<table>
<thead>
<tr>
<th>Section</th>
<th>Class Reference</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.23</td>
<td>sparksee.ExportManager Class</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>Detailed Description</td>
<td>59</td>
</tr>
<tr>
<td>3.23.2</td>
<td>Member Function Documentation</td>
<td>59</td>
</tr>
<tr>
<td>3.24</td>
<td>sparksee.ExportType Class</td>
<td>61</td>
</tr>
<tr>
<td>3.24.1</td>
<td>Detailed Description</td>
<td>62</td>
</tr>
<tr>
<td>3.24.2</td>
<td>Member Data Documentation</td>
<td>62</td>
</tr>
<tr>
<td>3.25</td>
<td>sparksee.Graph Class</td>
<td>62</td>
</tr>
<tr>
<td>3.25.1</td>
<td>Detailed Description</td>
<td>66</td>
</tr>
<tr>
<td>3.25.2</td>
<td>Member Function Documentation</td>
<td>66</td>
</tr>
<tr>
<td>3.26</td>
<td>sparksee.GraphExport Class</td>
<td>82</td>
</tr>
<tr>
<td>3.26.1</td>
<td>Detailed Description</td>
<td>82</td>
</tr>
<tr>
<td>3.26.2</td>
<td>Member Function Documentation</td>
<td>82</td>
</tr>
<tr>
<td>3.27</td>
<td>sparksee.Int32List Class</td>
<td>83</td>
</tr>
<tr>
<td>3.27.1</td>
<td>Detailed Description</td>
<td>83</td>
</tr>
<tr>
<td>3.27.2</td>
<td>Member Function Documentation</td>
<td>83</td>
</tr>
<tr>
<td>3.28</td>
<td>sparksee.Int32ListIterator Class</td>
<td>84</td>
</tr>
<tr>
<td>3.28.1</td>
<td>Detailed Description</td>
<td>84</td>
</tr>
<tr>
<td>3.28.2</td>
<td>Member Function Documentation</td>
<td>85</td>
</tr>
<tr>
<td>3.29</td>
<td>sparksee.LogLevel Class</td>
<td>85</td>
</tr>
<tr>
<td>3.29.1</td>
<td>Detailed Description</td>
<td>86</td>
</tr>
<tr>
<td>3.29.2</td>
<td>Member Data Documentation</td>
<td>86</td>
</tr>
<tr>
<td>3.30</td>
<td>sparksee.NodeExport Class</td>
<td>86</td>
</tr>
<tr>
<td>3.30.1</td>
<td>Detailed Description</td>
<td>88</td>
</tr>
<tr>
<td>3.30.2</td>
<td>Member Function Documentation</td>
<td>88</td>
</tr>
<tr>
<td>3.31</td>
<td>sparksee.NodeShape Class</td>
<td>91</td>
</tr>
<tr>
<td>3.31.1</td>
<td>Detailed Description</td>
<td>91</td>
</tr>
<tr>
<td>3.32</td>
<td>sparksee.NodeTypeExporter Class</td>
<td>91</td>
</tr>
<tr>
<td>3.32.1</td>
<td>Detailed Description</td>
<td>92</td>
</tr>
<tr>
<td>3.32.2</td>
<td>Member Function Documentation</td>
<td>92</td>
</tr>
<tr>
<td>3.33</td>
<td>sparksee.NodeTypeLoader Class</td>
<td>94</td>
</tr>
<tr>
<td>3.33.1</td>
<td>Detailed Description</td>
<td>96</td>
</tr>
<tr>
<td>3.33.2</td>
<td>Member Function Documentation</td>
<td>96</td>
</tr>
<tr>
<td>3.34</td>
<td>sparksee.Objects Class</td>
<td>99</td>
</tr>
<tr>
<td>3.34.1</td>
<td>Detailed Description</td>
<td>101</td>
</tr>
<tr>
<td>3.34.2</td>
<td>Member Function Documentation</td>
<td>101</td>
</tr>
<tr>
<td>3.35</td>
<td>sparksee.ObjectsIterator Class</td>
<td>106</td>
</tr>
<tr>
<td>Section</td>
<td>Class Reference</td>
<td>Detailed Description</td>
</tr>
<tr>
<td>---------</td>
<td>------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>3.35.1</td>
<td>sparksee.ObjectType</td>
<td>.</td>
</tr>
<tr>
<td>3.35.2</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.36</td>
<td>sparksee.OIDList</td>
<td>.</td>
</tr>
<tr>
<td>3.36.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.37</td>
<td>sparksee.OIDListIterator</td>
<td>.</td>
</tr>
<tr>
<td>3.37.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.38</td>
<td>sparksee.Order</td>
<td>.</td>
</tr>
<tr>
<td>3.38.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.39.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.40.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.41</td>
<td>sparksee.RowReader</td>
<td>.</td>
</tr>
<tr>
<td>3.41.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.42</td>
<td>sparksee.RowWriter</td>
<td>.</td>
</tr>
<tr>
<td>3.42.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.43</td>
<td>sparksee.ScriptParser</td>
<td>.</td>
</tr>
<tr>
<td>3.43.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.44</td>
<td>sparksee.Session</td>
<td>.</td>
</tr>
<tr>
<td>3.44.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.45</td>
<td>sparksee.ShortestPath</td>
<td>.</td>
</tr>
<tr>
<td>3.45.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.46</td>
<td>sparksee.SinglePairShortestPath</td>
<td>.</td>
</tr>
<tr>
<td>3.46.1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3.47</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>
3.60 sparksee.TraversalDFS Class Reference .................................................. 172
3.60.1 Detailed Description ................................................................. 174
3.60.2 Member Function Documentation ................................................. 174
3.61 sparksee.Type Class Reference ......................................................... 176
3.61.1 Detailed Description ................................................................. 177
3.61.2 Member Function Documentation ................................................. 177
3.62 sparksee.TypeExporter Class Reference ............................................. 179
3.62.1 Detailed Description ................................................................. 180
3.62.2 Member Function Documentation ................................................. 180
3.63 sparksee.TypeExporterEvent Class Reference ..................................... 181
3.63.1 Detailed Description ................................................................. 182
3.63.2 Member Function Documentation ................................................. 182
3.64 sparksee.TypeExporterListener Class Reference .................................. 183
3.64.1 Detailed Description ................................................................. 183
3.64.2 Member Function Documentation ................................................. 183
3.65 sparksee.TypeList Class Reference ..................................................... 183
3.65.1 Detailed Description ................................................................. 184
3.65.2 Member Function Documentation ................................................. 184
3.66 sparksee.TypeListIterator Class Reference ....................................... 185
3.66.1 Detailed Description ................................................................. 185
3.66.2 Member Function Documentation ................................................. 185
3.67 sparksee.TypeLoader Class Reference ............................................... 186
3.67.1 Detailed Description ................................................................. 187
3.67.2 Member Function Documentation ................................................. 187
3.68 sparksee.TypeLoaderEvent Class Reference ...................................... 190
3.68.1 Detailed Description ................................................................. 191
3.68.2 Member Function Documentation ................................................. 191
3.69 sparksee.TypeLoaderListener Class Reference .................................... 192
3.69.1 Detailed Description ................................................................. 192
3.69.2 Member Function Documentation ................................................. 193
3.70 sparksee.Value Class Reference ....................................................... 193
3.70.1 Detailed Description ................................................................. 195
3.70.2 Member Function Documentation ................................................. 195
3.71 sparksee.Values Class Reference ...................................................... 201
3.71.1 Detailed Description ................................................................. 202
3.71.2 Member Function Documentation ................................................. 202
1 Class Index

1.1 Class Hierarchy

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

sparksee.Attribute 6
sparksee.AttributeKind 8
sparksee.AttributeList 9
sparksee.AttributeListIterator 10
sparksee.AttributeStatistics 11
sparksee.BooleanList 15
sparksee.BooleanListIterator 16
sparksee.Condition 17
sparksee.ConnectedComponents 20
sparksee.Connectivity 22
sparksee.StrongConnectivity 155
    sparksee.StrongConnectivityGabow 158
sparksee.WeakConnectivity 204
    sparksee.WeakConnectivityDFS 208
sparksee.Context 26
sparksee.Database 36
sparksee.DatabaseStatistics 38
sparksee.DataType 39
1.1 Class Hierarchy

```
sparksee.EdgeData 43
sparksee.EdgeExport 44
sparksee.EdgesDirection 48
sparksee.ExportManager 58
  sparksee.DefaultExport 41
sparksee.ExportType 61
sparksee.Graph 62
sparksee.GraphExport 82
sparksee.Int32List 83
sparksee.Int32ListIterator 84
sparksee.LogLevel 85
sparksee.NodeExport 86
sparksee.NodeShape 91
sparksee.Objects 99
sparksee.ObjectsIterator 106
sparksee.ObjectType 107
sparksee.OIDList 108
sparksee.OIDListIterator 109
sparksee.Order 110
sparksee.Platform 111
sparksee.PlatformStatistics 111
sparksee.RowReader 113
  sparksee.CSVReader 29
sparksee.RowWriter 115
  sparksee.CSVWriter 33
sparksee.ScriptParser 116
sparksee.Session 118
sparksee.ShortestPath 120
  sparksee.SinglePairShortestPath 123
```
<table>
<thead>
<tr>
<th>Class Hierarchy</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>sparksee.SinglePairShortestPathBFS</td>
<td>127</td>
</tr>
<tr>
<td>sparksee.SinglePairShortestPathDijkstra</td>
<td>132</td>
</tr>
<tr>
<td>sparksee.Sparksee</td>
<td>136</td>
</tr>
<tr>
<td>sparksee.SparkseeConfig</td>
<td>138</td>
</tr>
<tr>
<td>sparksee.SparkseeProperties</td>
<td>150</td>
</tr>
<tr>
<td>sparksee.StringList</td>
<td>152</td>
</tr>
<tr>
<td>sparksee.StringListIterator</td>
<td>153</td>
</tr>
<tr>
<td>sparksee.TextStream</td>
<td>162</td>
</tr>
<tr>
<td>sparksee.Traversal</td>
<td>164</td>
</tr>
<tr>
<td>sparksee.TraversalBFS</td>
<td>168</td>
</tr>
<tr>
<td>sparksee.TraversalDFS</td>
<td>172</td>
</tr>
<tr>
<td>sparksee.Type</td>
<td>176</td>
</tr>
<tr>
<td>sparksee.TypeExporter</td>
<td>179</td>
</tr>
<tr>
<td>sparksee.EdgeTypeExporter</td>
<td>48</td>
</tr>
<tr>
<td>sparksee.NodeTypeExporter</td>
<td>91</td>
</tr>
<tr>
<td>sparksee.TypeExporterEvent</td>
<td>181</td>
</tr>
<tr>
<td>sparksee.TypeExporterListener</td>
<td>183</td>
</tr>
<tr>
<td>sparksee.TypeList</td>
<td>183</td>
</tr>
<tr>
<td>sparksee.TypeListIterator</td>
<td>185</td>
</tr>
<tr>
<td>sparksee.TypeLoader</td>
<td>186</td>
</tr>
<tr>
<td>sparksee.EdgeTypeLoader</td>
<td>52</td>
</tr>
<tr>
<td>sparksee.NodeTypeLoader</td>
<td>94</td>
</tr>
<tr>
<td>sparksee.TypeLoaderEvent</td>
<td>190</td>
</tr>
<tr>
<td>sparksee.TypeLoaderListener</td>
<td>192</td>
</tr>
<tr>
<td>sparksee.Value</td>
<td>193</td>
</tr>
<tr>
<td>sparksee.Values</td>
<td>201</td>
</tr>
<tr>
<td>sparksee.ValuesIterator</td>
<td>203</td>
</tr>
</tbody>
</table>
2 Class Index

2.1 Class List

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

- **sparksee.Attribute** *(Attribute data class)*  
- **sparksee.AttributeKind** *(Attribute kind enumeration)*  
- **sparksee.AttributeList** *(Sparksee attribute identifier list)*  
- **sparksee.AttributeListIterator** *(AttributeList iterator class)*  
- **sparksee.AttributeStatistics** *(Attribute statistics class)*  
- **sparksee.BooleanList** *(Boolean list)*  
- **sparksee.BooleanListIterator** *(BooleanList iterator class)*  
- **sparksee.Condition** *(Condition operators enumeration)*  
- **sparksee.ConnectedComponents** *(ConnectedComponents class)*  
- **sparksee.Connectivity** *(Connectivity class)*  
- **sparksee.Context** *(Context class)*  
- **sparksee.CSVReader** *(CSVReader interface)*  
- **sparksee.CSVWriter** *(CSVWriter interface)*  
- **sparksee.Database** *(Database class)*  
- **sparksee.DatabaseStatistics** *(Database statistics)*  
- **sparksee.DataType** *(Data type (domain) enumeration)*  
- **sparksee.DefaultExport** *(Default implementation for ExportManager class)*  
- **sparksee.EdgeData** *(Edge data class)*  
- **sparksee.EdgeExport** *(Stores edge exporting values)*  
- **sparksee.EdgesDirection** *(Edges direction enumeration)*  
- **sparksee.EdgeTypeExporter** *(EdgeTypeExporter class)*  
- **sparksee.EdgeTypeLoader** *(EdgeTypeLoader class)*  
- **sparksee.ExportManager** *(Defines how to export a graph to an external format)*  
- **sparksee.ExportType** *(Export type)*  
- **sparksee.Graph** *(Graph class)*  
- **sparksee.GraphExport** *(Stores the graph exporting values)*
sparksee.Int32List (Sparksee 32-bit signed integer list ) 83
sparksee.Int32ListIterator (Int32List iterator class ) 84
sparksee.LogLevel (Log level enumeration ) 85
sparksee.NodeExport (Stores the node exporting values ) 86
sparksee.NodeShape (Node shape ) 91
sparksee.NodeTypeExporter (NodeTypeExporter class ) 91
sparksee.NodeTypeLoader (NodeTypeLoader class ) 94
sparksee.Objects (Object identifier set class ) 99
sparksee.ObjectsIterator (ObjectsIterator class ) 106
sparksee.ObjectType (Object type enumeration ) 107
sparksee.OIDList (Sparksee object identifier list ) 108
sparksee.OIDListIterator (OIDList iterator class ) 109
sparksee.Order (Order enumeration ) 110
sparksee.Platform (Platform class ) 111
sparksee.PlatformStatistics (Platform data and statistics ) 111
sparksee.RowReader (RowReader interface ) 113
sparksee.RowWriter (RowWriter interface ) 115
sparksee.ScriptParser (ScriptParser ) 116
sparksee.Session (Session class ) 118
sparksee.ShortestPath (ShortestPath class ) 120
sparksee.SinglePairShortestPath (SinglePairShortestPath class ) 123
sparksee.SinglePairShortestPathBFS (SinglePairShortestPathBFS class ) 127
sparksee.SinglePairShortestPathDijkstra (SinglePairShortestPathDijkstra class ) 132
sparksee.Sparksee (Sparksee class ) 136
sparksee.SparkseeConfig (Sparksee configuration class ) 138
sparksee.SparkseeProperties (Sparksee properties file ) 150
sparksee.StringList (String list ) 152
sparksee.StringListIterator (StringList iterator class ) 153
sparksee.StrongConnectivity (StrongConnectivity class ) 155
3 Class Documentation

3.1 sparksee.Attribute Class Reference

Attribute data class.

Public Member Functions

* def get_kind
  
  Gets the attribute kind.
3.1 sparksee.Attribute Class Reference

- def get_count
  
  Gets the number of non-NULL values.

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

- def get_size
  
  Gets the number of different values.

- def get_type_id
  
  Gets the Sparksee type identifier.

- def get_data_type
  
  Gets the data type.

- def get_id
  
  Gets the Sparksee attribute identifier.

- def get_name
  
  Gets the unique attribute name.

Static Public Attributes

- int INVALID_ATTRIBUTE = 0
  
  Invalid attribute identifier constant.

3.1.1 Detailed Description

Attribute data class.
It contains information about an attribute.

Author:

Sparsity Technologies [http://www.sparsity-technologies.com](http://www.sparsity-technologies.com)

3.1.2 Member Function Documentation

3.1.2.1 def sparksee.Attribute.get_kind ( self )

Gets the attribute kind.

Returns:

The AttributeKind.
3.1.2.2  def sparksee.Attribute.get_count (self)
Gets the number of non-NULL values.

Returns:
   The number of non-NULL values.

3.1.2.3  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.

3.1.2.4  def sparksee.Attribute.get_size (self)
Gets the number of different values.

Returns:
   The number of different values.

3.1.2.5  def sparksee.Attribute.get_type_id (self)
Gets the Sparksee type identifier.

Returns:
   The Sparksee type identifier.

3.1.2.6  def sparksee.Attribute.get_data_type (self)
Gets the data type.

Returns:
   The DataType.

3.1.2.7  def sparksee.Attribute.get_id (self)
Gets the Sparksee attribute identifier.

Returns:
   The Sparksee attribute identifier.

3.1.2.8  def sparksee.Attribute.get_name (self)
Gets the unique attribute name.

Returns:
   The unique attribute name.
3.2 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).

3.2.1 Detailed Description

Attribute kind enumeration.
It determines the indexing-capabilities of an attribute.

Author:
Sparsity Technologies [http://www.sparsity-technologies.com](http://www.sparsity-technologies.com)

3.2.2 Member Data Documentation

3.2.2.1 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.

3.3 sparksee.AttributeList Class Reference

Sparksee attribute identifier list.

Public Member Functions

- `def clear`
  Clears the list.
- `def __iter__`
  Gets a new `TypeListIterator`.
- `def __init__`
  Constructor.
- `def iterator`
  Gets a new `AttributeListIterator`.
• def count
   
   Number of elements in the list.

• def add
   
   Adds a Sparksee attribute identifier at the end of the list.

### 3.3.1 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](http://www.sparsity-technologies.com)

### 3.3.2 Member Function Documentation

#### 3.3.2.1 def sparksee.AttributeList.__iter__ (self)

Gets a new TypeListIterator.

**Returns:**
TypeListIterator instance

#### 3.3.2.2 def sparksee.AttributeList.__init__ (self)

Constructor.
This creates an empty list.

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

Gets a new AttributeListIterator.

**Returns:**
AttributeListIterator instance.

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

Number of elements in the list.

**Returns:**
Number of elements in the list.
3.3.2.5 def sparksee.AttributeList.add (self, attr)
Adds a Sparksee attribute identifier at the end of the list.

Parameters:

3.4 sparksee.AttributeListIterator Class Reference

AttributeList iterator class.

Public Member Functions

- def has_next
  Gets if there are more elements.

- def next
  Moves to the next element.

- def __next__
  Used in next().

3.4.1 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

3.4.2 Member Function Documentation

3.4.2.1 def sparksee.AttributeListIterator.has_next (self)
Gets if there are more elements.

Returns:

TRUE if there are more elements, FALSE otherwise.

3.4.2.2 def sparksee.AttributeListIterator.next (self)
Moves to the next element.

Returns:

The next element.
3.4.2.3   def sparksee.AttributeListIterator.__next__(self)

Used in next().

Returns:

The next element

3.5   sparksee.AttributeStatistics Class Reference

Attribute statistics class.

Public Member Functions

- def get_min
  
  Gets the minimum existing value (BASIC statistics).

- def get_min_length_string
  
  Gets the minimum length.

- def get_variance
  
  Gets the variance.

- def get_mode
  
  Gets the mode.

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

- def get_distinct
  
  Gets the number of distinct values (BASIC statistics).

- def get_mean
  
  Gets the mean or average.

- def get_max
  
  Gets the maximum existing value (BASIC statistics).

- def get_median
  
  Gets the median.

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

- def get_max_length_string
  
  Gets the maximum length.

- def get_avg_length_string
  
  Gets the average length.
3.5 sparksee.AttributeStatistics Class Reference

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

3.5.1 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 getAttributeStatistics or check out the SPARKSEE User Manual for more details on this.

Author:

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

3.5.2 Member Function Documentation

3.5.2.1 def sparksee.AttributeStatistics.get_min (self)

Gets the minimum existing value (BASIC statistics).

Returns:

The minimum existing value.

3.5.2.2 def sparksee.AttributeStatistics.get_min_length_string (self)

Gets the minimum length.

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

Returns:

The minimum length.

3.5.2.3 def sparksee.AttributeStatistics.get_variance (self)

Gets the variance.

It is computed just for numerical attributes.

Returns:

The variance.

3.5.2.4 def sparksee.AttributeStatistics.get_mode (self)

Gets the mode.

Mode: Most frequent Value.

Returns:

The mode.
3.5.2.5  def sparksee.AttributeStatistics.get_null (self)
Gets the number of objects NULL a Value (BASIC statistics).

Returns:
The number of objects NULL a Value.

3.5.2.6  def sparksee.AttributeStatistics.get_distinct (self)
Gets the number of distinct values (BASIC statistics).

Returns:
The number of distinct values.

3.5.2.7  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.

3.5.2.8  def sparksee.AttributeStatistics.get_max (self)
Gets the maximum existing value (BASIC statistics).

Returns:
The maximum existing value.

3.5.2.9  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.

3.5.2.10  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.
3.5.2.11  def sparksee.AttributeStatistics.get_max_length_string ( self)
Gets the maximum length.
If the attribute is not an string attribute, it just returns 0.

**Returns:**
- The maximum length.

3.5.2.12  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.

3.5.2.13  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.

3.6  sparksee:BooleanList Class Reference

Boolean list.

**Public Member Functions**

- def clear
  
  Clears the list.

- def __iter__
  
  Gets a new TypeListIterator.

- def add
  
  Adds a Boolean at the end of the list.

- def iterator
  
  Gets a new BooleanListIterator.

- def count
  
  Number of elements in the list.

- def __init__
  
  Constructor.
3.6.1 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

3.6.2 Member Function Documentation

3.6.2.1 def sparksee.BooleanList.__iter__ (self)
Gets a new TypeListIterator.

Returns:
TypeListIterator instance

3.6.2.2 def sparksee.BooleanList.add (self, b)
Adds a Boolean at the end of the list.

Parameters:
   b [in] Boolean.

3.6.2.3 def sparksee.BooleanList.iterator (self)
Gets a new BooleanListIterator.

Returns:
BooleanListIterator instance.

3.6.2.4 def sparksee.BooleanList.count (self)
Number of elements in the list.

Returns:
Number of elements in the list.

3.6.2.5 def sparksee.BooleanList.__init__ (self)
Constructor.
This creates an empty list.

3.7 sparksee.BooleanListIterator Class Reference

BooleanList iterator class.
Public Member Functions

- def has_next
  Gets if there are more elements.

- def next
  Moves to the next element.

- def __next__
  Used in next().

3.7.1 Detailed Description

BooleanList iterator class.
Iterator to traverse all the strings into a BooleanList instance.

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

3.7.2 Member Function Documentation

3.7.2.1 def sparksee.BooleanListIterator.has_next (self)
Gets if there are more elements.

Returns:
TRUE if there are more elements, FALSE otherwise.

3.7.2.2 def sparksee.BooleanListIterator.next (self)
Moves to the next element.

Returns:
The next element.

3.7.2.3 def sparksee.BooleanListIterator.__next__ (self)
Used in next().

Returns:
The next element

3.8 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.

### 3.8.1 Detailed Description

**Condition** operators enumeration.

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

**Author:**

Sparsity Technologies [http://www.sparsity-technologies.com](http://www.sparsity-technologies.com)

### 3.8.2 Member Data Documentation

#### 3.8.2.1 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.
3.8.2.2 int sparksee.Condition.GREATER_EQUAL = 1 [static]
Greater or equal condition (\(\geq\)).
Null values cannot be used together with this condition.

3.8.2.3 int sparksee.Condition.GREATER_THAN = 2 [static]
Greater than condition (\(>\)).
Null values cannot be used together with this condition.

3.8.2.4 int sparksee.Condition.LESS_EQUAL = 3 [static]
Less or equal condition (\(\leq\)).
Null values cannot be used together with this condition.

3.8.2.5 int sparksee.Condition.LESS_THAN = 4 [static]
Less than condition (\(<\)).
Null values cannot be used together with this condition.

3.8.2.6 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.

3.8.2.7 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

3.8.2.8 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
3.8.2.9  int sparksee.Condition.BETWEEN = 8  [static]
In range operator condition ([x,y]).
Null values cannot be used together with this condition.

3.8.2.10 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 sensi-
tive.
See the `Regular expressions` section in the `SPARKSEE User Manual` for details.

3.9 sparksee.ConnectedComponents Class Reference

ConnectedComponents class.

Public Member Functions

- def __init__
  Creates a new instance of ConnectedComponents.

- def get_size
  Returns the number of nodes contained in the given connected component.

- def get_count
  Returns the number of connected components found in the graph.

- def is_closed
  Gets if ConnectedComponents has been closed or not.

- def get_connected_component
  Returns the connected component where the given node belongs to.

- def get_nodes
  Returns the collection of nodes contained in the given connected component.

- def close
  Closes the ConnectedComponents instance.

3.9.1 Detailed Description

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

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

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

Author:

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

3.9.2 Member Function Documentation

3.9.2.1 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:

s [in] Session to get the graph Graph on which the information will be retrieved just by getting the values contained in the given common attribute type for all the nodes in the graph and processing them.

materializedattribute [in] The common attribute type for all the nodes in the graph where data will be retrieved in order to process the results related to the connected components found in the graph.

3.9.2.2 def sparksee.ConnectedComponents.get_size (self, idConnectedComponent)
Returns the number of nodes contained in the given connected component.

Parameters:

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

Returns:

The number of nodes contained in the given connected component.

3.9.2.3 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.
3.9.2.4  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.

3.9.2.5  def sparksee.ConnectedComponents.get_connected_component ( self, idNode)

Returns the connected component where the given node belongs to.

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

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

3.9.2.6  def sparksee.ConnectedComponents.get_nodes ( self, idConnectedComponent)

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

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

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

3.9.2.7  def sparksee.ConnectedComponents.close ( self)

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

3.10  sparksee.Connectivity Class Reference

Connectivity class.
Inheritance diagram for sparksee.Connectivity:

```
sparksee.Connectivity
sparksee.StrongConnectivity   sparksee.WeakConnectivity
sparksee.StrongConnectivityGabow sparksee.WeakConnectivityDFS
```

Public Member Functions

- **def add_all_node_types**
  
  Allows connectivity through all node types of the graph.

- **def run**
  
  Runs the algorithm in order to find the connected components.

- **def exclude_nodes**
  
  Set which nodes can’t be used.

- **def is_closed**
  
  Gets if Connectivity has been closed or not.

- **def add_node_type**
  
  Allows connectivity through nodes of the given type.

- **def set_materialized_attribute**
  
  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 exclude_edges**
  
  Set which edges can’t be used.

- **def close**
  
  Closes the Connectivity instance.

- **def get_connected_components**
  
  Returns the results generated by the execution of the algorithm.
3.10.1 Detailed Description

Connectivity class.

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

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

Author:

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

3.10.2 Member Function Documentation

3.10.2.1 def sparksee.Connectivity.run ( self )

Runs the algorithm in order to find the connected components.

This method can be called only once.

Reimplemented in sparksee.WeakConnectivityDFS, sparksee.WeakConnectivity, sparksee.StrongConnectivity, and sparksee.StrongConnectivityGabow.

3.10.2.2 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:

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

Reimplemented in sparksee.WeakConnectivityDFS, sparksee.WeakConnectivity, sparksee.StrongConnectivity, and sparksee.StrongConnectivityGabow.

3.10.2.3 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.

3.10.2.4 def sparksee.Connectivity.add_node_type ( self, t )

Allows connectivity through nodes of the given type.

