JavaScript is disabled on your browser.

* [Overview](http://docs.google.com/overview-summary.html)
* [Package](http://docs.google.com/package-summary.html)
* Class
* [Use](http://docs.google.com/class-use/Aggregator.html)
* [Tree](http://docs.google.com/package-tree.html)
* [Deprecated](http://docs.google.com/deprecated-list.html)
* [Index](http://docs.google.com/index-files/index-1.html)
* [Help](http://docs.google.com/help-doc.html)
* [Prev Class](http://docs.google.com/simplenlg/aggregation/AggregationRule.html)
* [Next Class](http://docs.google.com/simplenlg/aggregation/BackwardConjunctionReductionRule.html)
* [Frames](http://docs.google.com/index.html?simplenlg/aggregation/Aggregator.html)
* [No Frames](http://docs.google.com/Aggregator.html)
* [All Classes](http://docs.google.com/allclasses-noframe.html)
* Summary:
* Nested |
* Field |
* [Constr](#id.3znysh7)|
* [Method](#id.2et92p0)
* Detail:
* Field |
* [Constr](#id.1t3h5sf)|
* [Method](#id.2s8eyo1)

simplenlg.aggregation

## Class Aggregator

* java.lang.Object
  + [simplenlg.framework.NLGModule](http://docs.google.com/simplenlg/framework/NLGModule.html)
    - simplenlg.aggregation.Aggregator
* public class Aggregator  
  extends [NLGModule](http://docs.google.com/simplenlg/framework/NLGModule.html)  
  An Aggregator performs aggregation on clauses, by applying a set of prespecified rules on them and returning the result.Author: Albert Gatt, University of Malya & University of Aberdeen

### Constructor SummaryConstructors

|  |
| --- |
| * + Constructor and Description |
| * + [**Aggregator**](http://docs.google.com/simplenlg/aggregation/Aggregator.html#Aggregator())() Creates an instance of Aggregator |

### Method SummaryMethods

|  |  |
| --- | --- |
| * + Modifier and Type | * + Method and Description |
| * + void | * + [**addRule**](http://docs.google.com/simplenlg/aggregation/Aggregator.html#addRule(simplenlg.aggregation.AggregationRule))([AggregationRule](http://docs.google.com/simplenlg/aggregation/AggregationRule.html) rule) Add a rule to this aggregator. |
| * + java.util.List<[AggregationRule](http://docs.google.com/simplenlg/aggregation/AggregationRule.html)> | * + [**getRules**](http://docs.google.com/simplenlg/aggregation/Aggregator.html#getRules())() Get the rules in this aggregator. |
| * + void | * + [**initialise**](http://docs.google.com/simplenlg/aggregation/Aggregator.html#initialise())() Performs one-time initialisation of the module. |
| * + java.util.List<[NLGElement](http://docs.google.com/simplenlg/framework/NLGElement.html)> | * + [**realise**](http://docs.google.com/simplenlg/aggregation/Aggregator.html#realise(java.util.List))(java.util.List<[NLGElement](http://docs.google.com/simplenlg/framework/NLGElement.html)> elements) Apply aggregation to a list of elements. |
| * + [NLGElement](http://docs.google.com/simplenlg/framework/NLGElement.html) | * + [**realise**](http://docs.google.com/simplenlg/aggregation/Aggregator.html#realise(simplenlg.framework.NLGElement))([NLGElement](http://docs.google.com/simplenlg/framework/NLGElement.html) element) Apply aggregation to a single phrase. |
| * + void | * + [**setFactory**](http://docs.google.com/simplenlg/aggregation/Aggregator.html#setFactory(simplenlg.framework.NLGFactory))([NLGFactory](http://docs.google.com/simplenlg/framework/NLGFactory.html) factory) Set the factory that this aggregator should use to create phrases. |

