iv\_in\_hex)  
*Restores a [Database](#) from an encrypted backup file.*
- def [decrypt](#) (self, path)  
*Converts a database WITH encryption into a database WITHOUT encryption.*
- def [restore](#) (self, path, backup\_file, config)  
*Restores a [Database](#) from a backup file.*
- def [create](#) (self, path, alias)  
*Creates a new [Database](#) instance.*
- def [open](#) (self, path, read\_only)  
*Opens an existing [Database](#) instance.*
- def [\\_\\_init\\_\\_](#) (self, config)  
*Creates a new instance.*
- def [open](#) (self, path, read\_only, config)  
*Opens an existing [Database](#) instance.*
- def [set\\_unrecoverable\\_error\\_callback](#) (self, callback)  
*Sets the unrecoverable error callback.*
- def [encrypt](#) (self, path)  
*Converts a database WITHOUT encryption into an encrypted database.*
- def [close](#) (self)  
*Closes the [Sparksee](#) instance.*
- def [get\\_in\\_memory\\_pool\\_capacity](#) (self)  
*Gets the capacity of the in-memory pool (in Megabytes)*

### Static Public Attributes

- float [VERSION](#) = 5.0  
*[Sparksee](#) version.*

#### 5.68.1 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>



## 5.68.2 Constructor & Destructor Documentation

### 5.68.2.1 `def sparksee.Sparksee.__init__( self, config )`

Creates a new instance.

#### Parameters

<i>config</i>	[in/out] <a href="#">Sparksee</a> configuration (may be updated).
---------------	---

## 5.68.3 Member Function Documentation

### 5.68.3.1 `def sparksee.Sparksee.add_checksums( self, 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::FileNotFoundException`  
`sparksee.gdb::Error`

#### Parameters

<i>path</i>	The path to the database image file
-------------	-------------------------------------

#### Returns

Returns true if the checksum could be added or false otherwise.

#### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

### 5.68.3.2 `def sparksee.Sparksee.close( self )`

Closes the [Sparksee](#) instance.

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

### 5.68.3.3 `def sparksee.Sparksee.create( self, path, alias, config )`

Creates a new [Database](#) instance.

#### Parameters

<i>path</i>	[in] <a href="#">Database</a> storage file.
<i>alias</i>	[in] <a href="#">Database</a> alias name.
<i>config</i>	[in] Use a specific configuration instead of the one in this <a href="#">Sparksee</a>

**Returns**

A [Database](#) instance.

**Exceptions**

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file cannot be created.

Referenced by `sparksee.Sparksee.create()`.

#### 5.68.3.4 `def sparksee.Sparksee.create ( self, path, alias )`

Creates a new [Database](#) instance.

**Parameters**

<i>path</i>	[in] <a href="#">Database</a> storage file.
<i>alias</i>	[in] <a href="#">Database</a> alias name.

**Returns**

A [Database](#) instance.

**Exceptions**

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file cannot be created.

References `sparksee.Sparksee.create()`.

#### 5.68.3.5 `def sparksee.Sparksee.decrypt ( self, 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::FileNotFoundsparksee::gdb::Error`

**Parameters**

<i>path</i>	The path to the database image file
-------------	-------------------------------------

**Returns**

Returns true if the database could be unencrypted. The errors are returned with an exception.

**Exceptions**

<i>RuntimeError</i>	null
<i>IOError</i>	null

### 5.68.3.6 def sparksee.Sparksee.encrypt ( self, path )

Converts a database WITHOUT encryption into an encrypted database.

Any error found is logged. The database (and it's recovery log if present) does is NOT encrypted. The encryption had already been configured using [SparkseeConfig](#). sparksee::gdb::FileNotFoundExceptionsparksee::gdb::Error

#### Parameters

<i>path</i>	The path to the database image file
-------------	-------------------------------------

#### Returns

Returns true if the database could be encrypted. The errors are returned with an exception.

#### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

### 5.68.3.7 def sparksee.Sparksee.get\_in\_memory\_pool\_capacity ( self )

Gets the capacity of the in-memory pool (in Megabytes)

#### Returns

Returns the capacity of the in memory pool

### 5.68.3.8 def sparksee.Sparksee.open ( self, path, read\_only )

Opens an existing [Database](#) instance.

#### Parameters

<i>path</i>	[in] <a href="#">Database</a> storage file.
<i>read_only</i>	[in] If TRUE, open <a href="#">Database</a> in read-only mode.

#### Returns

A [Database](#) instance.

#### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file does not exist.

Referenced by sparksee.Sparksee.open().

5.68.3.9 `def sparksee.Sparksee.open ( self, path, read_only, 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.

#### Parameters

<i>path</i>	[in] <a href="#">Database</a> storage file.
<i>read_only</i>	[in] If TRUE, open <a href="#">Database</a> in read-only mode.
<i>config</i>	[in] Use a specific configuration instead of the one in this <a href="#">Sparksee</a>

#### Returns

A [Database](#) instance.

#### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file does not exist.

References `sparksee.Sparksee.open()`.

5.68.3.10 `def sparksee.Sparksee.remove_checksums ( self, 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↔`  
`NotFoundExceptions``sparksee::gdb::Error`

#### Parameters

<i>path</i>	The path to the database image file
-------------	-------------------------------------

#### Returns

Returns true if the checksum could be removed. The errors are returned with an exception.

#### Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

5.68.3.11 `def sparksee.Sparksee.resize_in_memory_pool ( self, new_capacity )`

Resizes the in memory pool allocator.

## Parameters

<i>new_capacity</i>	The new capacity of the in memory pool allocator
---------------------	--

## Returns

Returns true if was successful

5.68.3.12 `def sparksee.Sparksee.restore ( self, path, backup_file )`

Restores a [Database](#) from a backup file.

See the [Graph](#) class Backup method.

## Parameters

<i>path</i>	[in] <a href="#">Database</a> storage file.
<i>backup_file</i>	[in] The Backup file to be restored.

## Returns

A [Database](#) instance.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file cannot be created, or the exported data file does not exists.

Referenced by `sparksee.Sparksee.restore()`.

5.68.3.13 `def sparksee.Sparksee.restore ( self, path, backup_file, config )`

Restores a [Database](#) from a backup file.

See the [Graph](#) class Backup method.

## Parameters

<i>path</i>	[in] <a href="#">Database</a> storage file.
<i>backup_file</i>	[in] The Backup file to be restored.
<i>config</i>	[in] Use a specific configuration instead of the one in this <a href="#">Sparksee</a>

## Returns

A [Database](#) instance.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file cannot be created, or the exported data file does not exists.

References `sparksee.Sparksee.restore()`.

5.68.3.14 `def sparksee.Sparksee.restore_encrypted_backup ( self, path, backup_file, config, key_in_hex, iv_in_hex )`

Restores a [Database](#) from an encrypted backup file.

See the [Graph](#) class `EncryptedBackup` method.

## Parameters

<i>path</i>	[in] <a href="#">Database</a> storage file.
<i>backup_file</i>	[in] The Backup file to be restored.
<i>config</i>	[in] Use a specific configuration instead of the one in this <a href="#">Sparksee</a>
<i>key_in_hex</i>	[In] The AES encryption Key of the backup file as a hexadecimal string (8, 16 or 32 bytes).
<i>iv_in_hex</i>	[In] The AES Initialization Vector of the backup file as a hexadecimal string (16 bytes).

## Returns

A [Database](#) instance.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file cannot be created, or the exported data file does not exists.

Referenced by `sparksee.Sparksee.restore_encrypted_backup()`.

5.68.3.15 `def sparksee.Sparksee.restore_encrypted_backup ( self, path, backup_file, key_in_hex, iv_in_hex )`

Restores a [Database](#) from an encrypted backup file.

See the [Graph](#) class `EncryptedBackup` method.

## Parameters

<i>path</i>	[in] <a href="#">Database</a> storage file.
<i>backup_file</i>	[in] The Backup file to be restored.
<i>key_in_hex</i>	[In] The AES encryption Key of the backup file as a hexadecimal string (8, 16 or 32 bytes).
<i>iv_in_hex</i>	[In] The AES Initialization Vector of the backup file as a hexadecimal string (16 bytes).

## Returns

A [Database](#) instance.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If the given file cannot be created, or the exported data file does not exists.

References `sparksee.Sparksee.restore_encrypted_backup()`.

5.68.3.16 `def sparksee.Sparksee.set_unrecoverable_error_callback ( self, callback )`

Sets the unrecoverable error callback.

## Parameters

<i>callback</i>	The unrecoverable error callback to set
-----------------	---

5.68.3.17 `def sparksee.Sparksee.verify_checksums ( self, 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 function returns false, in any other error it throws an exception.

Any error found is logged as a WARNING.

`sparksee::gdb::FileNotFoundExceptions``sparksee::gdb::Error`

## Parameters

<i>path</i>	The path to the image file
-------------	----------------------------

## Returns

Returns true if the checksum verification succeeded. Returns false if an invalid checksum was detected.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

5.69 `sparksee.SparkseeConfig` Class Reference

[Sparksee](#) configuration class.

## Public Member Functions

- def [set\\_recovery\\_enabled](#) (self, status)  
*Enables or disables the recovery.*
- def [set\\_license\\_pre\\_download\\_days](#) (self, days)  
*Start trying to automatically download a new license at the specified number of days before expiration.*
- def [get\\_extent\\_pages](#) (self)  
*Gets the number of pages per extent.*
- def [save\\_all](#) (self)  
*Save all the current configuration settings in the specified config file.*
- def [get\\_a\\_e\\_s\\_key](#) (self)  
*Get the AES encryption key in a hexadecimal encoded string.*
- def [get\\_pool\\_persistent\\_max\\_size](#) (self)  
*Gets the maximum size for the persistent pool in number of frames.*
- def [get\\_pool\\_temporary\\_min\\_size](#) (self)  
*Gets the minimum size for the temporary pool in number of frames.*
- def [get\\_call\\_stack\\_dump](#) (self)  
*Gets whether the signals will be captured to dump the call stack or not.*
- def [set\\_recovery\\_cache\\_max\\_size](#) (self, extents)  
*Sets the maximum size for the recovery log cache in extents.*
- def [set\\_a\\_e\\_s\\_encryption\\_enabled](#) (self, key\_in\_hex, iv\_in\_hex)  
*Enables storage encryption using AES with the given key and iv.*
- def [get\\_high\\_availability\\_enabled](#) (self)  
*Gets whether high availability mode is enabled or disabled.*
- def [get\\_pool\\_persistent\\_min\\_size](#) (self)  
*Gets the minimum size for the persistent pool in number of frames.*
- def [set\\_checksum\\_enabled](#) (self, status)  
*Enables or disables the storage checksum usage.*
- def [set\\_pool\\_persistent\\_min\\_size](#) (self, frames)  
*Sets the minimum size for the persistent pool in number of frames.*
- def [set\\_cache\\_statistics\\_enabled](#) (self, status)  
*Enables or disables cache statistics.*
- def [set\\_recovery\\_checkpoint\\_time](#) (self, micro\_seconds)  
*Sets the delay time (in microseconds) between automatic checkpoints.*
- def [set\\_encryption\\_disabled](#) (self)  
*Disables storage encryption.*
- def [set\\_recovery\\_log\\_file](#) (self, file\_path)  
*Sets the recovery log file.*
- def [set\\_pool\\_frame\\_size](#) (self, extents)  
*Sets the size of a pool frame in number of extents.*
- def [get\\_recovery\\_log\\_file](#) (self)  
*Gets the recovery log file.*
- def [set\\_high\\_availability\\_enabled](#) (self, status)  
*Enables or disables high availability mode.*
- def [get\\_cache\\_statistics\\_snapshot\\_time](#) (self)  
*Gets the cache statistics snapshot time in microseconds.*
- def [get\\_extent\\_size](#) (self)  
*Gets the size of a extent.*
- def [get\\_sparksee\\_config\\_file](#) (self)  
*Gets the config file path.*
- def [set\\_extent\\_size](#) (self, k\_bytes)



- Sets the size of the extents in KB.*
- def [set\\_high\\_availability\\_i\\_p](#) (self, ip)  
*Sets the IP address and port of the instance.*
- def [get\\_pool\\_temporary\\_max\\_size](#) (self)  
*Gets the maximum size for the temporary pool in number of frames.*
- def [set\\_in\\_mem\\_alloc\\_size](#) (self, size)  
*Sets the in-memory allocator size.*
- def [\\_\\_init\\_\\_](#) (self, path)  
*Creates a new instance with a specific config file.*
- def [set\\_high\\_availability\\_master\\_history](#) (self, file\_path)  
*Sets the master's history log.*
- def [set\\_tmp\\_enabled](#) (self, enabled)  
*Sets whether to use temporary storage for computations or not.*
- def [get\\_log\\_file](#) (self)  
*Gets the log file.*
- def [get\\_tmp\\_enabled](#) (self)  
*Gets whether using temporary storage for computations is enabled.*
- def [get\\_pool\\_frame\\_size](#) (self)  
*Gets the size of a pool frame in number of extents.*
- def [get\\_cache\\_statistics\\_enabled](#) (self)  
*Gets whether cache statistics are enabled or disabled.*
- def [set\\_high\\_availability\\_coordinators](#) (self, ip)  
*Sets the coordinators address and port list.*
- def [set\\_high\\_availability\\_synchronization](#) (self, micro\_seconds)  
*Sets the synchronization polling time.*
- def [get\\_client\\_id](#) (self)  
*Gets the client identifier.*
- def [get\\_cache\\_statistics\\_file](#) (self)  
*Gets the cache statistics log file.*
- def [set\\_sparksee\\_config\\_file](#) (self, path)  
*Sets the config file path.*
- def [save](#) (self)  
*Save the current configuration in the specified config file.*
- def [set\\_pool\\_persistent\\_max\\_size](#) (self, frames)  
*Sets the maximum size for the persistent pool in number of frames.*
- def [get\\_log\\_level](#) (self)  
*Gets the log level.*
- def [set\\_license](#) (self, key)  
*Sets the license key.*
- def [get\\_download\\_status](#) (self)  
*Gets a message with the license download result.*
- def [set\\_license\\_id](#) (self, license\_id)  
*Set the license identifier.*
- def [get\\_license](#) (self)  
*Gets the license key.*
- def [get\\_high\\_availability\\_i\\_p](#) (self)  
*Gets the IP address and port of the instance.*
- def [set\\_a\\_e\\_s\\_encryption\\_enabled](#) (self, key\_size)  
*Enables storage encryption using AES and GENERATES a key and an iv.*
- def [get\\_in\\_mem\\_alloc\\_size](#) (self)  
*Gets the in-memory allocator size.*

- def [get\\_license\\_pre\\_download\\_days](#) (self)  
*Get the number of days before expiration when a new license will be downloaded.*
- def [set\\_call\\_stack\\_dump](#) (self, status)  
*Sets whether the signals will be captured to dump the call stack or not.*
- def [download\\_license](#) (self)  
*Try to download a license key Can be used to explicitly download a license when the autodownload is disabled (LicensePreDownloadDays == -1).*
- def [get\\_recovery\\_enabled](#) (self)  
*Gets whether the recovery is enabled or disabled.*
- def [get\\_license\\_request](#) (self)  
*Get information useful to manually request a license.*
- def [set\\_cache\\_statistics\\_file](#) (self, file\_path)  
*Sets the cache statistics log file.*
- def [get\\_tmp\\_folder](#) (self)  
*Gets the temporary folder used for temporary staging.*
- def [get\\_pool\\_partitions](#) (self)  
*Gets the number of partitions in each PartitionedPool.*
- def [get\\_high\\_availability\\_coordinators](#) (self)  
*Gets the coordinators address and port list.*
- def [get\\_recovery\\_checkpoint\\_time](#) (self)  
*Gets the delay time (in microseconds) between automatic checkpoints.*
- def [set\\_log\\_file](#) (self, file\_path)  
*Sets the log file.*
- def [set\\_extent\\_pages](#) (self, pages)  
*Sets the number of pages per extent.*
- def [set\\_client\\_id](#) (self, client\_id)  
*Set the client identifier.*
- def [set\\_log\\_level](#) (self, level)  
*Sets the log level.*
- def [get\\_recovery\\_cache\\_max\\_size](#) (self)  
*Gets the maximum size for the recovery log cache in extents.*
- def [get\\_checksum\\_enabled](#) (self)  
*Gets whether the storage checksum usage is enabled or disabled.*
- def [get\\_high\\_availability\\_synchronization](#) (self)  
*Gets the synchronization polling time.*
- def [get\\_high\\_availability\\_master\\_history](#) (self)  
*Gets the master's history log.*
- def [get\\_license\\_id](#) (self)  
*Gets the license identifier.*
- def [\\_\\_init\\_\\_](#) (self, path, client\_id, license\_id)  
*Creates a new instance with a specific config file and IDs.*
- def [get\\_aes\\_iv](#) (self)  
*Get the AES initialization vector in a hexadecimal encoded string.*
- def [get\\_encryption\\_enabled](#) (self)  
*Gets whether the storage encryption is enabled or disabled.*
- def [set\\_pool\\_partitions](#) (self, pools)  
*Sets the number of pools in each PartitionedPool.*
- def [get\\_rollback\\_enabled](#) (self)  
*Gets whether the rollback is enabled or disabled.*
- def [set\\_pool\\_temporary\\_max\\_size](#) (self, frames)  
*Sets the maximum size for the temporary pool in number of frames.*

- def [get\\_cache\\_max\\_size](#) (self)  
*Gets the maximum size for the cache (all pools) in MB.*
- def [set\\_rollback\\_enabled](#) (self, status)  
*Enables or disables the rollback.*
- def [set\\_tmp\\_folder](#) (self, tmp\_folder)  
*Sets the temporary folder used for temporary staging.*
- def [set\\_cache\\_statistics\\_snapshot\\_time](#) (self, micro\_seconds)  
*Sets the cache statistics snapshot time.*
- def [set\\_cache\\_max\\_size](#) (self, mega\_bytes)  
*Sets the maximum size for the cache (all pools) in MB.*
- def [\\_\\_init\\_\\_](#) (self)  
*Creates a new instance.*
- def [set\\_pool\\_temporary\\_min\\_size](#) (self, frames)  
*Sets the minimum size for the temporary pool in number of frames.*
- def [download\\_expected](#) (self)  
*Check if a new license will be automatically downloaded with the current settings.*

### 5.69.1 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 overridden with values from a properties file (see [Sparksee↔ Properties](#) class). Also, this settings can be overridden 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 [SparkseeProperties](#)).

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>

### 5.69.2 Constructor & Destructor Documentation

#### 5.69.2.1 def sparksee.SparkseeConfig.\_\_init\_\_( self, 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 exist, it will be created.

#### Parameters

<i>path</i>	[in] File path to the config file.
-------------	------------------------------------

Referenced by `sparksee.SparkseeConfig.__init__()`.

5.69.2.2 `def sparksee.SparkseeConfig.__init__( self, path, client_id, license_id )`

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 exist, it will be created.

#### Parameters

<i>path</i>	[in] File path to the config file.
<i>client_id</i>	[in] The client identifier.
<i>license_id</i>	[in] The license identifier.

References `sparksee.SparkseeConfig.__init__()`.

5.69.2.3 `def sparksee.SparkseeConfig.__init__( self )`

Creates a new instance.

[Values](#) are set with default values.

References `sparksee.SparkseeConfig.__init__()`.

### 5.69.3 Member Function Documentation

5.69.3.1 `def sparksee.SparkseeConfig.download_license( self )`

Try to download a license key Can be used to explicitly download a license when the autdownload 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.

5.69.3.2 `def sparksee.SparkseeConfig.get_aes_iv( self )`

Get the AES initialization vector in a hexadecimal encoded string.

#### Returns

The AES initialization vector as an hexadecimal string.

5.69.3.3 `def sparksee.SparkseeConfig.get_aes_key ( self )`

Get the AES encryption key in a hexadecimal encoded string.

**Returns**

The AES encryption Key as an hexadecimal string.

5.69.3.4 `def sparksee.SparkseeConfig.get_cache_max_size ( self )`

Gets the maximum size for the cache (all pools) in MB.

**Returns**

The maximum size for the cache (all pools) in MB.

5.69.3.5 `def sparksee.SparkseeConfig.get_cache_statistics_enabled ( self )`

Gets whether cache statistics are enabled or disabled.

**Returns**

TRUE if cache statistics are enabled, FALSE otherwise.

5.69.3.6 `def sparksee.SparkseeConfig.get_cache_statistics_file ( self )`

Gets the cache statistics log file.

Useless if cache statistics are disabled.

**Returns**

The cache statistics log file.

5.69.3.7 `def sparksee.SparkseeConfig.get_cache_statistics_snapshot_time ( self )`

Gets the cache statistics snapshot time in microseconds.

Useless if cache statistics are disabled.

**Returns**

The cache statistics snapshot time in microseconds.

5.69.3.8 `def sparksee.SparkseeConfig.get_call_stack_dump ( self )`

Gets whether the signals will be captured to dump the call stack or not.

**Returns**

TRUE if the signals must be captured, FALSE otherwise.

5.69.3.9 `def sparksee.SparkseeConfig.get_checksum_enabled ( self )`

Gets whether the storage checksum usage is enabled or disabled.

**Returns**

TRUE if the checksum is enabled, FALSE otherwise.

5.69.3.10 `def sparksee.SparkseeConfig.get_client_id ( self )`

Gets the client identifier.

**Returns**

The client identifier.

5.69.3.11 `def sparksee.SparkseeConfig.get_download_status ( self )`

Gets a message with the license download result.

It can be used when the DownloadLicense call failed.

**Returns**

The log download result.

5.69.3.12 `def sparksee.SparkseeConfig.get_encryption_enabled ( self )`

Gets whether the storage encryption is enabled or disabled.

**Returns**

TRUE if the encryption is enabled, FALSE otherwise.

5.69.3.13 `def sparksee.SparkseeConfig.get_extent_pages ( self )`

Gets the number of pages per extent.

**Returns**

The number of pages per extent.

5.69.3.14 `def sparksee.SparkseeConfig.get_extent_size ( self )`

Gets the size of a extent.

**Returns**

The size of a extent in KB.

5.69.3.15 `def sparksee.SparkseeConfig.get_high_availability_coordinators ( self )`

Gets the coordinators address and port list.

**Returns**

The coordinators address and port list.

5.69.3.16 `def sparksee.SparkseeConfig.get_high_availability_enabled ( self )`

Gets whether high availability mode is enabled or disabled.

**Returns**

TRUE if high availability mode is enabled, FALSE otherwise.

5.69.3.17 `def sparksee.SparkseeConfig.get_high_availability_i_p ( self )`

Gets the IP address and port of the instance.

**Returns**

The IP address and port of the instance.

5.69.3.18 `def sparksee.SparkseeConfig.get_high_availability_master_history ( self )`

Gets the master's history log.

**Returns**

The master's history log.

5.69.3.19 `def sparksee.SparkseeConfig.get_high_availability_synchronization ( self )`

Gets the synchronization polling time.

**Returns**

The Synchronization polling time.

5.69.3.20 `def sparksee.SparkseeConfig.get_in_mem_alloc_size ( self )`

Gets the in-memory allocator size.

**Returns**

Returns the in-memory allocator size



5.69.3.21 `def sparksee.SparkseeConfig.get_license ( self )`

Gets the license key.

**Returns**

The license key.

5.69.3.22 `def sparksee.SparkseeConfig.get_license_id ( self )`

Gets the license identifier.

**Returns**

The license identifier.

5.69.3.23 `def sparksee.SparkseeConfig.get_license_pre_download_days ( self )`

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.

5.69.3.24 `def sparksee.SparkseeConfig.get_log_file ( self )`

Gets the log file.

**Returns**

The log file.

5.69.3.25 `def sparksee.SparkseeConfig.get_log_level ( self )`

Gets the log level.

**Returns**

The [LogLevel](#).

5.69.3.26 `def sparksee.SparkseeConfig.get_pool_frame_size ( self )`

Gets the size of a pool frame in number of extents.

**Returns**

The size of a pool frame in number of extents.

5.69.3.27 `def sparksee.SparkseeConfig.get_pool_partitions ( self )`

Gets the number of partitions in each PartitionedPool.

**Returns**

The number of partitions in each PartitionedPool.