Parameters:

t null

Reimplemented in sparksee.WeakConnectivityDFS, sparksee.WeakConnectivity, sparksee.StrongConnectivity, and sparksee.StrongConnectivityGabow.
3.10.2.5  `def` `sparksee.Connectivity.set_materialized_attribute (self, attributeName)`

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

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

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

**Parameters:**

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

Reimplemented in `sparksee.WeakConnectivityDFS`, `sparksee.WeakConnectivity`, `sparksee.StrongConnectivity`, and `sparksee.StrongConnectivityGabow`.

3.10.2.6  `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:**

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

Reimplemented in `sparksee.WeakConnectivityDFS`, `sparksee.WeakConnectivity`, `sparksee.StrongConnectivity`, and `sparksee.StrongConnectivityGabow`.

3.10.2.7  `def` `sparksee.Connectivity.close (self)`

Closes the `Connectivity` instance.

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

3.10.2.8  `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.

Reimplemented in `sparksee.WeakConnectivityDFS`, `sparksee.WeakConnectivity`, `sparksee.StrongConnectivity`, and `sparksee.StrongConnectivityGabow`. 

---

Generated on Fri Feb 7 13:22:10 2014 for SparkseePython by Doxygen
3.11 sparksee.Context Class Reference

Context class.

Public Member Functions

• def add_edge_type
  Allows for traversing edges of the given type.

• def exclude_nodes
  Set which nodes can’t be used.

• def compute
  Gets the resulting collection of nodes.

• def is_closed
  Gets if Context has been closed or not.

• def add_all_edge_types
  Allows for traversing all edge types of the graph.

• def add_node_type
  Allows for traversing nodes of the given type.

• def compute
  Helper method to easily compute a context from a node.

• def __init__
  Creates a new instance.

• def close
  Closes the Context instance.

• def add_all_node_types
  Allows for traversing all node types of the graph.

• def set_maximum_hops
  Sets the maximum hops restriction.

• def exclude_edges
  Set which edges can’t be used.

3.11.1 Detailed Description

Context class.

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

Check out the ’Algorithms’ section in the SPARKSEE User Manual for more details on this.
3.11 sparksee.Context Class Reference

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

3.11.2 Member Function Documentation

3.11.2.1 def sparksee.Context.add_edge_type (self, t, d)
Allows for traversing edges of the given type.

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

3.11.2.2 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:
   nodes [in] A set of node identifiers that must be kept intact until the destruction of the class.

3.11.2.3 def sparksee.Context.compute (self)
Gets the resulting collection of nodes.

Returns:
The resulting collection of nodes.

3.11.2.4 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.

3.11.2.5 def sparksee.Context.add_all_edge_types (self, d)
Allows for traversing all edge types of the graph.

Parameters:
   d [in] Edge direction.
3.11.2.6  def sparksee.Context.add_node_type (self, t)
Allows for traversing nodes of the given type.

Parameters:

   t null

3.11.2.7  def sparksee.Context.compute (self, s, node, nodeTypes, edgeTypes, dir, maxhops, include)
Helper method to easily compute a context from a node.

Parameters:

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

3.11.2.8  def sparksee.Context.__init__ (self, s, node)
Creates a new instance.

Parameters:

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

3.11.2.9  def sparksee.Context.close (self)
Closes the Context instance.
It must be called to ensure the integrity of all data.

3.11.2.10 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:

   maxhops [in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.
**3.12 sparksee.CSVReader Class Reference**

`include` [in] If TRUE, the resulting collection will include those nodes at distance less or equal than the given one, otherwise it will just include those nodes at distance equal than the given one. This parameter just makes sense if `maxhops` is different from 0; in that case it includes all nodes no matters the distance.

### 3.11.2.11 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:**

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

---

**3.12 sparksee.CSVReader Class Reference**

CSVReader interface.

Inheritance diagram for sparksee.CSVReader:

```
sparksee.CSVReader
sparksee.RowReader
```

Collaboration diagram for sparksee.CSVReader:

```
sparksee.CSVReader
sparksee.RowReader
```

---

**Public Member Functions**

- **def reset**
  
  *Moves the reader to the beginning.*

- **def close**
  
  *Closes the reader.*

- **def set_num_lines**
  
  *Used to limit the number of lines that will be read.*

- **def open**
  
  *Opens the source file path.*
• def set_multilines
  Allows the use of fields with more than one line.

• def set_separator
  Sets the character used to separate fields in the file.

• def set_quotes
  Sets the character used to quote fields.

• def set_single_line
  Only allows single line fields.

• def read
  Reads the next row as a string array.

• def set_start_line
  Sets the number of lines to be skipped from the beginning.

• def set_locale
  Sets the locale that will be used to read the file.

• def get_row
  The row number for the current row.

• def __init__
  Constructs CSVReader.

3.12.1 Detailed Description

CSVReader interface.

A very simple CSV reader.

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

Using the format RFC 4180.

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

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

Fields with multiple lines can be allowed (and the maximum lines specified), but the default is a single line.

The locale string can be used to set the language, country and the file encoding. The format must be ":[language_territory],codeset]". But only the file encoding is being used in the current version.

The languages supported are: "en_US", "es_ES" and "ca_ES".

The file encodings supported are: "utf8" and "iso88591".

For example:

To don’t change the default locales, use an empty string: "".
To read a file in utf8 with the default language settings use ".utf8".
To read a file in iso88591 with English language use: "en_US.iso88591". Check out the ‘Data import’ section in the SPARKSEE User Manual for more details on this.

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

3.12.2 Member Function Documentation

3.12.2.1 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:

IOException If bad things happen during the restart.

Reimplemented from sparksee.RowReader.

3.12.2.2 def sparksee.CSVReader.close ( self)
Closes the reader.

Exceptions:

IOException If the close fails.

Reimplemented from sparksee.RowReader.

3.12.2.3 def sparksee.CSVReader.set_num_lines ( self, numLines)
Used to limit the number of lines that will be read.

Parameters:

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

3.12.2.4 def sparksee.CSVReader.open ( self, f)
Opens the source file path.

Parameters:

f [in] CSV file path.

Exceptions:

IOException If bad things happen opening the file.
3.12.2.5  def sparksee.CSVReader.set_multilines (self, numExtralines)
Allows the use of fields with more than one line.

Parameters:

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

3.12.2.6  def sparksee.CSVReader.set_separator (self, sep)
Sets the character used to separate fields in the file.

Parameters:

sep  [in] Separator character.

Exceptions:

  RuntimeError  null

3.12.2.7  def sparksee.CSVReader.set_quotes (self, quotes)
Sets the character used to quote fields.

Parameters:

quotes  [in] Quote character.

Exceptions:

  RuntimeError  null

3.12.2.8  def sparksee.CSVReader.read (self, row)
Reads the next row as a string array.

Parameters:

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

Returns:

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

Exceptions:

  IOError  If bad things happen during the read.

Reimplemented from sparksee.RowReader.

3.12.2.9  def sparksee.CSVReader.set_start_line (self, startLine)
Sets the number of lines to be skiped from the beginning.

Parameters:

startLine  [in] The line number to skip for start reading
3.12.2.10  def sparksee.CSVReader.set_locale (self, localeStr)
Sets the locale that will be used to read the file.

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

3.12.2.11  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:
    IOError If it fails.

Reimplemented from sparksee.RowReader.

3.13  sparksee.CSVWriter Class Reference

CSVWriter interface.

Inheritance diagram for sparksee.CSVWriter:

```
sparksee.CSVWriter
sparksee.RowWriter
```

Collaboration diagram for sparksee.CSVWriter:

```
sparksee.RowWriter
sparksee.CSVWriter
```

Public Member Functions

- def set_auto_quotes

  Sets on/off the automatic quote mode.

- def set_separator

  Sets the character used to separate fields in the file.

- def set_quotes
Sets the character used to quote fields.

- `def set_locale`
  Sets the locale that will be used to write the file.

- `def write`
  Writes the next row.

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

- `def close`
  Closes the writer.

- `def __init__`
  Creates a new instance.

- `def open`
  Opens the output file path.

### 3.13.1 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.


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](http://www.sparsity-technologies.com)

### 3.13.2 Member Function Documentation

#### 3.13.2.1 `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:**

- `autoquotes`  [in] If TRUE it enables the automatic quote mode, if FALSE it disables it.
3.13.2.2  def sparksee.CSVWriter.set_separator (self, sep)
Sets the character used to separate fields in the file.

Parameters:
  
  sep  [in] Separator character.

Exceptions:
  
  RuntimeError null

3.13.2.3  def sparksee.CSVWriter.set_quotes (self, quotes)
Sets the character used to quote fields.

Parameters:
  
  quotes  [in] Quote character.

Exceptions:
  
  RuntimeError null

3.13.2.4  def sparksee.CSVWriter.set_locale (self, localeStr)
Sets the locale that will be used to write the file.

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

3.13.2.5  def sparksee.CSVWriter.write (self, row)
Writes the next row.

Parameters:
  
  row  [in] Row of data.

Exceptions:
  
  IOError If bad things happen during the write.
  
  RuntimeError null

Reimplemented from sparksee.RowWriter.

3.13.2.6  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:
  
  forcequotes  [in] A booleanList with the position for each column that must be quoted set to true.
3.13.2.7 def sparksee.CSVWriter.close (self)
Closes the writer.

Exceptions:

  IOError  If the close fails.
  RuntimeError  null

Reimplemented from sparksee.RowWriter.

3.13.2.8 def sparksee.CSVWriter.open (self, f)
Opens the output file path.

Parameters:

  f  [in] Output file path.

Exceptions:

  IOError  If bad things happen opening the file.

3.14 sparksee.Database Class Reference

Database class.

Public Member Functions

- def get_alias
  Gets the alias of the Database.

- def get_path
  Gets the path of the Database.

- def is_closed
  Gets if Database instance has been closed or not.

- def new_session
  Creates a new Session.

- def close
  Closes the Database instance.

- def get_statistics
  Gets Database statistics.
3.14.1 Detailed Description

Database class.

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

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

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

Author:

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

3.14.2 Member Function Documentation

3.14.2.1 def sparksee.Database.get_alias ( self)

Gets the alias of the Database.

Returns:

The alias of the Database.

3.14.2.2 def sparksee.Database.get_path ( self)

Gets the path of the Database.

Returns:

The path of the Database.

3.14.2.3 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.

3.14.2.4 def sparksee.Database.close ( self)

Closes the Database instance.

It must be called to ensure the integrity of all data.
3.14.2.5  def sparksee.Database.get_statistics (self, stats)

Gets Database statistics.

Parameters:

3.15  sparksee.DatabaseStatistics Class Reference

Database statistics.

Public Member Functions

• def get_write
   
   Gets total written data in KBytes.

• def get_sessions
   
   Gets the number of sessions.

• def get_data
   
   Gets database size in KBytes.

• def get_temp
   
   Gets temporary storage file size in KBytes.

• def get_read
   
   Gets total read data in KBytes.

• def get_cache
   
   Gets cache size in KBytes.

3.15.1  Detailed Description

Database statistics.

Author:

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

3.15.2  Member Function Documentation

3.15.2.1  def sparksee.DatabaseStatistics.get_write (self)

Gets total written data in KBytes.

Returns:

Total read written in KBytes.
3.15.2.2  def sparksee.DatabaseStatistics.get_sessions (self)
Gets the number of sessions.

Returns:
    The number of sessions.

3.15.2.3  def sparksee.DatabaseStatistics.get_data (self)
Gets database size in KBytes.

Returns:
    Database size in KBytes.

3.15.2.4  def sparksee.DatabaseStatistics.get_temp (self)
Gets temporary storage file size in KBytes.

Returns:
    Temporary storage file size in KBytes.

3.15.2.5  def sparksee.DatabaseStatistics.get_read (self)
Gets total read data in KBytes.

Returns:
    Total read data in KBytes.

3.15.2.6  def sparksee.DatabaseStatistics.get_cache (self)
Gets cache size in KBytes.

Returns:
    Cache size in KBytes.

3.16  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.
3.16 sparksee.DataType Class Reference

- 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.

3.16.1 Detailed Description
Data type (domain) enumeration.

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

3.16.2 Member Data Documentation

3.16.2.1 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].

3.16.2.2 int sparksee.DataType.STRING = 5 [static]
Unicode string data type.
2048 characters maximum length.

3.16.2.3 int sparksee.DataType.TEXT = 6 [static]
Large unicode character object (CLOB) data type.
TextStream
3.17  sparksee.DefaultExport Class Reference

Default implementation for ExportManager class.

Inheritance diagram for sparksee.DefaultExport:

```
sparksee.ExportManager
     ^
     | sparksee.DefaultExport
```

Collaboration diagram for sparksee.DefaultExport:

```
sparksee.ExportManager
     ^
     | sparksee.DefaultExport
```

Public Member Functions

- def enable_type
  Default implementation of the ExportManager class method EnableType.

- def get_edge
  Default implementation of the ExportManager class method GetEdge.

- def get_graph
  Default implementation of the ExportManager class method GetGraph.

- def get_edge_type
  Default implementation of the ExportManager class method GetEdgeType.

- def get_node_type
  Default implementation of the ExportManager class method GetNodeType.

- def release
  Default implementation of the ExportManager class method Release.

- def get_node
  Default implementation of the ExportManager class method GetNode.

- def prepare
  Default implementation of the ExportManager class method Prepare.

- def __init__
  Creates a new instance.
3.17 sparksee.DefaultExport Class Reference

3.17.1 Detailed Description

Default implementation for ExportManager class.
It uses the default values from GraphExport, NodeExport and EdgeExport to export all node and edge types.

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

3.17.2 Member Function Documentation

3.17.2.1 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:
  type null

Returns:
  TRUE.

Reimplemented from sparksee.ExportManager.

3.17.2.2 def sparksee.DefaultExport.get_edge ( self, edge, edgeExport)
Default implementation of the ExportManager class method GetEdge.
This sets the default EdgeExport values and sets the OID as the label. Also, it exports the edge as directed just if the edge is directed.

Parameters:
  edge null
  edgeExport null

Returns:
  TRUE.

Reimplemented from sparksee.ExportManager.

3.17.2.3 def sparksee.DefaultExport.get_graph ( self, graphExport)
Default implementation of the ExportManager class method GetGraph.
This sets the default GraphExport values and "Graph" as the label.

Parameters:
  graphExport null

Reimplemented from sparksee.ExportManager.
3.17.2.4  def sparksee.DefaultExport.get_edge_type ( self, type, edgeExport )
Default implementation of the ExportManager class method GetEdgeType.
This sets the default EdgeExport values.

Parameters:

   type  null
   edgeExport  null

Reimplemented from sparksee.ExportManager.

3.17.2.5  def sparksee.DefaultExport.get_node_type ( self, type, nodeExport )
Default implementation of the ExportManager class method GetNodeType.
This sets the default NodeExport values.

Parameters:

   type  null
   nodeExport  null

Reimplemented from sparksee.ExportManager.

3.17.2.6  def sparksee.DefaultExport.get_node ( self, node, nodeExport )
Default implementation of the ExportManager class method GetNode.
This sets the default NodeExport values and sets the OID as the label.

Parameters:

   node  null
   nodeExport  null

Returns:

   TRUE.

Reimplemented from sparksee.ExportManager.

3.17.2.7  def sparksee.DefaultExport.prepare ( self, graph )
Default implementation of the ExportManager class method Prepare.

Parameters:

   graph  null

Reimplemented from sparksee.ExportManager.

3.18  sparksee.EdgeData Class Reference

Edge data class.
Public Member Functions

- def get_head
  
  Gets the head of the edge.

- def get_tail
  
  Gets the tail of the edge.

- def get_edge
  
  Gets the edge identifier.

3.18.1 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](http://www.sparsity-technologies.com)

3.18.2 Member Function Documentation

3.18.2.1 def sparksee.EdgeData.get_head ( self)

Gets the head of the edge.

**Returns:**

The Sparksee edge identifier of the head of the edge.

3.18.2.2 def sparksee.EdgeData.get_tail ( self)

Gets the tail of the edge.

**Returns:**

The Sparksee edge identifier of the tail of the edge.

3.18.2.3 def sparksee.EdgeData.get_edge ( self)

Gets the edge identifier.

**Returns:**

The Sparksee edge identifier.

3.19 sparksee.EdgeExport Class Reference

Stores edge exporting values.

---

Generated on Fri Feb 7 13:22:10 2014 for SparkseePython by Doxygen
Public Member Functions

• def set_labelcolor_rgb
  Sets the edge label color.

• def get_font_size
  Gets the edge label font size.

• def set_width
  Sets the edge width.

• def set_defaults
  Sets to default values.

• def get_color_rgb
  Gets the edge color.

• def set_font_size
  Sets the edge label font size.

• def set_as_directed
  Sets if the edge should be managed as directed.

• def get_labelcolor_rgb
  Gets the edge label color.

• def get_width
  Gets the edge width.

• def set_label
  Sets the edge label.

• def as_directed
  Gets if the edge should be managed as directed.

• def set_color_rgb
  Sets the edge color.

• def __init__
  Creates a new instance.

• def get_label
  Gets the edge label.
3.19.1 Detailed Description

Stores edge exporting values.
Some properties may be ignored depending on the exportation type.
Default values are:
Label: "" (empty string).
As directed: TRUE.
Color: 13882323 (OxD3D3D3, Light gray).
Label color: 0 (Ox000000, Black).
Width: 5px.
Font size: 10.

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

3.19.2 Member Function Documentation

3.19.2.1 def sparksee.EdgeExport.set_labelcolor_rgb (self, c)
Sets the edge label color.

Parameters:

  c [in] The edge label color.

3.19.2.2 def sparksee.EdgeExport.get_font_size (self)
 Gets the edge label font size.

Returns:

  The edge label font size.

3.19.2.3 def sparksee.EdgeExport.set_width (self, w)
Sets the edge width.

Parameters:

  w [in] The edge width.

3.19.2.4 def sparksee.EdgeExport.get_color_rgb (self)
 Gets the edge color.

Returns:

  The edge color.
3.19.2.5  def sparksee.EdgeExport.set_font_size (self, s)
Sets the edge label font size.

Parameters:
   s [in] The edge label font size.

3.19.2.6  def sparksee.EdgeExport.set_as_directed (self, b)
Sets if the edge should be managed as directed.

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

3.19.2.7  def sparksee.EdgeExport.get_labelcolor_rgb (self)
Gets the edge label color.

Returns:
   The edge label color.

3.19.2.8  def sparksee.EdgeExport.get_width (self)
Gets the edge width.

Returns:
   The edge width.

3.19.2.9  def sparksee.EdgeExport.set_label (self, l)
Sets the edge label.

Parameters:
   l [in] The edge label.

3.19.2.10 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.
3.19.2.11  def sparksee.EdgeExport.set_color_rgb ( self, c)
Sets the edge color.

Parameters:
   c [in] The edge color.

3.19.2.12  def sparksee.EdgeExport.get_label ( self)
Gets the edge label.

Returns:
   The edge label.

3.20  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.

3.20.1  Detailed Description

Edges direction enumeration.

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

3.21  sparksee.EdgeTypeExporter Class Reference

EdgeTypeExporter class.

Inheritance diagram for sparksee.EdgeTypeExporter:
Collaboration diagram for sparksee.EdgeTypeExporter:

Public Member Functions

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

- `def set_frequency`
  Sets the frequency of listener notification.

- `def set_type`
  Sets the type to be exported.

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

- `def __init__`
  Creates a new instance.

- `def set_tail_position`
  Sets the position (index column) of the tail attribute in the exported data.

- `def set_graph`
  Sets the graph that will be exported.

- `def set_row_writer`
  Sets the output data destination.

- `def register`
  Registers a new listener.

- `def run`
  See the `TypeExporter` class `Run` method.

- `def __init__`
  Creates a new instance.

- `def set_header`
  Sets the presence of a header row.

- `def set_head_position`
  Sets the position (index column) of the head attribute in the exported data.
3.21 sparksee.EdgeTypeExporter Class Reference

- def set_attributes
  Sets the list of Attributes.

3.21.1 Detailed Description

EdgeTypeExporter class.
Specific TypeExporter implementation for edge types.
Check out the ‘Data export’ section in the SPARKSEE User Manual for more details on this.

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

3.21.2 Member Function Documentation

3.21.2.1 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:
  attr [in] Tail Attribute

3.21.2.2 def sparksee.EdgeTypeExporter.set_frequency ( self, freq)
Sets the frequency of listener notification.
freq [in] Frequency in number of rows managed to notify progress to all listeners

Parameters:
  freq null
Reimplemented from sparksee.TypeExporter.

3.21.2.3 def sparksee.EdgeTypeExporter.set_type ( self, t)
Sets the type to be exported.

Parameters:
  t [in] Type identifier.
Reimplemented from sparksee.TypeExporter.

3.21.2.4 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:
  attr [in] Head Attribute
3.21.2.5  def sparksee.EdgeTypeExporter.__init__ (self, rw, g, t, attrs, hPos, tPos, hAttr, tAttr)

Creates a new instance.

Parameters:
- **rw** [in] Output RowWriter.
- **g** [in] Graph.
- **t** [in] Type identifier.
- **attrs** [in] Attribute identifiers to be exported.
- **hPos** [in] The position (index column) for the head value.
- **tPos** [in] The position (index column) for the tail value.
- **hAttr** [in] The attribute identifier to get the value to be dumped for the head.
- **tAttr** [in] The attribute identifier to get the value to be dumped for the tail.

3.21.2.6  def sparksee.EdgeTypeExporter.set_tail_position (self, pos)

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

Parameters:
- **pos** [in] Tail position

3.21.2.7  def sparksee.EdgeTypeExporter.set_graph (self, g)

Sets the graph that will be exported.

Parameters:
- **g** [in] Graph.

Reimplemented from sparksee.TypeExporter.

3.21.2.8  def sparksee.EdgeTypeExporter.set_row_writer (self, rw)

Sets the output data destination.

Parameters:
- **rw** null

Reimplemented from sparksee.TypeExporter.

3.21.2.9  def sparksee.EdgeTypeExporter.register (self, tel)

Registers a new listener.

Parameters:
- **tel** [in] TypeExporterListener to be registered.

Reimplemented from sparksee.TypeExporter.
3.21.2.10  def sparksee.EdgeTypeExporter.run (self)
See the TypeExporter class Run method.

Exceptions:

   IOError  null
   RuntimeError  null

Reimplemented from sparksee.TypeExporter.

3.21.2.11  def sparksee.EdgeTypeExporter.set_header (self, header)
Sets the presence of a header row.

Parameters:

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

Reimplemented from sparksee.TypeExporter.

3.21.2.12  def sparksee.EdgeTypeExporter.set_head_position (self, pos)
Sets the position (index column) of the head attribute in the exported data.

Parameters:

   pos  [in] Head position

3.21.2.13  def sparksee.EdgeTypeExporter.set_attributes (self, attrs)
Sets the list of Attributes.

Parameters:

   attrs  [in] Attribute identifiers to be exported

Reimplemented from sparksee.TypeExporter.

3.22  sparksee.EdgeTypeLoader Class Reference

EdgeTypeLoader class.

Inheritance diagram for sparksee.EdgeTypeLoader:
Public Member Functions

- `def set_tail_attribute`
  
  Sets the attribute that will be used to find the tail of the edge.

- `def set_frequency`
  
  Sets the frequency of listener notification.

- `def set_log_off`
  
  Turn off all the error reporting.

- `def __init__`
  
  Creates a new instance.

- `def set_type`
  
  Sets the type to be loaded.

- `def set_log_error`
  
  Sets a log error file.

- `def set_head_attribute`
  
  Sets the attribute that will be used to find the head of the edge.

- `def run_two_phases`
  
  See the `TypeLoader` class `RunTwoPhases` method.

- `def set_tail_position`
  
  Sets the position of the tail attribute in the source data.

- `def set_graph`
  
  Sets the graph where the data will be loaded.

- `def set_row_reader`
  
  Sets the input data source.

- `def set_attribute_positions`
  
  Sets the list of attribute positions.

- `def register`
  
  Registers a new listener.
3.22 sparksee.EdgeTypeLoader Class Reference

• def set_locale
  Sets the locale that will be used to read the data.

• def run
  See the TypeLoader class Run method.

• def run_n_phases
  See the TypeLoader class RunNPhases method.

• def __init__
  Creates a new instance.

• def set_timestamp_format
  Sets a specific timestamp format.

• def set_head_position
  Sets the position of the head attribute in the source data.

• def set_attributes
  Sets the list of Attributes.

3.22.1 Detailed Description

EdgeTypeLoader class.

Specific TypeLoader implementation for edge types.

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

Author:

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

3.22.2 Member Function Documentation

3.22.2.1 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:

  attr [in] Tail Attribute

3.22.2.2 def sparksee.EdgeTypeLoader.set_frequency ( self, freq )

Sets the frequency of listener notification.

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

Parameters:

  freq null

Reimplemented from sparksee.TypeLoader.
3.22.2.3  
def sparksee.EdgeTypeLoader.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.
Reimplemented from sparksee.TypeLoader.

3.22.2.4  
def sparksee.EdgeTypeLoader.__init__ (self, rr, g, t, attrs, attrsPos, hPos, tPos, hAttr, tAttr)

Creates a new instance.
Parameters:

- **rr** [in] Input RowReader.
- **g** [in] Graph.
- **t** [in] Type identifier.
- **attrs** [in] Attribute identifiers to be loaded.
- **attrsPos** [in] Attribute positions (column index >=0). to all listeners.
- **hPos** [in] The position (index column) for the head value.
- **tPos** [in] The position (index column) for the tail value.
- **hAttr** [in] The attribute identifier for the head.
- **tAttr** [in] The attribute identifier for the tail.

3.22.2.5  
def sparksee.EdgeTypeLoader.set_type (self, t)

Sets the type to be loaded.
Parameters:

- **t** [in] Type identifier.

Reimplemented from sparksee.TypeLoader.

3.22.2.6  
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:

- **path** null

Exceptions:

- **IOError** If bad things happen opening the file.

Reimplemented from sparksee.TypeLoader.
3.22.7 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:
   attr [in] Head Attribute

3.22.8 def sparksee.EdgeTypeLoader.run_two_phases (self)
See the TypeLoader class RunTwoPhases method.

Exceptions:
   IOError null
   RuntimeError null

Reimplemented from sparksee.TypeLoader.

3.22.9 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:
   pos [in] Tail position

3.22.10 def sparksee.EdgeTypeLoader.set_graph (self, g)
Sets the graph where the data will be loaded.

Parameters:
   g [in] Graph.

Reimplemented from sparksee.TypeLoader.

3.22.11 def sparksee.EdgeTypeLoader.set_row_reader (self, rr)
Sets the input data source.

Parameters:
   rr [in] Input RowReader.

Reimplemented from sparksee.TypeLoader.
3.22.2.12 def sparksee.EdgeTypeLoader.set_attribute_positions (self, attrsPos)
Sets the list of attribute positions.

Parameters:

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

Reimplemented from sparksee.TypeLoader.

3.22.2.13 def sparksee.EdgeTypeLoader.register (self, tel)
Registers a new listener.

Parameters:

   tel TypeLoaderListener to be registered.

Reimplemented from sparksee.TypeLoader.

3.22.2.14 def sparksee.EdgeTypeLoader.set_locale (self, localeStr)
Sets the locale that will be used to read the data.

It should match the locale used in the rowreader.

Parameters:

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

Reimplemented from sparksee.TypeLoader.

3.22.2.15 def sparksee.EdgeTypeLoader.run (self)
See the TypeLoader class Run method.

Exceptions:

   IOError null
   RuntimeError null

Reimplemented from sparksee.TypeLoader.

3.22.2.16 def sparksee.EdgeTypeLoader.run_n_phases (self, partitions)
See the TypeLoader class RunNPhases method.

