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/InterrogativeType.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/features/InternalFeature.html)
* [Next Class](http://docs.google.com/simplenlg/features/LexicalFeature.html)
* [Frames](http://docs.google.com/index.html?simplenlg/features/InterrogativeType.html)
* [No Frames](http://docs.google.com/InterrogativeType.html)
* [All Classes](http://docs.google.com/allclasses-noframe.html)
* Summary:
* Nested |
* [Enum Constants](#id.3znysh7)|
* Field |
* [Method](#id.2et92p0)
* Detail:
* [Enum Constants](#id.1t3h5sf)|
* Field |
* [Method](#id.3j2qqm3)

simplenlg.features

## Enum InterrogativeType

* java.lang.Object
  + java.lang.Enum<[InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html)>
    - simplenlg.features.InterrogativeType

All Implemented Interfaces: java.io.Serializable, java.lang.Comparable<[InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html)>

* public enum InterrogativeType  
  extends java.lang.Enum<[InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html)>  
  An enumeration representing the different types of interrogatives or questions that SimpleNLG can realise. The interrogative type is recorded in the Feature.INTERROGATIVE\_TYPE feature and applies to clauses.Version: 4.0 Author: A. Gatt and D. Westwater, University of Aberdeen.

### Enum Constant SummaryEnum Constants

|  |
| --- |
| * + Enum Constant and Description |
| * + [**HOW**](http://docs.google.com/simplenlg/features/InterrogativeType.html#HOW) The type of interrogative relating to the manner in which an event happened. |
| * + [**HOW\_MANY**](http://docs.google.com/simplenlg/features/InterrogativeType.html#HOW_MANY) This represents a "how many" questions. |
| * + [**HOW\_PREDICATE**](http://docs.google.com/simplenlg/features/InterrogativeType.html#HOW_PREDICATE) A how question related to a predicative sentence, such as *John is fine*, which becomes *How is John?* |
| * + [**WHAT\_OBJECT**](http://docs.google.com/simplenlg/features/InterrogativeType.html#WHAT_OBJECT) This type of interrogative is a question pertaining to the object of a phrase. |
| * + [**WHAT\_SUBJECT**](http://docs.google.com/simplenlg/features/InterrogativeType.html#WHAT_SUBJECT) This type of interrogative is a question pertaining to the subject of a phrase. |
| * + [**WHERE**](http://docs.google.com/simplenlg/features/InterrogativeType.html#WHERE) This type of interrogative concerns the object of a verb that is to do with location. |
| * + [**WHO\_INDIRECT\_OBJECT**](http://docs.google.com/simplenlg/features/InterrogativeType.html#WHO_INDIRECT_OBJECT) This type of interrogative is a question pertaining to the indirect object of a phrase when the indirect object is a person. |
| * + [**WHO\_OBJECT**](http://docs.google.com/simplenlg/features/InterrogativeType.html#WHO_OBJECT) This type of interrogative is a question pertaining to the object of a phrase when the object is a person. |
| * + [**WHO\_SUBJECT**](http://docs.google.com/simplenlg/features/InterrogativeType.html#WHO_SUBJECT) This type of interrogative is a question pertaining to the subject of a phrase when the subject is a person. |
| * + [**WHY**](http://docs.google.com/simplenlg/features/InterrogativeType.html#WHY) The type of interrogative relating to the reason for an event happening. |
| * + [**YES\_NO**](http://docs.google.com/simplenlg/features/InterrogativeType.html#YES_NO) This represents a simple yes/no questions. |

### Method SummaryMethods

|  |  |
| --- | --- |
| * + Modifier and Type | * + Method and Description |
| * + java.lang.String | * + [**getString**](http://docs.google.com/simplenlg/features/InterrogativeType.html#getString())() Convenience method to return the String corresponding to the question word. |
| * + static boolean | * + [**isIndirectObject**](http://docs.google.com/simplenlg/features/InterrogativeType.html#isIndirectObject(java.lang.Object))(java.lang.Object type) A method to determine if the InterrogativeType is a question concerning an element with the discourse function of an indirect object. |
| * + static boolean | * + [**isObject**](http://docs.google.com/simplenlg/features/InterrogativeType.html#isObject(java.lang.Object))(java.lang.Object type) A method to determine if the InterrogativeType is a question concerning an element with the discourse function of an object. |
| * + static [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) | * + [**valueOf**](http://docs.google.com/simplenlg/features/InterrogativeType.html#valueOf(java.lang.String))(java.lang.String name) Returns the enum constant of this type with the specified name. |
| * + static [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html)[] | * + [**values**](http://docs.google.com/simplenlg/features/InterrogativeType.html#values())() Returns an array containing the constants of this enum type, in the order they are declared. |

### Methods inherited from class java.lang.EnumcompareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

### Methods inherited from class java.lang.ObjectgetClass, notify, notifyAll, wait, wait, wait

### Enum Constant Detail

#### HOW public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) HOW The type of interrogative relating to the manner in which an event happened. For example, *John kissed Mary* becomes *How did John kiss Mary?*

