constants

Class Constants

java.lang.Object
    constants.Constants

```
public class Constants
extends java.lang.Object
```

Constants used by the project

Field Summary

Fields

Modifier and Type	Field and Description
static java.lang.String	**CREATE_TESTQUERY** Fills a big database with a test record (example data).
static java.lang.String	**CREATE_TESTQUERY_LITTLE** Fills a simple database with a test record (example data).
static java.lang.String	**CREATE_TESTQUERY_VERY_LITTLE** Fills a simple database with a test record (example data).

Constructor Summary

Constructors

Constructor and Description
Constants()

Method Summary

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

CREATE_TESTQUERY

```
public static final java.lang.String CREATE_TESTQUERY
```

Fills a big database with a test record (example data).

See Also:

Constant Field Values

CREATE_TESTQUERY_LITTLE

```
public static final java.lang.String CREATE_TESTQUERY_LITTLE
```

Fills a simple database with a test record (example data).

See Also:

Constant Field Values

CREATE_TESTQUERY_VERY_LITTLE

```
public static final java.lang.String CREATE_TESTQUERY_VERY_LITTLE
```

Fills a simple database with a test record (example data).

See Also:

Constant Field Values

Constructor Detail

Constants

```
public Constants()
```

graph

Class Vertex

java.lang.Object
    graph.Vertex

```
public class Vertex
extends java.lang.Object
```

Represents vertices in a graph

Field Summary

Fields

Modifier and Type	Field and Description
`private java.lang.Integer`	**`id`** Id of the vertex
`private java.lang.String`	**`identifier`** Identifier of the vertex
`private java.util.List<Edge>`	**`incomingEdges`** List of incoming edges
`private java.lang.String`	**`label`** Label of the vertex
`private java.util.List<Edge>`	**`outgoingEdges`** List of outgoing edges
`private java.util.Map<java.lang.String,java.lang.String>`	**`properties`** A map with attributes of the vertex
`private boolean`	**`visited`** Can be used for special algorithms, which iterate about all nodes of a graph false by default

Constructor Summary

Constructors

Constructor and Description
`Vertex``(java.lang.String label, java.lang.String identifier, java.lang.Integer id, java.util.Map<java.lang.String,java.lang.String> attributes)` This method creates a new vertex.

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description

void	**addEdge**(**Vertex** vertex, **Edge** edge) This method adds an edge between this and another vertex
java.lang.Boolean	**equals**(org.neo4j.graphdb.Node b) This method compares Neo4J-node with node.
java.lang.Boolean	**equalsProp**(org.neo4j.graphdb.Node b) This method compares Neo4J-node with node considering the properties.
java.lang.Integer	**getId**() Returns the Id of the vertex.
java.lang.String	**getIdentifier**() Returns the identifier of the vertex.
java.util.List<**Edge**>	**getIncomingEdges**() Returns the incoming edges of the vertex.
java.lang.String	**getLabel**() Returns the label of the vertex.
java.util.List<**Edge**>	**getOutgoingEdges**() Returns the outgoing edges of the vertex.
java.util.Map<java.lang.String,java.lang.String>	**getProperties**() Returns the map of attributes of the vertex.
java.lang.Boolean	**isomorphic**(org.neo4j.graphdb.Node b) This method compares Neo4J-node with node for isomorphic
private boolean	**isomorphLabels**(org.neo4j.graphdb.Node b) This method compares Neo4J-node with node considering the properties.
private boolean	**isomorphProps**(org.neo4j.graphdb.Node b) This method compares Neo4J-nodes properties
java.lang.Boolean	**isomorphRelationships**(org.neo4j.graphdb.Node b) Checks if two Nodes are isomorph relating to relationsships of those node
boolean	**isVisited**() Was this vertex used by an algorithm
private void	**printArray**(java.lang.String[][] array) Returns a square array on the console.
void	**printVertex**() This method builds a field in which all nodes and edges are entered.
java.lang.String	**toString**() Returns the vertex as a formatted string
void	**visit**() Sets visited to true, can be useful for some special algorithms

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Detail

label

```
private java.lang.String label
```

Label of the vertex

identifier

```
private java.lang.String identifier
```

Identifier of the vertex

id

```
private java.lang.Integer id
```

Id of the vertex

properties

```
private java.util.Map<java.lang.String,java.lang.String> properties
```

A map with attributes of the vertex

incomingEdges

```
private java.util.List<Edge> incomingEdges
```

List of incoming edges

outgoingEdges

```
private java.util.List<Edge> outgoingEdges
```

List of outgoing edges

visited

```
private boolean visited
```

Can be used for special algorithms, which iterate about all nodes of a graph false by default

Constructor Detail

Vertex

```
public Vertex(java.lang.String label,
              java.lang.String identifier,
```

```
                java.lang.Integer id,
                java.util.Map<java.lang.String,java.lang.String> attributes)
```

This method creates a new vertex.