Parameters:

   partitions null

Exceptions:

   IOError null
   RuntimeError null

Reimplemented from sparksee.TypeLoader.
3.22.17  def sparksee.EdgeTypeLoader.set_timestamp_format (self, timestampFormat)
Sets a specific timestamp format.

Parameters:
  timestampFormat  null

Reimplemented from sparksee.TypeLoader.

3.22.18  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:
  pos  [in] Head position

3.22.19  def sparksee.EdgeTypeLoader.set_attributes (self, attrs)
Sets the list of Attributes.

Parameters:
  attrs  [in] Attribute identifiers to be loaded

Reimplemented from sparksee.TypeLoader.

3.23  sparksee.ExportManager Class Reference

Defines how to export a graph to an external format.

Inheritance diagram for sparksee.ExportManager:

```
sparksee.ExportManager
    ▷ sparksee.DefaultExport
```

Public Member Functions

- def get_node_type
  Gets the default node export definition for the given node type.

- def get_edge
  Gets the edge export definition for the given edge.

- def get_graph
  Gets the graph export definition.
3.23 sparksee.ExportManager Class Reference

• def get_edge_type
  
  Gets the default node export definition for the given edge type.

• def prepare
  
  Prepares the graph for the export process.

• def get_node
  
  Gets the node export definition for the given node.

• def release
  
  Ends the export process.

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

3.23.1 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 process, 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 different format. Nowadays, available formats are defined in the ExportType enum.

Author:

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

3.23.2 Member Function Documentation

3.23.2.1 def sparksee.ExportManager.get_node_type ( self, type, nodeExport)

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

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

Parameters:

type  [in] Node type identifier.

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

Returns:

TRUE.

Reimplemented in sparksee.DefaultExport.

3.23.2.2 def sparksee.ExportManager.get_edge ( self, edge, edgeExport)

Gets the edge export definition for the given edge.
Parameters:

edge Edge identifier.

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

Returns:

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

Reimplemented in sparksee.DefaultExport.

3.23.2.3 def sparksee.ExportManager.get_graph (self, graphExport)

Gets the graph export definition.

Parameters:

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

Returns:

TRUE.

Reimplemented in sparksee.DefaultExport.

3.23.2.4 def sparksee.ExportManager.get_edge_type (self, type, edgeExport)

Gets the default node export definition for the given edge type.
GetEdge has a higher priority than this function. That is, only if GetEdge returns FALSE, the EdgeExport
of this function will be used.

Parameters:

type [in] Edge type identifier.

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

Returns:

TRUE.

Reimplemented in sparksee.DefaultExport.

3.23.2.5 def sparksee.ExportManager.prepare (self, graph)

Prepares the graph for the export process.
It is called once before the export process.

Parameters:

graph Graph to be exported.

Reimplemented in sparksee.DefaultExport.
3.23.2.6  def sparksee.ExportManager.get_node (self, node, nodeExport)

Gets the node export definition for the given node.

Parameters:

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

Returns:

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

Reimplemented in sparksee.DefaultExport.

3.23.2.7  def sparksee.ExportManager.release (self)

Ends the export process.

It is called once after the export process.

Reimplemented in sparksee.DefaultExport.

3.23.2.8  def sparksee.ExportManager.enable_type (self, type)

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

Parameters:

- type  Node or edge type identifier.

Returns:

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

Reimplemented in sparksee.DefaultExport.

3.24  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.
3.25  sparksee.Graph Class Reference

3.24.1  Detailed Description

Export type.

Author:

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

3.24.2  Member Data Documentation

3.24.2.1  int sparksee.ExportType.GRAPHVIZ = 0  [static]
Export to Graphviz format.
Graphviz home page: http://www.graphviz.org

3.24.2.2  int sparksee.ExportType.GRAPHML = 1  [static]
Export to GraphML format.
GraphML home page: http://graphml.graphdrawing.org/

3.24.2.3  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").

3.25  sparksee.Graph Class Reference

Graph class.

Public Member Functions

- def set_attribute_text
  Sets the writable TextStream for the given text attribute and OID.

- def dump_data
  Dumps logical data to a file.

- def find_types
  Gets all existing Sparksee node and edge type identifiers.

- def set_attribute
  Sets the Value for the given attribute and OID.

- def select
  Selects all OIDs satisfying the given condition for the given attribute.
3.25 sparksee.Graph Class Reference

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

- def rename_attribute
  Renames an attribute.

- def get_attribute_interval_count
  Gets how many objects have a value into the given range for the given attribute.

- def tails_and_heads
  Gets all the tails and heads from the given edges collection.

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

- def rename_type
  Renames a type.

- def dump_storage
  Dumps internal storage data to a file.

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

- def export
  Exports the Graph.

- def get_attributes
  Gets all Sparksee attribute identifiers with a non-NULL value for the given Sparksee OID.

- def get_attribute_statistics
  Gets statistics from the given attribute.

- def new_node
  Creates a new node instance.

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

- def count_edges
  Gets the number of edges.

- def find_edge_types
  Gets all existing Sparksee edge type identifiers.

- def select
  Selects all OIDs satisfying the given condition for the given attribute.

- def index_attribute
  Updates the index of the given attribute.
• def get_type
  Gets information about the given type.

• def set_attribute_default_value
  Sets a default value for an attribute.

• def find_attribute
  Gets the Sparksee attribute identifier for the given type identifier and attribute name.

• def new_attribute
  Creates a new attribute.

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

• def select
  Selects all OIDs belonging to the given type.

• def find_node_types
  Gets all existing Sparksee node type identifiers.

• def get_attribute
  Gets the Value for the given attribute and OID.

• def remove_attribute
  Removes the given attribute.

• def backup
  Dumps all the data to a backup file.

• def new_session_attribute
  Creates a new Session attribute with a default value.

• def find_attributes
  Gets all existing Sparksee attribute identifiers for the given type identifier.

• def count_nodes
  Gets the number of nodes.

• def get_edge_data
  Gets information about an edge.

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

• def rename_type
  Renames a type.

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

- def new_node_type
  Creates a new node type.

- def new_session_attribute
  Creates a new Session attribute.

- def tails
  Gets all the tails from the given edges collection.

- def find_or_create_edge
  Gets any of the edges of the specified type between two given nodes (tail and head).

- def drop
  Drops the given OID.

- def new_edge_type
  Creates a new edge type.

- def heads
  Gets all the heads from the given edges collection.

- def find_edge
  Gets any of the edges of the given type between two given nodes (tail and head).

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

- def drop
  Drops all the OIDs from the given collection.

- def new_attribute
  Creates a new attribute with a default value.

- def get_object_type
  Gets the Sparksee node or edge type identifier for the given OID.

- def get_attribute
  Gets information about the given attribute.

- def new_edge
  Creates a new edge instance.

- def get_values
  Gets the Value collection for the given attribute.

- def get_edge_peer
  Gets the other end for the given edge.
3.25 sparksee.Graph Class Reference

* def find_object
  Finds one object having the given Value for the given attribute.

* def new_edge
  Creates a new edge instance.

* def select
  Selects all OIDs satisfying the given condition for the given attribute.

* def find_type
  Gets the Sparksee type identifier for the given type name.

* def get_attribute
  Gets the Value for the given attribute and OID.

* def find_or_create_object
  Finds one object having the given Value for the attribute or it creates one does not exist any.

* def remove_type
  Removes the given type.

* def new_restricted_edge_type
  Creates a new restricted edge type.

3.25.1 Detailed Description

Graph class.

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

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

Author:

Sparsity Technologies [http://www.sparsity-technologies.com](http://www.sparsity-technologies.com)

3.25.2 Member Function Documentation

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

Sets the writable TextStream for the given text attribute and OID.

Parameters:

  *oid*  [in] Sparksee OID.
  *attr*  [in] Sparksee attribute identifier.
  *tstream*  [in] New Text value. This corresponds to a TextStream to write.
3.25.2.2 def sparksee.Graph.dump_data (self, file)  
Dumps logical data to a file.

Parameters:
    file [in] Output file path.

Exceptions:
    IOError If the given file cannot be created.
    RuntimeError null

3.25.2.3 def sparksee.Graph.find_types (self)  
Gets all existing Sparksee node and edge type identifiers.

Returns:
    Sparksee node and edge type identifier list.

3.25.2.4 def sparksee.Graph.set_attribute (self, oid, attr, v)  
Sets the Value for the given attribute and OID.

Parameters:
    oid [in] Sparksee OID.
    attr [in] Sparksee attribute identifier.
    v [in] Value for the given attribute and for the given OID.

3.25.2.5 def sparksee.Graph.select (self, attr, cond, v)  
Selects all OIDs satisfying the given condition for the given attribute.

Parameters:
    attr [in] Sparksee attribute identifier.
    cond [in] Condition to be satisfied.
    v [in] Value to be satisfied.

Returns:
    Objects instance.

3.25.2.6 def sparksee.Graph.select (self, attr, cond, v, restriction)  
Selects all OIDs satisfying the given condition for the given attribute.

Parameters:
    attr [in] Sparksee attribute identifier.
cond [in] Condition to be satisfied.

\( v \) [in] Value to be satisfied.

\textit{restriction} [in] Objects to limit the select in this set of objects.

Returns:

Objects instance.

3.25.2.7 \texttt{def sparksee.Graph.rename_attribute ( self, \textit{attr}, \textit{newName})}

Renames an attribute.

The new name must be available.

Parameters:

\textit{attr} [in] Sparksee attribute identifier.

\textit{newName} [in] The new name for the attribute.

3.25.2.8 \texttt{def sparksee.Graph.get_attribute_interval_count ( self, \textit{attr}, \textit{lower}, \textit{includeLower}, \textit{higher}, \textit{includeHigher})}

Gets how many objects have a value into the given range for the given attribute.

This only works for the attributes with the AttributeKind Indexed or Unique.

Given values must belong to the same \texttt{DataType} than the attribute.

Parameters:

\textit{attr} [in] Sparksee attribute identifier.

\textit{lower} [in] Lower bound Value of the range.

\textit{includeLower} [in] If TRUE, include lower bound Value of the range.

\textit{higher} [in] Higher bound Value of the range.

\textit{includeHigher} [in] If TRUE, include higher bound Value of the range.

Returns:

Number of objects having a value into the given range.

3.25.2.9 \texttt{def sparksee.Graph.tails_and_heads ( self, \textit{edges}, \textit{tails}, \textit{heads})}

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

Parameters:

\textit{edges} [in] Sparksee edge identifier collection.

\textit{tails} [in|out] If not NULL, all the tails will be stored here.

\textit{heads} [in|out] If not NULL, all the heads will be stored here.
3.25.2.10  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:

oid  [in] Sparksee node OID.
etype  [in] Sparksee edge type identifier.
dir  [in] Direction.

Returns:

The number of edges.

3.25.2.11  def sparksee.Graph.rename_type (self, oldName, newName)
Renames a type.
The new name must be available.

Parameters:

oldName  [in] The current name of the type to be renamed.
newName  [in] The new name for the type.

3.25.2.12  def sparksee.Graph.dump_storage (self, file)
Dumps internal storage data to a file.

Parameters:

file  [in] Output file path.

Exceptions:

IOError  If the given file cannot be created.
RuntimeError  null

3.25.2.13  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:

objs  [in] Sparksee node OID collection.
etype  [in] Sparksee edge type identifier.
dir  [in] Direction.

Returns:

Objects instance.
3.25.2.14  def sparksee.Graph.export (self, file, type, e)
Exports the Graph.

Parameters:
   type  [in] Export type.
   e    [in] Defines how to do the export for each graph object.

Exceptions:
   IOError  null

3.25.2.15  def sparksee.Graph.get_attributes (self, oid)
Gets all Sparksee attribute identifiers with a non-NULL value for the given Sparksee OID.

Parameters:
   oid  [in] Sparksee OID.

Returns:
   Sparksee attribute identifier list.

3.25.2.16  def sparksee.Graph.get_attribute_statistics (self, attr, basic)
Gets statistics from the given attribute.

Parameters:
   attr  [in] Sparksee attribute identifier.
   basic [in] If FALSE all statistics are computed, if TRUE just those statistics marked as basic will
            be computed (see description of the AttributeStatistics class). Of course, computing just basic
            statistics will be faster than computing all of them.

Returns:
   An AttributeStatistics instace.

3.25.2.17  def sparksee.Graph.new_node (self, type)
Creates a new node instance.

Parameters:
   type  [in] Sparksee type identifier.

Returns:
   Unique OID of the new node instance.
3.25.2.18  def sparksee.Graph.get_attribute_text ( self, oid, attr)

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

Parameters:

    oid [in] Sparksee OID.
    attr [in] Sparksee attribute identifier.

Returns:

    A TextStream. This returns a TextStream to read.

3.25.2.19  def sparksee.Graph.count_edges ( self)

Gets the number of edges.

Returns:

    The number of edges.

3.25.2.20  def sparksee.Graph.find_edge_types ( self)

Gets all existing Sparksee edge type identifiers.

Returns:

    Sparksee edge type identifier list.

3.25.2.21  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:

    attr [in] Sparksee attribute identifier.
    cond [in] Condition to be satisfied. It must be the Between Condition.
    lower [in] Lower-bound Value to be satisfied.
    higher [in] Higher-bound Value to be satisfied.

Returns:

    Objects instance.

3.25.2.22  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:

    attr [in] Sparksee attribute identifier.
    kind [in] Attribute kind.
3.25.2.23  def sparksee.Graph.get_type (self, type)
Gets information about the given type.

Parameters:
    type [in] Sparksee type identifier.

Returns:
    The Type for the given Sparksee type identifier.

3.25.2.24  def sparksee.Graph.set_attribute_default_value (self, attr, v)
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:
    v [in] The default value to use for this attribute.

3.25.2.25  def sparksee.Graph.find_attribute (self, type, name)
Gets the Sparksee attribute identifier for the given type identifier and attribute name.

Parameters:
    type [in] Sparksee type identifier.
    name [in] Unique attribute name.

Returns:
    The Sparksee attribute identifier for the given type and attribute name or InvalidAttribute if there is no
    attribute with the given name for the given type.

3.25.2.26  def sparksee.Graph.new_attribute (self, type, name, dt, kind)
Creates a new attribute.

Parameters:
    type [in] Sparksee node or edge type identifier.
    name [in] Unique name for the new attribute.
    dt [in] Data type for the new attribute.
    kind [in] Attribute kind.

Returns:
    Unique Sparksee attribute identifier.
3.25.2.27  def sparksee.Graph.edges ( self, etype, tail, head)
Gets all the edges of the given type between two given nodes (tail and head).

Parameters:

etype  null

Parameters:

tail  [in] Tail node identifier.

head  [in] Head node identifier.

Returns:

Objects instance.

3.25.2.28  def sparksee.Graph.select ( self, type)
Selects all OIDs belonging to the given type.

Parameters:

Parameters:

type  [in] Sparksee type identifier.

Returns:

Objects instance.

3.25.2.29  def sparksee.Graph.find_node_types ( self)
Gets all existing Sparksee node type identifiers.

Returns:

Sparksee node type identifier list.

3.25.2.30  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:

oid  [in] Sparksee OID.

Parameters:

attr  [in] Sparksee attribute identifier.

Returns:

A new Value instance having the attribute value for the given OID.
3.25.2.31 def sparksee.Graph.remove_attribute (self, attr)
Removes the given attribute.

Parameters:
attr [in] Sparksee attribute identifier.

3.25.2.32 def sparksee.Graph.backup (self, file)
Dumps all the data to a backup file.
See the Sparksee class Restore method.

Parameters:
file [in] Output backup file path.

Exceptions:

IOError If the given file cannot be created.
RuntimeError null

3.25.2.33 def sparksee.Graph.new_session_attribute (self, type, dt, kind, defaultValue)
Creates a new Session attribute with a default value.
Session attributes are exclusive for the Session (just its Session can use the attribute) and are automatically
removed when the Session is closed (thus, attribute data is not persistent into the database).
Since they are not persistent, they cannot be retrieved from the database, so they do not have an identifier
name.

Parameters:
type [in] Sparksee node or edge type identifier.
dt [in] Data type for the new attribute.
kind [in] Attribute kind.
defaultValue [in] The default value to use in each new node/edge.

Returns:
Unique Sparksee attribute identifier.

3.25.2.34 def sparksee.Graph.find_attributes (self, type)
Gets all existing Sparksee attribute identifiers for the given type identifier.

Parameters:
type [in] Sparksee type identifier.

Returns:
Sparksee attribute identifier list.
3.25.2.35  def sparksee.Graph.count_nodes (self)
Gets the number of nodes.

Returns:
   The number of nodes.

3.25.2.36  def sparksee.Graph.get_edge_data (self, edge)
Gets information about an edge.

Parameters:
   edge [in] Sparksee edge identifier.

Returns:
   An EdgeData instance.

3.25.2.37  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:
   oid [in] Sparksee node OID.
   etype [in] Sparksee edge type identifier.
   dir [in] Direction.

Returns:
   Objects instance.

3.25.2.38  def sparksee.Graph.rename_type (self, type, newName)
Renames a type.
The new name must be available.

Parameters:
   type [in] The type to be renamed.
   newName [in] The new name for the type.

3.25.2.39  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:
   objs [in] Sparksee node OID collection.
   etype [in] Sparksee edge type identifier.
   dir [in] Direction.

Returns:
   Objects instance.
3.25.2.40  def sparksee.Graph.new_node_type (self, name)
Creates a new node type.

Parameters:
   name [in] Unique name for the new node type.

Returns:
   Unique Sparksee type identifier.

3.25.2.41  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:
   type [in] Sparksee node or edge type identifier.
   dt [in] Data type for the new attribute.
   kind [in] Attribute kind.

Returns:
   Unique Sparksee attribute identifier.

3.25.2.42  def sparksee.Graph.tails (self, edges)
Gets all the tails from the given edges collection.

Parameters:
   edges [in] Sparksee edge identifier collection.

Returns:
   The tails collection.

3.25.2.43  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:
   etype [in] Sparksee edge type identifier.
   tail [in] Tail node identifier.
   head [in] Head node identifier.

Returns:
   Any of the edges or the Objects InvalidOID.
3.25.2.44  def sparksee.Graph.drop (self, oid)
Drops the given OID. It also removes its edges as well as its attribute values.

Parameters:

oid  [in] Sparksee OID to be removed.

3.25.2.45  def sparksee.Graph.new_edge_type (self, name, directed, neighbors)
Creates a new edge type.

Parameters:

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

Returns:

Unique Sparksee type identifier.

3.25.2.46  def sparksee.Graph.heads (self, edges)
Gets all the heads from the given edges collection.

Parameters:

edges  [in] Sparksee edge identifier collection.

Returns:

The heads collection.

3.25.2.47  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:

etype  [in] Sparksee edge type identifier.
tail  [in] Tail node identifier.
head  [in] Head node identifier.

Returns:

Any of the edges or the Objects InvalidOID.
3.25.2.48  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:
   oid    [in] Sparksee node OID.
   etype  [in] Sparksee edge type identifier.
   dir    [in] Direction.

Returns:
    Objects instance.

3.25.2.49  def sparksee.Graph.drop ( self, objs)
Drops all the OIDs from the given collection.
See Drop method with the single OID parameter. This performs that call for all the elements into the collection.

Parameters:
   objs    [in] Objects collection with the OIDs to be removed.

3.25.2.50  def sparksee.Graph.new_attribute ( self, type, name, dt, kind, defaultValue)
Creates a new attribute with a default value.

Parameters:
   type    [in] Sparksee node or edge type identifier.
   name    [in] Unique name for the new attribute.
   dt       [in] Data type for the new attribute.
   kind    [in] Attribute kind.
   defaultValue    [in] The default value to use in each new node/edge.

Returns:
    Unique Sparksee attribute identifier.

3.25.2.51  def sparksee.Graph.get_object_type ( self, oid)
 Gets the Sparksee node or edge type identifier for the given OID.

Parameters:
   oid    [in] Sparksee OID.

Returns:
    Sparksee node or edge type identifier.
3.25.2.52  def sparksee.Graph.get_attribute ( self, attr)

Gets information about the given attribute.

Parameters:

   attr [in] Sparksee attribute identifier.

Returns:

The Attribute for the given Sparksee attribute identifier.

3.25.2.53  def sparksee.Graph.new_edge ( self, type, tailAttr, tailV, headAttr, headV)

Creates a new edge instance.

The tail of the edge will be any node having the given tailV Value for the given tailAttr attribute identifier,
and the head of the edge will be any node having the given headV Value for the given headAttr attribute identifier.

Parameters:

   type  [in] Sparksee type identifier.
   headAttr  [in] Sparksee attribute identifier.
   headV  [in] Value.

Returns:

Unique OID of the new edge instance.

3.25.2.54  def sparksee.Graph.get_values ( self, attr)

Gets the Value collection for the given attribute.

Parameters:

   attr  [in] Sparksee attribute identifier.

3.25.2.55  def sparksee.Graph.get_edge_peer ( self, edge, node)

Gets the other end for the given edge.

Parameters:

   edge  [in] Sparksee edge identifier.
   node  [in] Sparksee node identifier. It must be one of the ends of the edge.

Returns:

The other end of the edge.
3.25.2.56  def sparksee.Graph.find_object (self, attr, v)
Finds one object having the given Value for the given attribute.
If there are more than one, then any of them will be returned. And in case there are no object having this Value, the Objects InvalidOID will be returned.

Parameters:

    attr [in] Sparksee attribute identifier.
    v [in] Value.

Returns:

    Sparksee OID or the Objects InvalidOID.

3.25.2.57  def sparksee.Graph.new_edge (self, type, tail, head)
Creates a new edge instance.

Parameters:

    type [in] Sparksee type identifier.
    tail [in] Source Sparksee OID.
    head [in] Target Sparksee OID.

Returns:

    Unique OID of the new edge instance.

3.25.2.58  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:

    attr [in] Sparksee attribute identifier.
    cond [in] Condition to be satisfied. It must be the Between Condition.
    lower [in] Lower-bound Value to be satisfied.
    higher [in] Higher-bound Value to be satisfied.
    restriction [in] Objects to limit the select in this set of objects.

Returns:

    Objects instance.

3.25.2.59  def sparksee.Graph.find_type (self, name)
Gets the Sparksee type identifier for the given type name.

Parameters:

    name [in] Unique type name.
Returns:

The Sparksee type identifier for the given type name or the Type InvalidType if there is no type with the given name.

3.25.2.60 def sparksee.Graph.get_attribute (self, oid, attr, v)

Gets the Value for the given attribute and OID.

Parameters:

oid [in] Sparksee OID.
attr [in] Sparksee attribute identifier.
v [in|out] Value for the given attribute and for the given OID.

3.25.2.61 def sparksee.Graph.find_or_create_object (self, attr, v)

Finds one object having the given Value for the attribute or it creates one does not exist any.

If the attribute is a node or edge attribute and at least one node/edge with that value is found, it returns one of them. But if it does not exist, then: If it’s a node attribute it will create it and set the attribute. If it’s an edge attribute it will return the InvalidOID.

Using this method with a global attribute will return the InvalidOID.

Parameters:

attr [in] Sparksee attribute identifier.
v [in] Value.

Returns:

Sparksee OID or the Objects InvalidOID.

3.25.2.62 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:

type [in] Sparksee type identifier.

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

Creates a new restricted edge type.

Parameters:

name [in] Unique name for the new edge type.
tail [in] Tail Sparksee node type identifier.
head  [in]  Head Sparksee node type identifier.

neighbors  [in]  If TRUE, this indexes neighbor nodes, otherwise not.

Returns:

Unique Sparksee type identifier.

3.26  sparksee.GraphExport Class Reference

Stores the graph exporting values.

Public Member Functions

•  def  set_label
  
  Sets the graph label.

•  def  get_label
  
  Gets the graph label.

•  def  set_defaults
  
  Sets to default values.

•  def  __init__
  
  Creates a new GraphExport instance.

3.26.1  Detailed Description

Stores the graph exporting values.

Author:

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

3.26.2  Member Function Documentation

3.26.2.1  def  sparksee.GraphExport.set_label (self, l)

Sets the graph label.

Parameters:

  l  [in]  The graph label.

3.26.2.2  def  sparksee.GraphExport.get_label (self)

Gets the graph label.

Returns:

  The graph label.
3.27 sparksee.Int32List Class Reference

Sparksee 32-bit signed integer list.

Public Member Functions

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

• def clear
  Clears the list.

• def __iter__
  Gets a new TypeListIterator.

• def __init__
  Constructor.

• def iterator
  Gets a new Int32ListIterator.

• def count
  Number of elements in the list.

3.27.1 Detailed Description

Sparksee 32-bit signed integer list.
It stores a 32-bit signed integer list.
Use Int32ListIterator to access all elements into this collection.

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

3.27.2 Member Function Documentation

3.27.2.1 def sparksee.Int32List.add (self, value)
Adds an 32-bit signed integer at the end of the list.

Parameters:
  value null
3.28 sparksee.Int32ListIterator Class Reference

3.27.2.2 def sparksee.Int32List.__iter__(self)
Gets a new TypeListIterator.

Returns:
  TypeListIterator instance

3.27.2.3 def sparksee.Int32List.__init__(self)
Constructor.
This creates an empty list.

3.27.2.4 def sparksee.Int32List.iterator (self)
Gets a new Int32ListIterator.

Returns:
  Int32ListIterator instance.

3.27.2.5 def sparksee.Int32List.count (self)
Number of elements in the list.

Returns:
  Number of elements in the list.

3.28 sparksee.Int32ListIterator Class Reference

Int32List iterator class.

Public Member Functions

• def has_next
  Gets if there are more elements.

• def next
  Moves to the next element.

• def __next__
  Used in next().

3.28.1 Detailed Description

Int32List iterator class.
Iterator to traverse all the integer into a Int32List instance.

Author:
  Sparsity Technologies http://www.sparsity-technologies.com
3.28.2 Member Function Documentation

3.28.2.1 def sparksee.Int32ListIterator.hasNext (self)

Gets if there are more elements.

Returns:

TRUE if there are more elements, FALSE otherwise.

3.28.2.2 def sparksee.Int32ListIterator.next (self)

Moves to the next element.

Returns:

The next element.

3.28.2.3 def sparksee.Int32ListIterator.__next__ (self)

Used in next().

Returns:

The next element.

3.29 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.
3.29 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

3.29.2 Member Data Documentation

3.29.2.1 int sparksee.LogLevel.SEVERE = 1 [static]
Severe log level.
This is the lower log level, just errors are shown.

3.29.2.2 int sparksee.LogLevel.WARNING = 2 [static]
Warning log level.
Errors and warnings are shown.

3.29.2.3 int sparksee.LogLevel.INFO = 3 [static]
Info log level.
Errors, warnings and information messages are shown.

3.29.2.4 int sparksee.LogLevel.CONFIG = 4 [static]
Config log level.
Errors, warnings, information messages and configuration details of the different components are shown.

3.29.2.5 int sparksee.LogLevel.FINE = 5 [static]
Fine log level.
This is the higher and finest public log level, everything is dumped to the log.

3.29.2.6 int sparksee.LogLevel.DEBUG = 6 [static]
Debug log level.
This is for Sparksee development purposes and just works with debug versions of the library.

3.30 sparksee.NodeExport Class Reference

Stores the node exporting values.

Public Member Functions

- def get_shape
3.30  sparksee.NodeExport Class Reference

Gets the node shape.

• def set_labelcolor_rgb
  Sets the node label color.

• def get_font_size
  Gets the node label font size.

• def set_width
  Gets the node width.

• def set_defaults
  Sets to default values.

• def get_color_rgb
  Gets the node color.

• def set_font_size
  Sets the node label font size.

• def set_height
  Sets the node height.

• def is_fit
  Gets whether the node size is fitted to the label or not.

