IBM® Netezza® Analytics
Release 11.x

IBM Netezza Analytics Map/Reduce API Reference

Part Number 00J2394-03 Rev. 2

Note: Before using this information and the product that it supports, read the information in "Notices and Trademarks" on page 180.

Contents

Preface

1 Class Documentation

Notices and Trademarks

Index

Preface

This guide provides an API reference for IBM Netezza Analytics map/reduce programmers.

Audience for This Guide

The *IBM Netezza Analytics Map/Reduce API Reference* is written for programmers who intend to cre-ate IBM Netezza Analytics map/reduce jobs for IBM Netezza Analytics. This guide does not provide a tutorial on Map Reduce concepts. More information about Map/Reduce can be found in the *Map/Reduce Developer's Guide.*

Purpose of This Guide

This guide describes the IBM Netezza Analytics map/reduce API, which is a part of IBM Netezza Ana-lytics. The map/reduce API provides programmatic access to the IBM Netezza Analytics map/reduce product for Java programmers.

Conventions

Note on Terminology: The terms User-Defined Analytic Process (UDAP) and Analytic Executable (AE) are synonymous.

The following conventions apply:

Italics for emphasis on terms and user-defined values, such as user input.
Upper case for SQL commands, for example, INSERT or DELETE.
Bold for command line input, for example, **nzsystem stop**.
Bold to denote parameter names, argument names, or other named references.
Angle brackets (< >) to indicate a placeholder (variable) that should be replaced with actual text, for example, **nzmat <- nz.matrix("<matrix_name>")**.
A single backslash ("\") at the end of a line of code to denote a line continuation. Omit the back-slash when using the code at the command line, in a SQL command, or in a file.
When referencing a sequence of menu and submenu selections, the ">" character denotes the different menu options, for example *Menu Name* > *Submenu Name* > *Selection*.

If You Need Help

If you are having trouble using the IBM Netezza appliance, IBM Netezza Analytics or any of its com-ponents:

Retry the action, carefully following the instructions in the documentation.
Go to the IBM Support Portal at http://www.ibm.com/support. Log in using your IBM ID and password. You can search the Support Portal for solutions. To submit a support re-quest, click the 'Service Requests & PMRs' tab.
If you have an active service contract maintenance agreement with IBM, you can contact customer support teams via telephone. For individual countries, please visit the Technical

Support section of the IBM Directory of worldwide contacts http://www14.software.ibm.com/webapp/set2/sas/f/handbook/contacts.html#phone.

Comments on the Documentation

We welcome any questions, comments, or suggestions that you have for the IBM Netezza document-ation. Please send us an e-mail message at netezza-doc@wwpdl.vnet.ibm.com and include the fol-lowing information:

The name and version of the manual that you are using
Any comments that you have about the manual
Your name, address, and phone number

We appreciate your comments.

CHAPTER 1
Class Documentation

BooleanWritable Class Reference

A Writable for booleans.

Inherits Writable

Public Member Functions

BooleanWritable()
BooleanWritable(boolean value)
boolean equals(Object o)
Returns true iff o is a BooleanWritable with the same value.

Boolean get()
Returns the value of the BooleanWritable .

List<Class<?> > getStorageTypesList()
int hashCode()
void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

void set(Boolean value)
Set the value of the BooleanWritable .

String toString()
void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Detailed Description

A Writable for booleans.

Public Member Function Documentation

BooleanWritable()

BooleanWritable(boolean value)

boolean equals(Object o)
Returns true iff o is a BooleanWritable with the same value.

Boolean get()
Returns the value of the BooleanWritable .

List<Class<?> > getStorageTypesList()

Returns
list of classes of storage types. These classes are used by the framework for automatic con-version from database fields and for setting column types of output table.

int hashCode()

void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

Parameters
RecordInput in
RecordInput to read this object from.

Exceptions
IOException

void set(Boolean value)
Set the value of the BooleanWritable .

String toString()

void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Parameters
RecordOutput out
RecordOutput to write this object into.

Exceptions

IOException

Configurable Interface Reference

Something that may be configured with a Configuration .

Public Member Functions

void setConf(Configuration conf)
Set the configuration to be used by this object.

Configuration getConf()
Return the configuration used by this object.

Detailed Description

Something that may be configured with a Configuration .

Public Member Function Documentation

void setConf(Configuration conf)
Set the configuration to be used by this object.

Configuration getConf()
Return the configuration used by this object.

Returns
Configuration

Configuration Class Reference

Provides access to configuration parameters.