Parameters:

`label - Label of the vertex`

`identifier - Identifier of the vertex`

`id - Id of the vertex`

`attributes - the attributes of the vertex`

Method Detail

toString

```
public java.lang.String toString()
```

Returns the vertex as a formatted string

Overrides:

`toString in class java.lang.Object`

Returns:

`Formatted string [format: label:identifier]`

addEdge

```
public void addEdge(Vertex vertex,
                    Edge edge)
```

This method adds an edge between this and another vertex

Parameters:

`vertex - The vertex the added edge points to`

`edge - The added edge`

getLabel

```
public java.lang.String getLabel()
```

Returns the label of the vertex.

Returns:

`The label`

getIdentifier

```
public java.lang.String getIdentifier()
```

Returns the identifier of the vertex.

Returns:

`The identifier`

getProperties

```
public java.util.Map<java.lang.String,java.lang.String> getProperties()
```

Returns the map of attributes of the vertex.

Returns:

```
The map of attributes
```

getIncomingEdges

```
public java.util.List<Edge> getIncomingEdges()
```

Returns the incoming edges of the vertex.

Returns:

```
The list of the incoming edges
```

getOutgoingEdges

```
public java.util.List<Edge> getOutgoingEdges()
```

Returns the outgoing edges of the vertex.

Returns:

```
The list of the outgoing edges
```

getId

```
public java.lang.Integer getId()
```

Returns the Id of the vertex.

Returns:

```
The Id
```

equals

```
public java.lang.Boolean equals(org.neo4j.graphdb.Node b)
```

This method compares Neo4J-node with node.

Parameters:

```
b - The node for the comparison
```

Returns:

```
true if equal, false if not equal
```

isomorphic

```
public java.lang.Boolean isomorphic(org.neo4j.graphdb.Node b)
```

This method compares Neo4J-node with node for isomorphic

Parameters:

b - The node for the comparison

Returns:

true if equal, false if not equal

isomorphProps

```
private boolean isomorphProps(org.neo4j.graphdb.Node b)
```

This method compares Neo4J-nodes properties

Parameters:

b - The node for the comparison

Returns:

true if equal, false if not equal

isomorphLabels

```
private boolean isomorphLabels(org.neo4j.graphdb.Node b)
```

This method compares Neo4J-node with node considering the properties.

Parameters:

b - The node for the comparison

Returns:

true if equal, false if not equal

isomorphRelationships

```
public java.lang.Boolean isomorphRelationships(org.neo4j.graphdb.Node b)
```

Checks if two Nodes are isomorph relating to relationsships of those node

Parameters:

b - Node to compare relationsships with to this Vertex

equalsProp

```
public java.lang.Boolean equalsProp(org.neo4j.graphdb.Node b)
```

This method compares Neo4J-node with node considering the properties.

Parameters:

b - The node for the comparison

Returns:

true if equal, false if not equal

printVertex

```
public void printVertex()
```

This method builds a field in which all nodes and edges are entered. And prints a node with all connected nodes on the console.

printArray

```
private void printArray(java.lang.String[][] array)
```

Returns a square array on the console.

Parameters:

`array - Square array`

isVisited

```
public boolean isVisited()
```

Was this vertex used by an algorithm

Returns:

`The actual state of the visited variable`

visit

```
public void visit()
```

Sets visited to true, can be useful for some special algorithms

graph

Class Edge

java.lang.Object
    graph.Edge

```
public class Edge
extends java.lang.Object
```

Representation of a directed edge in a graph

Field Summary

Fields

Modifier and Type	Field and Description
`private java.lang.Integer`	**`id`** Unique id to identify the edge
`private java.lang.String`	**`label`** Name of the relation
`private java.util.Map<java.lang.String,java.lang.String>`	**`properties`** Properties
`private Vertex`	**`start`** Starting node
`private Vertex`	**`target`** Destination node
`private boolean`	**`visited`** Visited variable for some special algorithms, which iterate over the edges of a graph false by default

Constructor Summary

Constructors

Constructor and Description
`Edge``(Vertex start, Vertex target, java.lang.String relationLabel, java.util.Map<java.lang.String,java.lang.String> attributes)` Create a new edge.

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description
`boolean`	**`equals`**`(Edge edge)` Compares the ids of the given edges

java.lang.Boolean	**equalsProp**(org.neo4j.graphdb.Relationship rel) Compares edges with Neo4J Realtionships.
java.lang.Integer	**getId**() Returns the id of the given Edge
java.lang.String	**getLabel**() Returns the name of the relation.
java.util.Map<java.lang.String,java.lang.String>	**getProperties**() Returns the map of attributes.
Vertex	**getStart**() Returns the starting node.
Vertex	**getTarget**() Returns the destination node.
java.util.List<**Vertex**>	**getVertex**() The function returns the starting node **start** and the destination node **target** in a list of nodes.
boolean	**isVisited**() Was this Edge used by an algorithm
java.lang.String	**toString**() Output of the edge as a formatted string.
java.lang.String	**visit**() Sets visited to true, can be useful for some special algorithms

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Detail

start

```
private Vertex start
```

Starting node

label

```
private java.lang.String label
```

Name of the relation

target

```
private Vertex target
```

Destination node

properties

```
private java.util.Map<java.lang.String,java.lang.String> properties
```

Properties

visited

```
private boolean visited
```

Visited variable for some special algorithms, which iterate over the edges of a graph false by default

id

```
private java.lang.Integer id
```

Unique id to identify the edge

Constructor Detail

Edge

```
public Edge(Vertex start,
            Vertex target,
            java.lang.String relationLabel,
            java.util.Map<java.lang.String,java.lang.String> attributes)
```

Create a new edge.