• def set_fit
  Sets the node fit property.

• def get_labelcolor_rgb
  Gets the node label color.

• def get_width
  Gets the node width.

• def set_label
  Sets the node label.

• def set_color_rgb
  Sets the node color.

• def get_label
  Gets the node label.

• def set_shape
  Sets the node shape.

• def get_height
  Gets the node height.
3.3.0 sparksee.NodeExport Class Reference

- **def __init__**
  
  Creates a new instance.

3.3.1 Detailed Description

Stores the node exporting values.

When 'fit' is set to TRUE, then 'height' and 'width' will be ignored.

Some properties may be ignored depending on the exportation type.

Default values are:

- **Label**: "" (empty string).
- **Shape**: Box.
- **Color**: 10863606 (0xa5c3f6).
- **Label color**: 0 (Ox000000, Black).
- **Height**: 25px.
- **Width**: 25px.
- **Fit**: TRUE.
- **Font size**: 10.

**Author:**

Sparsity Technologies [http://www.sparsity-technologies.com](http://www.sparsity-technologies.com)

3.3.2 Member Function Documentation

3.3.2.1 **def sparksee.NodeExport.get_shape (self)**

Gets the node shape.

**Returns:**

The node shape.

3.3.2.2 **def sparksee.NodeExport.set_labelcolor_rgb (self, c)**

Sets the node label color.

**Parameters:**

- c [in] The node label color.

3.3.2.3 **def sparksee.NodeExport.get_font_size (self)**

Gets the node label font size.

**Returns:**

The node label font size.
3.30.2.4  def sparksee.NodeExport.set_width (self, w)
Gets the node width.

Parameters:
  w  The node width in pixels.

3.30.2.5  def sparksee.NodeExport.get_color_rgb (self)
Gets the node color.

Returns:
The node color.

3.30.2.6  def sparksee.NodeExport.set_font_size (self, s)
Sets the node label font size.

Parameters:
  s  [in] The node label font size.

3.30.2.7  def sparksee.NodeExport.set_height (self, h)
Sets the node height.

Parameters:
  h  [in] The node height in pixels.

3.30.2.8  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'.

3.30.2.9  def sparksee.NodeExport.set_fit (self, f)
Sets the node fit property.

Parameters:
  f  [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'.
3.30.2.10  def sparksee.NodeExport.get_labelcolor_rgb (self)
Gets the node label color.

Returns:
    The node label color.

3.30.2.11  def sparksee.NodeExport.get_width (self)
Gets the node width.

Returns:
    The node width in pixels.

3.30.2.12  def sparksee.NodeExport.set_label (self, l)
Sets the node label.

Parameters:
    l  [in] The node label.

3.30.2.13  def sparksee.NodeExport.set_color_rgb (self, c)
Sets the node color.

Parameters:
    c  The node color.

3.30.2.14  def sparksee.NodeExport.get_label (self)
Gets the node label.

Returns:
    The node label.

3.30.2.15  def sparksee.NodeExport.set_shape (self, s)
Sets the node shape.

Parameters:
    s  [in] The node shape.

3.30.2.16  def sparksee.NodeExport.get_height (self)
Gets the node height.

Returns:
    The node height in pixels.
3.31 sparksee.NodeShape Class Reference

Node shape.

Static Public Attributes

- int BOX = 0
  
  Box shape.

- int ROUND = 1
  
  Round shape.

3.31.1 Detailed Description

Node shape.

Author:

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

3.32 sparksee.NodeTypeExporter Class Reference

NodeTypeExporter class.

Inheritance diagram for sparksee.NodeTypeExporter:

Collaboration diagram for sparksee.NodeTypeExporter:

Public Member Functions

- def set_row_writer

  Sets the output data destination.

- def set_frequency

  Sets the frequency of listener notification.
• def register
  Registers a new listener.

• def run
  See the TypeExporter class Run method.

• def set_type
  Sets the type to be exported.

• def set_header
  Sets the presence of a header row.

• def __init__
  Creates a new instance.

• def set_graph
  Sets the graph that will be exported.

• def __init__
  Creates a new instance.

• def set_attributes
  Sets the list of Attributes.

### 3.32.1 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](http://www.sparsity-technologies.com)

### 3.32.2 Member Function Documentation

#### 3.32.2.1 def sparksee.NodeTypeExporter.set_row_writer (self, rw)

Sets the output data destination.

**Parameters:**

- rw null

Reimplemented from **sparksee.TypeExporter**.
3.32.2.2  def sparksee.NodeTypeExporter.set_frequency (self, freq)
Sets the frequency of listener notification.
freq [in] Frequency in number of rows managed to notify progress to all listeners

Parameters:

    freq  null

Reimplemented from sparksee.TypeExporter.

3.32.2.3  def sparksee.NodeTypeExporter.register (self, tel)
Registers a new listener.

Parameters:

    tel  [in] TypeExporterListener to be registered.

Reimplemented from sparksee.TypeExporter.

3.32.2.4  def sparksee.NodeTypeExporter.run (self)
See the TypeExporter class Run method.

Exceptions:

    IOError  null
    RuntimeError  null

Reimplemented from sparksee.TypeExporter.

3.32.2.5  def sparksee.NodeTypeExporter.set_type (self, t)
Sets the type to be exported.

Parameters:

    t  [in] Type identifier.

Reimplemented from sparksee.TypeExporter.

3.32.2.6  def sparksee.NodeTypeExporter.set_header (self, header)
Sets the presence of a header row.

Parameters:

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

Reimplemented from sparksee.TypeExporter.
3.32.2.7  def sparksee.NodeTypeExporter.__init__ (self, rw, g, t, attrs)

Creates a new instance.

Parameters:
- g   [in] Graph.
- t   [in] Type identifier.
- attrs [in] Attribute identifiers to be exported.

3.32.2.8  def sparksee.NodeTypeExporter.set_graph (self, g)

Sets the graph that will be exported.

Parameters:
- g  [in] Graph.

Reimplemented from sparksee.TypeExporter.

3.32.2.9  def sparksee.NodeTypeExporter.set_attributes (self, attrs)

Sets the list of Attributes.

Parameters:
- attrs [in] Attribute identifiers to be exported

Reimplemented from sparksee.TypeExporter.

3.33  sparksee.NodeTypeLoader Class Reference

NodeTypeLoader class.

Inheritance diagram for sparksee.NodeTypeLoader:

Collaboration diagram for sparksee.NodeTypeLoader:
Public Member Functions

- **def set_frequency**
  
  Sets the frequency of listener notification.

- **def set_log_off**
  
  Turns off all the error reporting.

- **def set_log_error**
  
  Sets a log error file.

- **def set_type**
  
  Sets the type to be loaded.

- **def run_two_phases**
  
  See the TypeLoader class RunTwoPhases method.

- **def set_graph**
  
  Sets the graph where the data will be loaded.

- **def set_row_reader**
  
  Sets the input data source.

- **def __init__**
  
  Creates a new instance.

- **def set_attribute_positions**
  
  Sets the list of attribute positions.

- **def register**
  
  Registers a new listener.

- **def __init__**
  
  Creates a new instance.

- **def set_locale**
  
  Sets the locale that will be used to read the data.

- **def run**
  
  See the TypeLoader class Run method.

- **def run_n_phases**
  
  See the TypeLoader class RunNPhases method.

- **def set_timestamp_format**
  
  Sets a specific timestamp format.

- **def set_attributes**
  
  Sets the list of Attributes.
3.33 sparksee.NodeTypeLoader Class Reference

3.33.1 Detailed Description

NodeTypeLoader class.
Specific TypeLoader implementation for node types.
Check out the ’Data import’ section in the SPARKSEE User Manual for more details on this.

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

3.33.2 Member Function Documentation

3.33.2.1 def sparksee.NodeTypeLoader.set_frequency ( self, freq)
Sets the frequency of listener notification.

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

Parameters:

freq null

Reimplemented from sparksee.TypeLoader.

3.33.2.2 def sparksee.NodeTypeLoader.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.

Reimplemented from sparksee.TypeLoader.

3.33.2.3 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:

path null

Exceptions:

IOError If bad things happen opening the file.

Reimplemented from sparksee.TypeLoader.

3.33.2.4 def sparksee.NodeTypeLoader.set_type ( self, t)
Sets the type to be loaded.

Parameters:

t [in] Type identifier.

Reimplemented from sparksee.TypeLoader.
3.33 sparksee.NodeTypeLoader Class Reference

3.33.2.5 def sparksee.NodeTypeLoader.run_two_phases (self)
See the TypeLoader class RunTwoPhases method.

Exceptions:

- IOError
- RuntimeError

Reimplemented from sparksee.TypeLoader.

3.33.2.6 def sparksee.NodeTypeLoader.set_graph (self, g)
Sets the graph where the data will be loaded.

Parameters:

- g [in] Graph.

Reimplemented from sparksee.TypeLoader.

3.33.2.7 def sparksee.NodeTypeLoader.set_row_reader (self, rr)
Sets the input data source.

Parameters:

- rr [in] Input RowReader.

Reimplemented from sparksee.TypeLoader.

3.33.2.8 def sparksee.NodeTypeLoader.set_attribute_positions (self, attrsPos)
Sets the list of attribute positions.

Parameters:

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

Reimplemented from sparksee.TypeLoader.

3.33.2.9 def sparksee.NodeTypeLoader.register (self, tel)
Registers a new listener.

Parameters:

- tel TypeLoaderListener to be registered.

Reimplemented from sparksee.TypeLoader.
3.33 sparksee.NodeTypeLoader Class Reference

3.33.2.10 def sparksee.NodeTypeLoader._init_ (self, rr, g, t, attrs, attrsPos)
Creates a new instance.

Parameters:

- **rr** [in] Input RowReader.
- **g** [in] Graph.
- **t** [in] Type identifier.
- **attrs** [in] Attribute identifiers to be loaded.
- **attrsPos** [in] Attribute positions (column index >=0).

3.33.2.11 def sparksee.NodeTypeLoader.set_locale (self, localeStr)
Sets the locale that will be used to read the data.

It should match the locale used in the rowreader.

Parameters:

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

Reimplemented from sparksee.TypeLoader.

3.33.2.12 def sparksee.NodeTypeLoader.run (self)
See the TypeLoader class Run method.

Exceptions:

- **IOError** null
- **RuntimeError** null

Reimplemented from sparksee.TypeLoader.

3.33.2.13 def sparksee.NodeTypeLoader.run_n_phases (self, partitions)
See the TypeLoader class RunNPhases method.

Parameters:

- **partitions** null

Exceptions:

- **IOError** null
- **RuntimeError** null

Reimplemented from sparksee.TypeLoader.
3.33.2.14  def sparksee.NodeTypeLoader.set_timestamp_format (self, timestampFormat)
Sets a specific timestamp format.

Parameters:

   timestampFormat  null

Reimplemented from sparksee.TypeLoader.

3.33.2.15  def sparksee.NodeTypeLoader.set_attributes (self, attrs)
Sets the list of Attributes.

Parameters:

   attrs  [in] Attribute identifiers to be loaded

Reimplemented from sparksee.TypeLoader.

3.34  sparksee.Objects Class Reference
Object identifier set class.

Public Member Functions

• def any
  Gets an element from the collection.

• def difference
  Performs the difference operation.

• def is_closed
  Gets if Objects instance has been closed or not.

• def combine_union
  Creates a new Objects instance which is the union of the two given.

• def iterator_from_element
  Gets an ObjectsIterator starting from the given element.

• def exists
  Gets if the given element exists into the collection.

• def equals
  Checks if the given Objects contains the same information.

• def combine_difference
  Creates a new Objects instance which is the difference of the two given.

• def copy
Performs the copy operation.

- **def sample**
  
  Creates a new `Objects` instance which is a sample of the calling one.

- **def close**
  
  Closes the `Objects` instance.

- **def combine_intersection**
  
  Creates a new `Objects` instance which is the intersection of the two given.

- **def contains**
  
  Checks if the objects contains the other one.

- **def count**
  
  Gets the number of elements into the collection.

- **def clear**
  
  Clears the collection removing all its elements.

- **def __iter__**
  
  Gets a new `ObjectsIterator`.

- **def iterator_from_index**
  
  Gets an `ObjectsIterator` skipping index elements.

- **def remove**
  
  Removes an element from the collection.

- **def intersection**
  
  Performs the intersection operation.

- **def iterator**
  
  Gets an `ObjectsIterator`.

- **def union**
  
  Performs the union operation.

- **def add**
  
  Adds an element into the collection.

- **def copy**
  
  Creates a new `Objects` instance as a copy of the given one.

**Static Public Attributes**

- int `INVALID_OID = 0`
  
  Invalid OID constant.
3.34.1 Detailed Description

Object identifier set class.
It stores a collection of Sparksee object identifiers as a set. As a set, there is no order and no duplicated elements.
This class should be used just to store large collections. Otherwise, it is strongly recommended to use common classes from the language API.
This class is not thread-safe.
ObjectsIterator must be used to traverse all the elements into the set.
When the Objects instance is closed, it closes all existing and non-closed ObjectsIterator instances too.

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

3.34.2 Member Function Documentation

3.34.2.1 def sparksee.Objects.any ( self)
Gets an element from the collection.

Returns:
Any element from the collection.

Exceptions:

   RuntimeError whether the collection is empty.
   RuntimeError null

3.34.2.2 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:

   objs [in] Objects instance.

Returns:
Number of elements into the collection once the operation has been executed.

3.34.2.3 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.

3.34.2.4 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:


Returns:

New Objects instance.

3.34.2.5 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:

- e [in] The first element to traverse in the resulting ObjectsIterator instance.

Returns:

ObjectsIterator instance.

3.34.2.6 def sparksee.Objects.exists (self, e)

Gets if the given element exists into the collection.

Parameters:

- e [in] Element.

Returns:

TRUE if the element exists into the collection, FALSE otherwise.

3.34.2.7 def sparksee.Objects.equals (self, objs)

Checks if the given Objects contains the same information.

Parameters:


Returns:

True if the objects are equal or false otherwise.
3.34.2.8  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:
  
  objs1 [in] Objects instance.
  objs2 [in] Objects instance.

Returns:
  New Objects instance.

3.34.2.9  def sparksee.Objects.copy ( self, objs)
Performs the copy operation.
This updates the Objects calling instance and copies the given Objects instance.

Parameters:
  
  objs [in] Objects instance.

Returns:
  Number of elements into the collection once the operation has been executed.

3.34.2.10 def sparksee.Objects.sample ( self, exclude, samples)
Creates a new Objects instance which is a sample of the calling one.

Parameters:
  
  exclude [in] If not NULL, elements into this collection will be excluded from the resulting one.
  samples [in] Number of elements into the resulting collection.

Returns:
  Sample collection.

3.34.2.11 def sparksee.Objects.close ( self)
Closes the Objects instance.
It must be called to ensure the integrity of all data.

3.34.2.12 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:
  
  objs1 [in] Objects instance.
3.34 sparksee.Objects Class Reference

`objs2` [in] Objects instance.

**Returns:**
New Objects instance.

### 3.34.2.13 def sparksee.Objects.contains (self, objs)
Check if this objects contains the other one.

**Parameters:**
- `objs` Objects collection.

**Returns:**
True if it contains the given object.

### 3.34.2.14 def sparksee.Objects.count (self)
Gets the number of elements into the collection.

**Returns:**
The number of elements into the collection.

### 3.34.2.15 def sparksee.Objects.__iter__ (self)
Gets a new ObjectsIterator.

**Returns:**
ObjectsIterator instance

### 3.34.2.16 def sparksee.Objects.iterator_from_index (self, index)
Gets an ObjectsIterator skipping index elements.

**Parameters:**
- `index` [in] The number of elements to skip from the beginning. It must be in the range [0..Size).

**Returns:**
ObjectsIterator instance.
3.34.2.17 def sparksee.Objects.remove (self, e)
Removes an element from the collection.

Parameters:
  e [in] Element to be removed.

Returns:
  TRUE if the element is removed, FALSE if the element was not into the collection.

3.34.2.18 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:
  objs [in] Objects instance.

Returns:
  Number of elements into the collection once the operation has been executed.

3.34.2.19 def sparksee.Objects.iterator (self)
Gets an ObjectsIterator.

Returns:
  ObjectsIterator instance.

3.34.2.20 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:
  objs [in] Objects instance.

Returns:
  Number of elements into the collection once the operation has been executed.

3.34.2.21 def sparksee.Objects.add (self, e)
Adds an element into the collection.

Parameters:
  e [in] Element to be added.

Returns:
  TRUE if the element is added, FALSE if the element was already into the collection.
3.34.2.22  def sparksee.Objects.copy (self)
Creates a new Objects instance as a copy of the given one.

Returns:
    The new Objects instance.

3.35  sparksee.ObjectsIterator Class Reference

ObjectsIterator class.

Public Member Functions

• def has_next
  Gets if there are more elements to traverse.

• def next
  Gets the next element to traverse.

• def is_closed
  Gets if ObjectsIterator instance has been closed or not.

• def __next__
  Used in next().

• def close
  Closes the ObjectsIterator instance.

3.35.1 Detailed Description

ObjectsIterator class.
Iterator to traverse all the object identifiers from an Objects instance.

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

3.35.2 Member Function Documentation

3.35.2.1  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.
3.35.2.2  def sparksee.ObjectsIterator.next (self)
Gets the next element to traverse.

Returns:
The next element.

3.35.2.3  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.

3.35.2.4  def sparksee.ObjectsIterator.__next__ (self)
Used in next().

Returns:
The next element

3.35.2.5  def sparksee.ObjectsIterator.close (self)
Closes the ObjectsIterator instance.
It must be called to ensure the integrity of all data.

3.36  sparksee.ObjectType Class Reference
Object type enumeration.

Static Public Attributes

- int NODE = 0
  Node object type.
- int EDGE = 1
  Edge object type.

3.36.1  Detailed Description
Object type enumeration.

Author:
Sparsity Technologies http://www.sparsity-technologies.com
3.37 sparksee.OIDList Class Reference

Sparksee object identifier list.

Public Member Functions

- `__init__`  
  Constructor.

- `add`  
  Adds a Sparksee object identifier at the end of the list.

- `clear`  
  Clears the list.

- `__iter__`  
  Gets a new TypeListIterator.

- `set`  
  Sets a Sparksee object identifier at the specified position of the list.

- `iterator`  
  Gets a new OIDListIterator.

- `count`  
  Number of elements in the list.

- `__init__`  
  Constructor.

3.37.1 Detailed Description

Sparksee object identifier list.

It stores a Sparksee object identifier list.

Use OIDListIterator to access all elements into this collection.

Author:

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

3.37.2 Member Function Documentation

3.37.2.1 `def sparksee.OIDList.__init__ (self, N)`

Constructor.

This creates a list with N invalid oids.

Parameters:

- `N` null
3.37.2.2 def sparksee.OIDList.add (self, attr)
Adds a Sparksee object identifier at the end of the list.

Parameters:
   attr [in] Sparksee object identifier.

3.37.2.3 def sparksee.OIDList.__iter__ (self)
Gets a new TypeListIterator.

Returns:
   TypeListIterator instance

3.37.2.4 def sparksee.OIDList.set (self, pos, oid)
Sets a Sparksee object identifier at the specified position of the list.

Parameters:
   pos [in] List position [0..Count()-1].
   oid [in] Sparksee object identifier.

3.37.2.5 def sparksee.OIDList.iterator (self)
Gets a new OIDListIterator.

Returns:
   OIDListIterator instance.

3.37.2.6 def sparksee.OIDList.count (self)
Number of elements in the list.

Returns:
   Number of elements in the list.

3.37.2.7 def sparksee.OIDList.__init__ (self)
Constructor.
This creates an empty list.

3.38 sparksee.OIDListIterator Class Reference
OIDList iterator class.
Public Member Functions

- def has_next
  
  Gets if there are more elements.

- def next
  
  Moves to the next element.

- def __next__
  
  Used in next().

3.38.1 Detailed Description

OIDList iterator class.

Iterator to traverse all the Sparksee object identifier into a OIDList instance.

Author:

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

3.38.2 Member Function Documentation

3.38.2.1 def sparksee.OIDListIterator.has_next (self)

Gets if there are more elements.

Returns:

TRUE if there are more elements, FALSE otherwise.

3.38.2.2 def sparksee.OIDListIterator.next (self)

Moves to the next element.

Returns:

The next element.

3.38.2.3 def sparksee.OIDListIterator.__next__ (self)

Used in next().

Returns:

The next element

3.39 sparksee.Order Class Reference

Order enumeration.
Static Public Attributes

- int ASCENDENT = 0  
  From lower to higher.

- int DESCENDENT = 1  
  From higher to lower.

3.39.1 Detailed Description

Order enumeration.

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

3.40 sparksee.Platform Class Reference

Platform class.

Public Member Functions

- def get_statistics
  
  Gets platform data and statistics.

3.40.1 Detailed Description

Platform class.

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

3.40.2 Member Function Documentation

3.40.2.1 def sparksee.Platform.get_statistics ( self, stats)

Gets platform data and statistics.

Parameters:

  stats [in|out] This updates the given PlatformStatistics.

3.41 sparksee.PlatformStatistics Class Reference

Platform data and statistics.
Public Member Functions

• def get_real_time
  
  Gets time in microseconds (since epoch).

• def get_available_mem
  
  Gets available (free) memory size in Bytes.

• def get_total_mem
  
  Gets physical memory size in Bytes.

• def get_system_time
  
  Gets CPU system time.

• def get_user_time
  
  Gets CPU user time.

• def get_num_c_p_us
  
  Gets the number of CPUs.

• def __init__
  
  Creates a new instance setting all values to 0.

3.41.1 Detailed Description

Platform data and statistics.

Author:

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

3.41.2 Member Function Documentation

3.41.2.1 def sparksee.PlatformStatistics.get_real_time (self)

Gets time in microseconds (since epoch).

Returns:

Time in microseconds (since epoch).

3.41.2.2 def sparksee.PlatformStatistics.get_available_mem (self)

Gets available (free) memory size in Bytes.

Returns:

Avialable (free) memory size in Bytes.
3.41.2.3  def sparksee.PlatformStatistics.get_total_mem (self)
Gets physical memory size in Bytes.

Returns:
    Physical memory size in Bytes.

3.41.2.4  def sparksee.PlatformStatistics.get_system_time (self)
Gets CPU system time.

Returns:
    CPU system time.

3.41.2.5  def sparksee.PlatformStatistics.get_user_time (self)
Gets CPU user time.

Returns:
    CPU user time.

3.41.2.6  def sparksee.PlatformStatistics.get_num_c_p_us (self)
Gets the number of CPUs.

Returns:
    The number of CPUs.

3.42  sparksee.RowReader Class Reference

RowReader interface.
Inheritance diagram for sparksee.RowReader:

Public Member Functions

• def close
    Closes the reader.

• def get_row
    The row number for the current row.
3.42 sparksee.RowReader Class Reference

• def reset
  
  Moves the reader to the beginning.

• def read
  
  Reads the next row as a string array.

3.42.1 Detailed Description

RowReader interface.

Common interface for those readers which get the data as an string array.

It works as follows: perform as many read operations as necessary and call close once at the end. Once close is called no more read operations can be executed.

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

Author:

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

3.42.2 Member Function Documentation

3.42.2.1 def sparksee.RowReader.close ( self)

Closes the reader.

Exceptions:

  IOError  If the close fails.

Reimplemented in sparksee.CSVReader.

3.42.2.2 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:

  IOError  If it fails.

Reimplemented in sparksee.CSVReader.

3.42.2.3 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:

IOError If bad things happen during the restart.

Reimplemented in sparksee.CSVReader.

3.42.2.4 def sparksee.RowReader.read (self, row)
Reads the next row as a string array.

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

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

Exceptions:

IOError If bad things happen during the read.

Reimplemented in sparksee.CSVReader.

3.43 sparksee.RowWriter Class Reference

RowWriter interface.

Inheritance diagram for sparksee.RowWriter:

Public Member Functions

• def write
  Writes the next row.

• def close
  Closes the writer.
3.43.1 Detailed Description

RowWriter interface.

Common interface for those writers which dump the data from an string array.

It works as follows: perform as many write operations as necessary and call close once at the end. Once close is called no more write operations can be executed.

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

Author:

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

3.43.2 Member Function Documentation

3.43.2.1 def sparksee.RowWriter.write (self, row)
Writes the next row.

Parameters:

row [in] Row of data.

Exceptions:

IOError If bad things happen during the write.

RuntimeError null

Reimplemented in sparksee.CSVWriter.

3.43.2.2 def sparksee.RowWriter.close (self)
Closes the writer.

Exceptions:

IOError If the close fails.

RuntimeError null

Reimplemented in sparksee.CSVWriter.

3.44 sparksee.ScriptParser Class Reference

ScriptParser.

Public Member Functions

• def parse
    Parses the given input file.

• def set_error_log
    Sets the error log.
• def __init__
  Constructor:

• def generate_schema_script
  Writes an script with the schema definition for the given database.

• def set_output_log
  Sets the output log.

3.44.1 Detailed Description

ScriptParser.
The ScriptParser can create schemas and load data from a set of commands in a sparksee script.
A SPARKSEE script contains an ordered list of commands. ScriptParser will execute each one of them in order. Commands may create schemas, define nodes and edges, and load data into a previous defined SPARKSEE schema.
Check out the 'Scripting' chapter in the SPARKSEE User Manual for a comprehensive explanation on the grammar of the SPARKSEE commands and how they work.

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

3.44.2 Member Function Documentation

3.44.2.1 def sparksee.ScriptParser.parse (self, path, execute, localeStr)
Parses the given input file.

Parameters:
  path [in] Input file path.
  execute [in] If TRUE the script is executed, if FALSE it is just parsed.
  localeStr [in] The locale string for reading the input file. See CSVReader.

Returns:
  TRUE if ok, FALSE in case of error.

Exceptions:
  IOError If bad things happen opening the file.

3.44.2.2 def sparksee.ScriptParser.set_error_log (self, path)
Sets the error log.
If not set, error log corresponds to standard error output.
Parameters:

path  [in] Path of the error log.

Exceptions:

IOException If bad things happen opening the file.

3.44.2.3  def sparksee.ScriptParser.generate_schema_script ( self, path, db)

Writes an script with the schema definition for the given database.

Parameters:

path  [in] Filename of the script to be writen.

db  [in] Database.

Exceptions:

IOException If bad things happen opening or writing the file.

3.44.2.4  def sparksee.ScriptParser.set_output_log ( self, path)

Sets the output log.
If not set, output log corresponds to standard output.

Parameters:

path  [in] Path of the output log.

Exceptions:

IOException If bad things happen opening the file.

3.45  sparksee.Session Class Reference

Session class.

Public Member Functions

•  def is_closed

  Gets if Session instance has been closed or not.

•  def commit

  Commits a transaction.

•  def get_graph

  Gets the Graph instance.

•  def close

  Closes the Session instance.
• **def begin**
  
  *Begins a transaction.*

• **def new_objects**
  
  *Creates a new Objects instance.*

### 3.45.1 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](http://www.sparsity-technologies.com)

### 3.45.2 Member Function Documentation

#### 3.45.2.1 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.

#### 3.45.2.2 def sparksee.Session.get_graph (self)

Gets the **Graph** instance.

**Returns:**

The **Graph** instance.
3.45.2.3  def sparksee.Session.close (self)
Closes the Session instance.
It must be called to ensure the integrity of all data.

3.45.2.4  def sparksee.Session.new_objects (self)
Creates a new Objects instance.

Returns:
The new Objects instance.

3.46  sparksee.ShortestPath Class Reference

ShortestPath class.
Inheritance diagram for sparksee.ShortestPath:

Public Member Functions

• def add_edge_type
  Allows for traversing edges of the given type.