### Methods inherited from class simplenlg.framework.[**NLGModule**](http://docs.google.com/simplenlg/framework/NLGModule.html)[getLexicon](http://docs.google.com/simplenlg/framework/NLGModule.html#getLexicon()), [setLexicon](http://docs.google.com/simplenlg/framework/NLGModule.html#setLexicon(simplenlg.lexicon.Lexicon))

### Methods inherited from class java.lang.Objectequals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

### Constructor Detail

#### Aggregator public Aggregator() Creates an instance of Aggregator

### Method Detail

#### initialise public void initialise() Performs one-time initialisation of the module.**Specified by:** [initialise](http://docs.google.com/simplenlg/framework/NLGModule.html#initialise()) in class [NLGModule](http://docs.google.com/simplenlg/framework/NLGModule.html)

#### setFactory public void setFactory([NLGFactory](http://docs.google.com/simplenlg/framework/NLGFactory.html) factory) Set the factory that this aggregator should use to create phrases. The factory will be passed on to all the component rules.Parameters:factory - the phrase factory

#### addRule public void addRule([AggregationRule](http://docs.google.com/simplenlg/aggregation/AggregationRule.html) rule) Add a rule to this aggregator. Aggregation rules are applied in the order in which they are supplied.Parameters:rule - the rule

#### getRules public java.util.List<[AggregationRule](http://docs.google.com/simplenlg/aggregation/AggregationRule.html)> getRules() Get the rules in this aggregator.Returns:the rules

#### realise public [NLGElement](http://docs.google.com/simplenlg/framework/NLGElement.html) realise([NLGElement](http://docs.google.com/simplenlg/framework/NLGElement.html) element) Apply aggregation to a single phrase. This will only work if the phrase is a coordinated phrase, whose children can be further aggregated.**Specified by:** [realise](http://docs.google.com/simplenlg/framework/NLGModule.html#realise(simplenlg.framework.NLGElement)) in class [NLGModule](http://docs.google.com/simplenlg/framework/NLGModule.html) Parameters:element - the NLGElement to be realised. Returns:the NLGElement representing the realised state. This may be the initial element in a changed form or be a completely new element.

#### realise public java.util.List<[NLGElement](http://docs.google.com/simplenlg/framework/NLGElement.html)> realise(java.util.List<[NLGElement](http://docs.google.com/simplenlg/framework/NLGElement.html)> elements) Apply aggregation to a list of elements. This method iterates through the rules supplied via [addRule(AggregationRule)](http://docs.google.com/simplenlg/aggregation/Aggregator.html#addRule(simplenlg.aggregation.AggregationRule)) and applies them to the elements.**Specified by:** [realise](http://docs.google.com/simplenlg/framework/NLGModule.html#realise(java.util.List)) in class [NLGModule](http://docs.google.com/simplenlg/framework/NLGModule.html) Parameters:elements - the list of elements to aggregate Returns:a list of the elements that remain after the aggregation rules have been applied

* [Overview](http://docs.google.com/overview-summary.html)
* [Package](http://docs.google.com/package-summary.html)
* Class
* [Use](http://docs.google.com/class-use/Aggregator.html)
* [Tree](http://docs.google.com/package-tree.html)
* [Deprecated](http://docs.google.com/deprecated-list.html)
* [Index](http://docs.google.com/index-files/index-1.html)
* [Help](http://docs.google.com/help-doc.html)
* [Prev Class](http://docs.google.com/simplenlg/aggregation/AggregationRule.html)
* [Next Class](http://docs.google.com/simplenlg/aggregation/BackwardConjunctionReductionRule.html)
* [Frames](http://docs.google.com/index.html?simplenlg/aggregation/Aggregator.html)
* [No Frames](http://docs.google.com/Aggregator.html)
* [All Classes](http://docs.google.com/allclasses-noframe.html)
* Summary:
* Nested |
* Field |
* [Constr](#id.3znysh7)|
* [Method](#id.2et92p0)
* Detail:
* Field |
* [Constr](#id.1t3h5sf)|
* [Method](#id.2s8eyo1)