Class DefaultRepresentativeFraction

Object
  Number
    DefaultRepresentativeFraction

All Implemented Interfaces:

Serializable, Cloneable, Emptiable, IdentifiedObject, RepresentativeFraction

---

```
public class DefaultRepresentativeFraction
extends Number
implements RepresentativeFraction, IdentifiedObject, Emptiable, Cloneable
```

A scale defined as the inverse of a denominator. Scale is defined as a kind of Number. The following property is mandatory in a well-formed metadata according ISO 19115:

```
MD_RepresentativeFraction
  └─denominator…… The number below the line in a vulgar fraction.
```

In addition to the standard properties, SIS provides the following methods:

- setScale(double) for computing the denominator from a scale value.

Limitations

- Instances of this class are not synchronized for multi-threading. Synchronization, if needed, is caller's responsibility.
- Serialized objects of this class are not guaranteed to be compatible with future Apache SIS releases. Serialization support is appropriate for short term storage or RMI between applications running the same version of Apache SIS. For long term storage, use XML instead.

Since:

0.3

See Also:

DefaultResolution.getEquivalentScale(),
Serialized Form

Constructor Summary

Constructors

Constructor	Description
DefaultRepresentativeFraction()	Creates a uninitialized representative fraction.

DefaultRepresentativeFraction(long denominator)	Creates a new representative fraction from the specified denominator.
DefaultRepresentativeFraction(RepresentativeFraction object)	Constructs a new representative fraction initialized to the value of the given object.

Method Summary

All Methods | Static Methods | Instance Methods | Concrete Methods

Modifier and Type	Method	Description
static DefaultRepresentativeFraction	**castOrCopy**(RepresentativeFraction object)	Returns a SIS metadata implementation with the same values as the given arbitrary implementation.
DefaultRepresentativeFraction	**clone**()	Returns a modifiable copy of this representative fraction.
double	**doubleValue**()	Returns the scale value of this representative fraction.
boolean	**equals**(Object object)	Compares this object with the specified value for equality.
float	**floatValue**()	Returns the scale as a `float` type.
void	**freeze**()	Makes this representative fraction unmodifiable.
long	**getDenominator**()	Returns the denominator of this representative fraction.
IdentifierMap	**getIdentifierMap**()	Returns a map view of the identifiers collection as (*authority*, *code*) entries.
Collection<Identifier>	**getIdentifiers**()	Returns all identifiers associated to this object, or an empty collection if none.
int	**hashCode**()	Returns a hash value for this representative fraction.
int	**intValue**()	Returns 1 if the denominator is equal to 1, or 0 otherwise.

boolean	**isEmpty**()	Returns true if no scale is defined.
long	**longValue**()	Returns 1 if the denominator is equal to 1, or 0 otherwise.
void	**setDenominator**(long denominator)	Sets the denominator value.
void	**setScale**(double scale)	Sets the denominator from a scale in the (0 … 1] range.
String	**toString**()	Returns a string representation of this scale, or NaN if undefined.

Methods inherited from class Number

byteValue, shortValue

Methods inherited from class Object

finalize, getClass, notify, notifyAll, wait, wait, wait

Constructor Details

DefaultRepresentativeFraction

```
public DefaultRepresentativeFraction()
```

Creates a uninitialized representative fraction. The denominator is initially zero and the double value is NaN.

DefaultRepresentativeFraction

```
public DefaultRepresentativeFraction(long denominator)
```

Creates a new representative fraction from the specified denominator.

Parameters:

denominator - the denominator as a positive number, or 0 if unspecified.

Throws:

IllegalArgumentException - if the given value is negative.

DefaultRepresentativeFraction

```
public DefaultRepresentativeFraction(RepresentativeFraction object)
```

Constructs a new representative fraction initialized to the value of the given object.

Note on properties validation

This constructor does not verify the property values of the given metadata (e.g. whether it contains unexpected negative values). This is because invalid metadata exist in practice, and verifying their validity in this copy constructor is often too late. Note that this is not the only hole, as invalid metadata instances can also be obtained by unmarshalling an invalid XML document.

Parameters:

`object` - the metadata to copy values from, or `null` if none.

Method Details

castOrCopy

```
public static DefaultRepresentativeFraction castOrCopy
(RepresentativeFraction object)
```

Returns a SIS metadata implementation with the same values as the given arbitrary implementation. If the given object is `null`, then this method returns `null`. Otherwise if the given object is already a SIS implementation, then the given object is returned unchanged. Otherwise a new SIS implementation is created and initialized to the property values of the given object, using a *shallow* copy operation (i.e. properties are not cloned).