• def add_all_node_types
  Allows for traversing all node types of the graph.

• def run
  Runs the algorithm.

• def exclude_nodes
  Set which nodes can’t be used.

• def is_closed
  Gets if ShortestPath has been closed or not.

• def add_node_type
  Allows for traversing nodes of the given type.
• def add_all_edge_types
  Allows for traversing all edge types of the graph.

• def exclude_edges
  Set which edges can’t be used.

• def set_maximum_hops
  Sets the maximum hops restriction.

• def close
  Closes the ShortestPath instance.

3.46.1 Detailed Description

ShortestPath class.
Classes implementing this abstract class solve the shortest path problem in a graph.
The user must set which node and edge types can be used for the traversal.
Check out the ‘Algorithms’ section in the SPARKSEE User Manual for more details on this.

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

3.46.2 Member Function Documentation

3.46.2.1 def sparksee.ShortestPath.add_edge_type ( self, t, d)
Allows for traversing edges of the given type.

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


3.46.2.2 def sparksee.ShortestPath.run ( self)
Runs the algorithm.
This method can only be called once.
3.46.2.3 `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:

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


3.46.2.4 `def sparksee.ShortestPath.is_closed (self)`
Gets if `ShortestPath` has been closed or not.

See also:
```python
close()
```

Returns:

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

3.46.2.5 `def sparksee.ShortestPath.add_node_type (self, t)`
Allows for traversing nodes of the given type.

Parameters:

- `t` null


3.46.2.6 `def sparksee.ShortestPath.add_all_edge_types (self, d)`
Allows for traversing all edge types of the graph.

Parameters:

- `d` [in] Edge direction.


3.46.2.7 `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:

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


3.46.2.8 def sparksee.ShortestPath.set_maximum_hops (self, maxhops)
Sets the maximum hops restriction.
All paths longer than the maximum hops restriction will be ignored. m[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

Parameters:

maxhops null


3.46.2.9 def sparksee.ShortestPath.close (self)
Closes the ShortestPath instance.
It must be called to ensure the integrity of all data.

3.47 sparksee.SinglePairShortestPath Class Reference

SinglePairShortestPath class.
Inheritance diagram for sparksee.SinglePairShortestPath:

Collaboration diagram for sparksee.SinglePairShortestPath:
Public Member Functions

• def exists
  
  Returns TRUE if a path exists or FALSE otherwise.

• def add_edge_type
  
  Allows for traversing edges of the given type.

• def exclude_nodes
  
  Set which nodes can’t be used.

• def run
  
  Runs the algorithm.

• def add_all_edge_types
  
  Allows for traversing all edge types of the graph.

• def add_node_type
  
  Allows for traversing nodes of the given type.

• def get_path_as_edges
  
  Gets the shortest path between the source node and the destination node as an ordered set of edges.

• def set_maximum_hops
  
  Sets the maximum hops restriction.

• def add_all_node_types
  
  Allows for traversing all node types of the graph.

• def get_cost
  
  Gets the cost of the shortest path.

• def get_path_as_nodes
  
  Gets the shortest path between the source node and the destination node as an ordered set of nodes.

• def exclude_edges
  
  Set which edges can’t be used.

• def is_closed
  
  Gets if ShortestPath has been closed or not.

• def close
  
  Closes the ShortestPath instance.
3.47 sparksee.SinglePairShortestPath Class Reference

3.47.1 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](http://www.sparsity-technologies.com)

3.47.2 Member Function Documentation

3.47.2.1 def sparksee.SinglePairShortestPath.add_edge_type (self, t, d)

Allows for traversing edges of the given type.

Parameters:

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

Reimplemented from sparksee.ShortestPath.

3.47.2.2 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:

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

Reimplemented from sparksee.ShortestPath.

3.47.2.3 def sparksee.SinglePairShortestPath.run (self)

Runs the algorithm.

This method can only be called once.

Reimplemented from sparksee.ShortestPath.

3.47.2.4 def sparksee.SinglePairShortestPath.add_all_edge_types (self, d)

Allows for traversing all edge types of the graph.

Parameters:

- d [in] Edge direction.
Reimplemented from `sparksee.ShortestPath`.

### 3.47.2.5 `def sparksee.SinglePairShortestPath.add_node_type (self, t)`

Allows for traversing nodes of the given type.

**Parameters:**

- `t` null

Reimplemented from `sparksee.ShortestPath`.

### 3.47.2.6 `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.


### 3.47.2.7 `def sparksee.SinglePairShortestPath.set_maximum_hops (self, maxhops)`

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored. m[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

**Parameters:**

- `maxhops` null

Reimplemented from `sparksee.ShortestPath`.

### 3.47.2.8 `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.

3.47.2.9 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.


3.47.2.10 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:

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

Reimplemented from sparksee.ShortestPath.


3.47.2.11 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.

3.47.2.12 def sparksee.ShortestPath.close ( self) [inherited]

Closes the ShortestPath instance.

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

3.48 sparksee.SinglePairShortestPathBFS Class Reference

SinglePairShortestPathBFS class.

Inheritance diagram for sparksee.SinglePairShortestPathBFS:
Public Member Functions

- def exists
  Returns TRUE if a path exists or FALSE otherwise.

- def add_edge_type
  Allows for traversing edges of the given type.

- def exclude_nodes
  Set which nodes can’t be used.

- def get_path_as_edges
  Gets the shortest path between the source node and the destination node as an ordered set of edges.

- def get_path_as_nodes
  Gets the shortest path between the source node and the destination node as an ordered set of nodes.

- def add_all_edge_types
  Allows for traversing all edge types of the graph.

- def add_node_type
  Allows for traversing nodes of the given type.

- def set_maximum_hops
  Sets the maximum hops restriction.

- def get_cost
  Gets the cost of the shortest path.

- def add_all_node_types
  Allows for traversing all node types of the graph.

- def run
  Executes the algorithm.

- def __init__
  Creates a new instance.

- def check_only_existence
Set that only the path existence must be calculated and not the path itself.

- **def exclude_edges**
  
  Set which edges can’t be used.

- **def is_closed**
  
  Gets if ShortestPath has been closed or not.

- **def close**
  
  Closes the ShortestPath instance.

### 3.48.1 Detailed Description

**SinglePairShortestPathBFS** class.

It solves the single-pair shortest path problem using a BFS-based implementation. It is an 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](http://www.sparsity-technologies.com)

### 3.48.2 Member Function Documentation

#### 3.48.2.1 **def sparksee.SinglePairShortestPathBFS.add_edge_type (self, t, d)**

Allows for traversing edges of the given type.

**Parameters:**

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

Reimplemented from `sparksee.SinglePairShortestPath`.

#### 3.48.2.2 **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:**

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

Reimplemented from `sparksee.SinglePairShortestPath`.
3.48.2.3  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.

Reimplemented from sparksee.SinglePairShortestPath.

3.48.2.4  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.

Reimplemented from sparksee.SinglePairShortestPath.

3.48.2.5  def sparksee.SinglePairShortestPathBFS.add_all_edge_types ( self, d)

Allows for traversing all edge types of the graph.

Parameters:
   d [in] Edge direction.

Reimplemented from sparksee.SinglePairShortestPath.

3.48.2.6  def sparksee.SinglePairShortestPathBFS.add_node_type ( self, t)

Allows for traversing nodes of the given type.

Parameters:
   t null

Reimplemented from sparksee.SinglePairShortestPath.

3.48.2.7  def sparksee.SinglePairShortestPathBFS.set_maximum_hops ( self, maxhops)

Sets the maximum hops restriction.

All paths longer than the maximum hops restriction will be ignored. m[in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

Parameters:
   maxhops null

Reimplemented from sparksee.SinglePairShortestPath.
3.48.2.8  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.

Reimplemented from sparksee.SinglePairShortestPath.

3.48.2.9  def sparksee.SinglePairShortestPathBFS.__init__ ( self, s, src, dst)
Creates a new instance.

**Parameters:**

  s  [in]  Session to get the graph from and perform traversal.
  src  [in]  Source node.
  dst  [dst]  Destination node.

3.48.2.10  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.

3.48.2.11  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:**

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

Reimplemented from sparksee.SinglePairShortestPath.

3.48.2.12  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.
3.49 sparksee.SinglePairShortestPathDijkstra Class Reference

3.48.2.13 def sparksee.ShortestPath.close (self) [inherited]
Closes the ShortestPath instance.
It must be called to ensure the integrity of all data.

3.49 sparksee.SinglePairShortestPathDijkstra Class Reference

SinglePairShortestPathDijkstra class.
Inheritance diagram for sparksee.SinglePairShortestPathDijkstra:

Collaboration diagram for sparksee.SinglePairShortestPathDijkstra:

Public Member Functions

• def exists
  Returns TRUE If a path exists or FALSE otherwise.

• def add_edge_type
  Allows for traversing edges of the given type.

• def exclude_nodes
  Set which nodes can’t be used.

• def get_path_as_edges
  Gets the shortest path between the source node and the destination node as an ordered set of edges.

• def get_path_as_nodes
  Gets the shortest path between the source node and the destination node as an ordered set of nodes.

• def add_weighted_edge_type
  Allows for traversing edges of the given type using the given attribute as the weight.
• def set_unweighted_edge_cost
   Sets the weight assigned to the unweighted edges.

• def add_all_edge_types
   Allows for traversing all edge types of the graph.

• def add_node_type
   Allows for traversing nodes of the given type.

• def set_maximum_hops
   Sets the maximum hops restriction.

• def get_cost
   Gets the cost of the shortest path.

• def add_all_node_types
   Allows for traversing all node types of the graph.

• def run
   Executes the algorithm.

• def exclude_edges
   Set which edges can’t be used.

• def __init__
   Creates a new instance.

• def is_closed
   Gets if ShortestPath has been closed or not.

• def close
   Closes the ShortestPath instance.

3.49.1 Detailed Description

SinglePairShortestPathDijkstra class.
It solves the single-pair shortest path problem using a Dijkstra-based implementation.
It is a weighted algorithm, so it takes into account the cost of the edges to compute a minimum-cost shortest path. That is, the user may set for each edge type which attribute should be used to retrieve the cost of the edge. If no attribute is given for an edge type, this will assume the edge has a fixed cost (the default is 1). Only numerical attribute can be set as weight attributes (that is Long, Integer or Double attributes are allowed).
Check out the ‘Algorithms’ section in the SPARKSEE User Manual for more details on this.

Author:

Sparsity Technologies http://www.sparsity-technologies.com
3.49.2 Member Function Documentation

3.49.2.1 `def sparksee.SinglePairShortestPathDijkstra.add_edge_type ( self, t, d)`
 Allows for traversing edges of the given type.

**Parameters:**
- `t` [in] Edge type.
- `d` [in] Edge direction.

Reimplemented from `sparksee.SinglePairShortestPath`.

3.49.2.2 `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:**
- `nodes` [in] A set of node identifiers that must be kept intact until the destruction of the class.

Reimplemented from `sparksee.SinglePairShortestPath`.

3.49.2.3 `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.

Reimplemented from `sparksee.SinglePairShortestPath`.

3.49.2.4 `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.

Reimplemented from `sparksee.SinglePairShortestPath`.

3.49.2.5 `def sparksee.SinglePairShortestPathDijkstra.add_weighted_edge_type (self, t, d, attr)`
 Allows for traversing edges of the given type using the given attribute as the weight.

**Parameters:**
- `t` [in] Edge type.
- `d` [in] Edge direction.
- `attr` [in] Attribute to be used as the weight. It must be a global attribute or an attribute of the given edge type.
3.49.2.6  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:
   weight  null

3.49.2.7  def sparksee.SinglePairShortestPathDijkstra.add_all_edge_types (self, d)
Allows for traversing all edge types of the graph.
Parameters:
   d  [in] Edge direction.
Reimplemented from sparksee.SinglePairShortestPath.

3.49.2.8  def sparksee.SinglePairShortestPathDijkstra.add_node_type (self, t)
Allows for traversing nodes of the given type.
Parameters:
   t  null
Reimplemented from sparksee.SinglePairShortestPath.

3.49.2.9  def sparksee.SinglePairShortestPathDijkstra.set_maximum_hops (self, maxhops)
Sets the maximum hops restriction.
All paths longer than the maximum hops restriction will be ignored. m[in] The maximum hops restriction.
It must be positive or zero. Zero, the default value, means unlimited.
Parameters:
   maxhops  null
Reimplemented from sparksee.SinglePairShortestPath.

3.49.2.10 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.
Reimplemented from sparksee.SinglePairShortestPath.
3.49.2.11 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:

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

Reimplemented from sparksee.SinglePairShortestPath.

3.49.2.12 def sparksee.SinglePairShortestPathDijkstra.__init__ (self, s, src, dst)
Creates a new instance.

Parameters:

s [in] Session to get the graph from and perform traversal.
src [in] Source node.
dst [dst] Destination node.

3.49.2.13 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.

3.49.2.14 def sparksee.ShortestPath.close (self) [inherited]
Closes the ShortestPath instance.
It must be called to ensure the integrity of all data.

3.50 sparksee.Sparksee Class Reference
Sparksee class.

Public Member Functions

- def create
  Creates a new Database instance.

- def restore
  Restores a Database from a backup file.
• def __init__
  Creates a new instance.

• def close
  Closes the Sparksee instance.

• def open
  Opens an existing Database instance.

Static Public Attributes

• float VERSION = 5.0
  Sparksee version.

3.50.1 Detailed Description

Sparksee class.
All Sparksee programs must have one single Sparksee instance to manage one or more Database instances. This class allows for the creation of new Databases or open an existing one.

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

3.50.2 Member Function Documentation

3.50.2.1 def sparksee.Sparksee.create (self, path, alias)
Creates a new Database instance.

Parameters:
  alias [in] Database alias name.

Returns:
  A Database instance.

Exceptions:
  IOError If the given file cannot be created.
  RuntimeError null

3.50.2.2 def sparksee.Sparksee.restore (self, path, backupFile)
Restores a Database from a backup file.
See the Graph class Backup method.
Parameters:

- **path** [in] Database storage file.
- **backupFile** [in] The Backup file to be restored.

Returns:

A Database instance.

Exceptions:

- **IOException** If the given file cannot be created, or the exported data file does not exist.
- **RuntimeError** null

### 3.50.2.3 def sparksee.Sparksee.__init__(self, config)

Creates a new instance.

Parameters:

- **config** [in] Sparksee configuration.

### 3.50.2.4 def sparksee.Sparksee.close(self)

Closes the Sparksee instance.

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

### 3.50.2.5 def sparksee.Sparksee.open(self, path, read)

Opens an existing Database instance.

Parameters:

- **path** [in] Database storage file.
- **read** [in] If TRUE, open Database in read-only mode.

Returns:

A Database instance.

Exceptions:

- **IOException** If the given file does not exist.
- **RuntimeError** null

### 3.51 sparksee.SparkseeConfig Class Reference

Sparksee configuration class.
Public Member Functions

- `def get_high_availability_synchronization
  Gets the synchronization polling time.

- `def set_recovery_cache_max_size
  Sets the maximum size for the recovery log cache in extents.

- `def set_high_availability_i_p
  Sets the IP address and port of the instance.

- `def set_pool_persistent_min_size
  Sets the minimum size for the persistent pool in number of frames.

- `def get_extent_size
  Gets the size of an extent.

- `def set_high_availability_enabled
  Enables or disables high availability mode.

- `def set_high_availability_coordinators
  Sets the coordinators address and port list.

- `def get_log_file
  Gets the log file.

- `def get_recovery_checkpoint_time
  Gets the delay time (in microseconds) between automatic checkpoints.

- `def set_cache_statistics_snapshot_time
  Sets the cache statistics snapshot time.

- `def get_cache_statistics_enabled
  Gets whether cache statistics are enabled or disabled.

- `def set_pool_persistent_max_size
  Gets the maximum size for the persistent pool in number of frames.

- `def set_log_level
  Sets the log level.

- `def set_extent_size
  Sets the size of a pool frame in number of extents.

- `def get_recovery_log_file
  Gets the recovery log file.

- `def get_extent_pages
  Gets the number of pages per extent.
• def set_pool_persistent_max_size
  Sets the maximum size for the persistent pool in number of frames.

• def set_cache_max_size
  Sets the maximum size for the cache (all pools) in MB.

• def get_high_availability_enabled
  Gets whether high availability mode is enabled or disabled.

• def set_recovery_log_file
  Sets the recovery log file.

• def get_recovery_enabled
  Gets whether the recovery is enabled or disabled.

• def __init__
  Creates a new instance.

• def get_high_availability_coordinators
  Gets the coordinators address and port list.

• def set_license
  Sets the license code.

• def set_recovery_enabled
  Enables or disables the recovery.

• def set_pool_temporary_max_size
  Sets the maximum size for the temporary pool in number of frames.

• def set_high_availability_synchronization
  Sets the synchronization polling time.

• def set_pool_frame_size
  Sets the size of a pool frame in number of extents.

• def get_cache_statistics_file
  Gets the cache statistics log file.

• def set_cache_statistics_file
  Sets the cache statistics log file.

• def set_pool_frame_size
  Gets the size of a pool frame in number of extents.

• def get_cache_max_size
  Gets the maximum size for the cache (all pools) in MB.

• def get_cache_statistics_snapshot_time
  Gets the cache statistics snapshot time in microseconds.
• def get_pool_temporary_max_size
  Gets the maximum size for the temporary pool in number of frames.

• def set_extent_pages
  Sets the number of pages per extent.

• def set_pool_temporary_min_size
  Sets the minimum size for the temporary pool in number of frames.

• def get_license
  Gets the license code.

• def get_pool_temporary_min_size
  Gets the minimum size for the temporary pool in number of frames.

• def set_high_availability_master_history
  Sets the master’s history log.

• def set_cache_statistics_enabled
  Enables or disables cache statistics.

• def get_recovery_cache_max_size
  Gets the maximum size for the recovery log cache in extents.

• def get_high_availability_master_history
  Gets the master’s history log.

• def get_high_availability_i_p
  Gets the IP address and port of the instance.

• def get_log_level
  Gets the log level.

• def set_recovery_checkpoint_time
  Sets the delay time (in microseconds) between automatic checkpoints.

• def set_log_file
  Sets the log file.

• def get_pool_persistent_min_size
  Gets the minimum size for the persistent pool in number of frames.

3.51.1 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 SparkseeProperties 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).
- **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).
- **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 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|m|s|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.
Author:
Sparsity Technologies http://www.sparsity-technologies.com

3.51.2 Member Function Documentation

3.51.2.1 def sparksee.SparkseeConfig.get_high_availability_synchronization (self)
Gets the synchronization polling time.

Returns:
The Synchronization polling time.

3.51.2.2 def sparksee.SparkseeConfig.set_recovery_cache_max_size (self, v)
Sets the maximum size for the recovery log cache in extents.

Parameters:

v [in] The maximum size for the recovery log cache in extents. A 0 sets the default value (extents up to 1MB).

3.51.2.3 def sparksee.SparkseeConfig.set_high_availability_i_p (self, v)
Sets the IP address and port of the instance.

Parameters:

v [in] The IP address and port of the instance.

3.51.2.4 def sparksee.SparkseeConfig.set_pool_persistent_min_size (self, v)
Sets the minimum size for the persistent pool in number of frames.

Parameters:

v [in] The minimum size for the persistent pool in number of frames. It must be non-negative.

3.51.2.5 def sparksee.SparkseeConfig.get_extent_size (self)
Gets the size of a extent.

Returns:
The size of a extent in KB.

3.51.2.6 def sparksee.SparkseeConfig.set_high_availability_enabled (self, v)
Enables or disables high availability mode.

Parameters:

v [in] If TRUE this enables high availability mode, if FALSE this disables high availability mode.
3.51.2.7 `def sparksee.SparkseeConfig.set_high_availability_coordinators (self, v)`
Sets the coordinators address and port list.

**Parameters:**

- `v` [in] The coordinators address and port list.

3.51.2.8 `def sparksee.SparkseeConfig.get_log_file (self)`
Gets the log file.

**Returns:**

The log file.

3.51.2.9 `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.

3.51.2.10 `def sparksee.SparkseeConfig.set_cache_statistics_snapshot_time (self, v)`
Sets the cache statistics snapshot time.
Useless if cache statistics are disabled.

**Parameters:**

- `v` [in] The cache statistics snapshot time in microseconds.

3.51.2.11 `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.

3.51.2.12 `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.
3.51.2.13 `def sparksee.SparkseeConfig.set_log_level (self, v)`
Sets the log level.

**Parameters:**

$v$ [in] The `LogLevel`.

3.51.2.14 `def sparksee.SparkseeConfig.set_extent_size (self, v)`
Sets the size of a pool frame in number of extents.

**Parameters:**

$v$ [in] The size of a extent size in KB. An extent can have a size between 4KB and 64KB, and it must be a power of 2.

3.51.2.15 `def sparksee.SparkseeConfig.get_recovery_log_file (self)`
Gets the recovery log file.

**Returns:**

The recovery log file.

3.51.2.16 `def sparksee.SparkseeConfig.get_extent_pages (self)`
Gets the number of pages per extent.

**Returns:**

The number of pages per extent.

3.51.2.17 `def sparksee.SparkseeConfig.set_pool_persistent_max_size (self, v)`
Sets the maximum size for the persistent pool in number of frames.

**Parameters:**

$v$ [in] The maximum size for the persistent pool in number of frames. It must be non-negative.

3.51.2.18 `def sparksee.SparkseeConfig.set_cache_max_size (self, v)`
Sets the maximum size for the cache (all pools) in MB.

**Parameters:**

$v$ [in] The maximum size for the cache (all pools) in MB. It must be non-negative.
3.51.2.19  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.

3.51.2.20  def sparksee.SparkseeConfig.set_recovery_log_file (self, v)  
Sets the recovery log file.  

Parameters:  
  
  v [in] The recovery log file. Left it empty for the default log file (same as <database_file_name>.log)

3.51.2.21  def sparksee.SparkseeConfig.get_recovery_enabled (self)  
Gets whether the recovery is enabled or disabled.  

Returns:  
TRUE if the recovery is enabled, FALSE otherwise.

3.51.2.22  def sparksee.SparkseeConfig.__init__ (self)  
Creates a new instance.  

Values are set with default values.

3.51.2.23  def sparksee.SparkseeConfig.get_high_availability_COORDINATORS (self)  
Gets the coordinators address and port list.  

Returns:  
The coordinators address and port list.

3.51.2.24  def sparksee.SparkseeConfig.set_license (self, v)  
Sets the license code.  

Parameters:  
  
  v [in] The license code.

3.51.2.25  def sparksee.SparkseeConfig.set_recovery_enabled (self, v)  
Enables or disables the recovery.  

Parameters:  
  
  v [in] If TRUE this enables the recovery, if FALSE then disables it.
3.51.2.26  def sparksee.SparkseeConfig.set_pool_temporary_max_size (self, v)
Sets the maximum size for the temporary pool in number of frames.

Parameters:

   v [in] The maximum size for the temporary pool in number of frames. It must be non-negative.

3.51.2.27  def sparksee.SparkseeConfig.set_high_availability_synchronization (self, v)
Sets the synchronization polling time.

Parameters:

   v [in] The synchronization polling time.

3.51.2.28  def sparksee.SparkseeConfig.set_pool_frame_size (self, v)
Sets the size of a pool frame in number of extents.

Parameters:

   v [in] The size of a pool frame in number of extents. It must be non-negative.

3.51.2.29  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.

3.51.2.30  def sparksee.SparkseeConfig.set_cache_statistics_file (self, v)
Sets the cache statistics log file.
Useless if cache statistics are disabled.

Parameters:


3.51.2.31  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.
3.51.2.32  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.

3.51.2.33  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.

3.51.2.34  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.

3.51.2.35  def sparksee.SparkseeConfig.set_extent_pages (self, v)
Sets the number of pages per extent.

Parameters:
   v [in] The number of pages. It must be at least 1 page and the page size must be greater than or equal to 4KB.

3.51.2.36  def sparksee.SparkseeConfig.set_pool_temporary_min_size (self, v)
Sets the minimum size for the temporary pool in number of frames.

Parameters:
   v [in] The minimum size for the temporary pool in number of frames. It must be non-negative.

3.51.2.37  def sparksee.SparkseeConfig.get_license (self)
Gets the license code.

Returns:
   The license code.
3.51.2.38  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.

3.51.2.39  def sparksee.SparkseeConfig.set_high_availability_master_history (self, v)
Sets the master's history log.

Parameters:
   v [in] The master's history log.

3.51.2.40  def sparksee.SparkseeConfig.set_cache_statistics_enabled (self, v)
Enables or disables cache statistics.

Parameters:
   v [in] If TRUE this enables cache statistics, if FALSE this disables cache statistics.

3.51.2.41  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.

3.51.2.42  def sparksee.SparkseeConfig.get_high_availability_master_history (self)
Gets the master's history log.

Returns:
   The master's history log.

3.51.2.43  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.
3.51.2.44  def sparksee.SparkseeConfig.get_log_level (self)
Gets the log level.

Returns:
    The LogLevel.

3.51.2.45  def sparksee.SparkseeConfig.set_recovery_checkpoint_time (self, v)
Sets the delay time (in microseconds) between automatic checkpoints.

Parameters:
    v [in] The delay time (in microseconds) between automatic checkpoints. A 0 forces a checkpoint after each committed transaction.

3.51.2.46  def sparksee.SparkseeConfig.set_log_file (self, v)
Sets the log file.

Parameters:

3.51.2.47  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.

3.52  sparksee.SparkseeProperties Class Reference

Sparksee properties file.

Public Member Functions

- def get
  Gets a property.

- def get_time_unit
  Gets a property as a time unit.

- def get_boolean
  Gets a property as a boolean.

- def load
  Loads properties from the given file path.

- def get_integer
  Gets a property as an integer.
3.52.1 Detailed Description

Sparksee properties file.

This class is implemented as a singleton, so all public methods are static.

It allows for getting the property values stored in a properties file. A properties file is a file where there is one line per property. A property is defined by a key and a value as follows: key=value

By default, this loads properties from the file './sparksee.cfg'. The user may choose to load a different file by calling the method Load().

If the default properties file or the one loaded by the user do not exist, then this behaves as loading an empty properties file.

3.52.2 Member Function Documentation

3.52.2.1 def sparksee.SparkseeProperties.get ( self, key, def)

Gets a property.

Parameters:

- **key** [in] The name of the property to lookup.
- **def** [in] Default value to be returned in case there is no property with the name key.

Returns:

- The value of the property, or def if the key is not found.

3.52.2.2 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:

- **key** [in] The name of the property to lookup.
- **def** [in] The default value (in microseconds) to be returned in case there is no property with the name key.

Returns:

- The time duration in microseconds, or def if the key is not found or in case of error.

3.52.2.3 def sparksee.SparkseeProperties.get_boolean ( self, key, def)

Gets a property as a boolean.
Parameters:
   key [in] The name of the property to lookup.
   def [in] Default value to be returned in case there is no property with the name key.

Returns:
   The property value, or def if the key is not found or in case of error.

3.52.2.4 def sparksee.SparkseeProperties.load (self, path)
Loads properties from the given file path.

Parameters:
   path [in] File path to load properties from.

3.52.2.5 def sparksee.SparkseeProperties.get_integer (self, key, def)
Gets a property as an integer.

Parameters:
   key [in] The name of the property to lookup.
   def [in] Default value to be returned in case there is no property with the name key.

Returns:
   The property value, or def if the key is not found or in case of error.

3.53 sparksee.StringList Class Reference

String list.