5.69.3.28 `def sparksee.SparkseeConfig.get_pool_persistent_max_size ( self )`

Gets the maximum size for the persistent pool in number of frames.

**Returns**

The maximum size for the persistent pool in number of frames.

5.69.3.29 `def sparksee.SparkseeConfig.get_pool_persistent_min_size ( self )`

Gets the minimum size for the persistent pool in number of frames.

**Returns**

The minimum size for the persistent pool in number of frames.

5.69.3.30 `def sparksee.SparkseeConfig.get_pool_temporary_max_size ( self )`

Gets the maximum size for the temporary pool in number of frames.

**Returns**

The maximum size for the temporary pool in number of frames.

5.69.3.31 `def sparksee.SparkseeConfig.get_pool_temporary_min_size ( self )`

Gets the minimum size for the temporary pool in number of frames.

**Returns**

The minimum size for the temporary pool in number of frames.

5.69.3.32 `def sparksee.SparkseeConfig.get_recovery_cache_max_size ( self )`

Gets the maximum size for the recovery log cache in extents.

**Returns**

The maximum size for the recovery log cache in extents.

5.69.3.33 `def sparksee.SparkseeConfig.get_recovery_checkpoint_time ( self )`

Gets the delay time (in microseconds) between automatic checkpoints.

**Returns**

The delay time (in microseconds) between automatic checkpoints.

5.69.3.34 `def sparksee.SparkseeConfig.get_recovery_enabled ( self )`

Gets whether the recovery is enabled or disabled.

**Returns**

TRUE if the recovery is enabled, FALSE otherwise.

5.69.3.35 `def sparksee.SparkseeConfig.get_recovery_log_file ( self )`

Gets the recovery log file.

**Returns**

The recovery log file.

5.69.3.36 `def sparksee.SparkseeConfig.get_rollback_enabled ( self )`

Gets whether the rollback is enabled or disabled.

**Returns**

TRUE if the rollback is enabled, FALSE otherwise.

5.69.3.37 `def sparksee.SparkseeConfig.get_sparksee_config_file ( self )`

Gets the config file path.

**Returns**

Returns the configured path or an empty string

5.69.3.38 `def sparksee.SparkseeConfig.get_tmp_enabled ( self )`

Gets whether using temporary storage for computations is enabled.

**Returns**

True if enabled

5.69.3.39 `def sparksee.SparkseeConfig.get_tmp_folder ( self )`

Gets the temporary folder used for temporary staging.

#### Returns

The temporary folder path

5.69.3.40 `def sparksee.SparkseeConfig.save ( self )`

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.

5.69.3.41 `def sparksee.SparkseeConfig.save_all ( self )`

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 CONTAIN 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.

5.69.3.42 `def sparksee.SparkseeConfig.set_a_e_s_encryption_enabled ( self, key_in_hex, iv_in_hex )`

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

<code>key_in_hex</code>	[In] The AES encryption Key as a hexadecimal string (8, 16 or 32 bytes).
<code>iv_in_hex</code>	[In] The AES Initialization Vector as a hexadecimal string (16 bytes).

#### Returns

Returns true if the encryption had been enabled with the given arguments or false otherwise.

Referenced by `sparksee.SparkseeConfig.set_a_e_s_encryption_enabled()`.

5.69.3.43 `def sparksee.SparkseeConfig.set_a_e_s_encryption_enabled ( self, key_size )`

Enables storage encryption using AES and GENERATES a key and an iv.

YOU MUST GET AND KEEP SAFE THE GENERATED KEY AND IV! The key and initialization vector will not be saved in the config file.

#### Parameters

<i>key_size</i>	[In] The key size in bytes (8, 16 or 32).
-----------------	---

#### Returns

Returns true if the encryption had been enabled.

References `sparksee.SparkseeConfig.set_a_e_s_encryption_enabled()`.

5.69.3.44 `def sparksee.SparkseeConfig.set_cache_max_size ( self, mega_bytes )`

Sets the maximum size for the cache (all pools) in MB.

#### Parameters

<i>mega_bytes</i>	[in] The maximum size for the cache (all pools) in MB. It must be non-negative.
-------------------	---

5.69.3.45 `def sparksee.SparkseeConfig.set_cache_statistics_enabled ( self, status )`

Enables or disables cache statistics.

#### Parameters

<i>status</i>	[in] If TRUE this enables cache statistics, if FALSE this disables cache statistics.
---------------	--

5.69.3.46 `def sparksee.SparkseeConfig.set_cache_statistics_file ( self, file_path )`

Sets the cache statistics log file.

Useless if cache statistics are disabled.

#### Parameters

<i>file_path</i>	[in] The cache statistics log file.
------------------	-------------------------------------

5.69.3.47 `def sparksee.SparkseeConfig.set_cache_statistics_snapshot_time ( self, micro_seconds )`

Sets the cache statistics snapshot time.

Useless if cache statistics are disabled.

## Parameters

<i>micro_seconds</i>	[in] The cache statistics snapshot time in microseconds.
----------------------	--

5.69.3.48 `def sparksee.SparkseeConfig.set_call_stack_dump ( self, status )`

Sets whether the signals will be captured to dump the call stack or not.

## Parameters

<i>status</i>	[in] If TRUE signals must be captured.
---------------	--

5.69.3.49 `def sparksee.SparkseeConfig.set_checksum_enabled ( self, status )`

Enables or disables the storage checksum usage.

## Parameters

<i>status</i>	[in] If TRUE this enables the checksum, if FALSE then disables it.
---------------	--

5.69.3.50 `def sparksee.SparkseeConfig.set_client_id ( self, client_id )`

Set the client identifier.

If you don't have a client identifier, you may want to register at <http://sparsity-technologies.com/#sparksee>

## Parameters

<i>client_id</i>	[in] The client identifier.
------------------	-----------------------------

5.69.3.51 `def sparksee.SparkseeConfig.set_extent_pages ( self, pages )`

Sets the number of pages per extent.

## Parameters

<i>pages</i>	[in] The number of pages. It must be at least 1 page and the page size must be greater than or equal to 4KB.
--------------	--

5.69.3.52 `def sparksee.SparkseeConfig.set_extent_size ( self, k_bytes )`

Sets the size of the extents in KB.

## Parameters

<i>k_bytes</i>	[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.
----------------	--

5.69.3.53 `def sparksee.SparkseeConfig.set_high_availability_coordinators ( self, ip )`

Sets the coordinators address and port list.

## Parameters

<i>ip</i>	[in] The coordinators address and port list.
-----------	--

5.69.3.54 `def sparksee.SparkseeConfig.set_high_availability_enabled ( self, status )`

Enables or disables high availability mode.

## Parameters

<i>status</i>	[in] If TRUE this enables high availability mode, if FALSE this disables high availability mode.
---------------	--

5.69.3.55 `def sparksee.SparkseeConfig.set_high_availability_i_p ( self, ip )`

Sets the IP address and port of the instance.

## Parameters

<i>ip</i>	[in] The IP address and port of the instance.
-----------	---

5.69.3.56 `def sparksee.SparkseeConfig.set_high_availability_master_history ( self, file_path )`

Sets the master's history log.

## Parameters

<i>file_path</i>	[in] The master's history log.
------------------	--------------------------------

5.69.3.57 `def sparksee.SparkseeConfig.set_high_availability_synchronization ( self, micro_seconds )`

Sets the synchronization polling time.

## Parameters

<i>micro_seconds</i>	[in] The synchronization polling time.
----------------------	--

5.69.3.58 `def sparksee.SparkseeConfig.set_in_mem_alloc_size ( self, size )`

Sets the in-memory allocator size.

Parameters

<i>size</i>	The size to set the in-memory allocator to
-------------	--

5.69.3.59 `def sparksee.SparkseeConfig.set_license ( self, key )`

Sets the license key.

Parameters

<i>key</i>	[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.
------------	---

5.69.3.60 `def sparksee.SparkseeConfig.set_license_id ( self, license_id )`

Set the license identifier.

If you don't have a license identifier, you may want to register at <http://sparsity-technologies.com/#sparksee>

Parameters

<i>license↔ _id</i>	[in] The license identifier.
-------------------------	------------------------------

5.69.3.61 `def sparksee.SparkseeConfig.set_license_pre_download_days ( self, days )`

Start trying to automatically download a new license at the specified number of days before expiration.

Parameters

<i>days</i>	[in] Number of days before expiration or -1 if the license should never be downloaded.
-------------	--

5.69.3.62 `def sparksee.SparkseeConfig.set_log_file ( self, file_path )`

Sets the log file.

Parameters

<i>file_path</i>	[in] The log file.
------------------	--------------------

5.69.3.63 `def sparksee.SparkseeConfig.set_log_level ( self, level )`

Sets the log level.



## Parameters

<i>level</i>	[in] The <a href="#">LogLevel</a> .
--------------	-------------------------------------

5.69.3.64 `def sparksee.SparkseeConfig.set_pool_frame_size ( self, extents )`

Sets the size of a pool frame in number of extents.

## Parameters

<i>extents</i>	[in] The size of a pool frame in number of extents. It must be non-negative.
----------------	--

5.69.3.65 `def sparksee.SparkseeConfig.set_pool_partitions ( self, pools )`

Sets the number of pools in each PartitionedPool.

## Parameters

<i>pools</i>	[in] The number of pools in each PartitionedPool. It must be non-negative.
--------------	--

5.69.3.66 `def sparksee.SparkseeConfig.set_pool_persistent_max_size ( self, frames )`

Sets the maximum size for the persistent pool in number of frames.

## Parameters

<i>frames</i>	[in] The maximum size for the persistent pool in number of frames. It must be non-negative.
---------------	---

5.69.3.67 `def sparksee.SparkseeConfig.set_pool_persistent_min_size ( self, frames )`

Sets the minimum size for the persistent pool in number of frames.

## Parameters

<i>frames</i>	[in] The minimum size for the persistent pool in number of frames. It must be non-negative.
---------------	---

5.69.3.68 `def sparksee.SparkseeConfig.set_pool_temporary_max_size ( self, frames )`

Sets the maximum size for the temporary pool in number of frames.

## Parameters

<i>frames</i>	[in] The maximum size for the temporary pool in number of frames. It must be non-negative.
---------------	--

5.69.3.69 `def sparksee.SparkseeConfig.set_pool_temporary_min_size ( self, frames )`

Sets the minimum size for the temporary pool in number of frames.

Parameters

<i>frames</i>	[in] The minimum size for the temporary pool in number of frames. It must be non-negative.
---------------	--

5.69.3.70 `def sparksee.SparkseeConfig.set_recovery_cache_max_size ( self, extents )`

Sets the maximum size for the recovery log cache in extents.

Parameters

<i>extents</i>	[in] The maximum size for the recovery log cache in extents. A 0 sets the default value (extents up to 1MB).
----------------	--

5.69.3.71 `def sparksee.SparkseeConfig.set_recovery_checkpoint_time ( self, micro_seconds )`

Sets the delay time (in microseconds) between automatic checkpoints.

Parameters

<i>micro_seconds</i>	[in] The delay time (in microseconds) between automatic checkpoints. A 0 forces a checkpoint after each committed transaction.
----------------------	--

5.69.3.72 `def sparksee.SparkseeConfig.set_recovery_enabled ( self, status )`

Enables or disables the recovery.

Parameters

<i>status</i>	[in] If TRUE this enables the recovery, if FALSE then disables it.
---------------	--

5.69.3.73 `def sparksee.SparkseeConfig.set_recovery_log_file ( self, file_path )`

Sets the recovery log file.

Parameters

<i>file_path</i>	[in] The recovery log file. Left it empty for the default log file (same as <database_file_name>.log)
------------------	---

5.69.3.74 `def sparksee.SparkseeConfig.set_rollback_enabled ( self, status )`

Enables or disables the rollback.

## Parameters

<i>status</i>	[in] If TRUE this enables the rollback, if FALSE then disables it.
---------------	--

5.69.3.75 `def sparksee.SparkseeConfig.set_sparksee_config_file ( self, 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 exist, it will be created.

## Parameters

<i>path</i>	[in] File path to the config file.
-------------	------------------------------------

5.69.3.76 `def sparksee.SparkseeConfig.set_tmp_enabled ( self, enabled )`

Sets whether to use temporary storage for computations or not.

## Parameters

<i>enabled</i>	True to use temporary storage for computation
----------------	---

5.69.3.77 `def sparksee.SparkseeConfig.set_tmp_folder ( self, tmp_folder )`

Sets the temporary folder used for temporary staging.

## Parameters

<i>tmp_folder</i>	The temporary folder to set
-------------------	-----------------------------

## 5.70 sparksee.SparkseeProperties Class Reference

[Sparksee](#) properties file.

### Public Member Functions

- def [clear](#) (self)  
*Remove all the properties.*
- def [get](#) (self, key, def)  
*Gets a property.*
- def [get\\_integer](#) (self, key, def)  
*Gets a property as an integer.*
- def [set\\_h\\_i](#) (self, value)  
*YOU SHOULD NOT USE THIS METHOD.*
- def [load](#) (self, path)

*Loads properties from the given file path.*

- def [get\\_time\\_unit](#) (self, key, def)  
*Gets a property as a time unit.*
- def [get\\_boolean](#) (self, key, def)  
*Gets a property as a boolean.*

### 5.70.1 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.

### 5.70.2 Member Function Documentation

#### 5.70.2.1 def sparksee.SparkseeProperties.get ( self, key, def )

Gets a property.

##### Parameters

<i>key</i>	[in] The name of the property to lookup.
<i>def</i>	[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.

#### 5.70.2.2 def sparksee.SparkseeProperties.get\_boolean ( self, key, def )

Gets a property as a boolean.

##### Parameters

<i>key</i>	[in] The name of the property to lookup.
<i>def</i>	[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.

### 5.70.2.3 `def sparksee.SparkseeProperties.get_integer ( self, key, def )`

Gets a property as an integer.

#### Parameters

<i>key</i>	[in] The name of the property to lookup.
<i>def</i>	[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.

### 5.70.2.4 `def sparksee.SparkseeProperties.get_time_unit ( self, key, 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 beginning 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' or '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

<i>key</i>	[in] The name of the property to lookup.
<i>def</i>	[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.

### 5.70.2.5 `def sparksee.SparkseeProperties.load ( self, path )`

Loads properties from the given file path.

#### Parameters

<i>path</i>	[in] File path to load properties from.
-------------	---

### 5.70.2.6 `def sparksee.SparkseeProperties.set_h_i ( self, value )`

YOU SHOULD NOT USE THIS METHOD.

This method exists only for a specific service.

## Parameters

<i>value</i>	null
--------------	------

## 5.71 sparksee.StringList Class Reference

String list.

### Public Member Functions

- def [clear](#) (self)  
*Clears the list.*
- def [\\_\\_init\\_\\_](#) (self)  
*Constructor.*
- def [\\_\\_iter\\_\\_](#) (self)  
*Gets a new [TypeListIterator](#).*
- def [iterator](#) (self)  
*Gets a new [StringListIterator](#).*
- def [add](#) (self, str)  
*Adds a String at the end of the list.*
- def [count](#) (self)  
*Number of elements in the list.*

### 5.71.1 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>

### 5.71.2 Constructor & Destructor Documentation

#### 5.71.2.1 def sparksee.StringList.\_\_init\_\_( self )

Constructor.

This creates an empty list.

### 5.71.3 Member Function Documentation

#### 5.71.3.1 def sparksee.StringList.\_\_iter\_\_( self )

Gets a new [TypeListIterator](#).

#### Returns

[TypeListIterator](#) instance

#### 5.71.3.2 def sparksee.StringList.add( self, str )

Adds a String at the end of the list.

**Parameters**

<i>str</i>	[in] String.
------------	--------------

**5.71.3.3 def sparksee.StringList.count ( self )**

Number of elements in the list.

**Returns**

Number of elements in the list.

**5.71.3.4 def sparksee.StringList.iterator ( self )**

Gets a new [StringListIterator](#).

**Returns**

[StringListIterator](#) instance.

**5.72 sparksee.StringListIterator Class Reference**

[StringList](#) iterator class.

**Public Member Functions**

- def [next](#) (self)  
*Moves to the next element.*
- def [has\\_next](#) (self)  
*Gets if there are more elements.*
- def [\\_\\_next\\_\\_](#) (self)  
*Used in [next\(\)](#)*

**5.72.1 Detailed Description**

[StringList](#) iterator class.

Iterator to traverse all the strings into a [StringList](#) instance.

**Author**

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

## 5.72.2 Member Function Documentation

## 5.72.2.1 def sparksee.StringListIterator.\_\_next\_\_( self )

Used in [next\(\)](#)

**Returns**

The next element

## 5.72.2.2 def sparksee.StringListIterator.has\_next ( self )

Gets if there are more elements.

**Returns**

TRUE if there are more elements, FALSE otherwise.

## 5.72.2.3 def sparksee.StringListIterator.next ( self )

Moves to the next element.

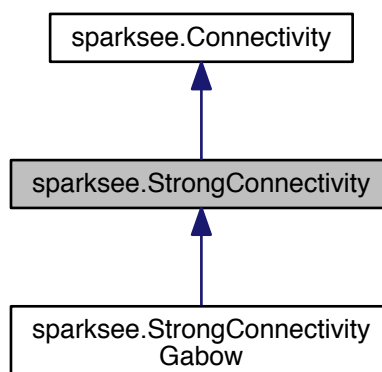
**Returns**

The next element.

## 5.73 sparksee.StrongConnectivity Class Reference

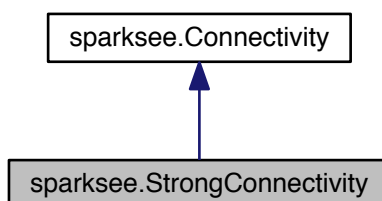
[StrongConnectivity](#) class.

Inheritance diagram for sparksee.StrongConnectivity:





Collaboration diagram for sparksee.StrongConnectivity:



### Public Member Functions

- def [add\\_edge\\_type](#) (self, type, dir)  
*Allows connectivity through edges of the given type.*
- def [run](#) (self)  
*Runs the algorithm in order to find the connected components.*
- def [exclude\\_nodes](#) (self, nodes)  
*Set which nodes can't be used.*
- def [get\\_connected\\_components](#) (self)  
*Returns the results generated by the execution of the algorithm.*
- def [add\\_all\\_edge\\_types](#) (self, dir)  
*Allows connectivity through all edge types of the graph.*
- def [exclude\\_edges](#) (self, edges)  
*Set which edges can't be used.*
- def [add\\_node\\_type](#) (self, t)  
*Allows connectivity through nodes of the given type.*
- def [set\\_materialized\\_attribute](#) (self, attribute\_name)  
*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.*
- def [add\\_all\\_node\\_types](#) (self)  
*Allows connectivity through all node types of the graph.*
- def [close](#) (self)  
*Closes the [Connectivity](#) instance.*
- def [is\\_closed](#) (self)  
*Gets if [Connectivity](#) has been closed or not.*

#### 5.73.1 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>

### 5.73.2 Member Function Documentation

#### 5.73.2.1 `def sparksee.StrongConnectivity.add_all_edge_types ( self, dir )`

Allows connectivity through all edge types of the graph.

##### Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

#### 5.73.2.2 `def sparksee.StrongConnectivity.add_edge_type ( self, type, dir )`

Allows connectivity through edges of the given type.

##### Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

#### 5.73.2.3 `def sparksee.StrongConnectivity.add_node_type ( self, t )`

Allows connectivity through nodes of the given type.

##### Parameters

<i>t</i>	null
----------	------

#### 5.73.2.4 `def sparksee.Connectivity.close ( self )` [inherited]

Closes the [Connectivity](#) instance.

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

#### 5.73.2.5 `def sparksee.StrongConnectivity.exclude_edges ( self, 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

<code>edges</code>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------------	---

5.73.2.6 `def sparksee.StrongConnectivity.exclude_nodes ( self, 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

<code>nodes</code>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------------	---

5.73.2.7 `def sparksee.StrongConnectivity.get_connected_components ( self )`

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.

5.73.2.8 `def sparksee.Connectivity.is_closed ( self )` [inherited]

Gets if [Connectivity](#) has been closed or not.

## See also

[close\(\)](#)

## Returns

TRUE if the [Connectivity](#) instance has been closed, FALSE otherwise.

5.73.2.9 `def sparksee.StrongConnectivity.run ( self )`

Runs the algorithm in order to find the connected components.

This method can be called only once.

5.73.2.10 `def sparksee.StrongConnectivity.set_materialized_attribute ( self, attribute_name )`

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.

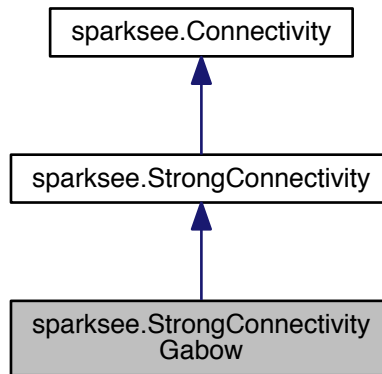
## Parameters

<i>attribute_name</i>	[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.
-----------------------	---

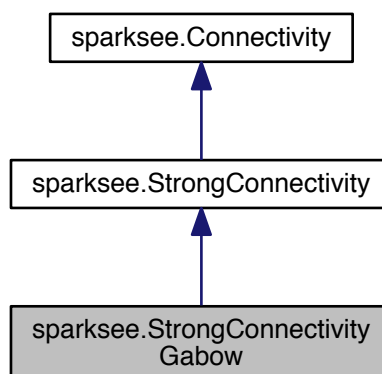
## 5.74 sparksee.StrongConnectivityGabow Class Reference

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

Inheritance diagram for sparksee.StrongConnectivityGabow:



Collaboration diagram for sparksee.StrongConnectivityGabow:



## Public Member Functions

- def [add\\_edge\\_type](#) (self, type, dir)  
*Allows connectivity through edges of the given type.*
- def [exclude\\_nodes](#) (self, nodes)  
*Set which nodes can't be used.*
- def [get\\_connected\\_components](#) (self)  
*Returns the results generated by the execution of the algorithm.*
- def [run](#) (self)  
*Executes the algorithm.*
- def [add\\_all\\_edge\\_types](#) (self, dir)  
*Allows connectivity through all edge types of the graph.*
- def [exclude\\_edges](#) (self, edges)  
*Set which edges can't be used.*
- def [add\\_node\\_type](#) (self, t)  
*Allows connectivity through nodes of the given type.*
- def [set\\_materialized\\_attribute](#) (self, attribute\_name)  
*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.*
- def [\\_\\_init\\_\\_](#) (self, session)  
*Creates a new instance of [StrongConnectivityGabow](#).*
- def [add\\_all\\_node\\_types](#) (self)  
*Allows connectivity through all node types of the graph.*
- def [close](#) (self)  
*Closes the [Connectivity](#) instance.*
- def [is\\_closed](#) (self)  
*Gets if [Connectivity](#) has been closed or not.*

### 5.74.1 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>

### 5.74.2 Constructor & Destructor Documentation

#### 5.74.2.1 `def sparksee.StrongConnectivityGabow.__init__( self, 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

<i>session</i>	[in] <a href="#">Session</a> to get the graph from and calculate the connectivity
----------------	---

## 5.74.3 Member Function Documentation

5.74.3.1 def sparksee.StrongConnectivityGabow.add\_all\_edge\_types ( *self*, *dir* )

Allows connectivity through all edge types of the graph.

## Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

5.74.3.2 def sparksee.StrongConnectivityGabow.add\_edge\_type ( *self*, *type*, *dir* )

Allows connectivity through edges of the given type.

## Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

5.74.3.3 def sparksee.StrongConnectivityGabow.add\_node\_type ( *self*, *t* )

Allows connectivity through nodes of the given type.

## Parameters

<i>t</i>	null
----------	------

5.74.3.4 def sparksee.Connectivity.close ( *self* ) [inherited]

Closes the [Connectivity](#) instance.

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

5.74.3.5 def sparksee.StrongConnectivityGabow.exclude\_edges ( *self*, *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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

#### 5.74.3.6 `def sparksee.StrongConnectivityGabow.exclude_nodes ( self, 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

<code>nodes</code>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------------	---

#### 5.74.3.7 `def sparksee.StrongConnectivityGabow.get_connected_components ( self )`

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.