Parameters:

`object` - the object to get as a SIS implementation, or `null` if none.

Returns:

a SIS implementation containing the values of the given object (may be the given object itself), or `null` if the argument was null.

getDenominator

```
@ValueRange(minimum=0.0)
public long getDenominator()
```

Returns the denominator of this representative fraction.

Specified by:

`getDenominator` in interface `RepresentativeFraction`

Returns:

the denominator.

setDenominator

```
public void setDenominator(long denominator)
```

Sets the denominator value.

Parameters:

`denominator` - the new denominator value, or 0 if none.

Throws:

`IllegalArgumentException` - if the given value is negative.

setScale

```
public void setScale(double scale)
```

Sets the denominator from a scale in the (0 … 1] range. The denominator is computed by `round(1 / scale)`.

The equivalent of a `getScale()` method is `doubleValue()`.

Parameters:

`scale` - the scale as a number between 0 exclusive and 1 inclusive, or NaN.

Throws:

`IllegalArgumentException` - if the given scale is our of range.

doubleValue

```
public double doubleValue()
```

Returns the scale value of this representative fraction. This method is the converse of `setScale(double)`.

Specified by:

`doubleValue` in interface `RepresentativeFraction`

Specified by:

`doubleValue` in class `Number`

Returns:

the scale value of this representative fraction, or NaN if none.

floatValue

```
public float floatValue()
```

Returns the scale as a `float` type.

Specified by:

`floatValue` in class `Number`

Returns:

the scale.

longValue

```
public long longValue()
```

Returns 1 if the denominator is equal to 1, or 0 otherwise. This method is implemented that way because scales smaller than 1 can only be cast to 0, and NaN values are also represented by 0.

Specified by:

`longValue` in class `Number`

Returns:

1 if the denominator is 1, or 0 otherwise.

intValue

```
public int intValue()
```

Returns 1 if the denominator is equal to 1, or 0 otherwise. This method is implemented that way because scales smaller than 1 can only be cast to 0, and NaN values are also represented by 0.

Specified by:

`intValue` in class `Number`

Returns:

1 if the denominator is 1, or 0 otherwise.

isEmpty

```
public boolean isEmpty()
```

Returns `true` if no scale is defined. The following relationship shall hold:

```
assert isEmpty() == Double.isNaN(doubleValue());
```

Specified by:

`isEmpty` in interface `Emptiable`

Returns:

`true` if no scale is defined.

Since:

0.6

See Also:

`doubleValue()`, `floatValue()`

freeze

```
public void freeze()
```

Makes this representative fraction unmodifiable. After invocation to this method, any call to a setter method will throw an `UnmodifiableMetadataException`.

Since:

0.7

See Also:

`ModifiableMetadata.transitionTo(ModifiableMetadata.State)`

clone

```
public DefaultRepresentativeFraction clone()
```

Returns a modifiable copy of this representative fraction.

Overrides:

`clone` in class `Object`

Returns:

a modifiable copy of this representative fraction.

equals

```
public boolean equals(Object object)
```

Compares this object with the specified value for equality.

Specified by:

`equals` in interface `RepresentativeFraction`

Overrides:

`equals` in class `Object`

Parameters:

`object` - the object to compare with.

Returns:

`true` if both objects are equal.

hashCode

```
public int hashCode()
```

Returns a hash value for this representative fraction.

Specified by:

`hashCode` in interface `RepresentativeFraction`

Overrides:

`hashCode` in class `Object`

Returns:

A hash code value for this representative fraction.

toString

```
public String toString()
```

Returns a string representation of this scale, or `NaN` if undefined. If defined, the string representation uses the colon as in "1:20000".

Overrides:

`toString` in class `Object`

Returns:

a string representation of this scale.

getIdentifiers

```
public Collection<Identifier> getIdentifiers()
```

Returns all identifiers associated to this object, or an empty collection if none. Those identifiers are marshalled in XML as `id` or `uuid` attributes.

Specified by:

`getIdentifiers` in interface `IdentifiedObject`

Returns:

all identifiers associated to this object, or an empty collection if none.

See Also:

`DefaultCitation.getIdentifiers()`,
`DefaultObjective.getIdentifiers()`,

AbstractIdentifiedObject.getIdentifiers()

getIdentifierMap

```
public IdentifierMap getIdentifierMap()
```

Returns a map view of the identifiers collection as (*authority*, *code*) entries. That map is *live*: changes in the identifiers list will be reflected in the map, and conversely.

Specified by:

getIdentifierMap in interface IdentifiedObject

Returns:

the identifiers as a map of (*authority*, *code*) entries, or an empty map if none.