Inherits CoreWritable

Public Member Functions

[instance initializer]
public<U> Class<? extends U> getClass(String name, Class<?extends U > defaultValue, Class< U > xface) Get the value of the name property as a Class implementing the interface specified by xface.

public<T extends Enum<T> > T getEnum(String name, T defaultValue) Return value matching this enumerated type.

public<T extends Enum<T> > void setEnum(String name, T value) Set the value of the name property to the given type.

void addResource(String name)

Add a configuration resource.

void addResource(URL url)
Add a configuration resource.

void addResource(File file)
Add a configuration resource.

void addResource(InputStream
in) Add a configuration resource.

void clear()
Clears all keys from the configuration.

Configuration(boolean loadDefaults)
A new configuration where the behavior of reading from the default resources can be turned off.

Configuration()
A new configuration.

Configuration(Configuration other)
A new configuration with the same settings cloned from another.

String get(String name)
Get the value of the name property, null if no such property exists.

String get(String name, String defaultValue)
Get the value of the name property.

boolean getBoolean(String name, boolean defaultValue)
Get the value of the name property as a boolean.

Class<?> getClass(String name, Class<?> defaultValue)
Get the value of the name property as a Class.

Class<?> getClassByName(String
name) Load a class by name.

Class<?> [] getClasses(String name, Class<?>...defaultValue)
Get the value of the name property as an array of Class.

ClassLoader getClassLoader()
Get the ClassLoader for this job.

InputStream getConfResourceAsInputStream(String name)
Get an input stream attached to the configuration resource with the given name.

Reader getConfResourceAsReader(String name)
Get a Reader attached to the configuration resource with the given name.

File getFile(String dirsProp, String path)
Get a local file name under a directory named in *dirsProp* with the given *path*.

float getFloat(String name, float defaultValue)
Get the value of the name property as a float.

int getInt(String name, int defaultValue)
Get the value of the name property as an int.

long getLong(String name, long defaultValue)
Get the value of the name property as a long.

String getRaw(String name)
Get the value of the name property, without doing

.

URL getResource(String name)
Get the URL for the named resource.

Collection<String> getStringCollection(String name)
Get the comma delimited values of the name property as a collection of Strings.

String [] getStrings(String name)
Get the comma delimited values of the name property as an array of Strings.

String [] getStrings(String name, String...defaultValue)
Get the comma delimited values of the name property as an array of Strings.

Map<String,String> getValByRegex(String
regex) get keys matching the the regex

Iterator<Map.Entry<String, String> > iterator()
Get an Iterator to go through the list of String key-value pairs in the configuration.

void readFields(DataInput in)
Deserialize the fields of this object from in.

synchronized void reloadConfiguration()
Reload configuration from previously added resources.

void set(String name, String value)
Set the value of the name property.

void setBoolean(String name, boolean value) Set
the value of the name property to a boolean.

void setBooleanIfUnset(String name, boolean value)
Set the given property, if it is currently unset.

void setClass(String name, Class<?> theClass, Class<?> xface)
Set the value of the name property to the name of a theClass implementing the given interface xface.

void setClassLoader(ClassLoader classLoader)
Set the class loader that will be used to load the various objects.

void setFloat(String name, float value)
Set the value of the name property to a float.

void setIfUnset(String name, String value)
Sets a property if it is currently unset.

void setInt(String name, int value)
Set the value of the name property to an int.

void setLong(String name, long value)

Set the value of the name property to a long.

void setStrings(String name, String...values)
Set the array of string values for the name property as as comma delimited values.

int size()
Return the number of keys in the configuration.

String toString()
synchronized void unset(String name)
Unset a previously set property.

void write(DataOutput out)
Serialize the fields of this object to out.

void writeXml(OutputStream out)
Write out the non-default properties in this configuration to the give OutputStream .

Static Public Member Functions

static synchronized void addDefaultResource(String name) Add a default resource.

static void main(String[] args) For debugging.

Detailed Description

Provides access to configuration parameters.

Resources

Configurations are specified by resources. A resource contains a set of name/value pairs as XML data. Each resource is named by either a String or by a File . If named by a String, then the classpath is examined for a file with that name. If named by a File, then the local filesystem is ex-amined directly, without referring to the classpath.

Unless explicitly turned off, INZA MapReduce by default specifies two resources, loaded in-order from the classpath:

mapreduce-default.xml: Read-only defaults.
mapreduce-site.xml: Site-specific configuration for a given installation.

Applications may add additional resources, which are loaded subsequent to these resources in the order they are added.

Final Parameters

Configuration parameters may be declared *final*. Once a resource declares a value final, no sub-sequently-loaded resource can alter that value. For example, one might define a final parameter with:

```
<property>
  <name>mapreduce.deploy.dir</name>
  <value>/nz/export/jobs</value>
  <final>true</final>
</property>
```

Administrators typically define parameters as final in mapreduce-site.xml for values that user applications may not alter.