#### 5.74.3.8 `def sparksee.Connectivity.is_closed ( self )` [inherited]

Gets if [Connectivity](#) has been closed or not.

##### See also

[close\(\)](#)

##### Returns

TRUE if the [Connectivity](#) instance has been closed, FALSE otherwise.

#### 5.74.3.9 `def sparksee.StrongConnectivityGabow.set_materialized_attribute ( self, attribute_name )`

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

<i>attribute_name</i>	[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.
-----------------------	---

## 5.75 sparksee.TextStream Class Reference

[TextStream](#) class.

## Public Member Functions

- def [write](#) (self, data\_i\_n, length)  
*Write data to the stream.*
- def [close](#) (self)  
*Closes the stream.*
- def [read](#) (self, length)  
*Read data from the stream.*
- def [is\\_null](#) (self)  
*Returns TRUE if the stream is not available.*
- def [\\_\\_init\\_\\_](#) (self, append)  
*Creates a new instance.*

## 5.75.1 Detailed Description

[TextStream](#) class.

It allows for reading and writing Text attribute values.

It is very important to close the stream once no more reading or writing 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) execute 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 GetAttributeText 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>

## 5.75.2 Constructor &amp; Destructor Documentation

## 5.75.2.1 def sparksee.TextStream.\_\_init\_\_( self, append )

Creates a new instance.

A [TextStream](#) only can be created by the user to write data.



## Parameters

<i>append</i>	[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 beginning of the stream.
---------------	--

## 5.75.3 Member Function Documentation

5.75.3.1 def sparksee.TextStream.close ( *self* )

Closes the stream.

Once the Stream is closed, it cannot be used again.

Closing the stream is mandatory when the stream is not null and strongly recommended when it's null to avoid deallocation problems in some platforms.

5.75.3.2 def sparksee.TextStream.is\_null ( *self* )

Returns TRUE if the stream is not available.

It returns for reading or writing data.

## Returns

FALSE if the stream is ready

5.75.3.3 def sparksee.TextStream.read ( *self*, *length* )

Read data from the stream.

## Parameters

<i>length</i>	[in] Length of the given data buffer. It must be > 0.
---------------	---

## Returns

The read data.

5.75.3.4 def sparksee.TextStream.write ( *self*, *data\_i\_n*, *length* )

Write data to the stream.

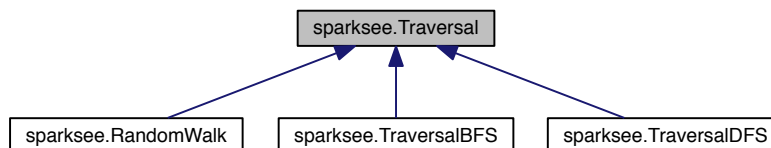
## Parameters

<i>data_i↔<sub>n</sub></i>	The string to write data from.
<i>length</i>	Number of characters to write. It must be > 0.

## 5.76 sparksee.Traversal Class Reference

[Traversal](#) class.

Inheritance diagram for sparksee.Traversal:



### Public Member Functions

- def [exclude\\_nodes](#) (self, nodes)  
*Set which nodes can't be used.*
- def [has\\_next](#) (self)  
*Gets if there are more objects to be traversed.*
- def [get\\_current\\_depth](#) (self)  
*Returns the depth of the current node.*
- def [\\_\\_iter\\_\\_](#) (self)  
*Gets a new `TraversalIterator`.*
- def [add\\_all\\_edge\\_types](#) (self, dir)  
*Allows for traversing all edge types of the graph.*
- def [exclude\\_edges](#) (self, edges)  
*Set which edges can't be used.*
- def [\\_\\_next\\_\\_](#) (self)  
*Used in `next()`*
- def [add\\_edge\\_type](#) (self, type, dir)  
*Allows for traversing edges of the given type.*
- def [add\\_node\\_type](#) (self, type)  
*Allows for traversing nodes of the given type.*
- def [next](#) (self)  
*Gets the next object of the traversal.*
- def [close](#) (self)  
*Closes the `Traversal` instance.*
- def [add\\_all\\_node\\_types](#) (self)  
*Allows for traversing all node types of the graph.*
- def [set\\_maximum\\_hops](#) (self, maxhops)  
*Sets the maximum hops restriction.*
- def [is\\_closed](#) (self)  
*Gets if `Traversal` has been closed or not.*

### 5.76.1 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>

### 5.76.2 Member Function Documentation

#### 5.76.2.1 `def sparksee.Traversal.__iter__( self )`

Gets a new TraversalIterator.

#### Returns

TraversalIterator instance

#### 5.76.2.2 `def sparksee.Traversal.__next__( self )`

Used in [next\(\)](#)

#### Returns

The next element

#### 5.76.2.3 `def sparksee.Traversal.add_all_edge_types( self, dir )`

Allows for traversing all edge types of the graph.

#### Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

#### 5.76.2.4 `def sparksee.Traversal.add_edge_type( self, type, dir )`

Allows for traversing edges of the given type.

#### Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

5.76.2.5 `def sparksee.Traversal.add_node_type ( self, type )`

Allows for traversing nodes of the given type.

Parameters

<i>type</i>	The node type to add
-------------	----------------------

5.76.2.6 `def sparksee.Traversal.close ( self )`

Closes the [Traversal](#) instance.

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

5.76.2.7 `def sparksee.Traversal.exclude_edges ( self, 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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

5.76.2.8 `def sparksee.Traversal.exclude_nodes ( self, 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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

5.76.2.9 `def sparksee.Traversal.get_current_depth ( self )`

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.

5.76.2.10 `def sparksee.Traversal.has_next ( self )`

Gets if there are more objects to be traversed.

Returns

TRUE if there are more objects, FALSE otherwise.

5.76.2.11 `def sparksee.Traversal.is_closed ( self )`

Gets if [Traversal](#) has been closed or not.

See also

[close\(\)](#)

Returns

TRUE if the [Traversal](#) instance has been closed, FALSE otherwise.

5.76.2.12 `def sparksee.Traversal.next ( self )`

Gets the next object of the traversal.

Returns

A node or edge identifier.

5.76.2.13 `def sparksee.Traversal.set_maximum_hops ( self, maxhops )`

Sets the maximum hops restriction.

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

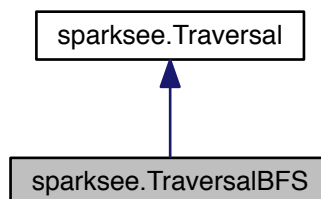
Parameters

<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
----------------	---

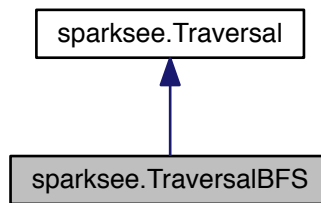
## 5.77 `sparksee.TraversalBFS` Class Reference

Breadth-First Search implementation of [Traversal](#).

Inheritance diagram for `sparksee.TraversalBFS`:



Collaboration diagram for sparksee.TraversalBFS:



### Public Member Functions

- def [add\\_edge\\_type](#) (self, type, dir)  
*Allows for traversing edges of the given type.*
- def [exclude\\_nodes](#) (self, nodes)  
*Set which nodes can't be used.*
- def [add\\_all\\_edge\\_types](#) (self, dir)  
*Allows for traversing all edge types of the graph.*
- def [add\\_node\\_type](#) (self, type)  
*Allows for traversing nodes of the given type.*
- def [exclude\\_edges](#) (self, edges)  
*Set which edges can't be used.*
- def [next](#) (self)  
*Gets the next object of the traversal.*
- def [has\\_next](#) (self)  
*Gets if there are more objects to be traversed.*
- def [get\\_current\\_depth](#) (self)  
*Returns the depth of the current node.*
- def [\\_\\_init\\_\\_](#) (self, session, node)  
*Creates a new instance.*
- def [add\\_all\\_node\\_types](#) (self)  
*Allows for traversing all node types of the graph.*
- def [set\\_maximum\\_hops](#) (self, maxhops)  
*Sets the maximum hops restriction.*
- def [\\_\\_iter\\_\\_](#) (self)  
*Gets a new TraversalIterator.*
- def [\\_\\_next\\_\\_](#) (self)  
*Used in [next\(\)](#)*
- def [close](#) (self)  
*Closes the [Traversal](#) instance.*
- def [is\\_closed](#) (self)  
*Gets if [Traversal](#) has been closed or not.*

### 5.77.1 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>

### 5.77.2 Constructor & Destructor Documentation

#### 5.77.2.1 `def sparksee.TraversalBFS.__init__( self, session, node )`

Creates a new instance.

#### Parameters

<i>session</i>	[in] <a href="#">Session</a> to get the graph from and perform traversal.
<i>node</i>	[in] Node to start traversal from.

### 5.77.3 Member Function Documentation

#### 5.77.3.1 `def sparksee.Traversal.__iter__( self )` [inherited]

Gets a new TraversalIterator.

#### Returns

TraversalIterator instance

#### 5.77.3.2 `def sparksee.Traversal.__next__( self )` [inherited]

Used in [next\(\)](#)

#### Returns

The next element

#### 5.77.3.3 `def sparksee.TraversalBFS.add_all_edge_types( self, dir )`

Allows for traversing all edge types of the graph.

#### Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

5.77.3.4 `def sparksee.TraversalBFS.add_edge_type ( self, type, dir )`

Allows for traversing edges of the given type.

Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

5.77.3.5 `def sparksee.TraversalBFS.add_node_type ( self, type )`

Allows for traversing nodes of the given type.

Parameters

<i>type</i>	The node type to add
-------------	----------------------

5.77.3.6 `def sparksee.Traversal.close ( self )` [inherited]

Closes the [Traversal](#) instance.

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

5.77.3.7 `def sparksee.TraversalBFS.exclude_edges ( self, 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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

5.77.3.8 `def sparksee.TraversalBFS.exclude_nodes ( self, 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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

5.77.3.9 `def sparksee.TraversalBFS.get_current_depth ( self )`

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.

#### 5.77.3.10 `def sparksee.TraversalBFS.has_next ( self )`

Gets if there are more objects to be traversed.

#### Returns

TRUE if there are more objects, FALSE otherwise.

#### 5.77.3.11 `def sparksee.Traversal.is_closed ( self )` [inherited]

Gets if [Traversal](#) has been closed or not.

#### See also

[close\(\)](#)

#### Returns

TRUE if the [Traversal](#) instance has been closed, FALSE otherwise.

#### 5.77.3.12 `def sparksee.TraversalBFS.next ( self )`

Gets the next object of the traversal.

#### Returns

A node or edge identifier.

#### 5.77.3.13 `def sparksee.TraversalBFS.set_maximum_hops ( self, maxhops )`

Sets the maximum hops restriction.

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

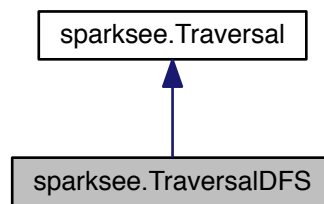
#### Parameters

<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
----------------	---

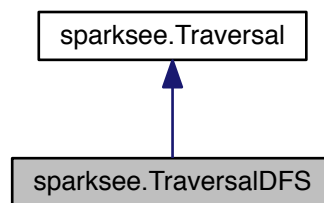
## 5.78 `sparksee.TraversalDFS` Class Reference

Depth-First Search (DFS) implementation of [Traversal](#).

Inheritance diagram for sparksee.TraversalDFS:



Collaboration diagram for sparksee.TraversalDFS:



### Public Member Functions

- def `add_edge_type` (self, type, dir)  
*Allows for traversing edges of the given type.*
- def `exclude_nodes` (self, nodes)  
*Set which nodes can't be used.*
- def `add_all_edge_types` (self, dir)  
*Allows for traversing all edge types of the graph.*
- def `add_node_type` (self, type)  
*Allows for traversing nodes of the given type.*
- def `exclude_edges` (self, edges)  
*Set which edges can't be used.*
- def `next` (self)  
*Gets the next object of the traversal.*
- def `has_next` (self)  
*Gets if there are more objects to be traversed.*
- def `get_current_depth` (self)  
*Returns the depth of the current node.*
- def `__init__` (self, session, node)  
*Creates a new instance.*

- def `add_all_node_types` (self)  
*Allows for traversing all node types of the graph.*
- def `set_maximum_hops` (self, maxhops)  
*Sets the maximum hops restriction.*
- def `__iter__` (self)  
*Gets a new `TraversalIterator`.*
- def `__next__` (self)  
*Used in `next()`*
- def `close` (self)  
*Closes the `Traversal` instance.*
- def `is_closed` (self)  
*Gets if `Traversal` has been closed or not.*

### 5.78.1 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>

### 5.78.2 Constructor & Destructor Documentation

#### 5.78.2.1 `def sparksee.TraversalDFS.__init__( self, session, node )`

Creates a new instance.

#### Parameters

<code>session</code>	[in] <code>Session</code> to get the graph from and perform traversal.
<code>node</code>	[in] Node to start traversal from.

### 5.78.3 Member Function Documentation

#### 5.78.3.1 `def sparksee.Traversal.__iter__( self )` [inherited]

Gets a new `TraversalIterator`.

#### Returns

`TraversalIterator` instance

5.78.3.2 `def sparksee.Traversal.__next__( self )` [inherited]

Used in [next\(\)](#)

#### Returns

The next element

5.78.3.3 `def sparksee.TraversalDFS.add_all_edge_types ( self, dir )`

Allows for traversing all edge types of the graph.

#### Parameters

<i>dir</i>	[in] Edge direction.
------------	----------------------

5.78.3.4 `def sparksee.TraversalDFS.add_edge_type ( self, type, dir )`

Allows for traversing edges of the given type.

#### Parameters

<i>type</i>	[in] Edge type.
<i>dir</i>	[in] Edge direction.

5.78.3.5 `def sparksee.TraversalDFS.add_node_type ( self, type )`

Allows for traversing nodes of the given type.

#### Parameters

<i>type</i>	The node type to add
-------------	----------------------

5.78.3.6 `def sparksee.Traversal.close ( self )` [inherited]

Closes the [Traversal](#) instance.

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

5.78.3.7 `def sparksee.TraversalDFS.exclude_edges ( self, 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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

#### 5.78.3.8 `def sparksee.TraversalDFS.exclude_nodes ( self, 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

<code>nodes</code>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------------	---

#### 5.78.3.9 `def sparksee.TraversalDFS.get_current_depth ( self )`

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.

#### 5.78.3.10 `def sparksee.TraversalDFS.has_next ( self )`

Gets if there are more objects to be traversed.

##### Returns

TRUE if there are more objects, FALSE otherwise.

#### 5.78.3.11 `def sparksee.Traversal.is_closed ( self )` [inherited]

Gets if [Traversal](#) has been closed or not.

##### See also

[close\(\)](#)

##### Returns

TRUE if the [Traversal](#) instance has been closed, FALSE otherwise.

#### 5.78.3.12 `def sparksee.TraversalDFS.next ( self )`

Gets the next object of the traversal.

##### Returns

A node or edge identifier.

#### 5.78.3.13 `def sparksee.TraversalDFS.set_maximum_hops ( self, maxhops )`

Sets the maximum hops restriction.

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

## Parameters

<i>maxhops</i>	[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
----------------	---

## 5.79 sparksee.Type Class Reference

Type data class.

## Public Member Functions

- def [get\\_object\\_type](#) (self)  
*Gets the object type.*
- def [get\\_id](#) (self)  
*Gets the Sparksee type identifier.*
- def [get\\_restricted\\_from](#) (self)  
*Gets the tail or source type identifier for restricted edge types.*
- def [get\\_num\\_objects](#) (self)  
*Gets the number of objects belonging to the type.*
- def [get\\_are\\_neighbors\\_indexed](#) (self)  
*Gets if this is an edge type with neighbors index.*
- def [get\\_restricted\\_to](#) (self)  
*Gets the head or target type identifier for restricted edge types.*
- def [get\\_is\\_restricted](#) (self)  
*Gets if this is a restricted edge type.*
- def [get\\_name](#) (self)  
*Gets the unique type name.*
- def [get\\_is\\_directed](#) (self)  
*Gets if this is a directed edge type.*

## Static Public Attributes

- int [INVALID\\_TYPE](#) = 0  
*Invalid type identifier.*
- int [NODES\\_TYPE](#) = 2  
*Identifier for all nodeType attributes.*
- int [GLOBAL\\_TYPE](#) = 1  
*Global type identifier.*
- int [EDGES\\_TYPE](#) = 3  
*Identifier for all edgeType attributes.*

## 5.79.1 Detailed Description

Type data class.

It contains information about a node or edge type.

## Author

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

## 5.79.2 Member Function Documentation

### 5.79.2.1 `def sparksee.Type.get_are_neighbors_indexed ( self )`

Gets if this is an edge type with neighbors index.

#### Returns

TRUE for edges types with neighbors index, FALSE otherwise.

### 5.79.2.2 `def sparksee.Type.get_id ( self )`

Gets the [Sparksee](#) type identifier.

#### Returns

The [Sparksee](#) type identifier.

### 5.79.2.3 `def sparksee.Type.get_is_directed ( self )`

Gets if this is a directed edge type.

#### Returns

TRUE for directed edge types, FALSE otherwise.

### 5.79.2.4 `def sparksee.Type.get_is_restricted ( self )`

Gets if this is a restricted edge type.

#### Returns

TRUE for restricted edge types, FALSE otherwise.

### 5.79.2.5 `def sparksee.Type.get_name ( self )`

Gets the unique type name.

#### Returns

The unique type name.

### 5.79.2.6 `def sparksee.Type.get_num_objects ( self )`

Gets the number of objects belonging to the type.

#### Returns

The number of objects belonging to the type.

5.79.2.7 `def sparksee.Type.get_object_type ( self )`

Gets the object type.

**Returns**

The object type.

5.79.2.8 `def sparksee.Type.get_restricted_from ( self )`

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.

5.79.2.9 `def sparksee.Type.get_restricted_to ( self )`

Gets the head or target type identifier for restricted edge types.

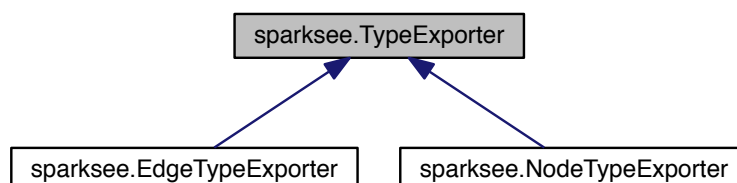
**Returns**

For restricted edge types, the head or target type identifier, the [Type InvalidType](#) otherwise.

## 5.80 sparksee.TypeExporter Class Reference

Base [TypeExporter](#) class.

Inheritance diagram for sparksee.TypeExporter:





## Public Member Functions

- def `run` (self)  
*Runs export process.*
- def `set_type` (self, type)  
*Sets the type to be exported.*
- def `set_header` (self, header)  
*Sets the presence of a header row.*
- def `set_row_writer` (self, rw)  
*Sets the output data destination.*
- def `set_graph` (self, graph)  
*Sets the graph that will be exported.*
- def `register` (self, tel)  
*Registers a new listener.*
- def `set_attributes` (self, attrs)  
*Sets the list of Attributes.*
- def `set_frequency` (self, freq)  
*Sets the frequency of listener notification.*

### 5.80.1 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>

### 5.80.2 Member Function Documentation

#### 5.80.2.1 `def sparksee.TypeExporter.register ( self, tel )`

Registers a new listener.

#### Parameters

<code>tel</code>	[in] <code>TypeExporterListener</code> to be registered.
------------------	--

#### 5.80.2.2 `def sparksee.TypeExporter.run ( self )`

Runs export process.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	If bad things happen writing to the <a href="#">RowWriter</a> .

## 5.80.2.3 def sparksee.TypeExporter.set\_attributes ( self, attrs )

Sets the list of Attributes.

## Parameters

<i>attrs</i>	[in] <a href="#">Attribute</a> identifiers to be exported
--------------	---

## 5.80.2.4 def sparksee.TypeExporter.set\_frequency ( self, freq )

Sets the frequency of listener notification.

## Parameters

<i>freq</i>	[in] Frequency in number of rows managed to notify progress to all listeners
-------------	--

## 5.80.2.5 def sparksee.TypeExporter.set\_graph ( self, graph )

Sets the graph that will be exported.

## Parameters

<i>graph</i>	[in] <a href="#">Graph</a> .
--------------	------------------------------

## 5.80.2.6 def sparksee.TypeExporter.set\_header ( self, header )

Sets the presence of a header row.

## Parameters

<i>header</i>	[in] If TRUE, a header row is dumped with the name of the attributes.
---------------	---

## 5.80.2.7 def sparksee.TypeExporter.set\_row\_writer ( self, rw )

Sets the output data destination.

## Parameters

<i>rw</i>	[in] Input <a href="#">RowWriter</a> .
-----------	--

### 5.80.2.8 `def sparksee.TypeExporter.set_type ( self, type )`

Sets the type to be exported.

#### Parameters

<code>type</code>	[in] <a href="#">Type</a> identifier.
-------------------	---------------------------------------

## 5.81 `sparksee.TypeExporterEvent` Class Reference

Provides information about the progress of an `TypeExproter` instance.

### Public Member Functions

- `def get\_type\_id (self)`  
*Gets the type identifier.*
- `def get\_total (self)`  
*Gets the total number of objects exported.*
- `def get\_count (self)`  
*Gets the current number of objects exported.*
- `def is\_last (self)`  
*Gets if this is the last event or not.*

### 5.81.1 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>

### 5.81.2 Member Function Documentation

#### 5.81.2.1 `def sparksee.TypeExporterEvent.get_count ( self )`

Gets the current number of objects exported.

#### Returns

The current number of objects exported.

#### 5.81.2.2 def sparksee.TypeExporterEvent.get\_total ( self )

Gets the total number of objects exported.

##### Returns

The total number of objects exported.

#### 5.81.2.3 def sparksee.TypeExporterEvent.get\_type\_id ( self )

Gets the type identifier.

##### Returns

The type identifier.

#### 5.81.2.4 def sparksee.TypeExporterEvent.is\_last ( self )

Gets if this is the last event or not.

##### Returns

TRUE if this is the last event, FALSE otherwise.

## 5.82 sparksee.TypeExporterListener Class Reference

Interface to be implemented to receive [TypeExporterEvent](#) events from a [TypeExporter](#).

### Public Member Functions

- def [notify\\_event](#) (self, tee)  
*Method to be notified from a [TypeExporter](#).*

#### 5.82.1 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>

#### 5.82.2 Member Function Documentation

##### 5.82.2.1 def sparksee.TypeExporterListener.notify\_event ( self, tee )

Method to be notified from a [TypeExporter](#).

## Parameters

<code>tee</code>	[in] Notified event.
------------------	----------------------

## 5.83 sparksee.TypeList Class Reference

[Sparksee](#) type identifier list.

### Public Member Functions

- def [clear](#) (self)  
*Clears the list.*
- def [\\_\\_iter\\_\\_](#) (self)  
*Gets a new [TypeListIterator](#).*
- def [iterator](#) (self)  
*Gets a new [TypeListIterator](#).*
- def [\\_\\_init\\_\\_](#) (self)  
*Constructor.*
- def [count](#) (self)  
*Number of elements in the list.*
- def [add](#) (self, type)  
*Adds a [Sparksee](#) type identifier at the end of the list.*

### 5.83.1 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>

### 5.83.2 Constructor & Destructor Documentation

#### 5.83.2.1 def sparksee.TypeList.\_\_init\_\_ ( self )

Constructor.

This creates an empty list.

### 5.83.3 Member Function Documentation

#### 5.83.3.1 def sparksee.TypeList.\_\_iter\_\_ ( self )

Gets a new [TypeListIterator](#).

#### Returns

[TypeListIterator](#) instance

#### 5.83.3.2 def sparksee.TypeList.add ( self, type )

Adds a [Sparksee](#) type identifier at the end of the list.

## Parameters

<i>type</i>	[in] <a href="#">Sparksee</a> type identifier.
-------------	--

5.83.3.3 def sparksee.TypeList.count ( *self* )

Number of elements in the list.

## Returns

Number of elements in the list.

5.83.3.4 def sparksee.TypeList.iterator ( *self* )

Gets a new [TypeListIterator](#).

## Returns

[TypeListIterator](#) instance.