Public Member Functions
   • def clear
     Clears the list.
   • def __init__
     Constructor.
   • def __iter__
     Gets a new TypeListIterator.
   • def iterator
     Gets a new StringListIterator.
   • def count
     Number of elements in the list.
   • def add
     Adds a String at the end of the list.
3.53.1 Detailed Description

String list.
It stores a String (unicode) list.
Use StringListIterator to access all elements into this collection.

Author:

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

3.53.2 Member Function Documentation

3.53.2.1 def sparksee.StringList.__init__ (self)
Constructor.
This creates an empty list.

3.53.2.2 def sparksee.StringList.__iter__ (self)
Gets a new TypeListIterator.

Returns:
TypeListIterator instance

3.53.2.3 def sparksee.StringList.iterator (self)
Gets a new StringListIterator.

Returns:
StringListIterator instance.

3.53.2.4 def sparksee.StringList.count (self)
Number of elements in the list.

Returns:
Number of elements in the list.

3.53.2.5 def sparksee.StringList.add (self, str)
Adds a String at the end of the list.

Parameters:

3.54 sparksee.StringListIterator Class Reference

StringList iterator class.
Public Member Functions

- def has_next
  Gets if there are more elements.

- def next
  Moves to the next element.

- def __next__
  Used in next().

3.54.1 Detailed Description

StringList iterator class.
Iterator to traverse all the strings into a StringList instance.

Author:
Sparsity Technologies [http://www.sparsity-technologies.com](http://www.sparsity-technologies.com)

3.54.2 Member Function Documentation

3.54.2.1 def sparksee.StringListIterator.has_next (self)
Gets if there are more elements.

Returns:
TRUE if there are more elements, FALSE otherwise.

3.54.2.2 def sparksee.StringListIterator.next (self)
Moves to the next element.

Returns:
The next element.

3.54.2.3 def sparksee.StringListIterator.__next__ (self)
Used in next().

Returns:
The next element
3.55 sparksee.StrongConnectivity Class Reference

StrongConnectivity class.

Inheritance diagram for sparksee.StrongConnectivity:

Collaboration diagram for sparksee.StrongConnectivity:

Public Member Functions

- def add_edge_type
  Allows connectivity through edges of the given type.

- def add_all_node_types
  Allows connectivity through all node types of the graph.

- def run
  Runs the algorithm in order to find the connected components.

- def exclude_nodes
  Set which nodes can’t be used.

- def add_node_type
  Allows connectivity through nodes of the given type.

- def add_all_edge_types
  Allows connectivity through all edge types of the graph.

- def set_materialized_attribute
  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 exclude_edges
  Set which edges can’t be used.

- def get_connected_components
Returns the results generated by the execution of the algorithm.

- **def is_closed**
  
  Gets if Connectivity has been closed or not.

- **def close**
  
  Closes the Connectivity instance.

### 3.55.1 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](http://www.sparsity-technologies.com)

### 3.55.2 Member Function Documentation

#### 3.55.2.1 **def** sparksee.StrongConnectivity.add_edge_type (self, t, d)

Allows connectivity through edges of the given type.

**Parameters:**

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

Reimplemented in **sparksee.StrongConnectivityGabow**.

#### 3.55.2.2 **def** sparksee.StrongConnectivity.run (self)

Runs the algorithm in order to find the connected components.

This method can be called only once.

Reimplemented from **sparksee.Connectivity**.

Reimplemented in **sparksee.StrongConnectivityGabow**.
3.55.2.3  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:

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

Reimplemented from sparksee.Connectivity.
Reimplemented in sparksee.StrongConnectivityGabow.

3.55.2.4  def sparksee.StrongConnectivity.add_node_type (self, t)
Allows connectivity through nodes of the given type.

Parameters:

   t  null

Reimplemented from sparksee.Connectivity.
Reimplemented in sparksee.StrongConnectivityGabow.

3.55.2.5  def sparksee.StrongConnectivity.add_all_edge_types (self, d)
Allows connectivity through all edge types of the graph.

Parameters:

   d  [in] Edge direction.

Reimplemented in sparksee.StrongConnectivityGabow.

3.55.2.6  def sparksee.StrongConnectivity.set_materialized_attribute (self, attributeName)
Creates a new common attribute type for all node types in the graph in order to store, persistently, the
results related to the connected components found while executing this algorithm.
Whenever the user wants to retrieve the results, even when the graph has been closed and opened again,
it is only necessary to create a new instance of the class ConnectedComponents indicating the graph and
the name of the common attribute type which stores the results. This instance will have all the information
related to the connected components found in the moment of the execution of the algorithm that stored this
data.
It is possible to run the algorithm without specifying this parameter in order to avoid materializing the
results of the execution.

Parameters:

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

Reimplemented from sparksee.Connectivity.
Reimplemented in sparksee.StrongConnectivityGabow.
3.55.2.7  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:

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

Reimplemented from sparksee.Connectivity.
Reimplemented in sparksee.StrongConnectivityGabow.

3.55.2.8  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.

Reimplemented from sparksee.Connectivity.
Reimplemented in sparksee.StrongConnectivityGabow.

3.55.2.9  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.

3.55.2.10  def sparksee.Connectivity.close (self)  [inherited]
Closes the Connectivity instance.
It must be called to ensure the integrity of all data.

3.56  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:

```
sparksee.StrongConnectivityGabow
  sparksee.StrongConnectivity
  sparksee.Connectivity
```

Collaboration diagram for sparksee.StrongConnectivityGabow:

```
sparksee.StrongConnectivityGabow
  sparksee.StrongConnectivity
  sparksee.Connectivity
```

**Public Member Functions**

- def add_edge_type
  
  Allows connectivity through edges of the given type.

- def add_all_node_types
  
  Allows connectivity through all node types of the graph.

- def exclude_nodes
  
  Set which nodes can’t be used.

- def run
  
  Executes the algorithm.

- def add_node_type
  
  Allows connectivity through nodes of the given type.

- def add_all_edge_types
  
  Allows connectivity through all edge types of the graph.

- def __init__
  
  Creates a new instance of StrongConnectivityGabow.

- def set_materialized_attribute
  
  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 exclude_edges
Set which edges can’t be used.

- def get_connected_components
  Returns the results generated by the execution of the algorithm.

- def is_closed
  Gets if Connectivity has been closed or not.

- def close
  Closes the Connectivity instance.

3.56.1 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](http://www.sparsity-technologies.com)

3.56.2 Member Function Documentation

3.56.2.1 def sparksee.StrongConnectivityGabow.add_edge_type (self, t, d)

Allows connectivity through edges of the given type.

Parameters:

- \(t\) [in] Edge type.
- \(d\) [in] Edge direction.

Reimplemented from sparksee.StrongConnectivity.

3.56.2.2 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:

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

Reimplemented from sparksee.StrongConnectivity.
3.56.2.3  def sparksee.StrongConnectivityGabow.add_node_type ( self, t )

Allows connectivity through nodes of the given type.

Parameters:
   t null

Reimplemented from sparksee.StrongConnectivity.

3.56.2.4  def sparksee.StrongConnectivityGabow.add_all_edge_types ( self, d )

Allows connectivity through all edge types of the graph.

Parameters:
   d [in] Edge direction.

Reimplemented from sparksee.StrongConnectivity.

3.56.2.5  def sparksee.StrongConnectivityGabow.__init__ ( self, s )

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:
   s [in] Session to get the graph from and calculate the connectivity

3.56.2.6  def sparksee.StrongConnectivityGabow.set_materialized_attribute ( self, attributeName )

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

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

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

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

Reimplemented from sparksee.StrongConnectivity.

3.56.2.7  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:

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

Reimplemented from sparksee.StrongConnectivity.

3.56.2.8 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.

Reimplemented from sparksee.StrongConnectivity.

3.56.2.9 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.

3.56.2.10 def sparksee.Connectivity.close (self) [inherited]

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

3.57 sparksee.TextStream Class Reference

TextStream class.

Public Member Functions

- def read
  
  Read data from the stream.

- def is_null
  
  Returns TRUE if the stream is not available.

- def write
  
  Write data to the stream.
3.57 sparksee.TextStream Class Reference

- def __init__
  Creates a new instance.

- def close
  Closes the stream.

3.57.1 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

3.57.2 Member Function Documentation

3.57.2.1 def sparksee.TextStream.read ( self, length)
Read data from the stream.

Parameters:

  length [in] Length of the given data buffer. It must be > 0.

Returns:

  The read data.

3.57.2.2 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
3.57.2.3  def sparksee.TextStream.write (self, dataIN, length)
Write data to the stream.

Parameters:

  dataIN  The string to write data from.
  length  Number of characters to write. It must be > 0.

3.57.2.4  def sparksee.TextStream.__init__ (self, append)
Creates a new instance.
A TextStream only can be created by the user to write data.

Parameters:

  append [in] If TRUE, the it is created in append mode to write from the end of the stream, otherwise it is created to write from the beginning of the stream.

3.57.2.5  def sparksee.TextStream.close (self)
Closes the stream.
Once the Stream is closed, it cannot be used again.

3.58 sparksee.Traversal Class Reference

Traversal class.
Inheritance diagram for sparksee.Traversal:

```
sparksee.Traversal
sparksee.TraversalBFS
sparksee.TraversalDFS
```

Public Member Functions

* def add_edge_type
  Allows for traversing edges of the given type.

* def has_next
  Gets if there are more objects to be traversed.

* def exclude_nodes
  Set which nodes can’t be used.

* def is_closed
  Gets if Traversal has been closed or not.
3.58 `sparksee.Traversal` Class Reference

- `def add_all_edge_types`
  
  Allows for traversing all edge types of the graph.

- `def add_node_type`
  
  Allows for traversing nodes of the given type.

- `def set_maximum_hops`
  
  Sets the maximum hops restriction.

- `def __next__`
  
  Used in `next()`.

- `def get_current_depth`
  
  Returns the depth of the current node.

- `def close`
  
  Closes the `Traversal` instance.

- `def add_all_node_types`
  
  Allows for traversing all node types of the graph.

- `def next`
  
  Gets the next object of the traversal.

- `def __iter__`
  
  Gets a new `TraversalIterator`.

- `def exclude_edges`
  
  Set which edges can’t be used.

3.58.1 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](http://www.sparsity-technologies.com)

3.58.2 Member Function Documentation

3.58.2.1 `def sparksee.Traversal.add_edge_type ( self, t, d)`

Allows for traversing edges of the given type.
Parameters:

\begin{itemize}
  \item \texttt{t} \ [\text{in}] \text{ Edge type.}
  \item \texttt{d} \ [\text{in}] \text{ Edge direction.}
\end{itemize}

Reimplemented in \texttt{sparksee.TraversalDFS}, and \texttt{sparksee.TraversalBFS}.

### 3.58.2.2 \texttt{def sparksee.Traversal.has_next (self)}

Gets if there are more objects to be traversed.

**Returns:**

TRUE if there are more objects, FALSE otherwise.

Reimplemented in \texttt{sparksee.TraversalDFS}, and \texttt{sparksee.TraversalBFS}.

### 3.58.2.3 \texttt{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:**

\begin{itemize}
  \item \texttt{nodes} \ [\text{in}] \text{ A set of node identifiers that must be kept intact until the destruction of the class.}
\end{itemize}

Reimplemented in \texttt{sparksee.TraversalDFS}, and \texttt{sparksee.TraversalBFS}.

### 3.58.2.4 \texttt{def sparksee.Traversal.is_closed (self)}

Gets if \texttt{Traversal} has been closed or not.

**See also:**

\texttt{close()}

**Returns:**

TRUE if the \texttt{Traversal} instance has been closed, FALSE otherwise.

### 3.58.2.5 \texttt{def sparksee.Traversal.add_all_edge_types (self, d)}

Allows for traversing all edge types of the graph.

**Parameters:**

\begin{itemize}
  \item \texttt{d} \ [\text{in}] \text{ Edge direction.}
\end{itemize}

Reimplemented in \texttt{sparksee.TraversalDFS}, and \texttt{sparksee.TraversalBFS}.
3.58.2.6  def sparksee.Traversal.add_node_type (self, t)
Allows for traversing nodes of the given type.

Parameters:

    t null

Reimplemented in sparksee.TraversalDFS, and sparksee.TraversalBFS.

3.58.2.7  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:

    maxhops [in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

Reimplemented in sparksee.TraversalDFS, and sparksee.TraversalBFS.

3.58.2.8  def sparksee.Traversal.__next__ (self)
Used in next().

Returns:

    The next element

3.58.2.9  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.

Reimplemented in sparksee.TraversalDFS, and sparksee.TraversalBFS.

3.58.2.10 def sparksee.Traversal.close (self)
Closes the Traversal instance.
It must be called to ensure the integrity of all data.

3.58.2.11 def sparksee.Traversal.next (self)
Gets the next object of the traversal.

Returns:

    A node or edge identifier.

Reimplemented in sparksee.TraversalDFS, and sparksee.TraversalBFS.
3.58.2.12  def sparksee.Traversal.__iter__ (self)
Gets a new TraversalIterator.

Returns:
   TraversalIterator instance

3.58.2.13  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:
   edges  [in] A set of edge identifiers that must be kept intact until the destruction of the class.

Reimplemented in sparksee.TraversalDFS, and sparksee.TraversalBFS.

3.59  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
  Allows for traversing edges of the given type.

- def add_all_node_types
  Allows for traversing all node types of the graph.

- def has_next
  Gets if there are more objects to be traversed.
• def **exclude_nodes**
  
  Set which nodes can’t be used.

• def **__init__**
  
  Creates a new instance.

• def **next**
  
  Gets the next object of the traversal.

• def **add_node_type**
  
  Allows for traversing nodes of the given type.

• def **add_all_edge_types**
  
  Allows for traversing all edge types of the graph.

• def **get_current_depth**
  
  Returns the depth of the current node.

• def **set_maximum_hops**
  
  Sets the maximum hops restriction.

• def **exclude_edges**
  
  Set which edges can’t be used.

• def **is_closed**
  
  Gets if Traversal has been closed or not.

• def **__next__**
  
  Used in next().

• def **close**
  
  Closes the Traversal instance.

• def **__iter__**
  
  Gets a new TraversalIterator.

### 3.59.1 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](http://www.sparsity-technologies.com)
3.59.2 Member Function Documentation

3.59.2.1 def sparksee.TraversalBFS.add_edge_type (self, t, d)

Allows for traversing edges of the given type.

Parameters:

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

Reimplemented from sparksee.Traversal.

3.59.2.2 def sparksee.TraversalBFS.has_next (self)

Gets if there are more objects to be traversed.

Returns:

  TRUE if there are more objects, FALSE otherwise.

Reimplemented from sparksee.Traversal.

3.59.2.3 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:

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

Reimplemented from sparksee.Traversal.

3.59.2.4 def sparksee.TraversalBFS.__init__ (self, s, node)

Creates a new instance.

Parameters:

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

3.59.2.5 def sparksee.TraversalBFS.next (self)

Gets the next object of the traversal.

Returns:

  A node or edge identifier.

Reimplemented from sparksee.Traversal.
3.59.2.6  def sparksee.TraversalBFS.add_node_type (self, t)
Allows for traversing nodes of the given type.

Parameters:
- t [null]

Reimplemented from sparksee.Traversal.

3.59.2.7  def sparksee.TraversalBFS.add_all_edge_types (self, d)
Allows for traversing all edge types of the graph.

Parameters:
- d [in] Edge direction.

Reimplemented from sparksee.Traversal.

3.59.2.8  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.

Reimplemented from sparksee.Traversal.

3.59.2.9  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:
- maxhops [in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

Reimplemented from sparksee.Traversal.

3.59.2.10 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:
- edges [in] A set of edge identifiers that must be kept intact until the destruction of the class.

Reimplemented from sparksee.Traversal.
3.59.2.11 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.

3.59.2.12 def sparksee.Traversal.__next__ (self)  [inherited]
Used in next().

Returns:
   The next element

3.59.2.13 def sparksee.Traversal.close (self)  [inherited]
Closes the Traversal instance.
It must be called to ensure the integrity of all data.

3.59.2.14 def sparksee.Traversal.__iter__ (self)  [inherited]
Gets a new TraversalIterator.

Returns:
   TraversalIterator instance

3.60 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
  Allows for traversing edges of the given type.

- def add_all_node_types
  Allows for traversing all node types of the graph.

- def has_next
  Gets if there are more objects to be traversed.

- def exclude_nodes
  Set which nodes can’t be used.

- def next
  Gets the next object of the traversal.

- def add_node_type
  Allows for traversing nodes of the given type.

- def add_all_edge_types
  Allows for traversing all edge types of the graph.

- def get_current_depth
  Returns the depth of the current node.

- def set_maximum_hops
  Sets the maximum hops restriction.

- def exclude_edges
  Set which edges can’t be used.

- def __init__
  Creates a new instance.

- def is_closed
  Gets if Traversal has been closed or not.

- def __next__
  Used in next().

- def close
  Closes the Traversal instance.

- def __iter__
  Gets a new TraversalIterator.
3.60.1 Detailed Description

Depth-First Search (DFS) implementation of Traversal.
Starting from a source or root node, it visits as far as possible along each branch before backtracking.
Check out the ‘Algorithms’ section in the SPARKSEE User Manual for more details on this.

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

3.60.2 Member Function Documentation

3.60.2.1 def sparksee.TraversalDFS.add_edge_type ( self, t, d)
Allows for traversing edges of the given type.

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

Reimplemented from sparksee.Traversal.

3.60.2.2 def sparksee.TraversalDFS.has_next ( self)
Gets if there are more objects to be traversed.

Returns:
TRUE if there are more objects, FALSE otherwise.

Reimplemented from sparksee.Traversal.

3.60.2.3 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:
  nodes [in] A set of node identifiers that must be kept intact until the destruction of the class.

Reimplemented from sparksee.Traversal.

3.60.2.4 def sparksee.TraversalDFS.next ( self)
Gets the next object of the traversal.

Returns:
A node or edge identifier.

Reimplemented from sparksee.Traversal.
3.60.2.5 def sparksee.TraversalDFS.add_node_type (self, t)
Allows for traversing nodes of the given type.

Parameters:
   
   $t$ null

Reimplemented from sparksee.Traversal.

3.60.2.6 def sparksee.TraversalDFS.add_all_edge_types (self, d)
Allows for traversing all edge types of the graph.

Parameters:
   
   $d$ [in] Edge direction.

Reimplemented from sparksee.Traversal.

3.60.2.7 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.

Reimplemented from sparksee.Traversal.

3.60.2.8 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:
   
   $maxhops$ [in] The maximum hops restriction. It must be positive or zero. Zero, the default value, means unlimited.

Reimplemented from sparksee.Traversal.

3.60.2.9 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:
   
   $edges$ [in] A set of edge identifiers that must be kept intact until the destruction of the class.

Reimplemented from sparksee.Traversal.
3.60.10  def sparksee.TraversalDFS.__init__ (self, s, node)
Creates a new instance.

Parameters:
  s  [in]  Session to get the graph from and perform traversal.
  node  [in]  Node to start traversal from.

3.60.11  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.

3.60.12  def sparksee.Traversal.__next__ (self)  [inherited]
Used in next().

Returns:
  The next element

3.60.13  def sparksee.Traversal.close (self)  [inherited]
Closes the Traversal instance.
It must be called to ensure the integrity of all data.

3.60.14  def sparksee.Traversal.__iter__ (self)  [inherited]
Gets a new TraversalIterator.

Returns:
  TraversalIterator instance

3.61  sparksee.Type Class Reference

Type data class.

Public Member Functions

  - def get_restricted_from
    
    *Gets the tail or source type identifier for restricted edge types.*
• def get_are_neighbors_indexed
  Gets if this is an edge type with neighbors index.

• def get_object_type
  Gets the object type.

• def get_restricted_to
  Gets the head or target type identifier for restricted edge types.

• def get_is_restricted
  Gets if this is a restricted edge type.

• def get_num_objects
  Gets the number of objects belonging to the type.

• def get_id
  Gets the Sparksee type identifier.

• def get_is_directed
  Gets if this is a directed edge type.

• def get_name
  Gets the unique type name.

Static Public Attributes

• int GLOBAL_TYPE = 1
  Global type identifier constant.

• int INVALID_TYPE = 0
  Invalid type identifier constant.

3.61.1 Detailed Description

Type data class.
It contains information about a node or edge type.

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

3.61.2 Member Function Documentation

3.61.2.1 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.
3.61.2.2  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.

3.61.2.3  def sparksee.Type.get_object_type ( self)  
Gets the object type.

Returns:  
The object type.

3.61.2.4  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.

3.61.2.5  def sparksee.Type.get_is_restricted ( self)  
Gets if this is a restricted edge type.

Returns:  
TRUE for restricted edge types, FALSE otherwise.

3.61.2.6  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.

3.61.2.7  def sparksee.Type.get_id ( self)  
Gets the Sparksee type identifier.

Returns:  
The Sparksee type identifier.
3.62 sparksee.TypeExporter Class Reference

3.61.2.8 def sparksee.Type.get_is_directed (self)
Gets if this is a directed edge type.

**Returns:**

TRUE for directed edge types, FALSE otherwise.

3.61.2.9 def sparksee.Type.get_name (self)
Gets the unique type name.

**Returns:**

The unique type name.

3.62 sparksee.TypeExporter Class Reference

Base TypeExporter class.

Inheritance diagram for sparksee.TypeExporter:

```
sparksee.TypeExporter
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sparksee.EdgeTypeExporter</td>
</tr>
<tr>
<td></td>
<td>sparksee.NodeTypeExporter</td>
</tr>
</tbody>
</table>
```

**Public Member Functions**

- def set_row_writer
  
  *Sets the output data destination.*

- def set_frequency
  
  *Sets the frequency of listener notification.*

- def run
  
  *Runs export process.*

- def register
  
  *Registers a new listener.*

- def set_type
  
  *Sets the type to be exported.*

- def set_header
  
  *Sets the presence of a header row.*

- def set_graph
  
  *Sets the graph that will be exported.*

- def set_attributes
  
  *Sets the list of Attributes.*
3.62 sparksee.TypeExporter Class Reference

3.62.1 Detailed Description

Base TypeExporter class.

Base class to export a node or edge type from a graph using a RowWriter.

TypeExporterListener can be registered to receive information about the progress of the export process by means of TypeExporterEvent. The default frequency of notification to listeners is 100000.

By default no header row is created.

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

Author:

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

3.62.2 Member Function Documentation

3.62.2.1 def sparksee.TypeExporter.set_row_writer (self, rw)

Sets the output data destination.

rr[in] Input RowWriter.

Parameters:

rw null

Reimplemented in sparksee.NodeTypeExporter, and sparksee.EdgeTypeExporter.

3.62.2.2 def sparksee.TypeExporter.set_frequency (self, freq)

Sets the frequency of listener notification.

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

Parameters:

freq null

Reimplemented in sparksee.NodeTypeExporter, and sparksee.EdgeTypeExporter.

3.62.2.3 def sparksee.TypeExporter.run (self)

Runs export process.

Exceptions:

IOError If bad things happen writing to the RowWriter.

RuntimeError null

Reimplemented in sparksee.NodeTypeExporter, and sparksee.EdgeTypeExporter.

3.62.2.4 def sparksee.TypeExporter.register (self, tel)

Registers a new listener.
3.63 sparksee.TypeExporterEvent Class Reference

Parameters:

\[ \text{tel} \ [\text{in}] \ TypeExporterListener \] to be registered.

Reimplemented in sparksee.NodeTypeExporter, and sparksee.EdgeTypeExporter.

3.62.2.5 \textbf{def} sparksee.TypeExporter.set_type ( \textit{self}, \textit{t} )

Sets the type to be exported.

Parameters:

\[ \text{t} \ [\text{in}] \ Type \ identifier. \]

Reimplemented in sparksee.NodeTypeExporter, and sparksee.EdgeTypeExporter.

3.62.2.6 \textbf{def} sparksee.TypeExporter.set_header ( \textit{self}, \textit{header} )

Sets the presence of a header row.

Parameters:

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

Reimplemented in sparksee.NodeTypeExporter, and sparksee.EdgeTypeExporter.

3.62.2.7 \textbf{def} sparksee.TypeExporter.set_graph ( \textit{self}, \textit{g} )

Sets the graph that will be exported.

Parameters:

\[ \text{g} \ [\text{in}] \ Graph. \]

Reimplemented in sparksee.NodeTypeExporter, and sparksee.EdgeTypeExporter.

3.62.2.8 \textbf{def} sparksee.TypeExporter.set_attributes ( \textit{self}, \textit{attrs} )

Sets the list of Attributes.

Parameters:

\[ \text{attrs} \ [\text{in}] \ Attribute \ identifiers \ to \ be \ exported \]

Reimplemented in sparksee.NodeTypeExporter, and sparksee.EdgeTypeExporter.

3.63 sparksee.TypeExporterEvent Class Reference

Provides information about the progress of an TypeExproter instance.
Public Member Functions

- def get_count
  
  Gets the current number of objects exported.

- def is_last
  
  Gets if this is the last event or not.

- def get_type_id
  
  Gets the type identifier.

- def get_total
  
  Gets the total number of objects exported.

### 3.63.1 Detailed Description

Provides information about the progress of a 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](http://www.sparsity-technologies.com)

### 3.63.2 Member Function Documentation

#### 3.63.2.1 def sparksee.TypeExporterEvent.get_count (self)

Gets the current number of objects exported.

**Returns:**

- The current number of objects exported.

#### 3.63.2.2 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.

#### 3.63.2.3 def sparksee.TypeExporterEvent.get_type_id (self)

Gets the type identifier.

**Returns:**

- The type identifier.
3.64 sparksee.TypeExporterListener Class Reference

Interface to be implemented to receive TypeExporterEvent events from a TypeExporter.

Public Member Functions

- def notify_event
  Method to be notified from a TypeExporter.

3.64.1 Detailed Description

Interface to be implemented to receive TypeExporterEvent events from a TypeExporter.
Check out the ‘Data export’ section in the SPARKSEE User Manual for more details on this.

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

3.64.2 Member Function Documentation

3.64.2.1 def sparksee.TypeExporterListener.notify_event (self, tee)
Method to be notified from a TypeExporter.

Parameters:

 tee [in] Notified event.

3.65 sparksee.TypeList Class Reference

Sparksee type identifier list.

Public Member Functions

- def add
  Adds a Sparksee type identifier at the end of the list.

- def clear
  Clears the list.

- def __iter__
Gets a new TypeListIterator.

- def iterator
  Gets a new TypeListIterator.

- def __init__
  Constructor.

- def count
  Number of elements in the list.

3.65.1 Detailed Description

Sparksee type identifier list.

It stores a Sparksee node or edge type identifier list.

Use TypeListIterator to access all elements into this collection.

Author:

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

3.65.2 Member Function Documentation

3.65.2.1 def sparksee.TypeList.add (self, type)

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

Parameters:

  type [in] Sparksee type identifier.

3.65.2.2 def sparksee.TypeList.__iter__ (self)

Gets a new TypeListIterator.

Returns:

  TypeListIterator instance

3.65.2.3 def sparksee.TypeList.iterator (self)

Gets a new TypeListIterator.

Returns:

  TypeListIterator instance.

3.65.2.4 def sparksee.TypeList.__init__ (self)

Constructor.

This creates an empty list.
3.65.2.5  def sparksee.TypeList.count (self)
Number of elements in the list.

Returns:
Number of elements in the list.

3.66  sparksee.TypeListIterator Class Reference

TypeList iterator class.

Public Member Functions

- def has_next
  Gets if there are more elements.

- def next
  Moves to the next element.

- def __next__
  Used in next().

3.66.1  Detailed Description

TypeList iterator class.
Iterator to traverse all the Sparksee node or edge type identifiers into a TypeList instance.

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

3.66.2  Member Function Documentation

3.66.2.1  def sparksee.TypeListIterator.has_next (self)
Gets if there are more elements.