Variable Expansion

Value strings are first processed for *variable expansion*. The available properties are:

Other properties defined in this Configuration; and, if a name is undefined here,
Properties in System#getProperties().

For example, if a configuration resource contains the following property definitions:

```
<property>
  <name>basedir</name>
  <value>/user/${user.name}</value>
</property>

<property>
  <name>tempdir</name>
  <value>${basedir}/tmp</value>
</property>
```

When conf.get("tempdir") is called, then ${ *basedir*} will be resolved to another property in this Configura-tion , while ${ *user.name*} would then ordinarily be resolved to the value of the System property with that name.

Public Member Function Documentation

[instance initializer]

public<U> Class<? extends U> getClass(String name, Class<?extends U > defaultValue, Class< U > xface)

Get the value of the name property as a Class implementing the interface specified by xface.

Parameters

name

the class name.

defaultValue

default value.

xface

the interface implemented by the named class.

Returns

property value as a Class, or defaultValue.

If no such property is specified, then defaultValue is returned.

An exception is thrown if the returned class does not implement the named interface.

public<T extends Enum<T> > T getEnum(String name, T defaultValue) Return value matching this enumerated type.

Parameters

name Property name

defaultValue
Value returned if no mapping exists

Exceptions

IllegalArgumentException

public<T extends Enum<T> > void setEnum(String name, T value) Set the value of the name property to the given type.

Parameters

name property name

value new value

This is equivalent to set(<name>, value.toString()).

void addResource(String name)
Add a configuration resource.

Parameters

name
resource to be added, the classpath is examined for a file with that name.

The properties of this resource will override properties of previously added resources, unless they were marked

.

void addResource(URL url)
Add a configuration resource.

Parameters

url
url of the resource to be added, the local filesystem is examined directly to find the re-source, without referring to the classpath.

The properties of this resource will override properties of previously added resources, unless they were marked

.

void addResource(File file)
Add a configuration resource.

Parameters
file
file-path of resource to be added, the local filesystem is examined directly to find the resource, without referring to the classpath.

The properties of this resource will override properties of previously added resources, unless they were marked

.

void addResource(InputStream in) Add a configuration resource.

Parameters
in
InputStream to deserialize the object from.

The properties of this resource will override properties of previously added resources, unless they were marked

.

void clear()
Clears all keys from the configuration.

Configuration(boolean loadDefaults)
A new configuration where the behavior of reading from the default resources can be turned off.

Parameters
loadDefaults
specifies whether to load from the default files

If the parameter loadDefaults is false, the new instance will not load resources from the default files.

Configuration()
A new configuration.

Configuration(Configuration other)
A new configuration with the same settings cloned from another.

Parameters
Configuration other
the configuration from which to clone settings.

String get(String name)
Get the value of the name property, null if no such property exists.

Parameters
name
the property name.

Returns
the value of the name property, or null if no such property exists.

Values are processed for

before being returned.

String get(String name, String defaultValue)
Get the value of the name property.

Parameters
name property
name.

defaultValue
default value.

Returns
property value, or defaultValue if the property doesn't exist.

If no such property exists, then defaultValue is returned.

boolean getBoolean(String name, boolean defaultValue)
Get the value of the name property as a boolean.

Parameters
name property
name.

defaultValue
default value.

Returns
property value as a boolean, or defaultValue.

If no such property is specified, or if the specified value is not a valid boolean, then default-Value is returned.

Class<?> getClass(String name, Class<?> defaultValue)
Get the value of the name property as a Class.

Parameters
name
the class name.

defaultValue
default value.

Returns
property value as a Class, or defaultValue.

If no such property is specified, then defaultValue is returned.

Class<?> getClassByName(String name) Load a class by name.

Parameters
name
the class name.

Returns
the class object.

Exceptions
ClassNotFoundException

Class<?> [] getClasses(String name, Class<?>...defaultValue)
Get the value of the name property as an array of Class.

Parameters
name
the property name.

defaultValue
default value.

Returns
property value as a Class[], or defaultValue.

The value of the property specifies a list of comma separated class names. If no such property is spe-cified, then defaultValue is returned.

ClassLoader getClassLoader()
Get the ClassLoader for this job.

Returns
the correct class loader.

InputStream getConfResourceAsInputStream(String name)
Get an input stream attached to the configuration resource with the given name.

Parameters
name
configuration resource name.

Returns
an input stream attached to the resource.