Parameters:

`start - Starting node`

`target - Destination node`

`relationLabel - Name of the transitional relation`

Method Detail

toString

```
public java.lang.String toString()
```

Output of the edge as a formatted string.

Overrides:

`toString in class java.lang.Object`

Returns:

`Formatted string [format: Starting node + Destination node + Name of the relation]`

getVertex

```
public java.util.List<Vertex> getVertex()
```

The function returns the starting node `start` and the destination node `target` in a list of nodes.

Returns:

List with start and end nodes

getTarget

```
public Vertex getTarget()
```

Returns the destination node.

Returns:

The node the edge points to

getStart

```
public Vertex getStart()
```

Returns the starting node.

Returns:

The node from which the edge originates

getLabel

```
public java.lang.String getLabel()
```

Returns the name of the relation.

Returns:

The name of the transition

getProperties

```
public java.util.Map<java.lang.String,java.lang.String> getProperties()
```

Returns the map of attributes.

Returns:

The map of attributes

equalsProp

```
public java.lang.Boolean equalsProp(org.neo4j.graphdb.Relationship rel)
```

Compares edges with Neo4J Realtionships.

Parameters:

rel - relatinoship to compare to

Returns:

true if both have the same properties and name, otherwise false

isVisited

```
public boolean isVisited()
```

Was this Edge used by an algorithm

Returns:

`The actual state of the visited variable`

visit

```
public java.lang.String visit()
```

Sets visited to true, can be useful for some special algorithms

equals

```
public boolean equals(Edge edge)
```

Compares the ids of the given edges

Parameters:

`edge - edge no. 1`

Returns:

`are the ids the same`

getId

```
public java.lang.Integer getId()
```

Returns the id of the given Edge

Returns:

`unique id`

graph

Class Graph

java.lang.Object
    graph.Graph

```
public class Graph
extends java.lang.Object
```

Represents the data structure in a graph

Field Summary

Fields

Modifier and Type	Field and Description
`private java.util.List<Edge>`	`edges` List of all edges in the graph
`private java.util.List<Vertex>`	`vertices` List of all nodes in the graph

Constructor Summary

Constructors

Constructor and Description
`Graph()`

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description
`void`	`addEdge(Edge edge)` With this method a new edge between two nodes can be defined.
`void`	`addVertex(Vertex v)` With this method a new node can be inserted in the graph.
`Vertex`	`checkLabel(java.lang.String label)` This method checks whether a node from the list of all `vertices` nodes has the given label and returns the first match.
`java.util.List<Edge>`	`getEdges()` This method returns all edges of the graph.
`java.util.List<Vertex>`	`getVertices()`

	This method returns all nodes of the graph.
private java.lang.String	**graphToDOT**() This method converts a graph to a DOT graph, which can be displayed.
void	**printGraph**() Prints the graph as graphToDOT in the standard output stream
java.lang.String	**toString**() This method outputs the graph as a string, by calling **printGraph()**.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Detail

vertices

```
private java.util.List<Vertex> vertices
```

List of all nodes in the graph

edges

```
private java.util.List<Edge> edges
```

List of all edges in the graph

Constructor Detail

Graph

```
public Graph()
```

Method Detail

addVertex

```
public void addVertex(Vertex v)
```

With this method a new node can be inserted in the graph. (Into the list of nodes `vertices`.) This node still has no edges.

Parameters:

`v` - New node

addEdge

```
public void addEdge(Edge edge)
```

With this method a new edge between two nodes can be defined. The new edge gets a name (relation label).

Parameters:

`edge - The new edge`

checkLabel

```
public Vertex checkLabel(java.lang.String label)
```

This method checks whether a node from the list of all `vertices` nodes has the given label and returns the first match.

Parameters:

`label - Label to be searched for`

Returns:

`The node that was found first with the given label. If no node is found, null is returned.`

graphToDOT

```
private java.lang.String graphToDOT()
```

This method converts a graph to a DOT graph, which can be displayed.

Returns:

`String of the DOT graph`

printGraph

```
public void printGraph()
```

Prints the graph as graphToDOT in the standard output stream

toString

```
public java.lang.String toString()
```

This method outputs the graph as a string, by calling `printGraph()`.

Overrides:

`toString in class java.lang.Object`

Returns:

`All nodes and edges`

getVertices

```
public java.util.List<Vertex> getVertices()
```

This method returns all nodes of the graph.

Returns:

`Nodes of the graph from vertices`

getEdges

```
public java.util.List<Edge> getEdges()
```

This method returns all edges of the graph.