Returns:
TRUE if there are more elements, FALSE otherwise.

3.66.2.2  def sparksee.TypeListIterator.next (self)
Moves to the next element.

Returns:
The next element.
3.66.2.3  def sparksee.TypeListIterator.__next__(self)

Used in next().

Returns:

The next element

3.67  sparksee.TypeLoader Class Reference

Base TypeLoader class.

Inheritance diagram for sparksee.TypeLoader:

sparksee.TypeLoader
sparksee.EdgeTypeLoader
sparksee.NodeTypeLoader

Public Member Functions

- def run_two_phases
  
  Run the loader for two phases loading.

- def run_n_phases
  
  Run the loader for N phases loading.

- def set_frequency
  
  Sets the frequency of listener notification.

- def set_log_off
  
  Trans off all the error reporting.

- def run
  
  Run the loader.

- def set_log_error
  
  Sets a log error file.

- def set_type
  
  Sets the type to be loaded.

- def set_graph
  
  Sets the graph where the data will be loaded.

- def set_row_reader
  
  Sets the input data source.

- def register
  
  Registers a new listener.
• def set_attribute_positions
  
  Sets the list of attribute positions.

• def set_locale
  
  Sets the locale that will be used to read the data.

• def set_timestamp_format
  
  Sets a specific timestamp format.

• def set_attributes
  
  Sets the list of Attributes.

### 3.67.1 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](http://www.sparsity-technologies.com)

### 3.67.2 Member Function Documentation

#### 3.67.2.1 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:**

- `IOError` null
- `RuntimeError` null

Reimplemented in `sparksee.NodeTypeLoader`, and `sparksee.EdgeTypeLoader`.

#### 3.67.2.2 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:

partitions  [in] Number of horizontal partitions to perform the load.

Exceptions:

IOError  null
RuntimeError  null

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.3 def sparksee.TypeLoader.set_frequency ( self, freq)

Sets the frequency of listener notification.

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

Parameters:

freq  null

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.4 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.

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.5 def sparksee.TypeLoader.run ( self)

Run the loader.

Exceptions:

IOError  null
RuntimeError  null

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.6 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:

path  null

Exceptions:

IOError  If bad things happen opening the file.

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.
3.67.2.7 def sparksee.TypeLoader.set_type (self, t)
Sets the type to be loaded.

Parameters:

   t [in] Type identifier.

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.8 def sparksee.TypeLoader.set_graph (self, g)
Sets the graph where the data will be loaded.

Parameters:

   g [in] Graph.

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.9 def sparksee.TypeLoader.set_row_reader (self, rr)
Sets the input data source.

Parameters:

   rr [in] Input RowReader.

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.10 def sparksee.TypeLoader.register (self, tel)
Registers a new listener.

Parameters:

   tel TypeLoaderListener to be registered.

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.11 def sparksee.TypeLoader.set_attribute_positions (self, attrsPos)
Sets the list of attribute positions.

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

Parameters:

   attrsPos null

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.
3.67.2.12  def sparksee.TypeLoader.set_locale (self, localeStr)
Sets the locale that will be used to read the data.
It should match the locale used in the rowreader.

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

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.13  def sparksee.TypeLoader.set_timestamp_format (self, timestampFormat)
Sets a specific timestamp format.

Parameters:
   timestampFormat  null

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.67.2.14  def sparksee.TypeLoader.set_attributes (self, attrs)
Sets the list of Attributes.

Parameters:
   attrs  [in] Attribute identifiers to be loaded

Reimplemented in sparksee.NodeTypeLoader, and sparksee.EdgeTypeLoader.

3.68  sparksee.TypeLoaderEvent Class Reference

Provides information about the progress of a TypeLoader instance.

Public Member Functions

- def get_total_phases
  Gets the total number of phases.

- def get_count
  Gets the current number of objects created.

- def get_total_partition_steps
  Gets the total number of steps in the current partition.

- def is_last
  Gets if this is the last event or not.

- def get_partition
  Gets the current partition.
• def get_type_id
  
  Gets the type identifier.

• def get_total_partitions
  
  Gets the total number of partitions.

• def get_phase
  
  Gets the current phase.

3.68.1 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

3.68.2 Member Function Documentation

3.68.2.1 def sparksee.TypeLoaderEvent.get_total_phases (self)

 Gets the total number of phases.

 Returns:

 The total number of phases.

3.68.2.2 def sparksee.TypeLoaderEvent.get_count (self)

 Gets the current number of objects created.

 Returns:

 The current number of objects created.

3.68.2.3 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.

3.68.2.4 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.
3.68.2.5 def sparksee.TypeLoaderEvent.get_partition (self)

Gets the current partition.

Returns:
- The current partition.

3.68.2.6 def sparksee.TypeLoaderEvent.get_type_id (self)

Gets the type identifier.

Returns:
- The type identifier.

3.68.2.7 def sparksee.TypeLoaderEvent.get_total_partitions (self)

Gets the total number of partitions.

Returns:
- The total number of partitions.

3.68.2.8 def sparksee.TypeLoaderEvent.get_phase (self)

Gets the current phase.

Returns:
- The current phase.

3.69 sparksee.TypeLoaderListener Class Reference

Interface to be implemented to receive TypeLoaderEvent events from a TypeLoader.

Public Member Functions

- def notify_event

  Method to receive events from a Loader.

3.69.1 Detailed Description

Interface to be implemented to receive TypeLoaderEvent events from a TypeLoader.

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

Author:

Sparsity Technologies http://www.sparsity-technologies.com
3.69.2 Member Function Documentation

3.69.2.1 def sparksee.TypeLoaderListener.notify_event (self, ev)
Method to receive events from a Loader.

Parameters:

   ev  Loader.LoaderEvent with information from a running Loader.

3.70 sparksee.Value Class Reference

Value class.

Public Member Functions

- def set_oid
  Sets the Value.

- def set_long_void
  Sets the Value.

- def get_oid
  Gets OID Value.

- def set_boolean_void
  Sets the Value.

- def compare
  Compares with the given Value.

- def set_double_void
  Sets the Value.

- def get_boolean
  Gets Boolean Value.

- def set_timestamp_void
  Sets the Value.

- def set_null_void
  Sets the Value to NULL.

- def __init__
  Copy constructor.

- def __init__
  Creates a new instance.

- def set_oid_void
Sets the OID Value.

• def get_data_type
  Gets the DataType.

• def get_string
  Gets String Value.

• def set_string
  Sets the Value.

• def is_null
  Gets if this is a NULL Value.

• def set_double
  Sets the Value.

• def set_timestamp_void
  Sets the Value.

• def equals
  Compares with the given Value.

• def get_double
  Gets Double Value.

• def set_integer_void
  Sets the Value.

• def set_void
  Sets the Value.

• def set_boolean
  Sets the Value.

• def set_timestamp
  Sets the Value.

• def set_integer
  Sets the Value.

• def set_long
  Sets the Value.

• def set_string_void
  Sets the Value.

• def get_long
  Gets Long Value.
• def get_timestamp
  
  Gets Timestamp Value.

• def set_null
  
  Sets the Value to NULL.

• def get_integer
  
  Gets Integer Value.

• def set_timestamp
  
  Sets the Value.

• def operator
  
  Assignment operator.

Static Public Attributes

• int MAX_LENGTH_STRING = 2047
  
  Maximum number of characters allowed for a String.

3.70.1 Detailed Description

Value class.

It is a container which stores a value and its data type (domain). A Value can be NULL.

Author:

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

3.70.2 Member Function Documentation

3.70.2.1 def sparksee.Value.set_oid (self, v)

Sets the Value.

Parameters:

v [in] New OID Value.

Returns:

The calling instance.

3.70.2.2 def sparksee.Value.set_long_void (self, v)

Sets the Value.

Parameters:

v [in] New Long value.
3.70.2.3  def sparksee.Value.get_oid ( self)

Gets OID Value.

This must be an non-NULL OID Value.

**Returns:**

The OID Value.

3.70.2.4  def sparksee.Value.set_boolean_void ( self, v)

Sets the Value.

**Parameters:**

v [in] New Boolean value.

3.70.2.5  def sparksee.Value.compare ( self, v)

Compares with the given Value.

It does not work for Text or if given Value objects does not have the same DataType.

**Parameters:**

v 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.

3.70.2.6  def sparksee.Value.set_double_void ( self, v)

Sets the Value.

**Parameters:**

v [in] New Double value.

3.70.2.7  def sparksee.Value.get_boolean ( self)

Gets Boolean Value.

This must be a non-NULL Boolean Value.

**Returns:**

The Boolean Value.

3.70.2.8  def sparksee.Value.set_timestamp_void ( self, v)

Sets the Value.

**Parameters:**

3.70.2.9   def sparksee.Value.__init__ (self, v)
Copy constructor.

Parameters:

    v [in] Value to be copied.

3.70.2.10  def sparksee.Value.__init__ (self)
Creates a new instance.
It creates a NULL Value.

3.70.2.11  def sparksee.Value.set_oid_void (self, v)
Sets the OID Value.

Parameters:

    v [in] New OID value.

3.70.2.12  def sparksee.Value.get_data_type (self)
Gets the DataType.
Value cannot be NULL.

Returns:

    The DataType.

3.70.2.13  def sparksee.Value.get_string (self)
Gets String Value.
This must be a non-NULL String Value.

Returns:

    The String Value.

3.70.2.14  def sparksee.Value.set_string (self, v)
Sets the Value.

Parameters:

    v [in] New String value.

Returns:

    The calling instance.
3.70.2.15  def sparksee.Value.is_null ( self)

Gets if this is a NULL Value.

Returns:
TRUE if this is a NULL Value, FALSE otherwise.

3.70.2.16  def sparksee.Value.set_double ( self, v)

Sets the Value.

Parameters:
   v  [in] New Double value.

Returns:
The calling instance.

3.70.2.17  def sparksee.Value.set_timestamp_void ( self, year, month, day, hour, minutes, seconds, millisecs)

Sets the Value.

Parameters:
   month  [in] The month ([1..12]).
   day  [in] The of the month ([1..31]).
   hour  [in] The hour ([0..23]).
   minutes  [in] The minutes ([0..59]).
   seconds  [in] The seconds ([0..59]).
   millisecs  [in] The milliseconds ([0..999]).

3.70.2.18  def sparksee.Value.equals ( self, v)

Compares with the given Value.

It does not work for Text or if given Value objects does not have the same DataType.

Parameters:
   v  Given value to compare to.

Returns:
TRUE if this Value is equal to the given one; FALSE otherwise.
3.70.2.19  def sparksee.Value.get_double (self)

Gets Double Value.

This must be a non-NULL Double Value.

Returns:

The Double Value.

3.70.2.20  def sparksee.Value.set_integer_void (self, v)

Sets the Value.

Parameters:

v [in] New Integer value.

3.70.2.21  def sparksee.Value.set_void (self, v)

Sets the Value.

Parameters:

v [in] New value.

3.70.2.22  def sparksee.Value.set_boolean (self, v)

Sets the Value.

Parameters:

v [in] New Boolean value.

Returns:

The calling instance.

3.70.2.23  def sparksee.Value.set_timestamp (self, year, month, day, hour, minutes, seconds, milliseecs)

Sets the Value.

Parameters:

month [in] The month ([1..12]).
day [in] The of the month ([1..31]).
hour [in] The hour ([0..23]).
minutes [in] The minutes ([0..59]).
seconds [in] The seconds ([0..59]).
milliseecs [in] The milliseconds ([0..999]).

Returns:

The calling instance.
3.70.2.24  def sparksee.Value.set_integer ( self, v)
Sets the Value.

Parameters:
   v  [in] New Integer value.

Returns:
   The calling instance.

3.70.2.25  def sparksee.Value.set_long ( self, v)
Sets the Value.

Parameters:
   v  [in] New Long value.

Returns:
   The calling instance.

3.70.2.26  def sparksee.Value.set_string_void ( self, v)
Sets the Value.

Parameters:
   v  [in] New String value.

3.70.2.27  def sparksee.Value.get_long ( self)
Gets Long Value.
This must be a non-NULL Long Value.

Returns:
   The Long Value.

3.70.2.28  def sparksee.Value.get_timestamp ( self)
Gets Timestamp Value.
This must be a non-NULL Timestamp Value.

Returns:
   The Timestamp Value.
3.70.2.29  def sparksee.Value.set_null (self)
Sets the Value to NULL.

Returns:
    The calling instance.

3.70.2.30  def sparksee.Value.get_integer (self)
Gets Integer Value.
This must be a non-NULL Integer Value.

Returns:
    The Integer Value.

3.70.2.31  def sparksee.Value.set_timestamp (self, v)
Sets the Value.

Parameters:

Returns:
    The calling instance.

3.70.2.32  def sparksee.Value.operator (self, v)
Assignment operator.

Parameters:
    v [in] Value to be copied.

3.71   sparksee.Values Class Reference
Value set class.

Public Member Functions

  • def is_closed
    Gets if Values instance has been closed or not.

  • def count
    Gets the number of elements into the collection.

  • def __iter__
    Gets a new ValuesIterator.
• def iterator
  
  Gets a ValuesIterator.

• def close
  
  Closes the Values instance.

3.71.1 Detailed Description

Value set class.
This is a set of Value instances, that is there is no duplicated elements.
Use a ValuesIterator to traverse all the elements into the set.
When the Values instance is closed, it closes all existing and non-closed ValuesIterator instances too.

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

3.71.2 Member Function Documentation

3.71.2.1 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.

3.71.2.2 def sparksee.Values.count (self)

Gets the number of elements into the collection.

Returns:
  The number of elements into the collection.

3.71.2.3 def sparksee.Values.__iter__ (self)

Gets a new ValuesIterator.

Returns:
  ValuesIterator instance
3.71.2.4  def sparksee.Values.iterate(self, order)

Gets a ValuesIterator.

Parameters:

order  [in] Ascending or descending order.

Returns:

ValuesIterator instance.

3.71.2.5  def sparksee.Values.close(self)

Closes the Values instance.

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

3.72  sparksee.ValuesIterator Class Reference

Values iterator class.

Public Member Functions

• def has_next

Gets if there are more elements to traverse.

• def next

Gets the next element to traverse.

• def is_closed

Gets if ValuesIterator instance has been closed or not.

• def __next__

Used in next()?

• def close

Closes the ValuesIterator instance.

3.72.1  Detailed Description

Values iterator class.

It allows for traversing all the elements into a Values instance.

Author:

Sparsity Technologies http://www.sparsity-technologies.com
3.72.2 Member Function Documentation

3.72.2.1 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.

3.72.2.2 def sparksee.ValuesIterator.next (self)

Gets the next element to traverse.

Returns:
The next element.

3.72.2.3 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.

3.72.2.4 def sparksee.ValuesIterator.__next__ (self)

Used in next().

Returns:
The next element

3.72.2.5 def sparksee.ValuesIterator.close (self)

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

3.73 sparksee.WeakConnectivity Class Reference

WeakConnectivity class.
Inheritance diagram for `sparksee.WeakConnectivity`:

Collaboration diagram for `sparksee.WeakConnectivity`:

**Public Member Functions**

- def `add_edge_type`
  Allows connectivity through edges of the given type.

- def `add_all_node_types`
  Allows connectivity through all node types of the graph.

- def `add_all_edge_types`
  Allows connectivity through all edge types of the graph.

- def `run`
  Runs the algorithm in order to find the connected components.

- def `exclude_nodes`
  Set which nodes can’t be used.

- def `add_node_type`
  Allows connectivity through nodes of the given type.

- def `set_materialized_attribute`
  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 `exclude_edges`
  Set which edges can’t be used.

- def `get_connected_components`
  Returns the results generated by the execution of the algorithm.

- def `is_closed`
3.73.1 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](http://www.sparsity-technologies.com)

3.73.2 Member Function Documentation

3.73.2.1 `def sparksee.WeakConnectivity.add_edge_type (self, t)`

Allows connectivity through edges of the given type.

In a weak connectivity the edges can be used in Any direction. \(d[in]\) Edge direction.

Parameters:

\(t \ [in]\) Edge type.

Reimplemented in `sparksee.WeakConnectivityDFS`.

3.73.2.2 `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. \(d[in]\) Edge direction.

Reimplemented in `sparksee.WeakConnectivityDFS`.

3.73.2.3 `def sparksee.WeakConnectivity.run (self)`

Runs the algorithm in order to find the connected components.

This method can be called only once.

Reimplemented from `sparksee.Connectivity`.

Reimplemented in `sparksee.WeakConnectivityDFS`. 

---

*Generated on Fri Feb 7 13:22:10 2014 for SparkseePython by Doxygen*
3.73.2.4  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:**

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

Reimplemented from sparksee.Connectivity.
Reimplemented in sparksee.WeakConnectivityDFS.

3.73.2.5  def sparksee.WeakConnectivity.add_node_type (self, t)
Allows connectivity through nodes of the given type.

**Parameters:**

*t*  null

Reimplemented from sparksee.Connectivity.
Reimplemented in sparksee.WeakConnectivityDFS.

3.73.2.6  def sparksee.WeakConnectivity.set_materialized_attribute (self, attributeName)
Creates a new common attribute type for all node types in the graph in order to store, persistently, the
results related to the connected components found while executing this algorithm.
Whenever the user wants to retrieve the results, even when the graph has been closed and opened again,
it is only necessary to create a new instance of the class ConnectedComponents indicating the graph and
the name of the common attribute type which stores the results. This instance will have all the information
related to the connected components found in the moment of the execution of the algorithm that stored this
data.
It is possible to run the algorithm without specifying this parameter in order to avoid materializing the
results of the execution.

**Parameters:**

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

Reimplemented from sparksee.Connectivity.
Reimplemented in sparksee.WeakConnectivityDFS.

3.73.2.7  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:**

*edges*  [in] A set of edge identifiers that must be kept intact until the destruction of the class.
Reimplemented from `sparksee.Connectivity`.
Reimplemented in `sparksee.WeakConnectivityDFS`.

3.73.2.8 **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.

Reimplemented from `sparksee.Connectivity`.
Reimplemented in `sparksee.WeakConnectivityDFS`.

3.73.2.9 **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.

3.73.2.10 **def sparksee.Connectivity.close ( self) [inherited]**
Closes the `Connectivity` instance.
It must be called to ensure the integrity of all data.

3.74 **sparksee.WeakConnectivityDFS Class Reference**

`WeakConnectivityDFS` class.
Inheritance diagram for `sparksee.WeakConnectivityDFS`:

```plaintext
diagram relationship
sparksee.Connectivity
sparksee.WeakConnectivity
sparksee.WeakConnectivityDFS
```

Generated on Fri Feb 7 13:22:10 2014 for SparkseePython by Doxygen
Collaboration diagram for sparksee.WeakConnectivityDFS:

```
sparksee.WeakConnectivityDFS

sparksee.WeakConnectivity

sparksee.Connectivity
```

Public Member Functions

- **def** `add_edge_type`
  
  Allows connectivity through edges of the given type.

- **def** `add_all_node_types`
  
  Allows connectivity through all node types of the graph.

- **def** `add_all_edge_types`
  
  Allows connectivity through all edge types of the graph.

- **def** `exclude_nodes`
  
  Set which nodes can’t be used.

- **def** `run`
  
  Executes the algorithm.

- **def** `__init__`
  
  Creates a new instance of `WeakConnectivityDFS`.

- **def** `add_node_type`
  
  Allows connectivity through nodes of the given type.

- **def** `set_materialized_attribute`
  
  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** `exclude_edges`
  
  Set which edges can’t be used.

- **def** `get_connected_components`
  
  Returns the results generated by the execution of the algorithm.

- **def** `is_closed`
  
  Gets if `Connectivity` has been closed or not.

- **def** `close`
  
  Closes the `Connectivity` instance.
3.74.1 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](http://www.sparsity-technologies.com)

3.74.2 Member Function Documentation

3.74.2.1 def sparksee.WeakConnectivityDFS.add_edge_type ( self, t)

Allows connectivity through edges of the given type.

In a weak connectivity the edges can be used in Any direction. d[in] Edge direction.

Parameters:

\( t \) [in] Edge type.

Reimplemented from sparksee.WeakConnectivity.

3.74.2.2 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. d[in] Edge direction.

Reimplemented from sparksee.WeakConnectivity.

3.74.2.3 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:

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

Reimplemented from sparksee.WeakConnectivity.

3.74.2.4 def sparksee.WeakConnectivityDFS.__init__ ( self, s)
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:**

\[ s \text{ [in] Session} \] to get the graph from and calculate the connectivity

### 3.74.2.5 def sparksee.WeakConnectivityDFS.add_node_type (self, t)

Allows connectivity through nodes of the given type.

**Parameters:**

\[ t \text{ null} \]

Reimplemented from `sparksee.WeakConnectivity`.

### 3.74.2.6 def sparksee.WeakConnectivityDFS.set_materialized_attribute (self, attributeName)

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

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

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

**Parameters:**

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

Reimplemented from `sparksee.WeakConnectivity`.

### 3.74.2.7 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:**

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

Reimplemented from `sparksee.WeakConnectivity`.

### 3.74.2.8 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.

Reimplemented from sparksee.WeakConnectivity.

3.74.2.9 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.

3.74.2.10 def sparksee.Connectivity.close (self) [inherited]

Closes the Connectivity instance.

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

__init__
sparksee::AttributeList, 10
sparksee::BooleanList, 16
sparksee::ConnectedComponents, 21
sparksee::Context, 28
sparksee::EdgeTypeExporter, 50
sparksee::EdgeTypeLoader, 55
sparksee::Int32List, 84
sparksee::NodeTypeExporter, 93
sparksee::NodeTypeLoader, 97
sparksee::OIDList, 108, 109
sparksee::SinglePairShortestPathBFS, 131
sparksee::SinglePairShortestPathDijkstra, 136
sparksee::Sparksee, 138
sparksee::SparkseeConfig, 146
sparksee::StringList, 153
sparksee::StrongConnectivityGabow, 161
sparksee::TextStream, 164
sparksee::TraversalBFS, 170
sparksee::TraversalDFS, 175
sparksee::TypeList, 184
sparksee::Value, 196, 197
sparksee::WeakConnectivityDFS, 210

__iter__
sparksee::AttributeList, 10
sparksee::BooleanList, 16
sparksee::Int32List, 83
sparksee::Objects, 105
sparksee::OIDList, 108
sparksee::StringList, 153
sparksee::TypeList, 184

__next__
sparksee::AttributeListIterator, 11
sparksee::BooleanListIterator, 17
sparksee::Int32ListIterator, 85
sparksee::ObjectsIterator, 107
sparksee::OIDListIterator, 110
sparksee::StringListIterator, 154
sparksee::Traversal, 167
sparksee::TraversalBFS, 172
sparksee::TraversalDFS, 176
sparksee::TypeListIterator, 185
sparksee::ValuesIterator, 204

add
sparksee::AttributeList, 10
sparksee::BooleanList, 16
sparksee::Int32List, 83
sparksee::Objects, 105
sparksee::OIDList, 108
sparksee::StringList, 153
sparksee::TypeList, 184

add_all_edge_types
sparksee::Context, 27
sparksee::ShortestPath, 122
sparksee::SinglePairShortestPath, 125
sparksee::SinglePairShortestPathBFS, 130
sparksee::SinglePairShortestPathDijkstra, 135
sparksee::StrongConnectivity, 157
sparksee::StrongConnectivityGabow, 161
sparksee::Traversal, 166
sparksee::TraversalBFS, 171
sparksee::TraversalDFS, 175
sparksee::WeakConnectivity, 206
sparksee::WeakConnectivityDFS, 210

add_edge_type
sparksee::Context, 27
sparksee::ShortestPath, 121
sparksee::SinglePairShortestPath, 125
sparksee::SinglePairShortestPathBFS, 129
sparksee::SinglePairShortestPathDijkstra, 134
sparksee::StrongConnectivity, 156
sparksee::StrongConnectivityGabow, 160
sparksee::Traversal, 165
sparksee::TraversalBFS, 170
sparksee::TraversalDFS, 174
sparksee::WeakConnectivity, 206
sparksee::WeakConnectivityDFS, 210

add_node_type
sparksee::Connectivity, 24
sparksee::Context, 27
sparksee::ShortestPath, 122
sparksee::SinglePairShortestPath, 126
sparksee::SinglePairShortestPathBFS, 130
sparksee::SinglePairShortestPathDijkstra, 135
sparksee::StrongConnectivity, 157
sparksee::StrongConnectivityGabow, 160
sparksee::Traversal, 166
sparksee::TraversalBFS, 170
sparksee::TraversalDFS, 174
sparksee::WeakConnectivity, 207
sparksee::WeakConnectivityDFS, 211

add_weighted_edge_type
sparksee::SinglePairShortestPathDijkstra, 134

any
sparksee::Objects, 101

as_directed
<table>
<thead>
<tr>
<th>Function/Method</th>
<th>Index Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>backup</td>
<td>74</td>
</tr>
<tr>
<td>BETWEEN</td>
<td>19</td>
</tr>
<tr>
<td>close</td>
<td>22, 25</td>
</tr>
<tr>
<td>check_only_existence</td>
<td>131</td>
</tr>
<tr>
<td>close (sparksee::ConnectedComponents)</td>
<td>131</td>
</tr>
<tr>
<td>close (sparksee::Connectivity)</td>
<td>158</td>
</tr>
<tr>
<td>close (sparksee::Context)</td>
<td>164</td>
</tr>
<tr>
<td>close (sparksee::Traversal)</td>
<td>142</td>
</tr>
<tr>
<td>close (sparksee::TraversalBFS)</td>
<td>147</td>
</tr>
<tr>
<td>close (sparksee::TraversalDFS)</td>
<td>152</td>
</tr>
<tr>
<td>close (sparksee::Values)</td>
<td>192</td>
</tr>
<tr>
<td>close (sparksee::WeakConnectivity)</td>
<td>208</td>
</tr>
<tr>
<td>close (sparksee::WeakConnectivityDFS)</td>
<td>212</td>
</tr>
<tr>
<td>combine_difference</td>
<td>102</td>
</tr>
<tr>
<td>combine_intersection</td>
<td>103</td>
</tr>
<tr>
<td>combine_union</td>
<td>102</td>
</tr>
<tr>
<td>compare</td>
<td>196</td>
</tr>
<tr>
<td>compute</td>
<td>27, 28</td>
</tr>
<tr>
<td>CONFIG</td>
<td>86</td>
</tr>
<tr>
<td>contains</td>
<td>104</td>
</tr>
<tr>
<td>copy</td>
<td>103, 105</td>
</tr>
<tr>
<td>count</td>
<td>10</td>
</tr>
<tr>
<td>count_edges</td>
<td>71</td>
</tr>
<tr>
<td>count_nodes</td>
<td>74</td>
</tr>
<tr>
<td>create</td>
<td>137</td>
</tr>
<tr>
<td>DEBUG</td>
<td>86</td>
</tr>
<tr>
<td>degree</td>
<td>101</td>
</tr>
<tr>
<td>difference</td>
<td>68</td>
</tr>
<tr>
<td>drop</td>
<td>76, 78</td>
</tr>
<tr>
<td>dump_data</td>
<td>66</td>
</tr>
<tr>
<td>dump_storage</td>
<td>69</td>
</tr>
<tr>
<td>edges</td>
<td>72</td>
</tr>
<tr>
<td>enable_type</td>
<td>42</td>
</tr>
<tr>
<td>EQUAL</td>
<td>18</td>
</tr>
<tr>
<td>equals</td>
<td>102</td>
</tr>
<tr>
<td>exclude_edges</td>
<td>25</td>
</tr>
<tr>
<td>exclude_nodes</td>
<td>24</td>
</tr>
</tbody>
</table>