Reader getConfResourceAsReader(String name)
Get a Reader attached to the configuration resource with the given name.

Parameters
name
configuration resource name.

Returns
a reader attached to the resource.

File getFile(String dirsProp, String path)
Get a local file name under a directory named in *dirsProp* with the given *path*.

Parameters
dirsProp
directory in which to locate the file.

path file-path.

Returns
local file under the directory with the given path.

If *dirsProp* contains multiple directories, then one is chosen based on *path*'s hash code. If the selected directory does not exist, an attempt is made to create it.

float getFloat(String name, float defaultValue)
Get the value of the name property as a float.

Parameters
name property name.

defaultValue
default value.

Returns
property value as a float, or defaultValue.

If no such property is specified, or if the specified value is not a valid float, then defaultValue is returned.

int getInt(String name, int defaultValue) Get the value of the name property as an int.

Parameters
name property name.

defaultValue
default value.

Returns
property value as an int, or defaultValue.

If no such property exists, or if the specified value is not a valid int, then defaultValue is returned.

long getLong(String name, long defaultValue)
Get the value of the name property as a long.

Parameters
name property
name.

defaultValue
default value.

Returns
property value as a long, or defaultValue.

If no such property is specified, or if the specified value is not a valid long, then defaultValue is re-turned.

String getRaw(String name)
Get the value of the name property, without doing

.

Parameters
name
the property name.

Returns
the value of the name property, or null if no such property exists.

URL getResource(String name)
Get the URL for the named resource.

Parameters
name resource
name.

Returns
the url for the named resource.

Collection<String> getStringCollection(String name)
Get the comma delimited values of the name property as a collection of Strings.

Parameters
name property
name.

Returns
property value as a collection of Strings.

If no such property is specified then empty collection is returned.

This is an optimized version of getStrings

String [] getStrings(String name)
Get the comma delimited values of the name property as an array of Strings.

Parameters
name property
name.

Returns
property value as an array of Strings, or null.

If no such property is specified then null is returned.

String [] getStrings(String name, String...defaultValue)
Get the comma delimited values of the name property as an array of Strings.

Parameters
name property
name.

defaultValue The
default value

Returns
property value as an array of Strings, or default value.

If no such property is specified then default value is returned.

Map<String,String> getValByRegex(String regex) get keys matching the the regex

Parameters
- **regex**

Returns
Map<String,String> with matching keys

Iterator<Map.Entry<String, String> > iterator()
Get an Iterator to go through the list of String key-value pairs in the configuration.

Returns
an iterator over the entries.

void readFields(DataInput in)

Deserialize the fields of this object from in.

Parameters

in

DataInput to deseriablize this object from.

Exceptions

IOException

For efficiency, implementations should attempt to re-use storage in the existing object where possible.

synchronized void reloadConfiguration()

Reload configuration from previously added resources.

This method will clear all the configuration read from the added resources, and final parameters. This will make the resources to be read again before accessing the values. Values that are added via set methods will overlay values read from the resources.

void set(String name, String value)

Set the value of the name property.

Parameters

name property name.

value property value.

void setBoolean(String name, boolean value) Set the value of the name property to a boolean.

Parameters

name property name.

value

boolean value of the property.

void setBooleanIfUnset(String name, boolean value) Set the given property, if it is currently unset.

Parameters

name property name

value

new value

void setClass(String name, Class<?> theClass, Class<?> xface)

Set the value of the name property to the name of a theClass implementing the given interface xface.

Parameters

name property
name.

theClass
property value.

xface
the interface implemented by the named class.

An exception is thrown if theClass does not implement the interface xface.

void setClassLoader(ClassLoader classLoader)
Set the class loader that will be used to load the various objects.

Parameters

classLoader
the new class loader.

void setFloat(String name, float value)
Set the value of the name property to a float.

Parameters

name property
name.

value property
value.

void setIfUnset(String name, String value)
Sets a property if it is currently unset.

Parameters

name
the property name

value
the new value

void setInt(String name, int value)
Set the value of the name property to an int.

Parameters

name property
name.

value
int value of the property.

void setLong(String name, long value)

Set the value of the name property to a long.

Parameters

name property
name.

value
long value of the property.

void setStrings(String name, String...values)

Set the array of string values for the name property as as comma delimited values.

Parameters

name property
name.

values
The values

int size()

Return the number of keys in the configuration.

Returns
number of keys in the configuration.

String toString()

synchronized void unset(String name)

Unset a previously set property.

void write(DataOutput out)

Serialize the fields of this object to out.

Parameters

out
DataOuput to serialize this object into.