## 5.84 sparksee.TypeListIterator Class Reference

[TypeList](#) iterator class.

## Public Member Functions

- def [next](#) (self)  
*Moves to the next element.*
- def [has\\_next](#) (self)  
*Gets if there are more elements.*
- def [\\_\\_next\\_\\_](#) (self)  
*Used in [next\(\)](#)*

## 5.84.1 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>

### 5.84.2 Member Function Documentation

#### 5.84.2.1 `def sparksee.TypeListIterator.__next__( self )`

Used in [next\(\)](#)

##### Returns

The next element

#### 5.84.2.2 `def sparksee.TypeListIterator.has_next ( self )`

Gets if there are more elements.

##### Returns

TRUE if there are more elements, FALSE otherwise.

#### 5.84.2.3 `def sparksee.TypeListIterator.next ( self )`

Moves to the next element.

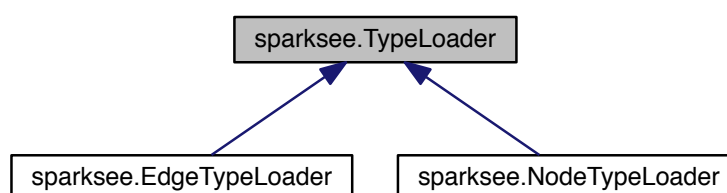
##### Returns

The next element.

## 5.85 `sparksee.TypeLoader` Class Reference

Base [TypeLoader](#) class.

Inheritance diagram for `sparksee.TypeLoader`:



## Public Member Functions

- def `run` (self)  
*Run the loader.*
- def `run_two_phases` (self)  
*Run the loader for two phases loading.*
- def `set_log_error` (self, path)  
*Sets a log error file.*
- def `set_type` (self, type)  
*Sets the type to be loaded.*
- def `set_locale` (self, locale\_str)  
*Sets the locale that will be used to read the data.*
- def `set_timestamp_format` (self, timestamp\_format)  
*Sets a specific timestamp format.*
- def `set_attributes` (self, attrs)  
*Sets the list of Attributes.*
- def `register` (self, tel)  
*Registers a new listener.*
- def `set_frequency` (self, freq)  
*Sets the frequency of listener notification.*
- def `set_row_reader` (self, rr)  
*Sets the input data source.*
- def `set_attribute_positions` (self, attrs\_pos)  
*Sets the list of attribute positions.*
- def `run_n_phases` (self, partitions)  
*Run the loader for N phases loading.*
- def `set_graph` (self, graph)  
*Sets the graph where the data will be loaded.*
- def `set_log_off` (self)  
*Truns off all the error reporting.*

### 5.85.1 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>

### 5.85.2 Member Function Documentation

#### 5.85.2.1 def sparksee.TypeLoader.register ( self, tel )

Registers a new listener.



## Parameters

<i>tel</i>	<a href="#">TypeLoaderListener</a> to be registered.
------------	--

5.85.2.2 `def sparksee.TypeLoader.run ( self )`

Run the loader.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

5.85.2.3 `def sparksee.TypeLoader.run_n_phases ( self, 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.

## Parameters

<i>partitions</i>	[in] Number of horizontal partitions to perform the load.
-------------------	---

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

5.85.2.4 `def sparksee.TypeLoader.run_two_phases ( self )`

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.

## Exceptions

<i>RuntimeError</i>	null
<i>IOError</i>	null

5.85.2.5 `def sparksee.TypeLoader.set_attribute_positions ( self, attrs_pos )`

Sets the list of attribute positions.

## Parameters

<i>attrs_pos</i>	[in] <a href="#">Attribute</a> positions (column index >=0).
------------------	--

## 5.85.2.6 def sparksee.TypeLoader.set\_attributes ( self, attrs )

Sets the list of Attributes.

## Parameters

<i>attrs</i>	[in] <a href="#">Attribute</a> identifiers to be loaded
--------------	---

## 5.85.2.7 def sparksee.TypeLoader.set\_frequency ( self, freq )

Sets the frequency of listener notification.

## Parameters

<i>freq</i>	[in] Frequency in number of rows managed to notify progress to all listeners
-------------	--

## 5.85.2.8 def sparksee.TypeLoader.set\_graph ( self, graph )

Sets the graph where the data will be loaded.

## Parameters

<i>graph</i>	[in] <a href="#">Graph</a> .
--------------	------------------------------

## 5.85.2.9 def sparksee.TypeLoader.set\_locale ( self, locale\_str )

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

It should match the locale used in the rowreader.

## Parameters

<i>locale_str</i>	[in] The locale string for the read data. See <a href="#">CSVReader</a> .
-------------------	---

## 5.85.2.10 def sparksee.TypeLoader.set\_log\_error ( self, 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

<i>path</i>	[in] The path to the error log file.
-------------	--------------------------------------

## Exceptions

<code>IOError</code>	If bad things happen opening the file.
----------------------	--

### 5.85.2.11 `def sparksee.TypeLoader.set_log_off ( self )`

Turns 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.

### 5.85.2.12 `def sparksee.TypeLoader.set_row_reader ( self, rr )`

Sets the input data source.

#### Parameters

<code>rr</code>	[in] Input <a href="#">RowReader</a> .
-----------------	--

### 5.85.2.13 `def sparksee.TypeLoader.set_timestamp_format ( self, timestamp_format )`

Sets a specific timestamp format.

#### Parameters

<code>timestamp_format</code>	[in] A string with the timestamp format definition.
-------------------------------	---

### 5.85.2.14 `def sparksee.TypeLoader.set_type ( self, type )`

Sets the type to be loaded.

#### Parameters

<code>type</code>	[in] <a href="#">Type</a> identifier.
-------------------	---------------------------------------

## 5.86 `sparksee.TypeLoaderEvent` Class Reference

Provides information about the progress of a [TypeLoader](#) instance.

### Public Member Functions

- `def get\_type\_id (self)`  
*Gets the type identifier.*
- `def get\_phase (self)`  
*Gets the current phase.*

- def `get_count` (self)  
*Gets the current number of objects created.*
- def `get_total_partition_steps` (self)  
*Gets the total number of steps in the current partition.*
- def `get_partition` (self)  
*Gets the current partition.*
- def `get_total_partitions` (self)  
*Gets the total number of partitions.*
- def `is_last` (self)  
*Gets if this is the last event or not.*
- def `get_total_phases` (self)  
*Gets the total number of phases.*

### 5.86.1 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>

### 5.86.2 Member Function Documentation

#### 5.86.2.1 def sparksee.TypeLoaderEvent.get\_count ( self )

Gets the current number of objects created.

#### Returns

The current number of objects created.

#### 5.86.2.2 def sparksee.TypeLoaderEvent.get\_partition ( self )

Gets the current partition.

#### Returns

The current partition.

#### 5.86.2.3 def sparksee.TypeLoaderEvent.get\_phase ( self )

Gets the current phase.

#### Returns

The current phase.

#### 5.86.2.4 `def sparksee.TypeLoaderEvent.get_total_partition_steps ( self )`

Gets the total number of steps in the current partition.

##### Returns

The total number steps in the current partition.

#### 5.86.2.5 `def sparksee.TypeLoaderEvent.get_total_partitions ( self )`

Gets the total number of partitions.

##### Returns

The total number of partitions.

#### 5.86.2.6 `def sparksee.TypeLoaderEvent.get_total_phases ( self )`

Gets the total number of phases.

##### Returns

The total number of phases.

#### 5.86.2.7 `def sparksee.TypeLoaderEvent.get_type_id ( self )`

Gets the type identifier.

##### Returns

The type identifier.

#### 5.86.2.8 `def sparksee.TypeLoaderEvent.is_last ( self )`

Gets if this is the last event or not.

##### Returns

TRUE if this is the last event, FALSE otherwise.

## 5.87 `sparksee.TypeLoaderListener` Class Reference

### Public Member Functions

- `def notify_event (self, ev)`  
*Method to receive events from a Loader.*

#### 5.87.1 Member Function Documentation

##### 5.87.1.1 `def sparksee.TypeLoaderListener.notify_event ( self, ev )`

Method to receive events from a Loader.

## Parameters

ev	Loader.LoaderEvent with information from a running Loader.
----	--

## 5.88 sparksee.Value Class Reference

[Value](#) class.

## Public Member Functions

- def [set\\_null](#) (self)  
*Sets the Value to NULL.*
- def [get\\_timestamp](#) (self)  
*Gets Timestamp Value.*
- def [get\\_long](#) (self)  
*Gets Long Value.*
- def [operator](#) (self, value)  
*Assignment operator.*
- def [set\\_oid](#) (self, value)  
*Sets the Value.*
- def [set\\_double\\_void](#) (self, value)  
*Sets the Value.*
- def [set\\_boolean](#) (self, value)  
*Sets the Value.*
- def [\\_\\_init\\_\\_](#) (self)  
*Creates a new instance.*
- def [get\\_integer](#) (self)  
*Gets Integer Value.*
- def [set\\_timestamp\\_void](#) (self, year, month, day, hour, minutes, seconds, millisecs)  
*Sets the Value.*
- def [equals](#) (self, value)  
*Compares with the given Value.*
- def [set\\_boolean\\_void](#) (self, value)  
*Sets the Value.*
- def [set\\_oid\\_void](#) (self, value)  
*Sets the OID Value.*
- def [set\\_long\\_void](#) (self, value)  
*Sets the Value.*
- def [set\\_string](#) (self, value)  
*Sets the Value.*
- def [compare](#) (self, value)  
*Compares with the given Value.*
- def [get\\_oid](#) (self)  
*Gets OID Value.*
- def [get\\_boolean](#) (self)  
*Gets Boolean Value.*
- def [set\\_timestamp](#) (self, value)  
*Sets the Value.*

- def `get_data_type` (self)  
*Gets the `DataType`.*
- def `set_long` (self, value)  
*Sets the `Value`.*
- def `set_integer` (self, value)  
*Sets the `Value`.*
- def `set_double` (self, value)  
*Sets the `Value`.*
- def `get_double` (self)  
*Gets Double `Value`.*
- def `__init__` (self, value)  
*Copy constructor.*
- def `set_timestamp` (self, year, month, day, hour, minutes, seconds, millisecs)  
*Sets the `Value`.*
- def `set_string_void` (self, value)  
*Sets the `Value`.*
- def `set_void` (self, value)  
*Sets the `Value`.*
- def `set_timestamp_void` (self, value)  
*Sets the `Value`.*
- def `set_integer_void` (self, value)  
*Sets the `Value`.*
- def `is_null` (self)  
*Gets if this is a `NULL Value`.*
- def `get_string` (self)  
*Gets String `Value`.*
- def `set_null_void` (self)  
*Sets the `Value` to `NULL`.*
- def `to_string` (self)  
*Returns a String representation of the `Value`, used in `unicode` and `str`*

#### Static Public Attributes

- int `MAX_LENGTH_STRING` = 2047  
*Maximum number of characters allowed for a String.*

#### 5.88.1 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>

### 5.88.2 Constructor & Destructor Documentation

#### 5.88.2.1 def sparksee.Value.\_\_init\_\_( self )

Creates a new instance.

It creates a NULL [Value](#).

Referenced by `sparksee.Value.__init__()`.

#### 5.88.2.2 def sparksee.Value.\_\_init\_\_( self, value )

Copy constructor.

##### Parameters

<i>value</i>	[in] <a href="#">Value</a> to be copied.
--------------	--

References `sparksee.Value.__init__()`.

### 5.88.3 Member Function Documentation

#### 5.88.3.1 def sparksee.Value.compare( self, value )

Compares with the given [Value](#).

It does not work if the given [Value](#) objects does not have the same [DataType](#).

##### Parameters

<i>value</i>	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.

#### 5.88.3.2 def sparksee.Value.equals( self, value )

Compares with the given [Value](#).

It does not work if the given [Value](#) objects does not have the same [DataType](#).

##### Parameters

<i>value</i>	Given value to compare to.
--------------	----------------------------



**Returns**

TRUE if this [Value](#) is equal to the given one; FALSE otherwise.

**5.88.3.3** `def sparksee.Value.get_boolean ( self )`

Gets Boolean [Value](#).

This must be a non-NULL Boolean [Value](#).

**Returns**

The Boolean [Value](#).

**5.88.3.4** `def sparksee.Value.get_data_type ( self )`

Gets the [DataType](#).

[Value](#) cannot be NULL.

**Returns**

The [DataType](#).

**5.88.3.5** `def sparksee.Value.get_double ( self )`

Gets Double [Value](#).

This must be a non-NULL Double [Value](#).

**Returns**

The Double [Value](#).

**5.88.3.6** `def sparksee.Value.get_integer ( self )`

Gets Integer [Value](#).

This must be a non-NULL Integer [Value](#).

**Returns**

The Integer [Value](#).

**5.88.3.7** `def sparksee.Value.get_long ( self )`

Gets Long [Value](#).

This must be a non-NULL Long [Value](#).

**Returns**

The Long [Value](#).

5.88.3.8 `def sparksee.Value.get_oid ( self )`

Gets OID [Value](#).

This must be a non-NULL OID [Value](#).

#### Returns

The OID [Value](#).

5.88.3.9 `def sparksee.Value.get_string ( self )`

Gets String [Value](#).

This must be a non-NULL String [Value](#).

#### Returns

The String [Value](#).

5.88.3.10 `def sparksee.Value.get_timestamp ( self )`

Gets Timestamp [Value](#).

This must be a non-NULL Timestamp [Value](#).

#### Returns

The Timestamp [Value](#).

5.88.3.11 `def sparksee.Value.is_null ( self )`

Gets if this is a NULL [Value](#).

#### Returns

TRUE if this is a NULL [Value](#), FALSE otherwise.

5.88.3.12 `def sparksee.Value.operator ( self, value )`

Assignment operator.

#### Parameters

<i>value</i>	[in] <a href="#">Value</a> to be copied.
--------------	--

#### Returns

Returns the [Value](#) reference.

5.88.3.13 `def sparksee.Value.set_boolean ( self, value )`

Sets the [Value](#).

Parameters

<i>value</i>	[in] Nex Boolean value.
--------------	-------------------------

Returns

The calling instance.

5.88.3.14 `def sparksee.Value.set_boolean_void ( self, value )`

Sets the [Value](#).

Parameters

<i>value</i>	[in] New Boolean value.
--------------	-------------------------

5.88.3.15 `def sparksee.Value.set_double ( self, value )`

Sets the [Value](#).

Parameters

<i>value</i>	[in] New Double value.
--------------	------------------------

Returns

The calling instance.

5.88.3.16 `def sparksee.Value.set_double_void ( self, value )`

Sets the [Value](#).

Parameters

<i>value</i>	[in] New Double value.
--------------	------------------------

5.88.3.17 `def sparksee.Value.set_integer ( self, value )`

Sets the [Value](#).

Parameters

<i>value</i>	[in] New Integer value.
--------------	-------------------------

**Returns**

The calling instance.

5.88.3.18 `def sparksee.Value.set_integer_void ( self, value )`

Sets the [Value](#).

**Parameters**

<i>value</i>	[in] New Integer value.
--------------	-------------------------

5.88.3.19 `def sparksee.Value.set_long ( self, value )`

Sets the [Value](#).

**Parameters**

<i>value</i>	[in] New Long value.
--------------	----------------------

**Returns**

The calling instance.

5.88.3.20 `def sparksee.Value.set_long_void ( self, value )`

Sets the [Value](#).

**Parameters**

<i>value</i>	[in] New Long value.
--------------	----------------------

5.88.3.21 `def sparksee.Value.set_null ( self )`

Sets the [Value](#) to NULL.

**Returns**

The calling instance.

5.88.3.22 `def sparksee.Value.set_oid ( self, value )`

Sets the [Value](#).

**Parameters**

<i>value</i>	[in] New OID <a href="#">Value</a> .
--------------	--------------------------------------

**Returns**

The calling instance.

5.88.3.23 `def sparksee.Value.set_oid_void ( self, value )`

Sets the OID [Value](#).

**Parameters**

<i>value</i>	[in] New OID value.
--------------	---------------------

5.88.3.24 `def sparksee.Value.set_string ( self, value )`

Sets the [Value](#).

**Parameters**

<i>value</i>	[in] New String value.
--------------	------------------------

**Returns**

The calling instance.

5.88.3.25 `def sparksee.Value.set_string_void ( self, value )`

Sets the [Value](#).

**Parameters**

<i>value</i>	[in] New String value.
--------------	------------------------

5.88.3.26 `def sparksee.Value.set_timestamp ( self, value )`

Sets the [Value](#).

**Parameters**

<i>value</i>	[in] New Timestamp value.
--------------	---------------------------

**Returns**

The calling instance.

Referenced by `sparksee.Value.set_timestamp()`.

5.88.3.27 `def sparksee.Value.set_timestamp ( self, year, month, day, hour, minutes, seconds, millisecs )`

Sets the [Value](#).

## Parameters

<i>year</i>	[in] The year ( $\geq 1970$ ).
<i>month</i>	[in] The month ([1..12]).
<i>day</i>	[in] The of the month ([1..31]).
<i>hour</i>	[in] The hour ([0..23]).
<i>minutes</i>	[in] The minutes ([0..59]).
<i>seconds</i>	[in] The seconds ([0..59]).
<i>millisecs</i>	[in] The milliseconds ([0..999]).

## Returns

The calling instance.

References `sparksee.Value.set_timestamp()`.

5.88.3.28 `def sparksee.Value.set_timestamp_void ( self, year, month, day, hour, minutes, seconds, millisecs )`

Sets the [Value](#).

## Parameters

<i>year</i>	[in] The year ( $\geq 1970$ ).
<i>month</i>	[in] The month ([1..12]).
<i>day</i>	[in] The of the month ([1..31]).
<i>hour</i>	[in] The hour ([0..23]).
<i>minutes</i>	[in] The minutes ([0..59]).
<i>seconds</i>	[in] The seconds ([0..59]).
<i>millisecs</i>	[in] The milliseconds ([0..999]).

Referenced by `sparksee.Value.set_timestamp_void()`.

5.88.3.29 `def sparksee.Value.set_timestamp_void ( self, value )`

Sets the [Value](#).

## Parameters

<i>value</i>	[in] New Timestamp value.
--------------	---------------------------

References `sparksee.Value.set_timestamp_void()`.

5.88.3.30 `def sparksee.Value.set_void ( self, value )`

Sets the [Value](#).

## Parameters

<i>value</i>	[in] New value.
--------------	-----------------

### 5.88.3.31 def sparksee.Value.to\_string ( self )

Returns a String representation of the [Value](#), used in **unicode** and **str**

## 5.89 sparksee.ValueArray Class Reference

[ValueArray](#) class.

### Public Member Functions

- def [set](#) (self, index, value)  
*Set a [Value](#) to a specific array position.*
- def [get\\_integer](#) (self)  
*Get all the values from this int array.*
- def [get\\_long](#) (self)  
*Get all the values from this long array.*
- def [set\\_timestamp](#) (self, values)  
*Set all the values of this timestamp array.*
- def [set\\_oid](#) (self, values)  
*Set all the values of this oid array.*
- def [get\\_oid](#) (self)  
*Get all the values from this oid array.*
- def [get](#) (self, index, value)  
*Get a [Value](#) from the array.*
- def [get\\_timestamp](#) (self)  
*Get all the values from this timestamp array.*
- def [get\\_double\\_range](#) (self, index, [size](#))  
*Get a subset of the values from this double array.*
- def [get\\_integer\\_range](#) (self, index, [size](#))  
*Get a subset of the values from this integer array.*
- def [get\\_boolean](#) (self)  
*Get all the values from this bool array.*
- def [set\\_boolean](#) (self, values)  
*Set all the values of this bool array.*
- def [set\\_double\\_range](#) (self, index, values)  
*Set a subset of the values from this double array.*
- def [size](#) (self)  
*Returns the array size.*
- def [get\\_long\\_range](#) (self, index, [size](#))  
*Get a subset of the values from this long array.*
- def [set\\_timestamp\\_range](#) (self, index, values)  
*Set a subset of the values from this timestamp array.*
- def [make\\_null](#) (self)  
*Sets the attribute array to Null.*
- def [set\\_integer\\_range](#) (self, index, values)  
*Set a subset of the values from this integer array.*
- def [get\\_oid\\_range](#) (self, index, [size](#))  
*Get a subset of the values from this oid array.*
- def [set\\_boolean\\_range](#) (self, index, values)

- Set a subset of the values from this boolean array.*

  - def [set\\_oid\\_range](#) (self, index, values)

*Set a subset of the values from this oid array.*
- def [get\\_double](#) (self)

*Get all the values from this double array.*
- def [set\\_long](#) (self, values)

*Set all the values of this long array.*
- def [set\\_double](#) (self, values)

*Set all the values of this double array.*
- def [set\\_integer](#) (self, values)

*Set all the values of this int array.*
- def [set\\_long\\_range](#) (self, index, values)

*Set a subset of the values from this long array.*
- def [set](#) (self, value)

*Set a [Value](#) to the whole array.*
- def [get\\_boolean\\_range](#) (self, index, size)

*Get a subset of the values from this boolean array.*
- def [get\\_timestamp\\_range](#) (self, index, size)

*Get a subset of the values from this timestamp array.*

### 5.89.1 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>

### 5.89.2 Member Function Documentation

#### 5.89.2.1 `def sparksee.ValueArray.get ( self, index, value )`

Get a [Value](#) from the array.

AppErrorIf the [ValueArray](#) is not available

#### Parameters

<i>index</i>	[in] Position of the element to get [0..N-1]
<i>value</i>	[out] <a href="#">Value</a> to get the array element



## Exceptions

<i>ValueError</i>	If the index is out of range or the <a href="#">Value</a> not valid
<i>RuntimeError</i>	null

5.89.2.2 `def sparksee.ValueArray.get_boolean ( self )`

Get all the values from this bool array.

UnsupportedOperationErrorIf array [DataType](#) is not bool

## Returns

All the array values

## Exceptions

<i>RuntimeError</i>	If the <a href="#">ValueArray</a> is not available
<i>RuntimeError</i>	null

5.89.2.3 `def sparksee.ValueArray.get_boolean_range ( self, index, size )`

Get a subset of the values from this boolean array.

Returns all the requested values

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>size</i>	[in] Number of elements to get [1..N]

5.89.2.4 `def sparksee.ValueArray.get_double ( self )`

Get all the values from this double array.

UnsupportedOperationErrorIf array [DataType](#) is not Double

## Returns

All the array values

## Exceptions

<i>RuntimeError</i>	If the <a href="#">ValueArray</a> is not available
<i>RuntimeError</i>	null

## 5.89.2.5 def sparksee.ValueArray.get\_double\_range ( self, index, size )

Get a subset of the values from this double array.

Returns all the requested values

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>size</i>	[in] Number of elements to get [1..N]

## 5.89.2.6 def sparksee.ValueArray.get\_integer ( self )

Get all the values from this int array.

UnsupportedOperationErrorIf array [DataType](#) is not int

## Returns

All the array values

## Exceptions

<i>RuntimeError</i>	If the <a href="#">ValueArray</a> is not available
<i>RuntimeError</i>	null

## 5.89.2.7 def sparksee.ValueArray.get\_integer\_range ( self, index, size )

Get a subset of the values from this integer array.

Returns all the requested values

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>size</i>	[in] Number of elements to get [1..N]

## 5.89.2.8 def sparksee.ValueArray.get\_long ( self )

Get all the values from this long array.

UnsupportedOperationErrorIf array [DataType](#) is not long

## Returns

All the array values

## Exceptions

<i>RuntimeError</i>	If the <a href="#">ValueArray</a> is not available
<i>RuntimeError</i>	null

5.89.2.9 `def sparksee.ValueArray.get_long_range ( self, index, size )`

Get a subset of the values from this long array.

Returns all the requested values

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>size</i>	[in] Number of elements to get [1..N]

5.89.2.10 `def sparksee.ValueArray.get_oid ( self )`

Get all the values from this oid array.

UnsupportedOperationErrorIf array [DataType](#) is not oid

## Returns

All the array values

## Exceptions

<i>RuntimeError</i>	If the <a href="#">ValueArray</a> is not available
<i>RuntimeError</i>	null

5.89.2.11 `def sparksee.ValueArray.get_oid_range ( self, index, size )`

Get a subset of the values from this oid array.