Returns:

`Edges of the graph from vertices`

matcher

Class Matcher

java.lang.Object
    matcher.Matcher

Direct Known Subclasses:

DualSimMatcher, DualSimMatcherProp, IsomorphicMatcher, TraceMatcher

---

```
public abstract class Matcher
extends java.lang.Object
```

Abstract matcher class

Field Summary

Fields

Modifier and Type	Field and Description
(package private) org.neo4j.graphdb.GraphDatabaseService	**db** Database service
(package private) Graph	**graph** Graph pattern

Constructor Summary

Constructors

Constructor and Description
Matcher() Default Matcher
Matcher(org.neo4j.graphdb.GraphDatabaseService database, Graph graph) Constructor to get to the database

Method Summary

All Methods Instance Methods Abstract Methods Concrete Methods

Modifier and Type	Method and Description
protected java.lang.Object	**clone**() Cloning of matchers finalized to dont get the suggestion to implem in specialised matchers
(package private) java.lang.Boolean	**compare**(org.neo4j.graphdb.Node a, org.neo4j.graphdb.N Compares the labels of two given nodes.
boolean	**equals**(java.lang.Object obj) Equals of the matcher finalized to dont get the suggestion to imple in specialised matchers
protected void	**finalize**() Finalizes the matcher finalized to dont get the suggestion to imple in specialised matchers
(package private) java.util.List<org.neo4j.graphdb.Node>	**findNodes**(Vertex vertex) Returns all nodes that have the same label as the vertex from the q graph.
(package private) java.util.List<org.neo4j.graphdb.Node>	**findNodesProp**(Vertex vertex) Returns all nodes that have the same label as the vertex from the q graph with consideration of the properties.
(package private) java.lang.Iterable<org.neo4j.graphdb.Relationship>	**getRelationships**(org.neo4j.graphdb.Node node) Returns the relationships of the given node.
(package private) java.lang.Iterable<org.neo4j.graphdb.Relationship>	**getRelationships**(org.neo4j.graphdb.Node node, org.neo4j.graphdb.Direction dir) Returns the relationships of the given node.
(package private) java.lang.Iterable<org.neo4j.graphdb.Relationship>	**getRelationships**(org.neo4j.graphdb.Node node, org.neo4j.graphdb.RelationshipType rel) Returns the relationships of the given node.
int	**hashCode**()

	Hash code for the matcher finalized to dont get the suggestion to implement in specialised matchers
abstract java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>>	**matchingAlgorithm**() This function must be overridden with the dualSimulation algorith
java.util.Set<java.util.Set<org.neo4j.graphdb.Node>>	**powerSet**(java.util.Set<org.neo4j.graphdb.Node> origin This method calculates the Power Set of a given Set
(package private) java.util.List<org.neo4j.graphdb.Node>	**previousNodes**(org.neo4j.graphdb.Node node) Returns all predecessors of the given node.
(package private) java.util.List<org.neo4j.graphdb.Node>	**previousNodes**(org.neo4j.graphdb.Node node, java.lang.String label) Returns all predecessors of the given node which have a given labe
(package private) java.util.List<org.neo4j.graphdb.Node>	**previousNodesProp**(org.neo4j.graphdb.Node node, **Edge** e Returns all predecessors of the given node which have a given labe consideration of the properties.
java.util.Set<org.neo4j.graphdb.Node>	**simulate**() This method executes the dualSimulation algorithm and formats t result for NEO4J.
(package private) java.util.List<org.neo4j.graphdb.Node>	**successingNodes**(org.neo4j.graphdb.Node node) Returns all successors of the given node.
(package private) java.util.List<org.neo4j.graphdb.Node>	**successingNodes**(org.neo4j.graphdb.Node node, java.lang.String label) Returns all successors of the given node which have a given label.
(package private) java.util.List<org.neo4j.graphdb.Node>	**successingNodesProp**(org.neo4j.graphdb.Node node, **Edge** Returns all successors of the given node which have a given label v consideration of the properties.
java.lang.String	**toString**() toString() for the matcher

Methods inherited from class java.lang.Object

getClass, notify, notifyAll, wait, wait, wait

Field Detail

db

```
org.neo4j.graphdb.GraphDatabaseService db
```

Database service

graph

```
Graph graph
```

Graph pattern

Constructor Detail

Matcher

```
public Matcher(org.neo4j.graphdb.GraphDatabaseService database,
               Graph graph)
```

Constructor to get to the database

Parameters:

`database - The database`

`graph - The graph`

Matcher

```
Matcher()
```

Default Matcher

Method Detail

hashCode

```
public final int hashCode()
```

Hash code for the matcher finalized to dont get the suggestion to implement in specialised matchers

Overrides:

`hashCode in class java.lang.Object`

Returns:

`hashCode of object`

equals

```
public final boolean equals(java.lang.Object obj)
```

Equals of the matcher finalized to dont get the suggestion to implement in specialised matchers

Overrides:

`equals in class java.lang.Object`

Parameters:

`obj - Object to compare`

Returns:

`Comparison of the given object and this`

clone

```
protected final java.lang.Object clone()
                                throws java.lang.CloneNotSupportedException
```

Cloning of matchers finalized to dont get the suggestion to implement in specialised matchers

Overrides:

`clone in class java.lang.Object`

Returns:

`cloned Object`

Throws:

`java.lang.CloneNotSupportedException - Error`

toString

```
public java.lang.String toString()
```

toString() for the matcher

Overrides:

`toString in class java.lang.Object`

Returns:

`Matcher information`

finalize

```
protected final void finalize()
                      throws java.lang.Throwable
```

Finalizes the matcher finalized to dont get the suggestion to implement in specialised matchers

Overrides:

`finalize in class java.lang.Object`

Throws:

`java.lang.Throwable - Error`

previousNodes

```
final java.util.List<org.neo4j.graphdb.Node> previousNodes(org.neo4j.graphdb.Node node)
```

Returns all predecessors of the given node.