Exceptions

IOException

void writeXml(OutputStream out)

Write out the non-default properties in this configuration to the give OutputStream .

Parameters

out
the output stream to write to.

Static Public Member Function Documentation

static synchronized void addDefaultResource(String name) Add a default resource.

Parameters

name

file name. File should be present in the classpath.

Resources are loaded in the order of the resources added.

static void main(String[] args) For debugging.

List non-default properties to the terminal and exit.

Configured Class Reference

Base class for things that may be configured with a Configuration .

Inherits Configurable

Public Member Functions

Configured()
Construct a Configured .

Configured(Configuration conf) Construct a Configured .

Configuration getConf()
Return the configuration used by this object.

void setConf(Configuration conf)
Set the configuration to be used by this object.

Detailed Description

Base class for things that may be configured with a Configuration .

Public Member Function Documentation

Configured()
Construct a Configured .

Configured(Configuration conf) Construct a Configured .

Configuration getConf()
Return the configuration used by this object.

Returns
Configuration

void setConf(Configuration conf)
Set the configuration to be used by this object.

Context Class Reference

Inherits ReduceContext< KEYIN, VALUEIN, KEYOUT, VALUEOUT >

Public Member Functions

Context(Configuration conf, ReducerRecordReader< KEYIN, VALUEIN > reader, RecordWriter< KEYOUT, VALUEOUT > writer, StatusReporter reporter)

Public Member Function Documentation

Context(Configuration conf, ReducerRecordReader< KEYIN, VALUEIN > reader, RecordWriter< KEY-OUT, VALUEOUT > writer, StatusReporter reporter)

ConversionException Class Reference

An exception class used for signaling failures of automatic conversion mechanism.

Inherits IOException

Public Member Functions

ConversionException(String message)
Constructs an ConversionException with the specified detail message.

ConversionException(String message, Throwable cause)
Constructs an ConversionException with the specified detail message and cause.

ConversionException(Throwable cause)
Constructs an ConversionException with the specified cause and a detail message of (cause==null ? null : cause.toString()) (which typically contains the class and detail message of cause).

Detailed Description

An exception class used for signaling failures of automatic conversion mechanism.

Public Member Function Documentation

ConversionException(String message)
Constructs an ConversionException with the specified detail message.

ConversionException(String message, Throwable cause)
Constructs an ConversionException with the specified detail message and cause.

ConversionException(Throwable cause)
Constructs an ConversionException with the specified cause and a detail message of (cause==null ? null : cause.toString()) (which typically contains the class and detail message of cause).

CoreText Class Reference

This class stores text using standard UTF8 encoding.

Inherits CoreWritable

Public Member Functions

void append(byte[] utf8, int start, int len)
Append a range of bytes to the end of the given text.

int charAt(int position)
Returns the Unicode Scalar Value (32-bit integer value) for the character at position.

void clear()
Clear the string to empty.

CoreText(String string)
Construct from a string.

CoreText(byte[] utf8)
Construct from a byte array.

CoreText(CoreText utf8)
Construct from another text.

CoreText()
int find(String what)
int find(String what, int start)

Finds any occurence of what in the backing buffer, starting as position start.

byte [] getBytes()
Returns the raw bytes; however, only data up to getLength is valid.

int getLength()
Returns the number of bytes in the byte array.

void readFields(DataInput
in) deserialize

void set(String string)
Set to contain the contents of a string.

void set(byte[] utf8, int start, int
len) Set the Text to range of bytes.

void set(byte[] utf8) Set
to a utf8 byte array.

void set(CoreText
other) copy a text.

String toString()
Convert text back to string.

void write(DataOutput out)
serialize write this object to out length uses zero-compressed encoding

Static Public Member Functions

static int bytesToCodePoint(ByteBuffer bytes)
Returns the next code point at the current position in the buffer.

static String decode(byte[] utf8)
STATIC UTILITIES FROM HERE DOWN.

static String decode(byte[] utf8, int start, int length, boolean replace)
Converts the provided byte array to a String using the UTF-8 encoding.

static String decode(byte[] utf8, int start, int length)
static ByteBuffer encode(String string)
Converts the provided String to bytes using the UTF-8 encoding.

static ByteBuffer encode(String string, boolean replace) Converts
the provided String to bytes using the UTF-8 encoding.

static String readString(DataInput in)
Read a UTF8 encoded string from in.

static void skip(DataInput in)
Skips over one Text in the input.

static int utf8Length(String string)
For the given string, returns the number of UTF-8 bytes required to encode the string.

static void validateUTF8(byte[] utf8)
Check if a byte array contains valid utf-8.

static void validateUTF8(byte[] utf8, int start, int len) Check to see if a byte array is valid utf-8.

static int writeString(DataOutput out, String s) Write a UTF8 encoded string to out.