Returns all the requested values

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>size</i>	[in] Number of elements to get [1..N]

5.89.2.12 `def sparksee.ValueArray.get_timestamp ( self )`

Get all the values from this timestamp array.

UnsupportedOperationErrorIf array [DataType](#) is not timestamp

## Returns

All the array values

## Exceptions

<i>RuntimeError</i>	If the <a href="#">ValueArray</a> is not available
<i>RuntimeError</i>	null

## 5.89.2.13 def sparksee.ValueArray.get\_timestamp\_range ( self, index, size )

Get a subset of the values from this timestamp array.

Returns all the requested values

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>size</i>	[in] Number of elements to get [1..N]

## 5.89.2.14 def sparksee.ValueArray.make\_null ( self )

Sets the attribute array to Null.

The [ValueArray](#) can not be used after this call.

## Exceptions

<i>RuntimeError</i>	null
---------------------	------

## 5.89.2.15 def sparksee.ValueArray.set ( self, index, value )

Set a [Value](#) to a specific array position.

AppErrorIf the [ValueArray](#) is not available

## Parameters

<i>index</i>	[in] Position of the element to set [0..N-1]
<i>value</i>	[in] <a href="#">Value</a> to set in the array element

## Exceptions

<i>ValueError</i>	If the index is out of range or the <a href="#">Value</a> not valid
<i>RuntimeError</i>	null

Referenced by sparksee.ValueArray.set().

5.89.2.16 `def sparksee.ValueArray.set ( self, value )`

Set a [Value](#) to the whole array.

AppErrorIf the [ValueArray](#) is not available

#### Parameters

<i>value</i>	[in] <a href="#">Value</a> to set in all the array elements
--------------	---

#### Exceptions

<i>ValueError</i>	If the <a href="#">Value</a> is not valid
<i>RuntimeError</i>	null

References `sparksee.ValueArray.set()`.

5.89.2.17 `def sparksee.ValueArray.set_boolean ( self, values )`

Set all the values of this bool array.

AppErrorIf the [ValueArray](#) is not available

#### Parameters

<i>values</i>	[in] All the values to set
---------------	----------------------------

#### Exceptions

<i>RuntimeError</i>	null
<i>RuntimeError</i>	If array <a href="#">DataType</a> is not bool

5.89.2.18 `def sparksee.ValueArray.set_boolean_range ( self, index, values )`

Set a subset of the values from this boolean array.

#### Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>values</i>	[in] All the values to set

5.89.2.19 `def sparksee.ValueArray.set_double ( self, values )`

Set all the values of this double array.

AppErrorIf the [ValueArray](#) is not available

## Parameters

<i>values</i>	[in] All the values to set
---------------	----------------------------

## Exceptions

<i>RuntimeError</i>	null
<i>RuntimeError</i>	If array <a href="#">DataType</a> is not Double

5.89.2.20 `def sparksee.ValueArray.set_double_range ( self, index, values )`

Set a subset of the values from this double array.

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>values</i>	[in] All the values to set

5.89.2.21 `def sparksee.ValueArray.set_integer ( self, values )`

Set all the values of this int array.

AppErrorIf the [ValueArray](#) is not available

## Parameters

<i>values</i>	[in] All the values to set
---------------	----------------------------

## Exceptions

<i>RuntimeError</i>	null
<i>RuntimeError</i>	If array <a href="#">DataType</a> is not int

5.89.2.22 `def sparksee.ValueArray.set_integer_range ( self, index, values )`

Set a subset of the values from this integer array.

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>values</i>	[in] All the values to set

5.89.2.23 `def sparksee.ValueArray.set_long ( self, values )`

Set all the values of this long array.

AppErrorIf the [ValueArray](#) is not available

## Parameters

<i>values</i>	[in] All the values to set
---------------	----------------------------

## Exceptions

<i>RuntimeError</i>	null
<i>RuntimeError</i>	If array <a href="#">DataType</a> is not long

5.89.2.24 `def sparksee.ValueArray.set_long_range ( self, index, values )`

Set a subset of the values from this long array.

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>values</i>	[in] All the values to set

5.89.2.25 `def sparksee.ValueArray.set_oid ( self, values )`

Set all the values of this oid array.

AppErrorIf the [ValueArray](#) is not available

## Parameters

<i>values</i>	[in] All the values to set
---------------	----------------------------

## Exceptions

<i>RuntimeError</i>	null
<i>RuntimeError</i>	If array <a href="#">DataType</a> is not oid

5.89.2.26 `def sparksee.ValueArray.set_oid_range ( self, index, values )`

Set a subset of the values from this oid array.

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>values</i>	[in] All the values to set

5.89.2.27 `def sparksee.ValueArray.set_timestamp ( self, values )`

Set all the values of this timestamp array.

AppErrorIf the [ValueArray](#) is not available

## Parameters

<i>values</i>	[in] All the values to set
---------------	----------------------------

## Exceptions

<i>RuntimeError</i>	null
<i>RuntimeError</i>	If array <a href="#">DataType</a> is not timestamp

5.89.2.28 `def sparksee.ValueArray.set_timestamp_range( self, index, values )`

Set a subset of the values from this timestamp array.

## Parameters

<i>index</i>	[in] Position of the first element to get [0..N-1]
<i>values</i>	[in] All the values to set

## 5.90 sparksee.ValueList Class Reference

[Value](#) list.

## Public Member Functions

- `def clear` (self)  
*Clears the list.*
- `def get` (self, index)  
*Returns the [Value](#) at the specified position in the list.*
- `def __iter__` (self)  
*Gets a new [ValueListIterator](#).*
- `def iterator` (self)  
*Gets a new [ValueListIterator](#).*
- `def __init__` (self)  
*Constructor.*
- `def add` (self, value)  
*Adds a value to the end of the list.*
- `def count` (self)  
*Number of elements in the list.*

## 5.90.1 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>



## 5.90.2 Constructor & Destructor Documentation

### 5.90.2.1 `def sparksee.ValueList.__init__( self )`

Constructor.

This creates an empty list.

## 5.90.3 Member Function Documentation

### 5.90.3.1 `def sparksee.ValueList.__iter__( self )`

Gets a new [ValueListIterator](#).

Returns

[ValueListIterator](#) instance

### 5.90.3.2 `def sparksee.ValueList.add( self, value )`

Adds a value to the end of the list.

Parameters

<i>value</i>	[in] The value to add
--------------	-----------------------

### 5.90.3.3 `def sparksee.ValueList.count( self )`

Number of elements in the list.

Returns

Number of elements in the list.

### 5.90.3.4 `def sparksee.ValueList.get( self, index )`

Returns the [Value](#) at the specified position in the list.

Parameters

<i>index</i>	[in] Index of the element to return, starting at 0.
--------------	---

### 5.90.3.5 `def sparksee.ValueList.iterator( self )`

Gets a new [ValueListIterator](#).

**Returns**

[ValueListIterator](#) instance.

**5.91 sparksee.ValueListIterator Class Reference**

[ValueList](#) iterator class.

**Public Member Functions**

- def [next](#) (self)  
*Moves to the next element.*
- def [has\\_next](#) (self)  
*Gets if there are more elements.*
- def [\\_\\_next\\_\\_](#) (self)  
*Used in [next\(\)](#)*

**5.91.1 Detailed Description**

[ValueList](#) iterator class.

Iterator to traverse all the values into a [ValueList](#) instance.

**Author**

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

**5.91.2 Member Function Documentation****5.91.2.1 def sparksee.ValueListIterator.\_\_next\_\_ ( self )**

Used in [next\(\)](#)

**Returns**

The next element

**5.91.2.2 def sparksee.ValueListIterator.has\_next ( self )**

Gets if there are more elements.

**Returns**

TRUE if there are more elements, FALSE otherwise.

### 5.91.2.3 `def sparksee.ValueListIterator.next ( self )`

Moves to the next element.

#### Returns

The next element.

## 5.92 `sparksee.Values` Class Reference

`Value` set class.

### Public Member Functions

- `def count` (self)  
*Gets the number of elements into the collection.*
- `def __iter__` (self)  
*Gets a new `ValuesIterator`.*
- `def iterator` (self, order)  
*Gets a `ValuesIterator`.*
- `def close` (self)  
*Closes the `Values` instance.*
- `def is_closed` (self)  
*Gets if `Values` instance has been closed or not.*

### 5.92.1 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>

### 5.92.2 Member Function Documentation

#### 5.92.2.1 `def sparksee.Values.__iter__ ( self )`

Gets a new `ValuesIterator`.

#### Returns

`ValuesIterator` instance

5.92.2.2 `def sparksee.Values.close ( self )`

Closes the [Values](#) instance.

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

5.92.2.3 `def sparksee.Values.count ( self )`

Gets the number of elements into the collection.

Returns

The number of elements into the collection.

5.92.2.4 `def sparksee.Values.is_closed ( self )`

Gets if [Values](#) instance has been closed or not.

See also

[close\(\)](#)

Returns

TRUE if the [Values](#) instance has been closed, FALSE otherwise.

5.92.2.5 `def sparksee.Values.iterator ( self, order )`

Gets a [ValuesIterator](#).

Parameters

<code>order</code>	[in] Ascending or descending order.
--------------------	-------------------------------------

Returns

[ValuesIterator](#) instance.

## 5.93 sparksee.ValuesIterator Class Reference

[Values](#) iterator class.

Public Member Functions

- `def next (self)`  
*Gets the next element to traverse.*
- `def has_next (self)`

- Gets if there are more elements to traverse.*
- def `close` (self)  
*Closes the `ValuesIterator` instance.*
- def `__next__` (self)  
*Used in `next()`*
- def `is_closed` (self)  
*Gets if `ValuesIterator` instance has been closed or not.*

### 5.93.1 Detailed Description

`Values` iterator class.

It allows for traversing all the elements into a `Values` instance.

#### Author

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

### 5.93.2 Member Function Documentation

#### 5.93.2.1 `def sparksee.ValuesIterator.__next__( self )`

Used in `next()`

#### Returns

The next element

#### 5.93.2.2 `def sparksee.ValuesIterator.close ( self )`

Closes the `ValuesIterator` instance.

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

#### 5.93.2.3 `def sparksee.ValuesIterator.has_next ( self )`

Gets if there are more elements to traverse.

#### Returns

TRUE if there are more elements to traverse, FALSE otherwise.

#### 5.93.2.4 `def sparksee.ValuesIterator.is_closed ( self )`

Gets if `ValuesIterator` instance has been closed or not.

#### See also

`close()`

#### Returns

TRUE if the `ValuesIterator` instance has been closed, FALSE otherwise.

## 5.93.2.5 def sparksee.ValuesIterator.next ( self )

Gets the next element to traverse.

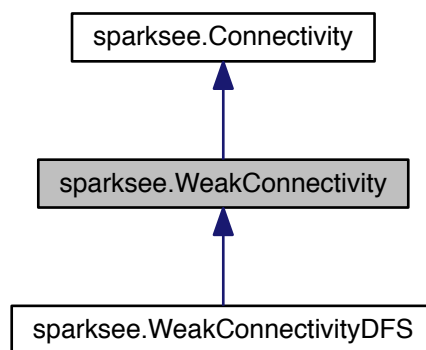
**Returns**

The next element.

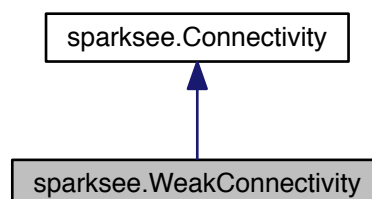
## 5.94 sparksee.WeakConnectivity Class Reference

[WeakConnectivity](#) class.

Inheritance diagram for sparksee.WeakConnectivity:



Collaboration diagram for sparksee.WeakConnectivity:



## Public Member Functions

- def `run` (self)  
*Runs the algorithm in order to find the connected components.*
- def `exclude_nodes` (self, nodes)  
*Set which nodes can't be used.*
- def `add_all_edge_types` (self)  
*Allows connectivity through all edge types of the graph.*
- def `get_connected_components` (self)  
*Returns the results generated by the execution of the algorithm.*
- def `add_edge_type` (self, type)  
*Allows connectivity through edges of the given type.*
- def `exclude_edges` (self, edges)  
*Set which edges can't be used.*
- def `add_node_type` (self, t)  
*Allows connectivity through nodes of the given type.*
- def `set_materialized_attribute` (self, attribute\_name)  
*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.*
- def `add_all_node_types` (self)  
*Allows connectivity through all node types of the graph.*
- def `close` (self)  
*Closes the `Connectivity` instance.*
- def `is_closed` (self)  
*Gets if `Connectivity` has been closed or not.*

### 5.94.1 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>

### 5.94.2 Member Function Documentation

#### 5.94.2.1 `def sparksee.WeakConnectivity.add_all_edge_types ( self )`

Allows connectivity through all edge types of the graph.

In a weak connectivity the edges can be used in Any direction.

#### 5.94.2.2 `def sparksee.WeakConnectivity.add_edge_type ( self, type )`

Allows connectivity through edges of the given type.

In a weak connectivity the edges can be used in Any direction.

## Parameters

<i>type</i>	[in] Edge type.
-------------	-----------------

5.94.2.3 def sparksee.WeakConnectivity.add\_node\_type ( *self*, *t* )

Allows connectivity through nodes of the given type.

## Parameters

<i>t</i>	null
----------	------

5.94.2.4 def sparksee.Connectivity.close ( *self* ) [inherited]

Closes the [Connectivity](#) instance.

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

5.94.2.5 def sparksee.WeakConnectivity.exclude\_edges ( *self*, *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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

5.94.2.6 def sparksee.WeakConnectivity.exclude\_nodes ( *self*, *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

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

5.94.2.7 def sparksee.WeakConnectivity.get\_connected\_components ( *self* )

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.



#### 5.94.2.8 `def sparksee.Connectivity.is_closed ( self ) [inherited]`

Gets if [Connectivity](#) has been closed or not.

See also

[close\(\)](#)

Returns

TRUE if the [Connectivity](#) instance has been closed, FALSE otherwise.

#### 5.94.2.9 `def sparksee.WeakConnectivity.run ( self )`

Runs the algorithm in order to find the connected components.

This method can be called only once.

#### 5.94.2.10 `def sparksee.WeakConnectivity.set_materialized_attribute ( self, attribute_name )`

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.

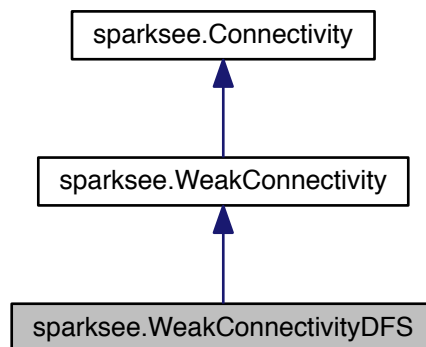
Parameters

<i>attribute_name</i>	[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.
-----------------------	---

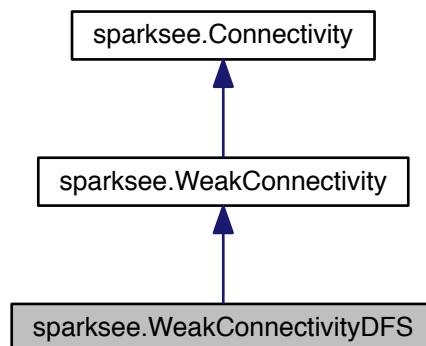
## 5.95 `sparksee.WeakConnectivityDFS` Class Reference

[WeakConnectivityDFS](#) class.

Inheritance diagram for sparksee.WeakConnectivityDFS:



Collaboration diagram for sparksee.WeakConnectivityDFS:



#### Public Member Functions

- def `__init__` (self, session)  
*Creates a new instance of [WeakConnectivityDFS](#).*
- def `exclude_nodes` (self, nodes)  
*Set which nodes can't be used.*
- def `add_all_edge_types` (self)  
*Allows connectivity through all edge types of the graph.*
- def `get_connected_components` (self)  
*Returns the results generated by the execution of the algorithm.*
- def `run` (self)  
*Executes the algorithm.*

- def `add_edge_type` (self, type)
  - Allows connectivity through edges of the given type.*
- def `exclude_edges` (self, edges)
  - Set which edges can't be used.*
- def `add_node_type` (self, t)
  - Allows connectivity through nodes of the given type.*
- def `set_materialized_attribute` (self, attribute\_name)
  - 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.*
- def `add_all_node_types` (self)
  - Allows connectivity through all node types of the graph.*
- def `close` (self)
  - Closes the `Connectivity` instance.*
- def `is_closed` (self)
  - Gets if `Connectivity` has been closed or not.*

### 5.95.1 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>

### 5.95.2 Constructor & Destructor Documentation

#### 5.95.2.1 def sparksee.WeakConnectivityDFS.\_\_init\_\_( self, 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

<code>session</code>	[in] <code>Session</code> to get the graph from and calculate the connectivity
----------------------	--

## 5.95.3 Member Function Documentation

## 5.95.3.1 def sparksee.WeakConnectivityDFS.add\_all\_edge\_types ( self )

Allows connectivity through all edge types of the graph.

In a weak connectivity the edges can be used in Any direction.

## 5.95.3.2 def sparksee.WeakConnectivityDFS.add\_edge\_type ( self, type )

Allows connectivity through edges of the given type.

In a weak connectivity the edges can be used in Any direction.

## Parameters

<i>type</i>	[in] Edge type.
-------------	-----------------

## 5.95.3.3 def sparksee.WeakConnectivityDFS.add\_node\_type ( self, t )

Allows connectivity through nodes of the given type.

## Parameters

<i>t</i>	null
----------	------

## 5.95.3.4 def sparksee.Connectivity.close ( self ) [inherited]

Closes the [Connectivity](#) instance.

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

## 5.95.3.5 def sparksee.WeakConnectivityDFS.exclude\_edges ( self, 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

<i>edges</i>	[in] A set of edge identifiers that must be kept intact until the destruction of the class.
--------------	---

## 5.95.3.6 def sparksee.WeakConnectivityDFS.exclude\_nodes ( self, 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**

<i>nodes</i>	[in] A set of node identifiers that must be kept intact until the destruction of the class.
--------------	---

**5.95.3.7 def sparksee.WeakConnectivityDFS.get\_connected\_components ( self )**

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.

**5.95.3.8 def sparksee.Connectivity.is\_closed ( self ) [inherited]**

Gets if [Connectivity](#) has been closed or not.

**See also**

[close\(\)](#)

**Returns**

TRUE if the [Connectivity](#) instance has been closed, FALSE otherwise.

**5.95.3.9 def sparksee.WeakConnectivityDFS.set\_materialized\_attribute ( self, attribute\_name )**

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**

<i>attribute_name</i>	[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.
-----------------------	---

## Index

- \_\_init\_\_
  - sparksee::AttributeList, 18
  - sparksee::BooleanList, 24
  - sparksee::CommunitiesSCD, 28
  - sparksee::ConnectedComponents, 37
  - sparksee::Context, 43
  - sparksee::DisjointCommunities, 63
  - sparksee::EdgeTypeExporter, 75
  - sparksee::EdgeTypeLoader, 80
  - sparksee::Int32List, 116
  - sparksee::KOpt, 121
  - sparksee::NodeTypeExporter, 131
  - sparksee::NodeTypeLoader, 135
  - sparksee::OIDList, 147, 148
  - sparksee::PageRank, 151
  - sparksee::RandomWalk, 162
  - sparksee::ResultSetList, 170
  - sparksee::SinglePairShortestPathBFS, 187
  - sparksee::SinglePairShortestPathDijkstra, 193
  - sparksee::Sparksee, 198
  - sparksee::SparkseeConfig, 209, 210
  - sparksee::StringList, 227
  - sparksee::StrongConnectivityGabow, 234
  - sparksee::TextStream, 237
  - sparksee::TraversalBFS, 244
  - sparksee::TraversalDFS, 248
  - sparksee::TypeList, 258
  - sparksee::Value, 269
  - sparksee::ValueList, 286
  - sparksee::WeakConnectivityDFS, 296
- \_\_iter\_\_
  - sparksee::AttributeList, 18
  - sparksee::BooleanList, 24
  - sparksee::Int32List, 116
  - sparksee::OIDList, 148
  - sparksee::Objects, 139
  - sparksee::RandomWalk, 162
  - sparksee::ResultSetList, 170
  - sparksee::StringList, 227
  - sparksee::Traversal, 240
  - sparksee::TraversalBFS, 244
  - sparksee::TraversalDFS, 248
  - sparksee::TypeList, 258
  - sparksee::ValueList, 286
  - sparksee::Values, 288
- \_\_next\_\_
  - sparksee::AttributeListIterator, 19
  - sparksee::BooleanListIterator, 25
  - sparksee::Int32ListIterator, 118
  - sparksee::KeyValues, 119
  - sparksee::OIDListIterator, 149
  - sparksee::ObjectsIterator, 145
  - sparksee::RandomWalk, 163
  - sparksee::ResultSetListIterator, 171
  - sparksee::StringListIterator, 229
  - sparksee::Traversal, 240
  - sparksee::TraversalBFS, 244
  - sparksee::TraversalDFS, 248
  - sparksee::TypeListIterator, 260
  - sparksee::ValueListIterator, 287
  - sparksee::ValuesIterator, 290
- add
  - sparksee::AttributeList, 18
  - sparksee::BooleanList, 24
  - sparksee::Int32List, 116
  - sparksee::OIDList, 148
  - sparksee::Objects, 139
  - sparksee::StringList, 227
  - sparksee::TypeList, 258
  - sparksee::ValueList, 286
- add\_all\_edge\_types
  - sparksee::CommunitiesSCD, 28
  - sparksee::Context, 43
  - sparksee::DisjointCommunityDetection, 66
  - sparksee::KOpt, 122
  - sparksee::PageRank, 151
  - sparksee::RandomWalk, 163
  - sparksee::ShortestPath, 180
  - sparksee::SinglePairShortestPath, 183
  - sparksee::SinglePairShortestPathBFS, 188
  - sparksee::SinglePairShortestPathDijkstra, 193
  - sparksee::StrongConnectivity, 231
  - sparksee::StrongConnectivityGabow, 235
  - sparksee::Traversal, 240
  - sparksee::TraversalBFS, 244
  - sparksee::TraversalDFS, 249
  - sparksee::WeakConnectivity, 292
  - sparksee::WeakConnectivityDFS, 297
- add\_checksums
  - sparksee::Sparksee, 198
- add\_edge\_type
  - sparksee::CommunitiesSCD, 28
  - sparksee::Context, 43
  - sparksee::DisjointCommunityDetection, 66
  - sparksee::KOpt, 122
  - sparksee::PageRank, 151
  - sparksee::RandomWalk, 163
  - sparksee::ShortestPath, 180
  - sparksee::SinglePairShortestPath, 183
  - sparksee::SinglePairShortestPathBFS, 188
  - sparksee::SinglePairShortestPathDijkstra, 193
  - sparksee::StrongConnectivity, 231
  - sparksee::StrongConnectivityGabow, 235
  - sparksee::Traversal, 240
  - sparksee::TraversalBFS, 245
  - sparksee::TraversalDFS, 249
  - sparksee::WeakConnectivity, 292
  - sparksee::WeakConnectivityDFS, 297
- add\_node\_type