Parameters:

`node - The given node`

Returns:

A list of all previous nodes

previousNodes

```
final java.util.List<org.neo4j.graphdb.Node> previousNodes(org.neo4j.graphdb.Node node,
                                                           java.lang.String label)
```

Returns all predecessors of the given node which have a given label.

Parameters:

node - The given node

label - The given label of the relationship

Returns:

A list of all previous nodes

previousNodesProp

```
final java.util.List<org.neo4j.graphdb.Node> previousNodesProp(org.neo4j.graphdb.Node node,
                                                               Edge edge)
```

Returns all predecessors of the given node which have a given label with consideration of the properties.

Parameters:

node - The given node

edge - The given edge from the query graph

Returns:

A list of all previous nodes

successingNodes

```
final java.util.List<org.neo4j.graphdb.Node> successingNodes(org.neo4j.graphdb.Node node)
```

Returns all successors of the given node.

Parameters:

node - The given node

Returns:

A list of all successing nodes

successingNodes

```
final java.util.List<org.neo4j.graphdb.Node> successingNodes(org.neo4j.graphdb.Node node,
                                                             java.lang.String label)
```

Returns all successors of the given node which have a given label.

Parameters:

node - The given node

label - The given label of the relationship

Returns:

A list of all successing nodes

successingNodesProp

```
final java.util.List<org.neo4j.graphdb.Node> successingNodesProp(org.neo4j.graphdb.Node node,
                                                                 Edge edge)
```

Returns all successors of the given node which have a given label with consideration of the properties.

Parameters:

node - The given node

edge - The given edge from the query graph

Returns:

A list of all successing nodes

compare

```
final java.lang.Boolean compare(org.neo4j.graphdb.Node a,
                                org.neo4j.graphdb.Node b)
```

Compares the labels of two given nodes.

Parameters:

a - The first node

b - The second node

Returns:

Returns true if the nodes have the same label otherwise false

getRelationships

```
final java.lang.Iterable<org.neo4j.graphdb.Relationship> getRelationships(org.neo4j.graphdb.Node node,
                                                                           org.neo4j.graphdb.Direction dir)
```

Returns the relationships of the given node.

Parameters:

node - The node you want the relationships from

dir - The direction of the relationship

Returns:

The list of the relationships

getRelationships

```
final java.lang.Iterable<org.neo4j.graphdb.Relationship> getRelationships(org.neo4j.graphdb.Node node,
                                                                           org.neo4j.graphdb.RelationshipType rel)
```

Returns the relationships of the given node.

Parameters:

node - The node you want the relationships from

rel - The type of the relationship

Returns:

The list of the relationships

getRelationships

```
final java.lang.Iterable<org.neo4j.graphdb.Relationship> getRelationships(org.neo4j.graphdb.Node node)
```

Returns the relationships of the given node.

Parameters:

node - The node you want the relationships from

Returns:

The list of the relationships

findNodes

```
final java.util.List<org.neo4j.graphdb.Node> findNodes(Vertex vertex)
```

Returns all nodes that have the same label as the vertex from the query graph.

Parameters:

vertex - The given Vertex

Returns:

Nodes for Vortex

findNodesProp

```
final java.util.List<org.neo4j.graphdb.Node> findNodesProp(Vertex vertex)
```

Returns all nodes that have the same label as the vertex from the query graph with consideration of the properties.

Parameters:

vertex - The given Vertex

Returns:

Nodes for Vortex

simulate

```
public final java.util.Set<org.neo4j.graphdb.Node> simulate()
```

This method executes the dualSimulation algorithm and formats the result for NEO4J.

Returns:

The result set

powerSet

```
public final java.util.Set<java.util.Set<org.neo4j.graphdb.Node>> powerSet(java.util.Set<org.neo4j.graphdb.Node> originalSet)
```

This method calculates the Power Set of a given Set

Parameters:

originalSet - Set to calculate Power Set about

Returns:

Power Set including empty set

matchingAlgorithm

```
public abstract java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>> matchingAlgorithm()
```

This function must be overridden with the dualSimulation algorithm

Returns:

The result of the algorithm

matcher

Class TraceMatcher

java.lang.Object
matcher.Matcher
matcher.TraceMatcher

```
public class TraceMatcher
extends Matcher
```

Field Summary

Fields

Modifier and Type	Field and Description
private java.util.Set<java.util.ArrayList<java.lang.String>>	**dbTraces** set of all traces of the database
private java.util.Set<java.util.ArrayList<java.lang.String>>	**patternTraces** set of all traces of a pattern

Fields inherited from class matcher.Matcher

db, graph

Constructor Summary

Constructors

Constructor and Description
TraceMatcher(org.neo4j.graphdb.GraphDatabaseService database, Graph graph) Creates a new TraceMatcher