Detailed Description

This class stores text using standard UTF8 encoding.

It provides methods to serialize, deserialize, and compare texts at byte level. The type of length is integer and is serialized using zero-compressed format.

In addition, it provides methods for string traversal without converting the byte array to a string.

Also includes utilities for serializing/deserialing a string, coding/decoding a string, checking if a byte array contains valid UTF8 code, calculating the length of an encoded string.

Public Member Function Documentation

void append(byte[] utf8, int start, int len)
Append a range of bytes to the end of the given text.

Parameters

utf8
the data to copy from

start
the first position to append from utf8

len
the number of bytes to append

int charAt(int position)
Returns the Unicode Scalar Value (32-bit integer value) for the character at position.

Returns
the Unicode scalar value at position or -1 if the position is invalid or points to a trailing byte

Note that this method avoids using the converter or doing String instatiation

void clear()
Clear the string to empty.

CoreText(String string)
Construct from a string.

CoreText(byte[] utf8)
Construct from a byte array.

CoreText(CoreText utf8)
Construct from another text.

CoreText()

int find(String what)

int find(String what, int start)
Finds any occurence of what in the backing buffer, starting as position start.

> Returns
> byte position of the first occurence of the search string in the UTF-8 buffer or -1 if not found

The starting position is measured in bytes and the return value is in terms of byte position in the buffer. The backing buffer is not converted to a string for this operation.

byte [] getBytes()
Returns the raw bytes; however, only data up to getLength is valid.

int getLength()
Returns the number of bytes in the byte array.

void readFields(DataInput in) deserialize

void set(String string)
Set to contain the contents of a string.

void set(byte[] utf8, int start, int len) Set the Text to range of bytes.

> Parameters
> > **utf8**
> > the data to copy from
> >
> > **start**
> > the first position of the new string
> >
> > **len**
> > the number of bytes of the new string

void set(byte[] utf8)
Set to a utf8 byte array.

**void set(CoreText
other)** copy a text.

String toString()
Convert text back to string.

See Also
java.lang.Object.toString()

void write(DataOutput out)
serialize write this object to out length uses zero-compressed encoding

See Also
write

Static Public Member Function Documentation

static int bytesToCodePoint(ByteBuffer bytes)
Returns the next code point at the current position in the buffer.

The buffer's position will be incremented. Any mark set on this buffer will be changed by this method!

static String decode(byte[] utf8)
STATIC UTILITIES FROM HERE DOWN.

Converts the provided byte array to a String using the UTF-8 encoding. If the input is mal-formed, replace by a default value.

static String decode(byte[] utf8, int start, int length, boolean replace)
Converts the provided byte array to a String using the UTF-8 encoding.

If replace is true, then malformed input is replaced with the substitution character, which is U+FFFD. Otherwise the method throws a MalformedInputException.

static String decode(byte[] utf8, int start, int length)

static ByteBuffer encode(String string)
Converts the provided String to bytes using the UTF-8

encoding. ▲ Returns

ByteBuffer: bytes stores at ByteBuffer.array() and length is ByteBuffer.limit()

If the input is malformed, invalid chars are replaced by a default value.

static ByteBuffer encode(String string, boolean replace)
Converts the provided String to bytes using the UTF-8 encoding.

Returns
ByteBuffer: bytes stores at ByteBuffer.array() and length is ByteBuffer.limit()

If replace is true, then malformed input is replaced with the substitution character, which is U+FFFD. Otherwise the method throws a MalformedInputException.

static String readString(DataInput in)
Read a UTF8 encoded string from in.

static void skip(DataInput in)
Skips over one Text in the input.

static int utf8Length(String string)
For the given string, returns the number of UTF-8 bytes required to encode the string.

Parameters
string
text to encode

Returns
number of UTF-8 bytes required to encode

static void validateUTF8(byte[] utf8)
Check if a byte array contains valid utf-8.

Parameters
utf8
byte array

Exceptions
MalformedInputException

static void validateUTF8(byte[] utf8, int start, int len) Check to see if a byte array is valid utf-8.

Parameters
utf8
the array of bytes

start
the offset of the first byte in the array

len

the length of the byte sequence

- Exceptions
  - MalformedInputException

static int writeString(DataOutput out, String s) Write a UTF8 encoded string to out.

CoreWritable Interface Reference

A serializable object which implements a simple, efficient, serialization protocol, based on DataIn-put and DataOutput .

Public Member Functions

void write(DataOutput out)
Serialize the fields of this object to out.

void readFields(DataInput in)
Deserialize the fields of this object from in.