- sparksee::CommunitiesSCD, 28
- sparksee::CommunityDetection, 33
- sparksee::Connectivity, 40
- sparksee::Context, 43
- sparksee::DisjointCommunityDetection, 66
- sparksee::KOpt, 122
- sparksee::PageRank, 152
- sparksee::RandomWalk, 163
- sparksee::ShortestPath, 180
- sparksee::SinglePairShortestPath, 184
- sparksee::SinglePairShortestPathBFS, 188
- sparksee::SinglePairShortestPathDijkstra, 193
- sparksee::StrongConnectivity, 231
- sparksee::StrongConnectivityGabow, 235
- sparksee::Traversal, 241
- sparksee::TraversalBFS, 245
- sparksee::TraversalDFS, 249
- sparksee::WeakConnectivity, 293
- sparksee::WeakConnectivityDFS, 297
- add\_weighted\_edge\_type
  - sparksee::SinglePairShortestPathDijkstra, 193
- any
  - sparksee::Objects, 140
- as\_directed
  - sparksee::EdgeExport, 71
- BETWEEN
  - sparksee::Condition, 35
- backup
  - sparksee::Graph, 92
- CONFIG
  - sparksee::LogLevel, 124
- calculate\_edge\_cost
  - sparksee::SinglePairShortestPathDijkstraDynamic↔  
Cost, 196
- check\_only\_existence
  - sparksee::SinglePairShortestPathBFS, 188
- close
  - sparksee::CSVReader, 47
  - sparksee::CSVWriter, 52
  - sparksee::CommunitiesSCD, 28
  - sparksee::CommunityDetection, 33
  - sparksee::ConnectedComponents, 38
  - sparksee::Connectivity, 40
  - sparksee::Context, 44
  - sparksee::Database, 54
  - sparksee::DisjointCommunities, 63
  - sparksee::DisjointCommunityDetection, 67
  - sparksee::KeyValues, 119
  - sparksee::Objects, 140
  - sparksee::ObjectsIterator, 145
  - sparksee::Query, 157
  - sparksee::RandomWalk, 163
  - sparksee::RowReader, 172
  - sparksee::RowWriter, 174
  - sparksee::Session, 178
  - sparksee::ShortestPath, 181
  - sparksee::SinglePairShortestPath, 184
  - sparksee::SinglePairShortestPathBFS, 188
  - sparksee::SinglePairShortestPathDijkstra, 194
  - sparksee::Sparksee, 198
  - sparksee::StrongConnectivity, 231
  - sparksee::StrongConnectivityGabow, 235
  - sparksee::TextStream, 238
  - sparksee::Traversal, 241
  - sparksee::TraversalBFS, 245
  - sparksee::TraversalDFS, 249
  - sparksee::Values, 288
  - sparksee::ValuesIterator, 290
  - sparksee::WeakConnectivity, 293
  - sparksee::WeakConnectivityDFS, 297
- combine\_difference
  - sparksee::Objects, 140
- combine\_intersection
  - sparksee::Objects, 140
- combine\_union
  - sparksee::Objects, 141
- compare
  - sparksee::Value, 269
- compute
  - sparksee::Context, 44
- contains
  - sparksee::Objects, 141
- copy
  - sparksee::Objects, 141, 142
- count
  - sparksee::AttributeList, 18
  - sparksee::BooleanList, 24
  - sparksee::Int32List, 117
  - sparksee::OIDList, 148
  - sparksee::Objects, 142
  - sparksee::ResultSetList, 170
  - sparksee::StringList, 228
  - sparksee::TypeList, 259
  - sparksee::ValueList, 286
  - sparksee::Values, 289
- count\_edges
  - sparksee::Graph, 92
- count\_nodes
  - sparksee::Graph, 92
- create
  - sparksee::Sparksee, 198, 199
- DEBUG
  - sparksee::LogLevel, 124
- decrypt
  - sparksee::Sparksee, 199
- degree
  - sparksee::Graph, 92
- difference
  - sparksee::Objects, 142
- download\_license
  - sparksee::SparkseeConfig, 210
- drop
  - sparksee::Graph, 92, 93
- dump\_data
  - sparksee::Graph, 93

- dump\_schema
  - sparksee::Database, [54](#)
- dump\_storage
  - sparksee::Graph, [93](#)
- EQUAL
  - sparksee::Condition, [35](#)
- edges
  - sparksee::Graph, [94](#)
- enable\_type
  - sparksee::DefaultExport, [60](#)
  - sparksee::ExportManager, [85](#)
- encrypt
  - sparksee::Sparksee, [200](#)
- encrypted\_backup
  - sparksee::Graph, [94](#)
- equals
  - sparksee::Objects, [142](#)
  - sparksee::Value, [269](#)
- exclude\_edges
  - sparksee::CommunitiesSCD, [28](#)
  - sparksee::CommunityDetection, [33](#)
  - sparksee::Connectivity, [40](#)
  - sparksee::Context, [44](#)
  - sparksee::DisjointCommunityDetection, [67](#)
  - sparksee::RandomWalk, [163](#)
  - sparksee::ShortestPath, [181](#)
  - sparksee::SinglePairShortestPath, [184](#)
  - sparksee::SinglePairShortestPathBFS, [188](#)
  - sparksee::SinglePairShortestPathDijkstra, [194](#)
  - sparksee::StrongConnectivity, [231](#)
  - sparksee::StrongConnectivityGabow, [235](#)
  - sparksee::Traversal, [241](#)
  - sparksee::TraversalBFS, [245](#)
  - sparksee::TraversalDFS, [249](#)
  - sparksee::WeakConnectivity, [293](#)
  - sparksee::WeakConnectivityDFS, [297](#)
- exclude\_nodes
  - sparksee::CommunitiesSCD, [30](#)
  - sparksee::CommunityDetection, [33](#)
  - sparksee::Connectivity, [41](#)
  - sparksee::Context, [45](#)
  - sparksee::DisjointCommunityDetection, [67](#)
  - sparksee::RandomWalk, [164](#)
  - sparksee::ShortestPath, [181](#)
  - sparksee::SinglePairShortestPath, [184](#)
  - sparksee::SinglePairShortestPathBFS, [190](#)
  - sparksee::SinglePairShortestPathDijkstra, [194](#)
  - sparksee::StrongConnectivity, [232](#)
  - sparksee::StrongConnectivityGabow, [236](#)
  - sparksee::Traversal, [241](#)
  - sparksee::TraversalBFS, [245](#)
  - sparksee::TraversalDFS, [250](#)
  - sparksee::WeakConnectivity, [293](#)
  - sparksee::WeakConnectivityDFS, [297](#)
- execute
  - sparksee::Query, [157](#)
- exists
  - sparksee::Objects, [142](#)
- explode
  - sparksee::Graph, [94, 95](#)
- export
  - sparksee::Graph, [95](#)
- FINE
  - sparksee::LogLevel, [124](#)
- fetch
  - sparksee::QueryStream, [160](#)
- find\_attribute
  - sparksee::Graph, [95](#)
- find\_attributes
  - sparksee::Graph, [95](#)
- find\_edge
  - sparksee::Graph, [96](#)
- find\_edge\_types
  - sparksee::Graph, [96](#)
- find\_node\_types
  - sparksee::Graph, [96](#)
- find\_object
  - sparksee::Graph, [96](#)
- find\_or\_create\_edge
  - sparksee::Graph, [97](#)
- find\_or\_create\_object
  - sparksee::Graph, [97](#)
- find\_type
  - sparksee::Graph, [97](#)
- find\_types
  - sparksee::Graph, [98](#)
- GRAPHML
  - sparksee::ExportType, [88](#)
- GRAPHVIZ
  - sparksee::ExportType, [88](#)
- GREATER\_EQUAL
  - sparksee::Condition, [35](#)
- GREATER\_THAN
  - sparksee::Condition, [35](#)
- generate\_schema\_script
  - sparksee::ScriptParser, [175](#)
- get
  - sparksee::ResultSetList, [170](#)
  - sparksee::SparkseeProperties, [225](#)
  - sparksee::ValueArray, [277](#)
  - sparksee::ValueList, [286](#)
- get\_a\_e\_s\_i\_v
  - sparksee::SparkseeConfig, [210](#)
- get\_a\_e\_s\_key
  - sparksee::SparkseeConfig, [210](#)
- get\_alias
  - sparksee::Database, [55](#)
- get\_are\_neighbors\_indexed
  - sparksee::Type, [252](#)
- get\_array\_attribute
  - sparksee::Graph, [98](#)
- get\_array\_size
  - sparksee::Attribute, [15](#)
- get\_attribute
  - sparksee::Graph, [98, 99](#)



- get\_attribute\_interval\_count
  - sparksee::Graph, 99
- get\_attribute\_statistics
  - sparksee::Graph, 100
- get\_attribute\_text
  - sparksee::Graph, 100
- get\_attributes
  - sparksee::Graph, 100
- get\_available\_mem
  - sparksee::PlatformStatistics, 156
- get\_avg\_length\_string
  - sparksee::AttributeStatistics, 21
- get\_boolean
  - sparksee::SparkseeProperties, 225
  - sparksee::Value, 270
  - sparksee::ValueArray, 278
- get\_boolean\_range
  - sparksee::ValueArray, 278
- get\_cache
  - sparksee::DatabaseStatistics, 57
- get\_cache\_max\_size
  - sparksee::Database, 55
  - sparksee::SparkseeConfig, 211
- get\_cache\_statistics\_enabled
  - sparksee::SparkseeConfig, 211
- get\_cache\_statistics\_file
  - sparksee::SparkseeConfig, 211
- get\_cache\_statistics\_snapshot\_time
  - sparksee::SparkseeConfig, 211
- get\_call\_stack\_dump
  - sparksee::SparkseeConfig, 211
- get\_checksum\_enabled
  - sparksee::SparkseeConfig, 211
- get\_client\_id
  - sparksee::SparkseeConfig, 212
- get\_color\_rgb
  - sparksee::EdgeExport, 71
  - sparksee::NodeExport, 127
- get\_column
  - sparksee::ResultSet, 167
- get\_column\_data\_type
  - sparksee::ResultSet, 167
- get\_column\_index
  - sparksee::ResultSet, 168
- get\_column\_name
  - sparksee::ResultSet, 168
- get\_communities
  - sparksee::CommunitiesSCD, 30
  - sparksee::DisjointCommunityDetection, 67
- get\_community
  - sparksee::DisjointCommunities, 63
- get\_connected\_component
  - sparksee::ConnectedComponents, 38
- get\_connected\_components
  - sparksee::Connectivity, 41
  - sparksee::StrongConnectivity, 232
  - sparksee::StrongConnectivityGabow, 236
  - sparksee::WeakConnectivity, 293
- sparksee::WeakConnectivityDFS, 298
- get\_cost
  - sparksee::SinglePairShortestPath, 184
  - sparksee::SinglePairShortestPathBFS, 190
  - sparksee::SinglePairShortestPathDijkstra, 194
- get\_count
  - sparksee::Attribute, 15
  - sparksee::ConnectedComponents, 38
  - sparksee::DisjointCommunities, 63
  - sparksee::TypeExporterEvent, 256
  - sparksee::TypeLoaderEvent, 265
- get\_current\_depth
  - sparksee::RandomWalk, 164
  - sparksee::Traversal, 241
  - sparksee::TraversalBFS, 245
  - sparksee::TraversalDFS, 250
- get\_data
  - sparksee::DatabaseStatistics, 57
- get\_data\_type
  - sparksee::Attribute, 15
  - sparksee::Value, 270
- get\_distinct
  - sparksee::AttributeStatistics, 21
- get\_double
  - sparksee::Value, 270
  - sparksee::ValueArray, 278
- get\_double\_range
  - sparksee::ValueArray, 278
- get\_download\_status
  - sparksee::SparkseeConfig, 212
- get\_edge
  - sparksee::DefaultExport, 60
  - sparksee::EdgeData, 69
  - sparksee::ExportManager, 85
- get\_edge\_data
  - sparksee::Graph, 100
- get\_edge\_peer
  - sparksee::Graph, 101
- get\_edge\_type
  - sparksee::DefaultExport, 60
  - sparksee::ExportManager, 86
- get\_encryption\_enabled
  - sparksee::SparkseeConfig, 212
- get\_extent\_pages
  - sparksee::SparkseeConfig, 212
- get\_extent\_size
  - sparksee::SparkseeConfig, 212
- get\_font\_size
  - sparksee::EdgeExport, 71
  - sparksee::NodeExport, 127
- get\_graph
  - sparksee::DefaultExport, 61
  - sparksee::ExportManager, 86
  - sparksee::Session, 178
- get\_head
  - sparksee::EdgeData, 69
- get\_height
  - sparksee::NodeExport, 127

- get\_high\_availability\_coordinators
  - sparksee::SparkseeConfig, 212
- get\_high\_availability\_enabled
  - sparksee::SparkseeConfig, 213
- get\_high\_availability\_i\_p
  - sparksee::SparkseeConfig, 213
- get\_high\_availability\_master\_history
  - sparksee::SparkseeConfig, 213
- get\_high\_availability\_synchronization
  - sparksee::SparkseeConfig, 213
- get\_id
  - sparksee::Attribute, 15
  - sparksee::Type, 252
- get\_in\_mem\_alloc\_size
  - sparksee::SparkseeConfig, 213
- get\_in\_memory\_pool\_capacity
  - sparksee::Session, 178
  - sparksee::Sparksee, 200
- get\_integer
  - sparksee::SparkseeProperties, 225
  - sparksee::Value, 270
  - sparksee::ValueArray, 279
- get\_integer\_range
  - sparksee::ValueArray, 279
- get\_is\_directed
  - sparksee::Type, 252
- get\_is\_restricted
  - sparksee::Type, 252
- get\_j\_s\_o\_n
  - sparksee::ResultSet, 168
- get\_kind
  - sparksee::Attribute, 15
- get\_label
  - sparksee::EdgeExport, 71
  - sparksee::GraphExport, 115
  - sparksee::NodeExport, 127
- get\_labelcolor\_rgb
  - sparksee::EdgeExport, 72
  - sparksee::NodeExport, 127
- get\_license
  - sparksee::SparkseeConfig, 213
- get\_license\_id
  - sparksee::SparkseeConfig, 214
- get\_license\_pre\_download\_days
  - sparksee::SparkseeConfig, 214
- get\_log\_file
  - sparksee::SparkseeConfig, 214
- get\_log\_level
  - sparksee::SparkseeConfig, 214
- get\_long
  - sparksee::Value, 270
  - sparksee::ValueArray, 279
- get\_long\_range
  - sparksee::ValueArray, 280
- get\_max
  - sparksee::AttributeStatistics, 21
- get\_max\_length\_string
  - sparksee::AttributeStatistics, 21
- get\_mean
  - sparksee::AttributeStatistics, 21
- get\_median
  - sparksee::AttributeStatistics, 21
- get\_min
  - sparksee::AttributeStatistics, 22
- get\_min\_length\_string
  - sparksee::AttributeStatistics, 22
- get\_mode
  - sparksee::AttributeStatistics, 22
- get\_mode\_count
  - sparksee::AttributeStatistics, 22
- get\_name
  - sparksee::Attribute, 16
  - sparksee::Type, 252
- get\_node
  - sparksee::DefaultExport, 61
  - sparksee::ExportManager, 86
- get\_node\_type
  - sparksee::DefaultExport, 61
  - sparksee::ExportManager, 87
- get\_nodes
  - sparksee::ConnectedComponents, 38
  - sparksee::DisjointCommunities, 63
- get\_null
  - sparksee::AttributeStatistics, 22
- get\_num\_c\_p\_us
  - sparksee::PlatformStatistics, 156
- get\_num\_columns
  - sparksee::ResultSet, 169
- get\_num\_objects
  - sparksee::Type, 252
- get\_object\_type
  - sparksee::Graph, 101
  - sparksee::Type, 252
- get\_oid
  - sparksee::Value, 270
  - sparksee::ValueArray, 280
- get\_oid\_range
  - sparksee::ValueArray, 280
- get\_partition
  - sparksee::TypeLoaderEvent, 265
- get\_path
  - sparksee::Database, 55
- get\_path\_as\_edges
  - sparksee::SinglePairShortestPath, 185
  - sparksee::SinglePairShortestPathBFS, 190
  - sparksee::SinglePairShortestPathDijkstra, 194
- get\_path\_as\_nodes
  - sparksee::SinglePairShortestPath, 185
  - sparksee::SinglePairShortestPathBFS, 190
  - sparksee::SinglePairShortestPathDijkstra, 194
- get\_phase
  - sparksee::TypeLoaderEvent, 265
- get\_pool\_frame\_size
  - sparksee::SparkseeConfig, 214
- get\_pool\_partitions
  - sparksee::SparkseeConfig, 214

- get\_pool\_persistent\_max\_size
  - sparksee::SparkseeConfig, 215
- get\_pool\_persistent\_min\_size
  - sparksee::SparkseeConfig, 215
- get\_pool\_temporary\_max\_size
  - sparksee::SparkseeConfig, 215
- get\_pool\_temporary\_min\_size
  - sparksee::SparkseeConfig, 215
- get\_read
  - sparksee::DatabaseStatistics, 57
- get\_real\_time
  - sparksee::PlatformStatistics, 156
- get\_recovery\_cache\_max\_size
  - sparksee::SparkseeConfig, 215
- get\_recovery\_checkpoint\_time
  - sparksee::SparkseeConfig, 215
- get\_recovery\_enabled
  - sparksee::SparkseeConfig, 216
- get\_recovery\_log\_file
  - sparksee::SparkseeConfig, 216
- get\_restricted\_from
  - sparksee::Type, 253
- get\_restricted\_to
  - sparksee::Type, 253
- get\_rollback\_enabled
  - sparksee::SparkseeConfig, 216
- get\_row
  - sparksee::CSVReader, 48
  - sparksee::RowReader, 173
- get\_sessions
  - sparksee::DatabaseStatistics, 57
- get\_shape
  - sparksee::NodeExport, 128
- get\_size
  - sparksee::Attribute, 16
  - sparksee::ConnectedComponents, 39
  - sparksee::DisjointCommunities, 64
- get\_sparksee\_config\_file
  - sparksee::SparkseeConfig, 216
- get\_statistics
  - sparksee::Database, 55
  - sparksee::Platform, 155
- get\_string
  - sparksee::Value, 271
- get\_system\_time
  - sparksee::PlatformStatistics, 156
- get\_tail
  - sparksee::EdgeData, 69
- get\_temp
  - sparksee::DatabaseStatistics, 57
- get\_time\_unit
  - sparksee::SparkseeProperties, 226
- get\_timestamp
  - sparksee::Value, 271
  - sparksee::ValueArray, 280
- get\_timestamp\_range
  - sparksee::ValueArray, 281
- get\_tmp\_enabled
  - sparksee::SparkseeConfig, 216
- get\_tmp\_folder
  - sparksee::SparkseeConfig, 216
- get\_total
  - sparksee::AttributeStatistics, 23
  - sparksee::TypeExporterEvent, 256
- get\_total\_mem
  - sparksee::PlatformStatistics, 156
- get\_total\_partition\_steps
  - sparksee::TypeLoaderEvent, 265
- get\_total\_partitions
  - sparksee::TypeLoaderEvent, 266
- get\_total\_phases
  - sparksee::TypeLoaderEvent, 266
- get\_type
  - sparksee::Graph, 101
- get\_type\_id
  - sparksee::Attribute, 16
  - sparksee::TypeExporterEvent, 257
  - sparksee::TypeLoaderEvent, 266
- get\_user\_time
  - sparksee::PlatformStatistics, 156
- get\_values
  - sparksee::Graph, 101
- get\_variance
  - sparksee::AttributeStatistics, 23
- get\_width
  - sparksee::EdgeExport, 72
  - sparksee::NodeExport, 128
- get\_write
  - sparksee::DatabaseStatistics, 57
- has\_next
  - sparksee::AttributeListIterator, 19
  - sparksee::BooleanListIterator, 25
  - sparksee::Int32ListIterator, 118
  - sparksee::KeyValues, 119
  - sparksee::OIDListIterator, 149
  - sparksee::ObjectsIterator, 146
  - sparksee::RandomWalk, 164
  - sparksee::ResultSetListIterator, 171
  - sparksee::StringListIterator, 229
  - sparksee::Traversal, 241
  - sparksee::TraversalBFS, 246
  - sparksee::TraversalDFS, 250
  - sparksee::TypeListIterator, 260
  - sparksee::ValueListIterator, 287
  - sparksee::ValuesIterator, 290
- heads
  - sparksee::Graph, 102
- INFO
  - sparksee::LogLevel, 124
- include\_edges
  - sparksee::CommunitiesSCD, 30
  - sparksee::CommunityDetection, 33
  - sparksee::DisjointCommunityDetection, 67
- include\_nodes
  - sparksee::CommunitiesSCD, 30

- sparksee::CommunityDetection, 34
- sparksee::DisjointCommunityDetection, 68
- index\_attribute
  - sparksee::Graph, 102
- index\_neighbors
  - sparksee::Graph, 102
- intersection
  - sparksee::Objects, 143
- is\_array\_attribute
  - sparksee::Attribute, 16
- is\_closed
  - sparksee::CommunitiesSCD, 31
  - sparksee::CommunityDetection, 34
  - sparksee::ConnectedComponents, 39
  - sparksee::Connectivity, 41
  - sparksee::Context, 45
  - sparksee::Database, 55
  - sparksee::DisjointCommunities, 64
  - sparksee::DisjointCommunityDetection, 68
  - sparksee::KeyValues, 119
  - sparksee::Objects, 143
  - sparksee::ObjectsIterator, 146
  - sparksee::Query, 157
  - sparksee::RandomWalk, 164
  - sparksee::ResultSet, 169
  - sparksee::Session, 178
  - sparksee::ShortestPath, 181
  - sparksee::SinglePairShortestPath, 185
  - sparksee::SinglePairShortestPathBFS, 190
  - sparksee::SinglePairShortestPathDijkstra, 194
  - sparksee::StrongConnectivity, 232
  - sparksee::StrongConnectivityGabow, 236
  - sparksee::Traversal, 241
  - sparksee::TraversalBFS, 246
  - sparksee::TraversalDFS, 250
  - sparksee::Values, 289
  - sparksee::ValuesIterator, 290
  - sparksee::WeakConnectivity, 293
  - sparksee::WeakConnectivityDFS, 298
- is\_fit
  - sparksee::NodeExport, 128
- is\_last
  - sparksee::TypeExporterEvent, 257
  - sparksee::TypeLoaderEvent, 266
- is\_null
  - sparksee::TextStream, 238
  - sparksee::Value, 271
- is\_session\_attribute
  - sparksee::Attribute, 16
- iterator
  - sparksee::AttributeList, 18
  - sparksee::BooleanList, 24
  - sparksee::Int32List, 117
  - sparksee::OIDList, 148
  - sparksee::Objects, 143
  - sparksee::ResultSetList, 170
  - sparksee::StringList, 228
  - sparksee::TypeList, 259
  - sparksee::ValueList, 286
  - sparksee::Values, 289
- iterator\_from\_element
  - sparksee::Objects, 143
- iterator\_from\_index
  - sparksee::Objects, 144
- LESS\_EQUAL
  - sparksee::Condition, 35
- LESS\_THAN
  - sparksee::Condition, 35
- LIKE\_NO\_CASE
  - sparksee::Condition, 36
- LIKE
  - sparksee::Condition, 35
- load
  - sparksee::SparkseeProperties, 226
- load\_edges\_csv
  - sparksee, 13
- load\_nodes\_csv
  - sparksee, 14
- make\_null
  - sparksee::ValueArray, 281
- NOT\_EQUAL
  - sparksee::Condition, 36
- neighbors
  - sparksee::Graph, 102, 103
- new\_array\_attribute
  - sparksee::Graph, 103
- new\_attribute
  - sparksee::Graph, 103, 104
- new\_edge
  - sparksee::Graph, 104, 105
- new\_edge\_type
  - sparksee::Graph, 105
- new\_node
  - sparksee::Graph, 105
- new\_node\_type
  - sparksee::Graph, 105
- new\_objects
  - sparksee::Session, 178
- new\_query
  - sparksee::Session, 178
- new\_restricted\_edge\_type
  - sparksee::Graph, 106
- new\_session\_array\_attribute
  - sparksee::Graph, 106
- new\_session\_attribute
  - sparksee::Graph, 106, 107
- next
  - sparksee::AttributeListIterator, 19
  - sparksee::BooleanListIterator, 25
  - sparksee::Int32ListIterator, 118
  - sparksee::KeyValues, 120
  - sparksee::OIDListIterator, 149
  - sparksee::ObjectsIterator, 146
  - sparksee::RandomWalk, 164