Generated on Fri Feb 7 13:22:10 2014 for SparkseePython by Doxygen
INDEX

sparksee::ShortestPath, 121
sparksee::SinglePairShortestPath, 125
sparksee::SinglePairShortestPathBFS, 129
sparksee::SinglePairShortestPathDijkstra, 134
sparksee::StrongConnectivity, 156
sparksee::StrongConnectivityGabow, 160
sparksee::Traversal, 166
sparksee::TraversalBFS, 170
sparksee::TraversalDFS, 174
sparksee::WeakConnectivity, 206
sparksee::WeakConnectivityDFS, 210
exists
sparksee::Objects, 102
explode
sparksee::Graph, 75, 77
export
sparksee::Graph, 69
find_attribute
sparksee::Graph, 72
find_attributes
sparksee::Graph, 74
find_edge
sparksee::Graph, 77
find_edge_types
sparksee::Graph, 71
find_node_types
sparksee::Graph, 73
find_object
sparksee::Graph, 79
find_or_create_edge
sparksee::Graph, 76
find_or_create_object
sparksee::Graph, 81
find_type
sparksee::Graph, 80
find_types
sparksee::Graph, 67
FINE
sparksee::LogLevel, 86

generate_schema_script
sparksee::ScriptParser, 118
get
sparksee::SparkseeProperties, 151
get_alias
sparksee::Database, 37
get_are_neighbors_indexed
sparksee::Type, 177
get_attribute
sparksee::Graph, 73, 78, 81
get_attribute_interval_count
sparksee::Graph, 68
get_attribute_statistics
sparksee::Graph, 70
get_attribute_text
sparksee::Graph, 70
get_attributes
sparksee::Graph, 70
get_available_mem
sparksee::PlatformStatistics, 112
get_avg_length_string
sparksee::AttributeStatistics, 15
get_boolean
sparksee::SparkseeProperties, 151
sparksee::Value, 196
get_cache
sparksee::DatabaseStatistics, 39
get_cache_max_size
sparksee::SparkseeConfig, 147
get_cache_statistics_enabled
sparksee::SparkseeConfig, 144
get_cache_statistics_file
sparksee::SparkseeConfig, 147
get_cache_statistics_snapshot_time
sparksee::SparkseeConfig, 148
get_color_rgb
sparksee::EdgeExport, 46
sparksee::NodeExport, 89
get_connected_component
sparksee::ConnectedComponents, 22
get_connected_components
sparksee::Connectivity, 25
sparksee::StrongConnectivity, 158
sparksee::StrongConnectivityGabow, 162
sparksee::WeakConnectivity, 208
sparksee::WeakConnectivityDFS, 211
get_cost
sparksee::SinglePairShortestPath, 126
sparksee::SinglePairShortestPathBFS, 130
sparksee::SinglePairShortestPathDijkstra, 135
get_count
sparksee::Attribute, 7
sparksee::ConnectedComponents, 21
sparksee::TypeExporterEvent, 182
sparksee::TypeLoaderEvent, 191
get_current_depth
sparksee::Traversal, 167
sparksee::TraversalBFS, 171
sparksee::TraversalDFS, 175
get_data
sparksee::DatabaseStatistics, 39
get_data_type
sparksee::Attribute, 8
sparksee::Value, 197
get_distinct
sparksee::AttributeStatistics, 13
get_double

Generated on Fri Feb 7 13:22:10 2014 for SparkseePython by Doxygen
INDEX 216

sparksee::Value, 198
get_edge
sparksee::DefaultExport, 42
sparksee::EdgeData, 44
sparksee::ExportManager, 59
get_edge_data
sparksee::Graph, 75
get_edge_peer
sparksee::Graph, 79
get_edge_type
sparksee::DefaultExport, 42
sparksee::ExportManager, 60
get_extent_pages
sparksee::SparkseeConfig, 145
get_extent_size
sparksee::SparkseeConfig, 143
get_font_size
sparksee::EdgeExport, 46
sparksee::NodeExport, 88
get_graph
sparksee::DefaultExport, 42
sparksee::ExportManager, 60
sparksee::Session, 119
get_head
sparksee::EdgeData, 44
get_height
sparksee::NodeExport, 90
get_high_availability_coordinators
sparksee::SparkseeConfig, 146
get_high_availability_enabled
sparksee::SparkseeConfig, 145
get_high_availability_i_p
sparksee::SparkseeConfig, 149
get_high_availability_master_history
sparksee::SparkseeConfig, 149
get_high_availability_synchronization
sparksee::SparkseeConfig, 143
get_id
sparksee::Attribute, 8
sparksee::Type, 178
get_integer
sparksee::SparkseeProperties, 152
sparksee::Value, 201
get_is_directed
sparksee::Type, 178
get_is_restricted
sparksee::Type, 178
get_kind
sparksee::Attribute, 7
get_label
sparksee::EdgeExport, 48
sparksee::GraphExport, 82
sparksee::NodeExport, 90
get_labelcolor_rgb
sparksee::EdgeExport, 47
sparksee::NodeExport, 89
get_license
sparksee::SparkseeConfig, 148
get_log_file
sparksee::SparkseeConfig, 144
get_log_level
sparksee::SparkseeConfig, 149
get_long
sparksee::Value, 200
get_max
sparksee::AttributeStatistics, 14
get_max_length_string
sparksee::AttributeStatistics, 14
get_mean
sparksee::AttributeStatistics, 14
get_median
sparksee::AttributeStatistics, 14
get_min
sparksee::AttributeStatistics, 13
get_min_length_string
sparksee::AttributeStatistics, 13
get_mode
sparksee::AttributeStatistics, 13
get_mode_count
sparksee::AttributeStatistics, 15
get_name
sparksee::Attribute, 8
sparksee::Type, 179
get_node
sparksee::DefaultExport, 43
sparksee::ExportManager, 60
get_node_type
sparksee::DefaultExport, 43
sparksee::ExportManager, 59
get_nodes
sparksee::ConnectedComponents, 22
get_null
sparksee::AttributeStatistics, 13
get_num_c_p_us
sparksee::PlatformStatistics, 113
get_num_objects
sparksee::Type, 178
get_object_type
sparksee::Graph, 78
sparksee::Type, 178
get_oid
sparksee::Value, 195
get_partition
sparksee::TypeLoaderEvent, 191
get_path
sparksee::Database, 37
get_path_as_edges
sparksee::SinglePairShortestPath, 126
INDEX 218

heads
sparksee::Graph, 77

index_attribute
sparksee::Graph, 71

INFO
sparksee::LogLevel, 86
intersection
sparksee::Objects, 105
is_closed
sparksee::ConnectedComponents, 21
sparksee::Connectivity, 24
sparksee::Context, 27
sparksee::Database, 37
sparksee::Objects, 101
sparksee::ObjectsIterator, 107
sparksee::Session, 119
sparksee::ShortestPath, 122
sparksee::SinglePairShortestPath, 127
sparksee::SinglePairShortestPathBFS, 131
sparksee::SinglePairShortestPathDijkstra, 136
sparksee::StrongConnectivity, 158
sparksee::StrongConnectivityGabow, 162
sparksee::Traversal, 166
sparksee::TraversalBFS, 171
sparksee::TraversalDFS, 176
sparksee::Values, 202
sparksee::ValuesIterator, 204
sparksee::WeakConnectivity, 208
sparksee::WeakConnectivityDFS, 212

is_fit
sparksee::NodeExport, 89

is_last
sparksee::TypeExporterEvent, 182
sparksee::TypeLoaderEvent, 191

is_null
sparksee::TextStream, 163
sparksee::Value, 197

is_session_attribute
sparksee::Attribute, 7

iterator
sparksee::AttributeList, 10
sparksee::BooleanList, 16
sparksee::Int32List, 84
sparksee::Objects, 105
sparksee::OIDList, 109
sparksee::StringList, 153
sparksee::TypeList, 184
sparksee::Values, 202
iterator_from_element
sparksee::Objects, 102
iterator_from_index
sparksee::Objects, 104

LESS_EQUAL
sparksee::Condition, 19

LESS_THAN
sparksee::Condition, 19

LIKE
sparksee::Condition, 19

LIKE_NO_CASE
sparksee::Condition, 19

load
sparksee::SparkseeProperties, 152

neighbors
sparksee::Graph, 69, 75
new_attribute
sparksee::Graph, 72, 78
new_edge
sparksee::Graph, 79, 80
new_edge_type
sparksee::Graph, 77
new_node
sparksee::Graph, 70
new_node_type
sparksee::Graph, 75
new_objects
sparksee::Session, 120
new_restricted_edge_type
sparksee::Graph, 81
new_session_attribute
sparksee::Graph, 74, 76

next
sparksee::AttributeListIterator, 11
sparksee::BooleanListIterator, 17
sparksee::Int32ListIterator, 85
sparksee::ObjectsIterator, 106
sparksee::OIDListIterator, 110
sparksee::StringListIterator, 154
sparksee::TraversalIterator, 167
sparksee::TraversalBFS, 170
sparksee::TraversalDFS, 174
sparksee::TypeListIterator, 185
sparksee::ValuesIterator, 204

NOT_EQUAL
sparksee::Condition, 19

notify_event
sparksee::TypeExporterListener, 183
sparksee::TypeLoaderListener, 193

open
sparksee::CSVReader, 31
sparksee::CSVWriter, 36
sparksee::Sparksee, 138

operator
sparksee::Value, 201
sparksee::SparkseeConfig, 145
set_fit
sparksee::NodeExport, 89
set_font_size
sparksee::EdgeExport, 46
sparksee::NodeExport, 89
set_forced_quotes
sparksee::CSVWriter, 35
set_frequency
sparksee::EdgeTypeExporter, 50
sparksee::EdgeTypeLoader, 54
sparksee::NodeTypeLoader, 92
sparksee::NodeTypeLoader, 96
sparksee::TypeExporter, 180
sparksee::TypeLoader, 188
set_graph
sparksee::EdgeTypeExporter, 51
sparksee::EdgeTypeLoader, 56
sparksee::NodeTypeLoader, 94
sparksee::NodeTypeLoader, 97
sparksee::TypeExporter, 181
sparksee::TypeLoader, 189
set_head_attribute
sparksee::EdgeTypeExporter, 50
sparksee::EdgeTypeLoader, 55
set_head_position
sparksee::EdgeTypeExporter, 52
sparksee::EdgeTypeLoader, 58
set_header
sparksee::EdgeTypeExporter, 52
sparksee::NodeTypeLoader, 93
sparksee::TypeExporter, 181
set_height
sparksee::NodeExport, 89
set_high_availability coordinators
sparksee::SparkseeConfig, 143
set_high_availability_enabled
sparksee::SparkseeConfig, 143
set_high_availability_i_p
sparksee::SparkseeConfig, 143
set_high_availability_master_history
sparksee::SparkseeConfig, 149
set_high_availability_synchronization
sparksee::SparkseeConfig, 147
set_integer
sparksee::Value, 199
set_integer_void
sparksee::Value, 199
set_label
sparksee::EdgeExport, 47
sparksee::GraphExport, 82
sparksee::NodeExport, 90
set_labelcolor_rgb
sparksee::EdgeExport, 46
sparksee::NodeExport, 88
set_license
sparksee::SparkseeConfig, 146
set_locale
sparksee::CSVReader, 32
sparksee::CSVWriter, 35
sparksee::EdgeTypeLoader, 57
sparksee::NodeTypeLoader, 98
sparksee::TypeLoader, 189
set_log_error
sparksee::EdgeTypeLoader, 55
sparksee::NodeTypeLoader, 96
sparksee::TypeLoader, 188
set_log_file
sparksee::SparkseeConfig, 150
set_log_level
sparksee::SparkseeConfig, 144
set_log_off
sparksee::EdgeTypeLoader, 54
sparksee::NodeTypeLoader, 96
sparksee::TypeLoader, 188
set_long
sparksee::Value, 200
set_long_void
sparksee::Value, 195
set_materialized attribute
sparksee::Connectivity, 24
sparksee::StrongConnectivity, 157
sparksee::StrongConnectivityGabow, 161
sparksee::WeakConnectivity, 207
sparksee::WeakConnectivityDFS, 211
set_maximum_hops
sparksee::Context, 28
sparksee::ShortestPath, 123
sparksee::SinglePairShortestPath, 126
sparksee::SinglePairShortestPathBFS, 130
sparksee::SinglePairShortestPathDijkstra, 135
sparksee::Traversal, 167
sparksee::TraversalBFS, 171
sparksee::TraversalDFS, 175
set_multilines
sparksee::CSVReader, 31
set_null
sparksee::Value, 200
set_num_lines
sparksee::CSVReader, 31
set_oid
sparksee::Value, 195
set_oid_void
sparksee::Value, 197
set_output_log
sparksee::ScriptParser, 118
set_pool_frame_size
sparksee::SparkseeConfig, 147

Generated on Fri Feb 7 13:22:10 2014 for SparkseePython by Doxygen
set_pool_persistent_max_size
sparksee::SparkseeConfig, 145
set_pool_persistent_min_size
sparksee::SparkseeConfig, 143
set_pool_temporary_max_size
sparksee::SparkseeConfig, 146
set_pool_temporary_min_size
sparksee::SparkseeConfig, 148
set_quotes
sparksee::CSVReader, 32
sparksee::CSVWriter, 35
set_recovery_cache_max_size
sparksee::SparkseeConfig, 143
set_recovery_checkpoint_time
sparksee::SparkseeConfig, 150
set_recovery_enabled
sparksee::SparkseeConfig, 146
set_recovery_log_file
sparksee::SparkseeConfig, 146
set_row_reader
sparksee::EdgeTypeLoader, 56
sparksee::NodeTypeLoader, 97
sparksee::TypeLoader, 189
set_row_writer
sparksee::EdgeTypeExporter, 51
sparksee::NodeTypeExporter, 92
sparksee::TypeExporter, 180
set_separator
sparksee::CSVReader, 32
sparksee::CSVWriter, 34
set_shape
sparksee::NodeExport, 90
set_start_line
sparksee::CSVReader, 32
set_string
sparksee::Value, 197
set_string_void
sparksee::Value, 200
set_tail_attribute
sparksee::EdgeTypeExporter, 50
sparksee::EdgeTypeLoader, 54
set_tail_position
sparksee::EdgeTypeExporter, 51
sparksee::EdgeTypeLoader, 56
set_timestamp
sparksee::Value, 199, 201
set_timestamp_format
sparksee::EdgeTypeLoader, 57
sparksee::NodeTypeLoader, 98
sparksee::TypeLoader, 190
set_timestamp_void
sparksee::Value, 196, 198
set_type
sparksee::EdgeTypeExporter, 50
sparksee::EdgeTypeLoader, 55
sparksee::NodeTypeExporter, 93
sparksee::NodeTypeLoader, 96
sparksee::TypeExporter, 181
sparksee::TypeLoader, 188
set_unweighted_edge_cost
sparksee::SinglePairShortestPathDijkstra, 134
set_void
sparksee::Value, 199
set_width
sparksee::EdgeExport, 46
sparksee::NodeExport, 88
SEVERE
sparksee::LogLevel, 86
sparksee::Attribute, 6
get_count, 7
get_data_type, 8
get_id, 8
get_kind, 7
get_name, 8
get_size, 7
get_type_id, 8
is_session_attribute, 7
sparksee::AttributeKind, 8
UNIQUE, 9
sparksee::AttributeList, 9
__init__, 10
__iter__, 10
add, 10
count, 10
iterator, 10
sparksee::AttributeListIterator, 10
__next__, 11
has_next, 11
next, 11
sparksee::AttributeStatistics, 11
get_avg_length_string, 15
get_distinct, 13
get_max, 14
get_max_length_string, 14
get_mean, 14
get_median, 14
get_min, 13
get_min_length_string, 13
get_mode, 13
get_mode_count, 15
gt_null, 13
gt_total, 14
gt_variance, 13
sparksee::BooleanList, 15
__init__, 16
__iter__, 16
add, 16
count, 16
sparksee::EdgeTypeExporter, 48
  __init__, 50
  register, 51
  run, 51
  set_attributes, 52
  set_frequency, 50
  set_graph, 51
  set_head_attribute, 50
  set_head_position, 52
  set_header, 52
  set_row_writer, 51
  set_tail_attribute, 50
  set_tail_position, 51
  set_type, 50
sparksee::EdgeTypeLoader, 52
  __init__, 55
  register, 57
  run, 57
  run_n_phases, 57
  run_two_phases, 56
  set_attribute_positions, 56
  set_attributes, 58
  set_frequency, 54
  set_graph, 56
  set_head_attribute, 55
  set_head_position, 58
  set_locale, 57
  set_log_error, 55
  set_log_off, 54
  set_row_reader, 56
  set_tail_attribute, 54
  set_tail_position, 56
  set_timestamp_format, 57
  set_type, 55
sparksee::ExportManager, 58
  enable_type, 61
  get_edge, 59
  get_edge_type, 60
  get_graph, 60
  get_node, 60
  get_node_type, 59
  prepare, 60
  release, 61
sparksee::ExportType, 61
  GRAPHML, 62
  GRAPHVIZ, 62
  YGRAPHML, 62
sparksee::Graph, 62
  backup, 74
  count_edges, 71
  count_nodes, 74
  degree, 68
  drop, 76, 78
  dump_data, 66
dump_storage, 69
  edges, 72
  explode, 75, 77
  export, 69
  find_attribute, 72
  find_attributes, 74
  find_edge, 77
  find_edge_types, 71
  find_node_types, 73
  find_object, 79
  find_or_create_edge, 76
  find_or_create_object, 81
  find_type, 80
  find_types, 67
  get_attribute, 73, 78, 81
  get_attribute_interval_count, 68
  get_attribute_statistics, 70
  get_attribute_text, 70
  get_attributes, 70
  get_edge_data, 75
  get_edge_peer, 79
  get_object_type, 78
  get_type, 71
  get_values, 79
  heads, 77
  index_attribute, 71
  neighbors, 69, 75
  new_attribute, 72, 78
  new_edge, 79, 80
  new_edge_type, 77
  new_node, 70
  new_node_type, 75
  new_restricted_edge_type, 81
  new_session_attribute, 74, 76
  remove_attribute, 73
  remove_type, 81
  rename_attribute, 68
  rename_type, 69, 75
  select, 67, 71, 73, 80
  set_attribute, 67
  set_attribute_default_value, 72
  set_attribute_text, 66
  tails, 76
  tails_and_heads, 68
sparksee::GraphExport, 82
  get_label, 82
  set_label, 82
sparksee::Int32List, 83
  __init__, 84
  __iter__, 83
  add, 83
  count, 84
  iterator, 84
sparksee::Int32ListIterator, 84
reset, 114
sparksee::RowWriter, 115
  close, 116
  write, 116
sparksee::ScriptParser, 116
  generate_schema_script, 118
  parse, 117
  set_error_log, 117
  set_output_log, 118
sparksee::Session, 118
  close, 119
  get_graph, 119
  is_closed, 119
  new_objects, 120
sparksee::ShortestPath, 120
  add_all_edge_types, 122
  add_edge_type, 121
  add_node_type, 122
  close, 123
  exclude_edges, 122
  exclude_nodes, 121
  is_closed, 122
  run, 121
  set_maximum_hops, 123
sparksee::SinglePairShortestPath, 123
  add_all_edge_types, 125
  add_edge_type, 125
  add_node_type, 126
  close, 127
  exclude_edges, 127
  exclude_nodes, 125
  get_cost, 126
  get_path_as_edges, 126
  get_path_as_nodes, 126
  is_closed, 127
  run, 125
  set_maximum_hops, 126
sparksee::SinglePairShortestPathBFS, 127
  __init__, 131
  add_all_edge_types, 130
  add_edge_type, 129
  add_node_type, 130
  check_only_existence, 131
  close, 131
  exclude_edges, 131
  exclude_nodes, 129
  get_cost, 130
  get_path_as_edges, 129
  get_path_as_nodes, 130
  is_closed, 131
  run, 130
  set_maximum_hops, 130
sparksee::SinglePairShortestPathDijkstra, 132
  __init__, 136
  add_all_edge_types, 135
  add_edge_type, 134
  add_node_type, 135
  add_weighted_edge_type, 134
  close, 136
  exclude_edges, 135
  exclude_nodes, 134
  get_cost, 135
  get_path_as_edges, 134
  get_path_as_nodes, 134
  is_closed, 136
  set_maximum_hops, 135
  set_unweighted_edge_cost, 134
sparksee::Sparksee, 136
  __init__, 138
  close, 138
  create, 137
  open, 138
  restore, 137
sparksee::SparkseeConfig, 138
  __init__, 146
  get_cache_max_size, 147
  get_cache_statistics_enabled, 144
  get_cache_statistics_file, 147
  get_cache_statistics_snapshot_time, 148
  get_extent_pages, 145
  get_extent_size, 143
  get_high_availability_coordinators, 146
  get_high_availability_enabled, 145
  get_high_availability_i_p, 149
  get_high_availability_master_history, 149
  get_high_availability_synchronization, 143
  get_license, 148
  get_log_file, 144
  get_log_level, 149
  get_pool_frame_size, 147
  get_pool_persistent_max_size, 144
  get_pool_persistent_min_size, 150
  get_pool_temporary_max_size, 148
  get_pool_temporary_min_size, 148
  get_recovery_cache_max_size, 149
  get_recovery_checkpoint_time, 144
  get_recovery_enabled, 146
  get_recovery_log_file, 145
  get_recovery_max_size, 145
  set_cache_max_size, 145
  set_cache_statistics_enabled, 149
  set_cache_statistics_file, 147
  set_cache_statistics_snapshot_time, 144
  set_extent_pages, 148
  set_extent_size, 145
  set_high_availability_coordinators, 143
  set_high_availability_enabled, 143
  set_high_availability_i_p, 143
  set_high_availability_master_history, 149
  set_high_availability_synchronization, 147
set_license, 146
set_log_file, 150
set_log_level, 144
set_pool_frame_size, 147
set_pool_persistent_max_size, 145
set_pool_persistent_min_size, 143
set_pool_temporary_max_size, 146
set_pool_temporary_min_size, 148
set_recovery_cache_max_size, 143
set_recovery_checkpoint_time, 150
set_recovery_enabled, 146
set_recovery_log_file, 146
sparksee::SparkseeProperties, 150
get, 151
get_boolean, 151
get_integer, 152
get_time_unit, 151
load, 152
sparksee::StringList, 152
__init__, 153
__iter__, 153
add, 153
count, 153
iterator, 153
sparksee::StringListIterator, 153
__next__, 154
has_next, 154
next, 154
sparksee::StrongConnectivity, 155
add_all_edge_types, 157
add_edge_type, 156
add_node_type, 157
close, 158
exclude_edges, 157
exclude_nodes, 156
get_connected_components, 158
is_closed, 158
run, 156
set_materialized_attribute, 157
sparksee::StrongConnectivityGabow, 158
__init__, 161
add_all_edge_types, 161
add_edge_type, 160
add_node_type, 160
close, 162
exclude_edges, 161
exclude_nodes, 160
get_connected_components, 162
is_closed, 162
set_materialized_attribute, 161
sparksee::TextStream, 162
__init__, 164
close, 164
is_null, 163
read, 163
write, 163
sparksee::Traversal, 164
__iter__, 167
__next__, 167
add_all_edge_types, 166
add_edge_type, 165
add_node_type, 166
close, 167
exclude_edges, 168
exclude_nodes, 166
get_current_depth, 167
has_next, 166
is_closed, 166
next, 167
set_maximum_hops, 167
sparksee::TraversalBFS, 168
__init__, 170
__iter__, 172
__next__, 172
add_all_edge_types, 171
add_edge_type, 170
add_node_type, 170
close, 172
exclude_edges, 171
exclude_nodes, 170
get_current_depth, 171
has_next, 170
is_closed, 171
next, 170
sparksee::TraversalDFS, 172
__init__, 175
__iter__, 176
__next__, 176
add_all_edge_types, 175
add_edge_type, 174
add_node_type, 174
close, 176
exclude_edges, 175
exclude_nodes, 174
get_current_depth, 175
has_next, 174
is_closed, 176
next, 174
set_maximum_hops, 175
sparksee::Type, 176
get_are_neighbors_indexed, 177
get_id, 178
get_is_directed, 178
get_is_restricted, 178
get_name, 179
get_num_objects, 178
get_object_type, 178
get_restricted_from, 177
get_restricted_to, 178
sparksee::TypeExporter, 179
  register, 180
  run, 180
  set_attributes, 181
  set_frequency, 180
  set_graph, 181
  set_header, 181
  set_row_writer, 180
  set_type, 181
sparksee::TypeExporterEvent, 181
  get_count, 182
  get_total, 182
  get_type_id, 182
  is_last, 182
sparksee::TypeExporterListener, 183
  notify_event, 183
sparksee::TypeList, 183
  __init__, 184
  __iter__, 184
  add, 184
  count, 184
  iterator, 184
sparksee::TypeListIterator, 185
  __next__, 185
  has_next, 185
  next, 185
sparksee::TypeLoader, 186
  register, 189
  run, 188
  run_n_phases, 187
  run_two_phases, 187
  set_attribute_positions, 189
  set_attributes, 190
  set_frequency, 188
  set_graph, 189
  set_locale, 189
  set_log_error, 188
  set_log_off, 188
  set_row_reader, 189
  set_timestamp_format, 190
  set_type, 188
sparksee::TypeLoaderEvent, 190
  get_count, 191
  get_partition, 191
  get_phase, 192
  get_total_partition_steps, 191
  get_total_partitions, 192
  get_total_phases, 191
  get_type_id, 192
  is_last, 191
sparksee::TypeLoaderListener, 192
  notify_event, 193
sparksee::Value, 193
  __init__, 196, 197
  compare, 196
  equals, 198
  get_boolean, 196
  get_data_type, 197
  get_double, 198
  get_integer, 201
  get_long, 200
  get_oid, 195
  get_string, 197
  get_timestamp, 200
  is_null, 197
  operator, 201
  set_boolean, 199
  set_boolean_void, 196
  set_double, 198
  set_double_void, 196
  set_integer, 199
  set_integer_void, 199
  set_long, 200
  set_long_void, 195
  set_null, 200
  set_oid, 195
  set_oid_void, 197
  set_string, 197
  set_string_void, 200
  set_timestamp, 199, 201
  set_timestamp_void, 196, 198
  set_void, 199
sparksee::Values, 201
  __iter__, 202
  close, 203
  count, 202
  is_closed, 202
  iterator, 202
sparksee::ValuesIterator, 203
  __next__, 204
  close, 204
  has_next, 204
  is_closed, 204
  next, 204
sparksee::WeakConnectivity, 204
  add_all_edge_types, 206
  add_edge_type, 206
  add_node_type, 207
  close, 208
  exclude_edges, 207
  exclude_nodes, 206
  get_connected_components, 208
  get_connected_components, 208
  is_closed, 208
  run, 206
  set_materialized_attribute, 207
sparksee::WeakConnectivityDFS, 208
__init__, 210
add_all_edge_types, 210
add_edge_type, 210
add_node_type, 211
close, 212
exclude_edges, 211
exclude_nodes, 210
get_connected_components, 211
is_closed, 212
set_materialized_attribute, 211
STRING
  sparksee::DataType, 40
tails
    sparksee::Graph, 76
tails_and_heads
    sparksee::Graph, 68
TEXT
  sparksee::DataType, 40
TIMESTAMP
  sparksee::DataType, 40
union
  sparksee::Objects, 105
UNIQUE
  sparksee::AttributeKind, 9
WARNING
  sparksee::LogLevel, 86
write
  sparksee::StreamWriter, 35
  sparksee::RowWriter, 116
  sparksee::TextStream, 163
YGRAPHML
  sparksee::ExportType, 62