Detailed Description

A serializable object which implements a simple, efficient, serialization protocol, based on DataIn-put and DataOutput .

Any key or value type in the Map-Reduce framework implements this interface.

Implementations typically implement a static read(DataInput) method which constructs a new in-stance, calls readFields and returns the instance.

Example:

```
public class MyWritable implements Writable {
    Some data private
 int counter; private
 long timestamp;

 public void write(DataOutput out) throws IOException {
   out.writeInt(counter);
   out.writeLong(timestamp);
 }

 public void readFields(DataInput in) throws
   IOException { counter = in.readInt();
   timestamp = in.readLong();
 }

 public static MyWritable read(DataInput in) throws IOException {
    MyWritable w = new MyWritable();
    w.readFields(in);
    return w;
  }
}
```

Public Member Function Documentation

void write(DataOutput out)
Serialize the fields of this object to out.

Parameters
out
DataOuput to serialize this object into.

Exceptions
IOException

void readFields(DataInput in)
Deserialize the fields of this object from in.

Parameters
in
DataInput to deseriablize this object from.

Exceptions
IOException

For efficiency, implementations should attempt to re-use storage in the existing object where possible.

Counter Class Reference

A named counter that tracks the progress of a map/reduce job.

Inherits CoreWritable

Public Member Functions

synchronized boolean equals(Object genericRight)
synchronized String getDisplayName()
Get the name of the counter.

synchronized String getName()
synchronized long getValue()
What is the current value of this counter?

synchronized int hashCode()
synchronized void increment(long incr)

Increment this counter by the given value.

synchronized void readFields(DataInput in)
Read the binary representation of the counter.

synchronized void setValue(long value)
Set this counter by the given value.

synchronized void write(DataOutput out) Write
the binary representation of the counter.

Protected Member Functions

Counter()
Counter(String name, String displayName)
synchronized void setDisplayName(String displayName)

Detailed Description

A named counter that tracks the progress of a map/reduce job.

Counters represent global counters, defined either by the Map-Reduce framework or applications. Each Counter is named by an Enum and has a long for the value.

Counters are bunched into Groups, each comprising of counters from a particular Enum class.

Public Member Function Documentation

synchronized boolean equals(Object genericRight)

synchronized String getDisplayName()
Get the name of the counter.

Returns
the user facing name of the counter

synchronized String getName()

synchronized long getValue()
What is the current value of this counter?

Returns
the current value

synchronized int hashCode()

synchronized void increment(long incr)
Increment this counter by the given value.

Parameters

incr

the value to increase this counter by

synchronized void readFields(DataInput in)

Read the binary representation of the counter.

synchronized void setValue(long value)

Set this counter by the given value.

Parameters

value

the value to set

synchronized void write(DataOutput out)

Write the binary representation of the counter.

Protected Member Function Documentation

Counter()

Counter(String name, String displayName)

synchronized void setDisplayName(String displayName)

CounterGroup Class Reference

A group of Counter s that logically belong together.

Inherits CoreWritable

Public Member Functions

synchronized boolean equals(Object genericRight)
synchronized Counter findCounter(String counterName)
synchronized String getDisplayName()
Get the display name of the group.

synchronized String getName()
Get the internal name of the group.

synchronized int hashCode()
synchronized void incrAllCounters(CounterGroup rightGroup)
synchronized Iterator<Counter> iterator()
synchronized void readFields(DataInput in)

Deserialize the fields of this object from in.

synchronized int size()
Returns the number of counters in this group.

synchronized void write(DataOutput out)
Serialize the fields of this object to out.

synchronized void addCounter(Counter counter)

Protected Member Functions

CounterGroup(String name)
CounterGroup(String name, String displayName)
Counter findCounter(String counterName, String displayName) Internal to find a counter in a group.

Detailed Description

A group of Counter s that logically belong together.

Typically, it is an Enum subclass and the counters are the values.

Public Member Function Documentation

synchronized boolean equals(Object genericRight)

synchronized Counter findCounter(String counterName)
Returns
Counter

synchronized String getDisplayName()
Get the display name of the group.

Returns
the human readable name

synchronized String getName() Get the internal name of the group.

Returns
the internal name

synchronized int hashCode()

synchronized void incrAllCounters(CounterGroup rightGroup)

synchronized Iterator<Counter> iterator()

Returns

Counter

synchronized void readFields(DataInput in)

Deserialize the fields of this object from in.

Parameters

in

DataInput to deseriablize this object from.