- sparksee::ResultSet, 169
- sparksee::ResultSetListIterator, 171
- sparksee::StringListIterator, 229
- sparksee::Traversal, 242
- sparksee::TraversalBFS, 246
- sparksee::TraversalDFS, 250
- sparksee::TypeListIterator, 260
- sparksee::ValueListIterator, 287
- sparksee::ValuesIterator, 290
- notify\_event
  - sparksee::TypeExporterListener, 257
  - sparksee::TypeLoaderListener, 266
- open
  - sparksee::CSVReader, 48
  - sparksee::CSVWriter, 52
  - sparksee::Sparksee, 200
- operator
  - sparksee::Value, 271
- parse
  - sparksee::ScriptParser, 176
- pre\_commit
  - sparksee::Session, 179
- prepare
  - sparksee::DefaultExport, 62
  - sparksee::ExportManager, 87
  - sparksee::QueryStream, 160
- REG\_EXP
  - sparksee::Condition, 36
- read
  - sparksee::CSVReader, 48
  - sparksee::RowReader, 173
  - sparksee::TextStream, 238
- redo\_precommitted
  - sparksee::Database, 55
- register
  - sparksee::EdgeTypeExporter, 75
  - sparksee::EdgeTypeLoader, 80
  - sparksee::NodeTypeExporter, 132
  - sparksee::NodeTypeLoader, 135
  - sparksee::TypeExporter, 254
  - sparksee::TypeLoader, 261
- release
  - sparksee::ExportManager, 87
- remove
  - sparksee::Objects, 144
- remove\_attribute
  - sparksee::Graph, 107
- remove\_checksums
  - sparksee::Sparksee, 201
- remove\_type
  - sparksee::Graph, 107
- rename\_attribute
  - sparksee::Graph, 108
- rename\_type
  - sparksee::Graph, 108
- reset
  - sparksee::CSVReader, 48
  - sparksee::RandomWalk, 165
  - sparksee::RowReader, 173
- resize\_in\_memory\_pool
  - sparksee::Sparksee, 201
- restore
  - sparksee::Sparksee, 202
- restore\_encrypted\_backup
  - sparksee::Sparksee, 203
- run
  - sparksee::CommunityDetection, 34
  - sparksee::Connectivity, 41
  - sparksee::DisjointCommunityDetection, 68
  - sparksee::EdgeTypeExporter, 76
  - sparksee::EdgeTypeLoader, 80
  - sparksee::NodeTypeExporter, 132
  - sparksee::NodeTypeLoader, 135
  - sparksee::PageRank, 152
  - sparksee::ShortestPath, 181
  - sparksee::SinglePairShortestPath, 185
  - sparksee::StrongConnectivity, 232
  - sparksee::TypeExporter, 254
  - sparksee::TypeLoader, 262
  - sparksee::WeakConnectivity, 294
- run\_n\_phases
  - sparksee::EdgeTypeLoader, 80
  - sparksee::NodeTypeLoader, 135
  - sparksee::TypeLoader, 262
- run\_two\_phases
  - sparksee::EdgeTypeLoader, 81
  - sparksee::NodeTypeLoader, 136
  - sparksee::TypeLoader, 262
- SEVERE
  - sparksee::LogLevel, 125
- STRING
  - sparksee::DataType, 58
- sample
  - sparksee::Objects, 144
- save
  - sparksee::SparkseeConfig, 217
- save\_all
  - sparksee::SparkseeConfig, 217
- select
  - sparksee::Graph, 108–110
- set
  - sparksee::OIDList, 148
  - sparksee::ValueArray, 281
- set\_a\_e\_s\_encryption\_enabled
  - sparksee::SparkseeConfig, 217
- set\_array\_attribute
  - sparksee::Graph, 110
- set\_array\_attribute\_void
  - sparksee::Graph, 111
- set\_as\_directed
  - sparksee::EdgeExport, 72
- set\_attribute
  - sparksee::Graph, 111
- set\_attribute\_default\_value

- sparksee::Graph, 112
- set\_attribute\_positions
  - sparksee::EdgeTypeLoader, 81
  - sparksee::NodeTypeLoader, 136
  - sparksee::TypeLoader, 262
- set\_attribute\_text
  - sparksee::Graph, 112
- set\_attributes
  - sparksee::EdgeTypeExporter, 76
  - sparksee::EdgeTypeLoader, 81
  - sparksee::NodeTypeExporter, 132
  - sparksee::NodeTypeLoader, 136
  - sparksee::TypeExporter, 255
  - sparksee::TypeLoader, 263
- set\_auto\_quotes
  - sparksee::CSVWriter, 52
- set\_boolean
  - sparksee::Value, 271
  - sparksee::ValueArray, 282
- set\_boolean\_range
  - sparksee::ValueArray, 282
- set\_boolean\_void
  - sparksee::Value, 272
- set\_cache\_max\_size
  - sparksee::Database, 56
  - sparksee::SparkseeConfig, 218
- set\_cache\_statistics\_enabled
  - sparksee::SparkseeConfig, 218
- set\_cache\_statistics\_file
  - sparksee::SparkseeConfig, 218
- set\_cache\_statistics\_snapshot\_time
  - sparksee::SparkseeConfig, 218
- set\_call\_stack\_dump
  - sparksee::SparkseeConfig, 219
- set\_checksum\_enabled
  - sparksee::SparkseeConfig, 219
- set\_client\_id
  - sparksee::SparkseeConfig, 219
- set\_color\_rgb
  - sparksee::EdgeExport, 72
  - sparksee::NodeExport, 128
- set\_current\_tour
  - sparksee::KOpt, 122
- set\_damping
  - sparksee::PageRank, 152
- set\_default\_weight
  - sparksee::PageRank, 152
  - sparksee::RandomWalk, 165
- set\_double
  - sparksee::Value, 272
  - sparksee::ValueArray, 282
- set\_double\_range
  - sparksee::ValueArray, 283
- set\_double\_void
  - sparksee::Value, 272
- set\_dynamic
  - sparksee::Query, 158
- set\_dynamic\_edge\_cost\_callback
  - sparksee::SinglePairShortestPathDijkstra, 195
- set\_edge\_weight\_attribute\_type
  - sparksee::KOpt, 123
  - sparksee::PageRank, 153
  - sparksee::RandomWalk, 165
- set\_error\_log
  - sparksee::ScriptParser, 176
- set\_extent\_pages
  - sparksee::SparkseeConfig, 219
- set\_extent\_size
  - sparksee::SparkseeConfig, 219
- set\_fit
  - sparksee::NodeExport, 128
- set\_font\_size
  - sparksee::EdgeExport, 72
  - sparksee::NodeExport, 129
- set\_forced\_quotes
  - sparksee::CSVWriter, 52
- set\_frequency
  - sparksee::EdgeTypeExporter, 76
  - sparksee::EdgeTypeLoader, 81
  - sparksee::NodeTypeExporter, 132
  - sparksee::NodeTypeLoader, 136
  - sparksee::TypeExporter, 255
  - sparksee::TypeLoader, 263
- set\_graph
  - sparksee::EdgeTypeExporter, 76
  - sparksee::EdgeTypeLoader, 81
  - sparksee::NodeTypeExporter, 132
  - sparksee::NodeTypeLoader, 136
  - sparksee::TypeExporter, 255
  - sparksee::TypeLoader, 263
- set\_h\_i
  - sparksee::SparkseeProperties, 226
- set\_head\_attribute
  - sparksee::EdgeTypeExporter, 76
  - sparksee::EdgeTypeLoader, 82
- set\_head\_mep
  - sparksee::EdgeTypeLoader, 82
- set\_head\_position
  - sparksee::EdgeTypeExporter, 77
  - sparksee::EdgeTypeLoader, 82
- set\_header
  - sparksee::EdgeTypeExporter, 77
  - sparksee::NodeTypeExporter, 132
  - sparksee::TypeExporter, 255
- set\_height
  - sparksee::NodeExport, 129
- set\_high\_availability\_coordinators
  - sparksee::SparkseeConfig, 220
- set\_high\_availability\_enabled
  - sparksee::SparkseeConfig, 220
- set\_high\_availability\_i\_p
  - sparksee::SparkseeConfig, 220
- set\_high\_availability\_master\_history
  - sparksee::SparkseeConfig, 220
- set\_high\_availability\_synchronization
  - sparksee::SparkseeConfig, 220

set\_in\_mem\_alloc\_size  
   sparksee::SparkseeConfig, 220  
 set\_in\_out\_parameter  
   sparksee::RandomWalk, 165  
 set\_initial\_page\_rank\_value  
   sparksee::PageRank, 153  
 set\_integer  
   sparksee::Value, 272  
   sparksee::ValueArray, 283  
 set\_integer\_range  
   sparksee::ValueArray, 283  
 set\_integer\_void  
   sparksee::Value, 273  
 set\_label  
   sparksee::EdgeExport, 73  
   sparksee::GraphExport, 115  
   sparksee::NodeExport, 129  
 set\_labelcolor\_rgb  
   sparksee::EdgeExport, 73  
   sparksee::NodeExport, 129  
 set\_license  
   sparksee::SparkseeConfig, 221  
 set\_license\_id  
   sparksee::SparkseeConfig, 221  
 set\_license\_pre\_download\_days  
   sparksee::SparkseeConfig, 221  
 set\_locale  
   sparksee::CSVReader, 49  
   sparksee::CSVWriter, 52  
   sparksee::EdgeTypeLoader, 82  
   sparksee::NodeTypeLoader, 137  
   sparksee::TypeLoader, 263  
 set\_log\_error  
   sparksee::EdgeTypeLoader, 82  
   sparksee::NodeTypeLoader, 137  
   sparksee::TypeLoader, 263  
 set\_log\_file  
   sparksee::SparkseeConfig, 221  
 set\_log\_level  
   sparksee::SparkseeConfig, 221  
 set\_log\_off  
   sparksee::EdgeTypeLoader, 83  
   sparksee::NodeTypeLoader, 137  
   sparksee::TypeLoader, 264  
 set\_long  
   sparksee::Value, 273  
   sparksee::ValueArray, 283  
 set\_long\_range  
   sparksee::ValueArray, 284  
 set\_long\_void  
   sparksee::Value, 273  
 set\_look\_ahead  
   sparksee::CommunitiesSCD, 31  
 set\_materialized\_attribute  
   sparksee::CommunitiesSCD, 31  
   sparksee::Connectivity, 41  
   sparksee::DisjointCommunityDetection, 68  
   sparksee::StrongConnectivity, 232  
   sparksee::StrongConnectivityGabow, 236  
   sparksee::WeakConnectivity, 294  
   sparksee::WeakConnectivityDFS, 298  
 set\_max\_iterations  
   sparksee::KOpt, 123  
 set\_maximum\_hops  
   sparksee::Context, 45  
   sparksee::RandomWalk, 166  
   sparksee::ShortestPath, 181  
   sparksee::SinglePairShortestPath, 185  
   sparksee::SinglePairShortestPathBFS, 190  
   sparksee::SinglePairShortestPathDijkstra, 195  
   sparksee::Traversal, 242  
   sparksee::TraversalBFS, 246  
   sparksee::TraversalDFS, 250  
 set\_multilines  
   sparksee::CSVReader, 49  
 set\_null  
   sparksee::Value, 273  
 set\_num\_iterations  
   sparksee::PageRank, 153  
 set\_num\_lines  
   sparksee::CSVReader, 49  
 set\_oid  
   sparksee::Value, 273  
   sparksee::ValueArray, 284  
 set\_oid\_range  
   sparksee::ValueArray, 284  
 set\_oid\_void  
   sparksee::Value, 274  
 set\_output\_attribute\_type  
   sparksee::PageRank, 153  
 set\_output\_log  
   sparksee::ScriptParser, 176  
 set\_pool\_frame\_size  
   sparksee::SparkseeConfig, 222  
 set\_pool\_partitions  
   sparksee::SparkseeConfig, 222  
 set\_pool\_persistent\_max\_size  
   sparksee::SparkseeConfig, 222  
 set\_pool\_persistent\_min\_size  
   sparksee::SparkseeConfig, 222  
 set\_pool\_temporary\_max\_size  
   sparksee::SparkseeConfig, 222  
 set\_pool\_temporary\_min\_size  
   sparksee::SparkseeConfig, 222  
 set\_quotes  
   sparksee::CSVReader, 49  
   sparksee::CSVWriter, 53  
 set\_recovery\_cache\_max\_size  
   sparksee::SparkseeConfig, 223  
 set\_recovery\_checkpoint\_time  
   sparksee::SparkseeConfig, 223  
 set\_recovery\_enabled  
   sparksee::SparkseeConfig, 223  
 set\_recovery\_log\_file  
   sparksee::SparkseeConfig, 223  
 set\_return\_parameter

- sparksee::RandomWalk, 166
- set\_rollback\_enabled
  - sparksee::SparkseeConfig, 223
- set\_row\_reader
  - sparksee::EdgeTypeLoader, 83
  - sparksee::NodeTypeLoader, 137
  - sparksee::TypeLoader, 264
- set\_row\_writer
  - sparksee::EdgeTypeExporter, 77
  - sparksee::NodeTypeExporter, 133
  - sparksee::TypeExporter, 255
- set\_seed
  - sparksee::RandomWalk, 166
- set\_separator
  - sparksee::CSVReader, 50
  - sparksee::CSVWriter, 53
- set\_shape
  - sparksee::NodeExport, 129
- set\_sparksee\_config\_file
  - sparksee::SparkseeConfig, 224
- set\_start\_line
  - sparksee::CSVReader, 50
- set\_starting\_node
  - sparksee::PageRank, 154
- set\_stream
  - sparksee::Query, 158
- set\_string
  - sparksee::Value, 274
- set\_string\_void
  - sparksee::Value, 274
- set\_tail\_attribute
  - sparksee::EdgeTypeExporter, 77
  - sparksee::EdgeTypeLoader, 83
- set\_tail\_mep
  - sparksee::EdgeTypeLoader, 83
- set\_tail\_position
  - sparksee::EdgeTypeExporter, 77
  - sparksee::EdgeTypeLoader, 83
- set\_time\_limit
  - sparksee::KOpt, 123
- set\_timestamp
  - sparksee::Value, 274
  - sparksee::ValueArray, 284
- set\_timestamp\_format
  - sparksee::EdgeTypeLoader, 84
  - sparksee::NodeTypeLoader, 137
  - sparksee::TypeLoader, 264
- set\_timestamp\_range
  - sparksee::ValueArray, 285
- set\_timestamp\_void
  - sparksee::Value, 275
- set\_tmp\_enabled
  - sparksee::SparkseeConfig, 224
- set\_tmp\_folder
  - sparksee::SparkseeConfig, 224
- set\_tolerance
  - sparksee::PageRank, 154
- set\_type
  - sparksee::EdgeTypeExporter, 77
  - sparksee::EdgeTypeLoader, 84
  - sparksee::NodeTypeExporter, 133
  - sparksee::NodeTypeLoader, 138
  - sparksee::TypeExporter, 255
  - sparksee::TypeLoader, 264
- set\_unrecoverable\_error\_callback
  - sparksee::Sparksee, 204
- set\_unweighted\_edge\_cost
  - sparksee::SinglePairShortestPathDijkstra, 195
- set\_void
  - sparksee::Value, 275
- set\_width
  - sparksee::EdgeExport, 73
  - sparksee::NodeExport, 129
- sparksee, 9
  - load\_edges\_csv, 13
  - load\_nodes\_csv, 14
- sparksee.Attribute, 14
- sparksee.AttributeKind, 17
- sparksee.AttributeList, 17
- sparksee.AttributeListIterator, 19
- sparksee.AttributeStatistics, 20
- sparksee.BooleanList, 23
- sparksee.BooleanListIterator, 25
- sparksee.CSVReader, 45
- sparksee.CSVWriter, 50
- sparksee.CommunitiesSCD, 26
- sparksee.CommunityDetection, 31
- sparksee.Condition, 34
- sparksee.ConnectedComponents, 36
- sparksee.Connectivity, 39
- sparksee.Context, 42
- sparksee.DataType, 58
- sparksee.Database, 53
- sparksee.DatabaseStatistics, 56
- sparksee.DefaultExport, 59
- sparksee.DisjointCommunities, 62
- sparksee.DisjointCommunityDetection, 64
- sparksee.EdgeData, 69
- sparksee.EdgeExport, 70
- sparksee.EdgeTypeExporter, 74
- sparksee.EdgeTypeLoader, 78
- sparksee.EdgesDirection, 73
- sparksee.ExportManager, 84
- sparksee.ExportType, 87
- sparksee.Graph, 88
- sparksee.GraphExport, 115
- sparksee.Int32List, 116
- sparksee.Int32ListIterator, 117
- sparksee.KOpt, 120
- sparksee.KeyValue, 118
- sparksee.KeyValues, 118
- sparksee.LogLevel, 123
- sparksee.MissingEndpoint, 125
- sparksee.NodeExport, 125
- sparksee.NodeShape, 130
- sparksee.NodeTypeExporter, 130



- sparksee.NodeTypeLoader, 133
- sparksee.OIDList, 147
- sparksee.OIDListIterator, 149
- sparksee.ObjectType, 146
- sparksee.Objects, 138
- sparksee.ObjectsIterator, 145
- sparksee.Order, 150
- sparksee.PageRank, 150
- sparksee.Platform, 154
- sparksee.PlatformStatistics, 155
- sparksee.Query, 157
- sparksee.QueryContext, 158
- sparksee.QueryLanguage, 159
- sparksee.QueryStream, 159
- sparksee.RandomWalk, 160
- sparksee.ResultSet, 166
- sparksee.ResultSetList, 169
- sparksee.ResultSetListIterator, 171
- sparksee.RowReader, 172
- sparksee.RowWriter, 174
- sparksee.ScriptParser, 175
- sparksee.Session, 177
- sparksee.ShortestPath, 179
- sparksee.SinglePairShortestPath, 182
- sparksee.SinglePairShortestPathBFS, 186
- sparksee.SinglePairShortestPathDijkstra, 191
- sparksee.SinglePairShortestPathDijkstraDynamicCost, 195
- sparksee.Sparksee, 196
- sparksee.SparkseeConfig, 204
- sparksee.SparkseeProperties, 224
- sparksee.StringList, 227
- sparksee.StringListIterator, 228
- sparksee.StrongConnectivity, 229
- sparksee.StrongConnectivityGabow, 233
- sparksee.TextStream, 237
- sparksee.Traversal, 239
- sparksee.TraversalBFS, 242
- sparksee.TraversalDFS, 246
- sparksee.Type, 251
- sparksee.TypeExporter, 253
- sparksee.TypeExporterEvent, 256
- sparksee.TypeExporterListener, 257
- sparksee.TypeList, 258
- sparksee.TypeListIterator, 259
- sparksee.TypeLoader, 260
- sparksee.TypeLoaderEvent, 264
- sparksee.TypeLoaderListener, 266
- sparksee.Value, 267
- sparksee.ValueArray, 276
- sparksee.ValueList, 285
- sparksee.ValueListIterator, 287
- sparksee.Values, 288
- sparksee.ValuesIterator, 289
- sparksee.WeakConnectivity, 291
- sparksee.WeakConnectivityDFS, 294
- sparksee::Attribute
  - get\_array\_size, 15
  - get\_count, 15
  - get\_data\_type, 15
  - get\_id, 15
  - get\_kind, 15
  - get\_name, 16
  - get\_size, 16
  - get\_type\_id, 16
  - is\_array\_attribute, 16
  - is\_session\_attribute, 16
- sparksee::AttributeKind
  - UNIQUE, 17
- sparksee::AttributeList
  - \_\_init\_\_, 18
  - \_\_iter\_\_, 18
  - add, 18
  - count, 18
  - iterator, 18
- sparksee::AttributeListIterator
  - \_\_next\_\_, 19
  - has\_next, 19
  - next, 19
- sparksee::AttributeStatistics
  - get\_avg\_length\_string, 21
  - get\_distinct, 21
  - get\_max, 21
  - get\_max\_length\_string, 21
  - get\_mean, 21
  - get\_median, 21
  - get\_min, 22
  - get\_min\_length\_string, 22
  - get\_mode, 22
  - get\_mode\_count, 22
  - get\_null, 22
  - get\_total, 23
  - get\_variance, 23
- sparksee::BooleanList
  - \_\_init\_\_, 24
  - \_\_iter\_\_, 24
  - add, 24
  - count, 24
  - iterator, 24
- sparksee::BooleanListIterator
  - \_\_next\_\_, 25
  - has\_next, 25
  - next, 25
- sparksee::CSVReader
  - close, 47
  - get\_row, 48
  - open, 48
  - read, 48
  - reset, 48
  - set\_locale, 49
  - set\_multilines, 49
  - set\_num\_lines, 49
  - set\_quotes, 49
  - set\_separator, 50
  - set\_start\_line, 50
- sparksee::CSVWriter

- close, 52
- open, 52
- set\_auto\_quotes, 52
- set\_forced\_quotes, 52
- set\_locale, 52
- set\_quotes, 53
- set\_separator, 53
- write, 53
- sparksee::CommunitiesSCD
  - \_\_init\_\_, 28
  - add\_all\_edge\_types, 28
  - add\_edge\_type, 28
  - add\_node\_type, 28
  - close, 28
  - exclude\_edges, 28
  - exclude\_nodes, 30
  - get\_communities, 30
  - include\_edges, 30
  - include\_nodes, 30
  - is\_closed, 31
  - set\_look\_ahead, 31
  - set\_materialized\_attribute, 31
- sparksee::CommunityDetection
  - add\_node\_type, 33
  - close, 33
  - exclude\_edges, 33
  - exclude\_nodes, 33
  - include\_edges, 33
  - include\_nodes, 34
  - is\_closed, 34
  - run, 34
- sparksee::Condition
  - BETWEEN, 35
  - EQUAL, 35
  - GREATER\_EQUAL, 35
  - GREATER\_THAN, 35
  - LESS\_EQUAL, 35
  - LESS\_THAN, 35
  - LIKE\_NO\_CASE, 36
  - LIKE, 35
  - NOT\_EQUAL, 36
  - REG\_EXP, 36
- sparksee::ConnectedComponents
  - \_\_init\_\_, 37
  - close, 38
  - get\_connected\_component, 38
  - get\_count, 38
  - get\_nodes, 38
  - get\_size, 39
  - is\_closed, 39
- sparksee::Connectivity
  - add\_node\_type, 40
  - close, 40
  - exclude\_edges, 40
  - exclude\_nodes, 41
  - get\_connected\_components, 41
  - is\_closed, 41
  - run, 41
  - set\_materialized\_attribute, 41
- sparksee::Context
  - \_\_init\_\_, 43
  - add\_all\_edge\_types, 43
  - add\_edge\_type, 43
  - add\_node\_type, 43
  - close, 44
  - compute, 44
  - exclude\_edges, 44
  - exclude\_nodes, 45
  - is\_closed, 45
  - set\_maximum\_hops, 45
- sparksee::DataType
  - STRING, 58
  - TEXT, 58
  - TIMESTAMP, 58
- sparksee::Database
  - close, 54
  - dump\_schema, 54
  - get\_alias, 55
  - get\_cache\_max\_size, 55
  - get\_path, 55
  - get\_statistics, 55
  - is\_closed, 55
  - redo\_precommitted, 55
  - set\_cache\_max\_size, 56
- sparksee::DatabaseStatistics
  - get\_cache, 57
  - get\_data, 57
  - get\_read, 57
  - get\_sessions, 57
  - get\_temp, 57
  - get\_write, 57
- sparksee::DefaultExport
  - enable\_type, 60
  - get\_edge, 60
  - get\_edge\_type, 60
  - get\_graph, 61
  - get\_node, 61
  - get\_node\_type, 61
  - prepare, 62
- sparksee::DisjointCommunities
  - \_\_init\_\_, 63
  - close, 63
  - get\_community, 63
  - get\_count, 63
  - get\_nodes, 63
  - get\_size, 64
  - is\_closed, 64
- sparksee::DisjointCommunityDetection
  - add\_all\_edge\_types, 66
  - add\_edge\_type, 66
  - add\_node\_type, 66
  - close, 67
  - exclude\_edges, 67
  - exclude\_nodes, 67
  - get\_communities, 67
  - include\_edges, 67