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method and Description
private void	**floodDatabaseSubset**(org.neo4j.graphdb.Node actualNode, java.util.ArrayList<java.lang.String> trace, java.util.Set<org.neo4j.graphdb.Relationship> usedEdges, java.util.Set<org.neo4j.graphdb.Node> subsetOfPowerSet) Find all possible traces starting from the starting node.
private void	**floodPattern**(Vertex actualNode, java.util.ArrayList<java.lang.String> trace, java.util.Set<java.lang.Integer> usedEdges) Find all possible traces starting from the starting node.
java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>>	**matchingAlgorithm**() The trace matching algorithm
private void	**trace**(java.util.Set<org.neo4j.graphdb.Node> set) Starts the trace for a set of nodes

Methods inherited from class matcher.Matcher

clone, compare, equals, finalize, findNodes, findNodesProp, getRelationships, getRelationships, getRelationships, hashCode, powerSet, previousNodes, previousNodes, previousNodesProp, simulate, successingNodes, successingNodes, successingNodesProp, toString

Methods inherited from class java.lang.Object

getClass, notify, notifyAll, wait, wait, wait

Field Detail

patternTraces

```
private java.util.Set<java.util.ArrayList<java.lang.String>> patternTraces
```

set of all traces of a pattern

dbTraces

```
private java.util.Set<java.util.ArrayList<java.lang.String>> dbTraces
```

set of all traces of the database

Constructor Detail

TraceMatcher

```
public TraceMatcher(org.neo4j.graphdb.GraphDatabaseService database,
                    Graph graph)
```

Creates a new TraceMatcher

Parameters:

database - Database to use

graph - Pattern to find in the database

Method Detail

matchingAlgorithm

```
public java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>> matchingAlgorithm()
```

The trace matching algorithm

Specified by:

matchingAlgorithm in class Matcher

Returns:

Result of the trace matching

trace

```
private void trace(java.util.Set<org.neo4j.graphdb.Node> set)
```

Starts the trace for a set of nodes

Parameters:

set - Set to calculate trace about

floodPattern

```
private void floodPattern(Vertex actualNode,
                          java.util.ArrayList<java.lang.String> trace,
                          java.util.Set<java.lang.Integer> usedEdges)
```

Find all possible traces starting from the starting node. Circles are ignored.

Parameters:

actualNode - starting node

trace - already traveled path

usedEdges - already used edges; circles should be avoided

floodDatabaseSubset

```
private void floodDatabaseSubset(org.neo4j.graphdb.Node actualNode,
                                 java.util.ArrayList<java.lang.String> trace,
                                 java.util.Set<org.neo4j.graphdb.Relationship> usedEdges,
                                 java.util.Set<org.neo4j.graphdb.Node> subsetOfPowerSet)
```

Find all possible traces starting from the starting node. Circles are ignored.

Parameters:

```
actualNode - starting node

trace - already traveled path

usedEdges - already used edges; circles should be avoided
```

matcher

Class DualSimMatcher

java.lang.Object
  matcher.Matcher
    matcher.DualSimMatcher

```
public class DualSimMatcher
extends Matcher
```

Dual simulation for graph databases Sample matcher that extends the abstract class `Matcher`.

Field Summary

Fields inherited from class matcher.Matcher

`db`, `graph`

Constructor Summary

Constructors

Constructor and Description
`DualSimMatcher(org.neo4j.graphdb.GraphDatabaseService db, Graph graph)` This method creates a new dual simulation matcher.

Method Summary

All Methods | Static Methods | Instance Methods | Concrete Methods

Modifier and Type	Method and Description
(package private) static void	`count(java.util.Map<java.lang.Integer,java.util.List<org.neo4j` Outsources method for the dualSimulation
`java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>>`	`matchingAlgorithm()` The dualSimulation algorithm to be used.

Methods inherited from class matcher.Matcher

`clone`, `compare`, `equals`, `finalize`, `findNodes`, `findNodesProp`, `getRelationships`, `getRelationships`, `getRelationships`, `hashCode`, `powerSet`, `previousNodes`, `previousNodes`, `previousNodesProp`, `simulate`, `successingNodes`, `successingNodes`, `successingNodesProp`, `toString`

Methods inherited from class java.lang.Object

`getClass`, `notify`, `notifyAll`, `wait`, `wait`, `wait`

Constructor Detail

DualSimMatcher

```
public DualSimMatcher(org.neo4j.graphdb.GraphDatabaseService db,
                      Graph graph)
```

This method creates a new dual simulation matcher.

Parameters:

`db - The database to be used`

`graph - The graph to be used`

Method Detail

count

```
static void count(java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>> sim)
```

Outsources method for the dualSimulation

Parameters:

sim - Map of nodes, mapped with key

matchingAlgorithm

```
public java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>> matchingAlgorithm()
```

The dualSimulation algorithm to be used.

Specified by:

`matchingAlgorithm` in class `Matcher`

Returns:

The simulation

matcher

Class DualSimMatcherProp

java.lang.Object
  matcher.Matcher
    matcher.DualSimMatcherProp

```
public class DualSimMatcherProp
extends Matcher
```

Sample matcher that extends the abstract class `Matcher`.

Field Summary

Fields inherited from class matcher.Matcher

`db, graph`

Constructor Summary

Constructors

Constructor and Description
`DualSimMatcherProp(org.neo4j.graphdb.GraphDatabaseService db, Graph graph)` This method creates a new dual simulation matcher with consideration of the properties.