#### HOW\_PREDICATE public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) HOW\_PREDICATE A how question related to a predicative sentence, such as *John is fine*, which becomes *How is John?*

#### WHAT\_OBJECT public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) WHAT\_OBJECT This type of interrogative is a question pertaining to the object of a phrase. For example, *John bought a horse* becomes *what did John buy?* while *John gave Mary a flower* becomes *What did John give Mary?*

#### WHAT\_SUBJECT public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) WHAT\_SUBJECT This type of interrogative is a question pertaining to the subject of a phrase. For example, *A hurricane destroyed the house* becomes *what destroyed the house?*

#### WHERE public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) WHERE This type of interrogative concerns the object of a verb that is to do with location. For example, *John went to the beach* becomes *Where did John go?*

#### WHO\_INDIRECT\_OBJECT public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) WHO\_INDIRECT\_OBJECT This type of interrogative is a question pertaining to the indirect object of a phrase when the indirect object is a person. For example, *John gave Mary a flower* becomes *Who did John give a flower to?*

#### WHO\_OBJECT public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) WHO\_OBJECT This type of interrogative is a question pertaining to the object of a phrase when the object is a person. For example, *John kissed Mary* becomes *who did John kiss?*

#### WHO\_SUBJECT public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) WHO\_SUBJECT This type of interrogative is a question pertaining to the subject of a phrase when the subject is a person. For example, *John kissed Mary* becomes *Who kissed Mary?* while *John gave Mary a flower* becomes *Who gave Mary a flower?*

#### WHY public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) WHY The type of interrogative relating to the reason for an event happening. For example, *John kissed Mary* becomes *Why did John kiss Mary?*

#### YES\_NO public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) YES\_NO This represents a simple yes/no questions. So taking the example phrases of *John is a professor* and *John kissed Mary* we can construct the questions *Is John a professor?* and *Did John kiss Mary?*

#### HOW\_MANY public static final [InterrogativeType](http://docs.google.com/simplenlg/features/InterrogativeType.html) HOW\_MANY This represents a "how many" questions. For example of *dogs chased John/em> becomes How many dogs chased John*

### ***Method Detail***

#### ***values*** *public static*[*InterrogativeType*](http://docs.google.com/simplenlg/features/InterrogativeType.html)*[] values()* *Returns an array containing the constants of this enum type, in the order they are declared. This method may be used to iterate over the constants as follows:* *for (InterrogativeType c : InterrogativeType.values())   System.out.println(c); Returns:an array containing the constants of this enum type, in the order they are declared*

#### ***valueOf*** *public static*[*InterrogativeType*](http://docs.google.com/simplenlg/features/InterrogativeType.html)*valueOf(java.lang.String name)* *Returns the enum constant of this type with the specified name. The string must match exactly an identifier used to declare an enum constant in this type. (Extraneous whitespace characters are not permitted.)Parameters:name - the name of the enum constant to be returned. Returns:the enum constant with the specified name Throws: java.lang.IllegalArgumentException - if this enum type has no constant with the specified name java.lang.NullPointerException - if the argument is null*

#### ***isObject*** *public static boolean isObject(java.lang.Object type)* *A method to determine if the InterrogativeType is a question concerning an element with the discourse function of an object.Parameters:type - the interrogative type to be checked Returns:true if the type concerns an object, false otherwise.*

#### ***isIndirectObject*** *public static boolean isIndirectObject(java.lang.Object type)* *A method to determine if the InterrogativeType is a question concerning an element with the discourse function of an indirect object.Parameters:type - the interrogative type to be checked Returns:true if the type concerns an indirect object, false otherwise.*

#### ***getString*** *public java.lang.String getString()* *Convenience method to return the String corresponding to the question word. Useful, since the types in this enum aren't all simply convertible to strings (e.g. WHO\_SUBJCT and WHO\_OBJECT both correspond to String Who)Returns:the string corresponding to the question word*

* [*Overview*](http://docs.google.com/overview-summary.html)
* [*Package*](http://docs.google.com/package-summary.html)
* *Class*
* [*Use*](http://docs.google.com/class-use/InterrogativeType.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/features/InternalFeature.html)
* [*Next Class*](http://docs.google.com/simplenlg/features/LexicalFeature.html)
* [*Frames*](http://docs.google.com/index.html?simplenlg/features/InterrogativeType.html)
* [*No Frames*](http://docs.google.com/InterrogativeType.html)
* [*All Classes*](http://docs.google.com/allclasses-noframe.html)
* *Summary:*
* *Nested |*
* [*Enum Constants*](#id.3znysh7)*|*
* *Field |*
* [*Method*](#id.2et92p0)
* *Detail:*
* [*Enum Constants*](#id.1t3h5sf)*|*
* *Field |*
* [*Method*](#id.3j2qqm3)