- include\_nodes, 68
- is\_closed, 68
- run, 68
- set\_materialized\_attribute, 68
- sparksee::EdgeData
  - get\_edge, 69
  - get\_head, 69
  - get\_tail, 69
- sparksee::EdgeExport
  - as\_directed, 71
  - get\_color\_rgb, 71
  - get\_font\_size, 71
  - get\_label, 71
  - get\_labelcolor\_rgb, 72
  - get\_width, 72
  - set\_as\_directed, 72
  - set\_color\_rgb, 72
  - set\_font\_size, 72
  - set\_label, 73
  - set\_labelcolor\_rgb, 73
  - set\_width, 73
- sparksee::EdgeTypeExporter
  - \_\_init\_\_, 75
  - register, 75
  - run, 76
  - set\_attributes, 76
  - set\_frequency, 76
  - set\_graph, 76
  - set\_head\_attribute, 76
  - set\_head\_position, 77
  - set\_header, 77
  - set\_row\_writer, 77
  - set\_tail\_attribute, 77
  - set\_tail\_position, 77
  - set\_type, 77
- sparksee::EdgeTypeLoader
  - \_\_init\_\_, 80
  - register, 80
  - run, 80
  - run\_n\_phases, 80
  - run\_two\_phases, 81
  - set\_attribute\_positions, 81
  - set\_attributes, 81
  - set\_frequency, 81
  - set\_graph, 81
  - set\_head\_attribute, 82
  - set\_head\_mep, 82
  - set\_head\_position, 82
  - set\_locale, 82
  - set\_log\_error, 82
  - set\_log\_off, 83
  - set\_row\_reader, 83
  - set\_tail\_attribute, 83
  - set\_tail\_mep, 83
  - set\_tail\_position, 83
  - set\_timestamp\_format, 84
  - set\_type, 84
- sparksee::ExportManager
  - enable\_type, 85
  - get\_edge, 85
  - get\_edge\_type, 86
  - get\_graph, 86
  - get\_node, 86
  - get\_node\_type, 87
  - prepare, 87
  - release, 87
- sparksee::ExportType
  - GRAPHML, 88
  - GRAPHVIZ, 88
  - YGRAPHML, 88
- sparksee::Graph
  - backup, 92
  - count\_edges, 92
  - count\_nodes, 92
  - degree, 92
  - drop, 92, 93
  - dump\_data, 93
  - dump\_storage, 93
  - edges, 94
  - encrypted\_backup, 94
  - explode, 94, 95
  - export, 95
  - find\_attribute, 95
  - find\_attributes, 95
  - find\_edge, 96
  - find\_edge\_types, 96
  - find\_node\_types, 96
  - find\_object, 96
  - find\_or\_create\_edge, 97
  - find\_or\_create\_object, 97
  - find\_type, 97
  - find\_types, 98
  - get\_array\_attribute, 98
  - get\_attribute, 98, 99
  - get\_attribute\_interval\_count, 99
  - get\_attribute\_statistics, 100
  - get\_attribute\_text, 100
  - get\_attributes, 100
  - get\_edge\_data, 100
  - get\_edge\_peer, 101
  - get\_object\_type, 101
  - get\_type, 101
  - get\_values, 101
  - heads, 102
  - index\_attribute, 102
  - index\_neighbors, 102
  - neighbors, 102, 103
  - new\_array\_attribute, 103
  - new\_attribute, 103, 104
  - new\_edge, 104, 105
  - new\_edge\_type, 105
  - new\_node, 105
  - new\_node\_type, 105
  - new\_restricted\_edge\_type, 106
  - new\_session\_array\_attribute, 106
  - new\_session\_attribute, 106, 107

- remove\_attribute, 107
- remove\_type, 107
- rename\_attribute, 108
- rename\_type, 108
- select, 108–110
- set\_array\_attribute, 110
- set\_array\_attribute\_void, 111
- set\_attribute, 111
- set\_attribute\_default\_value, 112
- set\_attribute\_text, 112
- tails, 112
- tails\_and\_heads, 112
- top\_k, 112–114
- sparksee::GraphExport
  - get\_label, 115
  - set\_label, 115
- sparksee::Int32List
  - \_\_init\_\_, 116
  - \_\_iter\_\_, 116
  - add, 116
  - count, 117
  - iterator, 117
- sparksee::Int32ListIterator
  - \_\_next\_\_, 118
  - has\_next, 118
  - next, 118
- sparksee::KOpt
  - \_\_init\_\_, 121
  - add\_all\_edge\_types, 122
  - add\_edge\_type, 122
  - add\_node\_type, 122
  - set\_current\_tour, 122
  - set\_edge\_weight\_attribute\_type, 123
  - set\_max\_iterations, 123
  - set\_time\_limit, 123
- sparksee::KeyValue
  - to\_string, 118
- sparksee::KeyValues
  - \_\_next\_\_, 119
  - close, 119
  - has\_next, 119
  - is\_closed, 119
  - next, 120
- sparksee::LogLevel
  - CONFIG, 124
  - DEBUG, 124
  - FINE, 124
  - INFO, 124
  - SEVERE, 125
  - WARNING, 125
- sparksee::NodeExport
  - get\_color\_rgb, 127
  - get\_font\_size, 127
  - get\_height, 127
  - get\_label, 127
  - get\_labelcolor\_rgb, 127
  - get\_shape, 128
  - get\_width, 128
- is\_fit, 128
- set\_color\_rgb, 128
- set\_fit, 128
- set\_font\_size, 129
- set\_height, 129
- set\_label, 129
- set\_labelcolor\_rgb, 129
- set\_shape, 129
- set\_width, 129
- sparksee::NodeTypeExporter
  - \_\_init\_\_, 131
  - register, 132
  - run, 132
  - set\_attributes, 132
  - set\_frequency, 132
  - set\_graph, 132
  - set\_header, 132
  - set\_row\_writer, 133
  - set\_type, 133
- sparksee::NodeTypeLoader
  - \_\_init\_\_, 135
  - register, 135
  - run, 135
  - run\_n\_phases, 135
  - run\_two\_phases, 136
  - set\_attribute\_positions, 136
  - set\_attributes, 136
  - set\_frequency, 136
  - set\_graph, 136
  - set\_locale, 137
  - set\_log\_error, 137
  - set\_log\_off, 137
  - set\_row\_reader, 137
  - set\_timestamp\_format, 137
  - set\_type, 138
- sparksee::OIDList
  - \_\_init\_\_, 147, 148
  - \_\_iter\_\_, 148
  - add, 148
  - count, 148
  - iterator, 148
  - set, 148
- sparksee::OIDListIterator
  - \_\_next\_\_, 149
  - has\_next, 149
  - next, 149
- sparksee::Objects
  - \_\_iter\_\_, 139
  - add, 139
  - any, 140
  - close, 140
  - combine\_difference, 140
  - combine\_intersection, 140
  - combine\_union, 141
  - contains, 141
  - copy, 141, 142
  - count, 142
  - difference, 142

- equals, 142
- exists, 142
- intersection, 143
- is\_closed, 143
- iterator, 143
- iterator\_from\_element, 143
- iterator\_from\_index, 144
- remove, 144
- sample, 144
- union, 144
- sparksee::ObjectsIterator
  - \_\_next\_\_, 145
  - close, 145
  - has\_next, 146
  - is\_closed, 146
  - next, 146
- sparksee::PageRank
  - \_\_init\_\_, 151
  - add\_all\_edge\_types, 151
  - add\_edge\_type, 151
  - add\_node\_type, 152
  - run, 152
  - set\_damping, 152
  - set\_default\_weight, 152
  - set\_edge\_weight\_attribute\_type, 153
  - set\_initial\_page\_rank\_value, 153
  - set\_num\_iterations, 153
  - set\_output\_attribute\_type, 153
  - set\_starting\_node, 154
  - set\_tolerance, 154
- sparksee::Platform
  - get\_statistics, 155
- sparksee::PlatformStatistics
  - get\_available\_mem, 156
  - get\_num\_c\_p\_us, 156
  - get\_real\_time, 156
  - get\_system\_time, 156
  - get\_total\_mem, 156
  - get\_user\_time, 156
- sparksee::Query
  - close, 157
  - execute, 157
  - is\_closed, 157
  - set\_dynamic, 158
  - set\_stream, 158
- sparksee::QueryStream
  - fetch, 160
  - prepare, 160
  - start, 160
- sparksee::RandomWalk
  - \_\_init\_\_, 162
  - \_\_iter\_\_, 162
  - \_\_next\_\_, 163
  - add\_all\_edge\_types, 163
  - add\_edge\_type, 163
  - add\_node\_type, 163
  - close, 163
  - exclude\_edges, 163
  - exclude\_nodes, 164
  - get\_current\_depth, 164
  - has\_next, 164
  - is\_closed, 164
  - next, 164
  - reset, 165
  - set\_default\_weight, 165
  - set\_edge\_weight\_attribute\_type, 165
  - set\_in\_out\_parameter, 165
  - set\_maximum\_hops, 166
  - set\_return\_parameter, 166
  - set\_seed, 166
- sparksee::ResultSet
  - get\_column, 167
  - get\_column\_data\_type, 167
  - get\_column\_index, 168
  - get\_column\_name, 168
  - get\_j\_s\_o\_n, 168
  - get\_num\_columns, 169
  - is\_closed, 169
  - next, 169
- sparksee::ResultSetList
  - \_\_init\_\_, 170
  - \_\_iter\_\_, 170
  - count, 170
  - get, 170
  - iterator, 170
- sparksee::ResultSetListIterator
  - \_\_next\_\_, 171
  - has\_next, 171
  - next, 171
- sparksee::RowReader
  - close, 172
  - get\_row, 173
  - read, 173
  - reset, 173
- sparksee::RowWriter
  - close, 174
  - write, 175
- sparksee::ScriptParser
  - generate\_schema\_script, 175
  - parse, 176
  - set\_error\_log, 176
  - set\_output\_log, 176
- sparksee::Session
  - close, 178
  - get\_graph, 178
  - get\_in\_memory\_pool\_capacity, 178
  - is\_closed, 178
  - new\_objects, 178
  - new\_query, 178
  - pre\_commit, 179
- sparksee::ShortestPath
  - add\_all\_edge\_types, 180
  - add\_edge\_type, 180
  - add\_node\_type, 180
  - close, 181
  - exclude\_edges, 181

- exclude\_nodes, 181
- is\_closed, 181
- run, 181
- set\_maximum\_hops, 181
- sparksee::SinglePairShortestPath
  - add\_all\_edge\_types, 183
  - add\_edge\_type, 183
  - add\_node\_type, 184
  - close, 184
  - exclude\_edges, 184
  - exclude\_nodes, 184
  - get\_cost, 184
  - get\_path\_as\_edges, 185
  - get\_path\_as\_nodes, 185
  - is\_closed, 185
  - run, 185
  - set\_maximum\_hops, 185
- sparksee::SinglePairShortestPathBFS
  - \_\_init\_\_, 187
  - add\_all\_edge\_types, 188
  - add\_edge\_type, 188
  - add\_node\_type, 188
  - check\_only\_existence, 188
  - close, 188
  - exclude\_edges, 188
  - exclude\_nodes, 190
  - get\_cost, 190
  - get\_path\_as\_edges, 190
  - get\_path\_as\_nodes, 190
  - is\_closed, 190
  - set\_maximum\_hops, 190
- sparksee::SinglePairShortestPathDijkstra
  - \_\_init\_\_, 193
  - add\_all\_edge\_types, 193
  - add\_edge\_type, 193
  - add\_node\_type, 193
  - add\_weighted\_edge\_type, 193
  - close, 194
  - exclude\_edges, 194
  - exclude\_nodes, 194
  - get\_cost, 194
  - get\_path\_as\_edges, 194
  - get\_path\_as\_nodes, 194
  - is\_closed, 194
  - set\_dynamic\_edge\_cost\_callback, 195
  - set\_maximum\_hops, 195
  - set\_unweighted\_edge\_cost, 195
- sparksee::SinglePairShortestPathDijkstraDynamicCost
  - calculate\_edge\_cost, 196
- sparksee::Sparksee
  - \_\_init\_\_, 198
  - add\_checksums, 198
  - close, 198
  - create, 198, 199
  - decrypt, 199
  - encrypt, 200
  - get\_in\_memory\_pool\_capacity, 200
  - open, 200
  - remove\_checksums, 201
  - resize\_in\_memory\_pool, 201
  - restore, 202
  - restore\_encrypted\_backup, 203
  - set\_unrecoverable\_error\_callback, 204
  - verify\_checksums, 204
- sparksee::SparkseeConfig
  - \_\_init\_\_, 209, 210
  - download\_license, 210
  - get\_a\_e\_s\_i\_v, 210
  - get\_a\_e\_s\_key, 210
  - get\_cache\_max\_size, 211
  - get\_cache\_statistics\_enabled, 211
  - get\_cache\_statistics\_file, 211
  - get\_cache\_statistics\_snapshot\_time, 211
  - get\_call\_stack\_dump, 211
  - get\_checksum\_enabled, 211
  - get\_client\_id, 212
  - get\_download\_status, 212
  - get\_encryption\_enabled, 212
  - get\_extent\_pages, 212
  - get\_extent\_size, 212
  - get\_high\_availability\_coordinators, 212
  - get\_high\_availability\_enabled, 213
  - get\_high\_availability\_i\_p, 213
  - get\_high\_availability\_master\_history, 213
  - get\_high\_availability\_synchronization, 213
  - get\_in\_mem\_alloc\_size, 213
  - get\_license, 213
  - get\_license\_id, 214
  - get\_license\_pre\_download\_days, 214
  - get\_log\_file, 214
  - get\_log\_level, 214
  - get\_pool\_frame\_size, 214
  - get\_pool\_partitions, 214
  - get\_pool\_persistent\_max\_size, 215
  - get\_pool\_persistent\_min\_size, 215
  - get\_pool\_temporary\_max\_size, 215
  - get\_pool\_temporary\_min\_size, 215
  - get\_recovery\_cache\_max\_size, 215
  - get\_recovery\_checkpoint\_time, 215
  - get\_recovery\_enabled, 216
  - get\_recovery\_log\_file, 216
  - get\_rollback\_enabled, 216
  - get\_sparksee\_config\_file, 216
  - get\_tmp\_enabled, 216
  - get\_tmp\_folder, 216
  - save, 217
  - save\_all, 217
  - set\_a\_e\_s\_encryption\_enabled, 217
  - set\_cache\_max\_size, 218
  - set\_cache\_statistics\_enabled, 218
  - set\_cache\_statistics\_file, 218
  - set\_cache\_statistics\_snapshot\_time, 218
  - set\_call\_stack\_dump, 219
  - set\_checksum\_enabled, 219
  - set\_client\_id, 219
  - set\_extent\_pages, 219

- set\_extent\_size, 219
- set\_high\_availability\_coordinators, 220
- set\_high\_availability\_enabled, 220
- set\_high\_availability\_i\_p, 220
- set\_high\_availability\_master\_history, 220
- set\_high\_availability\_synchronization, 220
- set\_in\_mem\_alloc\_size, 220
- set\_license, 221
- set\_license\_id, 221
- set\_license\_pre\_download\_days, 221
- set\_log\_file, 221
- set\_log\_level, 221
- set\_pool\_frame\_size, 222
- set\_pool\_partitions, 222
- set\_pool\_persistent\_max\_size, 222
- set\_pool\_persistent\_min\_size, 222
- set\_pool\_temporary\_max\_size, 222
- set\_pool\_temporary\_min\_size, 222
- set\_recovery\_cache\_max\_size, 223
- set\_recovery\_checkpoint\_time, 223
- set\_recovery\_enabled, 223
- set\_recovery\_log\_file, 223
- set\_rollback\_enabled, 223
- set\_sparksee\_config\_file, 224
- set\_tmp\_enabled, 224
- set\_tmp\_folder, 224
- sparksee::SparkseeProperties
  - get, 225
  - get\_boolean, 225
  - get\_integer, 225
  - get\_time\_unit, 226
  - load, 226
  - set\_h\_i, 226
- sparksee::StringList
  - \_\_init\_\_, 227
  - \_\_iter\_\_, 227
  - add, 227
  - count, 228
  - iterator, 228
- sparksee::StringListIterator
  - \_\_next\_\_, 229
  - has\_next, 229
  - next, 229
- sparksee::StrongConnectivity
  - add\_all\_edge\_types, 231
  - add\_edge\_type, 231
  - add\_node\_type, 231
  - close, 231
  - exclude\_edges, 231
  - exclude\_nodes, 232
  - get\_connected\_components, 232
  - is\_closed, 232
  - run, 232
  - set\_materialized\_attribute, 232
- sparksee::StrongConnectivityGabow
  - \_\_init\_\_, 234
  - add\_all\_edge\_types, 235
  - add\_edge\_type, 235
  - add\_node\_type, 235
  - close, 235
  - exclude\_edges, 235
  - exclude\_nodes, 236
  - get\_connected\_components, 236
  - is\_closed, 236
  - set\_materialized\_attribute, 236
- sparksee::TextStream
  - \_\_init\_\_, 237
  - close, 238
  - is\_null, 238
  - read, 238
  - write, 238
- sparksee::Traversal
  - \_\_iter\_\_, 240
  - \_\_next\_\_, 240
  - add\_all\_edge\_types, 240
  - add\_edge\_type, 240
  - add\_node\_type, 241
  - close, 241
  - exclude\_edges, 241
  - exclude\_nodes, 241
  - get\_current\_depth, 241
  - has\_next, 241
  - is\_closed, 241
  - next, 242
  - set\_maximum\_hops, 242
- sparksee::TraversalBFS
  - \_\_init\_\_, 244
  - \_\_iter\_\_, 244
  - \_\_next\_\_, 244
  - add\_all\_edge\_types, 244
  - add\_edge\_type, 245
  - add\_node\_type, 245
  - close, 245
  - exclude\_edges, 245
  - exclude\_nodes, 245
  - get\_current\_depth, 245
  - has\_next, 246
  - is\_closed, 246
  - next, 246
  - set\_maximum\_hops, 246
- sparksee::TraversalDFS
  - \_\_init\_\_, 248
  - \_\_iter\_\_, 248
  - \_\_next\_\_, 248
  - add\_all\_edge\_types, 249
  - add\_edge\_type, 249
  - add\_node\_type, 249
  - close, 249
  - exclude\_edges, 249
  - exclude\_nodes, 250
  - get\_current\_depth, 250
  - has\_next, 250
  - is\_closed, 250
  - next, 250
  - set\_maximum\_hops, 250
- sparksee::Type

- get\_are\_neighbors\_indexed, [252](#)
- get\_id, [252](#)
- get\_is\_directed, [252](#)
- get\_is\_restricted, [252](#)
- get\_name, [252](#)
- get\_num\_objects, [252](#)
- get\_object\_type, [252](#)
- get\_restricted\_from, [253](#)
- get\_restricted\_to, [253](#)
- sparksee::TypeExporter
  - register, [254](#)
  - run, [254](#)
  - set\_attributes, [255](#)
  - set\_frequency, [255](#)
  - set\_graph, [255](#)
  - set\_header, [255](#)
  - set\_row\_writer, [255](#)
  - set\_type, [255](#)
- sparksee::TypeExporterEvent
  - get\_count, [256](#)
  - get\_total, [256](#)
  - get\_type\_id, [257](#)
  - is\_last, [257](#)
- sparksee::TypeExporterListener
  - notify\_event, [257](#)
- sparksee::TypeList
  - \_\_init\_\_, [258](#)
  - \_\_iter\_\_, [258](#)
  - add, [258](#)
  - count, [259](#)
  - iterator, [259](#)
- sparksee::TypeListIterator
  - \_\_next\_\_, [260](#)
  - has\_next, [260](#)
  - next, [260](#)
- sparksee::TypeLoader
  - register, [261](#)
  - run, [262](#)
  - run\_n\_phases, [262](#)
  - run\_two\_phases, [262](#)
  - set\_attribute\_positions, [262](#)
  - set\_attributes, [263](#)
  - set\_frequency, [263](#)
  - set\_graph, [263](#)
  - set\_locale, [263](#)
  - set\_log\_error, [263](#)
  - set\_log\_off, [264](#)
  - set\_row\_reader, [264](#)
  - set\_timestamp\_format, [264](#)
  - set\_type, [264](#)
- sparksee::TypeLoaderEvent
  - get\_count, [265](#)
  - get\_partition, [265](#)
  - get\_phase, [265](#)
  - get\_total\_partition\_steps, [265](#)
  - get\_total\_partitions, [266](#)
  - get\_total\_phases, [266](#)
  - get\_type\_id, [266](#)
  - is\_last, [266](#)
- sparksee::TypeLoaderListener
  - notify\_event, [266](#)
- sparksee::Value
  - \_\_init\_\_, [269](#)
  - compare, [269](#)
  - equals, [269](#)
  - get\_boolean, [270](#)
  - get\_data\_type, [270](#)
  - get\_double, [270](#)
  - get\_integer, [270](#)
  - get\_long, [270](#)
  - get\_oid, [270](#)
  - get\_string, [271](#)
  - get\_timestamp, [271](#)
  - is\_null, [271](#)
  - operator, [271](#)
  - set\_boolean, [271](#)
  - set\_boolean\_void, [272](#)
  - set\_double, [272](#)
  - set\_double\_void, [272](#)
  - set\_integer, [272](#)
  - set\_integer\_void, [273](#)
  - set\_long, [273](#)
  - set\_long\_void, [273](#)
  - set\_null, [273](#)
  - set\_oid, [273](#)
  - set\_oid\_void, [274](#)
  - set\_string, [274](#)
  - set\_string\_void, [274](#)
  - set\_timestamp, [274](#)
  - set\_timestamp\_void, [275](#)
  - set\_void, [275](#)
  - to\_string, [276](#)
- sparksee::ValueArray
  - get, [277](#)
  - get\_boolean, [278](#)
  - get\_boolean\_range, [278](#)
  - get\_double, [278](#)
  - get\_double\_range, [278](#)
  - get\_integer, [279](#)
  - get\_integer\_range, [279](#)
  - get\_long, [279](#)
  - get\_long\_range, [280](#)
  - get\_oid, [280](#)
  - get\_oid\_range, [280](#)
  - get\_timestamp, [280](#)
  - get\_timestamp\_range, [281](#)
  - make\_null, [281](#)
  - set, [281](#)
  - set\_boolean, [282](#)
  - set\_boolean\_range, [282](#)
  - set\_double, [282](#)
  - set\_double\_range, [283](#)
  - set\_integer, [283](#)
  - set\_integer\_range, [283](#)
  - set\_long, [283](#)
  - set\_long\_range, [284](#)



- set\_oid, 284
- set\_oid\_range, 284
- set\_timestamp, 284
- set\_timestamp\_range, 285
- sparksee::ValueList
  - \_\_init\_\_, 286
  - \_\_iter\_\_, 286
  - add, 286
  - count, 286
  - get, 286
  - iterator, 286
- sparksee::ValueListIterator
  - \_\_next\_\_, 287
  - has\_next, 287
  - next, 287
- sparksee::Values
  - \_\_iter\_\_, 288
  - close, 288
  - count, 289
  - is\_closed, 289
  - iterator, 289
- sparksee::ValuesIterator
  - \_\_next\_\_, 290
  - close, 290
  - has\_next, 290
  - is\_closed, 290
  - next, 290
- sparksee::WeakConnectivity
  - add\_all\_edge\_types, 292
  - add\_edge\_type, 292
  - add\_node\_type, 293
  - close, 293
  - exclude\_edges, 293
  - exclude\_nodes, 293
  - get\_connected\_components, 293
  - is\_closed, 293
  - run, 294
  - set\_materialized\_attribute, 294
- sparksee::WeakConnectivityDFS
  - \_\_init\_\_, 296
  - add\_all\_edge\_types, 297
  - add\_edge\_type, 297
  - add\_node\_type, 297
  - close, 297
  - exclude\_edges, 297
  - exclude\_nodes, 297
  - get\_connected\_components, 298
  - is\_closed, 298
  - set\_materialized\_attribute, 298
- start
  - sparksee::QueryStream, 160
- TEXT
  - sparksee::DataType, 58
- TIMESTAMP
  - sparksee::DataType, 58
- tails
  - sparksee::Graph, 112
- tails\_and\_heads
  - sparksee::Graph, 112
- to\_string
  - sparksee::KeyValue, 118
  - sparksee::Value, 276
- top\_k
  - sparksee::Graph, 112–114
- UNIQUE
  - sparksee::AttributeKind, 17
- union
  - sparksee::Objects, 144
- verify\_checksums
  - sparksee::Sparksee, 204
- WARNING
  - sparksee::LogLevel, 125
- write
  - sparksee::CSVWriter, 53
  - sparksee::RowWriter, 175
  - sparksee::TextStream, 238
- YGRAPHML
  - sparksee::ExportType, 88