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method and Description
`java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>>`	`matchingAlgorithm()` The dualSimulation algorithm to be used.

Methods inherited from class matcher.Matcher

`clone, compare, equals, finalize, findNodes, findNodesProp, getRelationships, getRelationships, getRelationships, hashCode, powerSet, previousNodes, previousNodes, previousNodesProp, simulate, successingNodes, successingNodes, successingNodesProp, toString`

Methods inherited from class java.lang.Object

`getClass, notify, notifyAll, wait, wait, wait`

Constructor Detail

DualSimMatcherProp

```
public DualSimMatcherProp(org.neo4j.graphdb.GraphDatabaseService db,
                          Graph graph)
```

This method creates a new dual simulation matcher with consideration of the properties.

Parameters:

`db - The database to be used`

`graph - The graph to be used`

Method Detail

matchingAlgorithm

```
public java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>> matchingAlgorithm()
```

The dualSimulation algorithm to be used. (Currently: Dual simulation with consideration of the properties)

Specified by:

`matchingAlgorithm in class Matcher`

Returns:

`The simulation`

matcher

Class IsomorphicMatcher

java.lang.Object
matcher.Matcher
matcher.IsomorphicMatcher

```
public class IsomorphicMatcher
extends Matcher
```

Creates a new Isomorphic matcher

Field Summary

Fields inherited from class matcher.Matcher

db, graph

Constructor Summary

Constructors

Constructor and Description
IsomorphicMatcher(org.neo4j.graphdb.GraphDatabaseService database, Graph graph)

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method and Description
java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>>	matchingAlgorithm() Simple graph isomorphism of the pattern on all subgraphs of the database

Methods inherited from class matcher.Matcher

clone, compare, equals, finalize, findNodes, findNodesProp, getRelationships, getRelationships, getRelationships, hashCode, powerSet, previousNodes, previousNodes, previousNodesProp, simulate, successingNodes, successingNodes, successingNodesProp, toString

Methods inherited from class java.lang.Object

getClass, notify, notifyAll, wait, wait, wait

Constructor Detail

IsomorphicMatcher

```
public IsomorphicMatcher(org.neo4j.graphdb.GraphDatabaseService database,
                         Graph graph)
```

Method Detail

matchingAlgorithm

```
public java.util.Map<java.lang.Integer,java.util.List<org.neo4j.graphdb.Node>> matchingAlgorithm()
```

Simple graph isomorphism of the pattern on all subgraphs of the database

Specified by:

`matchingAlgorithm` in class `Matcher`

Returns:

isomorphic subsets

procedure.ressources

Class NodeResult

java.lang.Object
    procedure.ressources.NodeResult

```
public class NodeResult
extends java.lang.Object
```

Result constructor for NEO4J procedures.

Field Summary

Fields

Modifier and Type	Field and Description
org.neo4j.graphdb.Node	**node** Node of results.

Constructor Summary

Constructors

Constructor and Description
NodeResult(org.neo4j.graphdb.Node node) Result of the query.

Method Summary

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

node

```
public org.neo4j.graphdb.Node node
```

Node of results. (Must be public for NEO4J!)

Constructor Detail

NodeResult

```
public NodeResult(org.neo4j.graphdb.Node node)
```

Result of the query. (Must be public for NEO4J!)

Parameters:

```
node - The given node
```

procedure.ressources

Class ProcedureRessources

java.lang.Object
    procedure.ressources.ProcedureRessources

```
public class ProcedureRessources
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor and Description
`ProcedureRessources()`

Method Summary

All Methods Static Methods Concrete Methods

Modifier and Type	Method and Description
`static Graph`	`prepareQuery(org.neo4j.graphdb.GraphDatabaseService db, java.lang.String query)` Prepares query for execution

Methods inherited from class java.lang.Object

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

ProcedureRessources

```
public ProcedureRessources()
```

Method Detail

prepareQuery

```
public static Graph prepareQuery(org.neo4j.graphdb.GraphDatabaseService db,
                                 java.lang.String query)
```

Prepares query for execution

Parameters:

`db - database`

`query - pattern`

Returns:

`graph`

procedure

Class QueryBuilder

java.lang.Object
    procedure.QueryBuilder

```
public class QueryBuilder
extends java.lang.Object
```

Builds a graph from a query.

Field Summary

Fields

Modifier and Type	Field and Description
private java.lang.String	query Query from which the graph is built.

Constructor Summary

Constructors

Constructor and Description
QueryBuilder(java.lang.String queryParam) Default Constructor.

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description
Graph	build() This method builds a graph from a query.
private java.util.Map<java.lang.String,java.lang.String>	forgeProperties(java.lang.StringBuilder attString) This method builds the properties.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

query

```
private java.lang.String query
```

Query from which the graph is built.

Constructor Detail

QueryBuilder

```
public QueryBuilder(java.lang.String queryParam)
```

Default Constructor.

Method Detail

build

```
public Graph build()
```

This method builds a graph from a query.

Returns:

```
The built graph.
```

forgeProperties

```
private java.util.Map<java.lang.String,java.lang.String> forgeProperties(java.lang.StringBuilder attString)
```

This method builds the properties.

Returns:

```
The map of the properties
```

procedure

Class GraphProcedures

java.lang.Object
    procedure.GraphProcedures

```
public class GraphProcedures
extends java.lang.Object
```

Field Summary

Fields

Modifier and Type	Field and Description
org.neo4j.graphdb.GraphDatabaseService	**db** Access to the database.
org.neo4j.logging.Log	**log** Log

Constructor Summary

Constructors

Constructor and Description
GraphProcedures()

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description
java.util.stream.Stream<NodeResult>	**dualSim**(java.lang.String query) NEO4J Procedure that can be executed in the database.
java.util.stream.Stream<NodeResult>	**dualSimProp**(java.lang.String query) NEO4J Procedure that can be executed in the database.
java.util.stream.Stream<NodeResult>	**isomorphic**(java.lang.String query) NEO4J Procedure that can be executed in the database.
java.util.stream.Stream<NodeResult>	**trace**(java.lang.String query) NEO4J Procedure that can be executed in the database.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

db

```
public org.neo4j.graphdb.GraphDatabaseService db
```

Access to the database. (Must be public for NEO4J!)

log

```
public org.neo4j.logging.Log log
```

Log

Constructor Detail

GraphProcedures

```
public GraphProcedures()
```

Method Detail

dualSim

```
public java.util.stream.Stream<NodeResult> dualSim(java.lang.String query)
```

NEO4J Procedure that can be executed in the database. First, the query is converted to a graph. Then the matching algorithm is executed. Finally, the result is returned. The passed query is processed with this NEO4J procedure and a result set is returned.

Parameters:

query - The given query to execute

Returns:

Stream of NodeResults. Each NodeResult contains only one node, the one it represents in the result set.

dualSimProp

```
public java.util.stream.Stream<NodeResult> dualSimProp(java.lang.String query)
```

NEO4J Procedure that can be executed in the database.

First, the query is converted to a graph. Then the matching algorithm is executed. Finally, the result is returned.

The passed query is processed with this NEO4J procedure and a result set is returned.

Parameters:

`query - The given query to execute`

Returns:

`Stream of NodeResults. Each NodeResult contains only one node, the one it represents in the result set.`

isomorphic

```
public java.util.stream.Stream<NodeResult> isomorphic(java.lang.String query)
```

NEO4J Procedure that can be executed in the database.

First, the query is converted to a graph. Then the matching algorithm is executed. Finally, the result is returned.

The passed query is processed with this NEO4J procedure and a result set is returned.

Parameters:

`query - The given query to execute`

Returns:

`Stream of NodeResults. Each NodeResult contains only one node, the one it represents in the result set.`

trace

```
public java.util.stream.Stream<NodeResult> trace(java.lang.String query)
```

NEO4J Procedure that can be executed in the database.

First, the query is converted to a graph. Then the matching algorithm is executed. Finally, the result is returned.

The passed query is processed with this NEO4J procedure and a result set is returned.

Parameters:

`query - The given query to execute`

Returns:

`Stream of NodeResults. Each NodeResult contains only one node, the one it represents in the result set.`

tests

Class NeoTest

java.lang.Object
    tests.NeoTest

```
public class NeoTest
extends java.lang.Object
```

In this test some simple methods are called on a locally hosted database

Constructor Summary

Constructors

Constructor and Description
`NeoTest()`

Method Summary

All Methods | Static Methods | Concrete Methods

Modifier and Type	Method and Description
`private static void`	`createAndShow()` Display of data sets from a locally hosted database
`static void`	`main(java.lang.String[] args)` Test method to start a new query on a locally hosted database

Methods inherited from class java.lang.Object

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

NeoTest

```
public NeoTest()
```

Method Detail

main

```
public static void main(java.lang.String[] args)
```

Test method to start a new query on a locally hosted database

Parameters:

`args - program parameters`

createAndShow

```
private static void createAndShow()
```

Display of data sets from a locally hosted database

tests

Class ProcedureTest

java.lang.Object
    tests.ProcedureTest

```
public class ProcedureTest
extends java.lang.Object
```

Example call of a procedure in a class as test Using session.run(String query) a query can be used and called in NEO4J The results can be saved in a StatementResult sr and can be displayed individually using sr.peek()

Field Summary

Fields

Modifier and Type	Field and Description
org.neo4j.harness.junit.Neo4jRule	**neo4j** NEO4J rules

Constructor Summary

Constructors

Constructor and Description
ProcedureTest()

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description
void	**shouldAllowIndexingAndFindingANode**() NEO4J Test A new database with test data sets from the Constants (**Constants.CREATE_TESTQUERY**) is loaded and a procedure is called to test the functionality

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

neo4j

```
public final org.neo4j.harness.junit.Neo4jRule neo4j
```

NEO4J rules

Constructor Detail

ProcedureTest

```
public ProcedureTest()
```

Method Detail

shouldAllowIndexingAndFindingANode

```
public void shouldAllowIndexingAndFindingANode()
```

NEO4J Test A new database with test data sets from the Constants (`Constants.CREATE_TESTQUERY`) is loaded and a procedure is called to test the functionality