Exceptions

IOException

For efficiency, implementations should attempt to re-use storage in the existing object where possible.

synchronized int size()

Returns the number of counters in this group.

synchronized void write(DataOutput out)

Serialize the fields of this object to out.

Parameters

out

DataOuput to serialize this object into.

Exceptions

IOException

synchronized void addCounter(Counter counter)

Protected Member Function Documentation

CounterGroup(String name)

CounterGroup(String name, String displayName)

Counter findCounter(String counterName, String displayName) Internal to find a counter in a group.

Parameters

counterName

the name of the counter

displayName

the display name of the counter

Returns

Counter

the counter that was found or added

CounterReporter Class Reference

Inherits StatusReporter

Public Member Functions

CounterReporter(Counters counters)
Counter getCounter(String group, String name)
Get the Counter of the given group with the given name.

Counter getCounter(Enum<?> key)
Get the Counter identified by the given Enum type.

Public Member Function Documentation

CounterReporter(Counters counters)

Counter getCounter(String group, String name)
Get the Counter of the given group with the given name.

Parameters
group counter
group

name counter
name

Returns
Counter

the Counter of the given group/name.

Counter getCounter(Enum<?> key)
Get the Counter identified by the given Enum type.

Parameters
key
key to identify the counter

Returns
Counter

the Counter identified by the given key

Counters Class Reference

Inherits CoreWritable

Public Member Functions

synchronized int countCounters()
Returns the total number of counters, by summing the number of counters in each group.

Counters()
boolean equals(Object genericRight)
Counter findCounter(String groupName, String counterName)
synchronized Counter findCounter(Enum<?>
key) Find the counter for the given enum.

synchronized CounterGroup getGroup(String groupName)
Returns the named counter group, or an empty group if there is none with the specified name.

synchronized Collection<String> getGroupNames()
Returns the names of all counter classes.

int hashCode()
synchronized void incrAllCounters(Counters other)
Increments multiple counters by their amounts in another Counters instance.

Iterator<CounterGroup> iterator()
synchronized void readFields(DataInput
in) Read a set of groups.

synchronized String toString()
Return textual representation of the counter values.

synchronized void write(DataOutput
out) Write the set of groups.

Public Member Function Documentation

synchronized int countCounters()
Returns the total number of counters, by summing the number of counters in each group.

Counters()

boolean equals(Object genericRight)

Counter findCounter(String groupName, String counterName)
Returns
Counter

synchronized Counter findCounter(Enum<?> key)

Find the counter for the given enum.

Parameters

key

the counter key

Returns

Counter

the matching counter object

The same enum will always return the same counter.

synchronized CounterGroup getGroup(String groupName)

Returns the named counter group, or an empty group if there is none with the specified name.

Returns

CounterGroup

synchronized Collection<String> getGroupNames()

Returns the names of all counter classes.

Returns

Set of counter names.

int hashCode()

synchronized void incrAllCounters(Counters other)

Increments multiple counters by their amounts in another Counters instance.

Parameters

Counters other

the other Counters instance

Iterator<CounterGroup> iterator()

Returns

CounterGroup

synchronized void readFields(DataInput in) Read a set of groups.

synchronized String toString()

Return textual representation of the counter values.

synchronized void write(DataOutput out) Write the set of groups.

The external format is: groups (groupName group)*

i.e. the number of groups followed by 0 or more groups, where each group is of the

form: groupDisplayName counters (false | true counter)*

where each counter is of the form:

name (false | true displayName) value

CountersUtils Class Reference

Static Public Member Functions

static Counters readCounters(File f)
Read Counters from the given file.

Static Public Member Function Documentation

static Counters readCounters(File f)
Read Counters from the given file.

Parameters
f
file to read

Returns
Counters

counters

DataInputBuffer Class Reference

A reusable DataInput implementation that reads from an in-memory buffer.

Inherits DataInputStream

Public Member Functions

DataInputBuffer()
Constructs a new empty buffer.

byte [] getData()
int getLength()
Returns the length of the input.

int getPosition()

Returns the current position in the input.

void reset(byte[] input, int start, int length)
Resets the data that the buffer reads.

void reset(byte[] input, int length)
Resets the data that the buffer reads.

Detailed Description

A reusable DataInput implementation that reads from an in-memory buffer.

This saves memory over creating a new DataInputStream and ByteArrayInputStream each time data is read.

Typical usage is something like the following:

```
DataInputBuffer buffer = new DataInputBuffer();
 while (... loop condition ...) {
   byte[] data = ... get data ...;
   int dataLength = ... get data length ...;
   buffer.reset(data, dataLength);
   ... read buffer using DataInput methods ...
 }
```

Public Member Function Documentation

DataInputBuffer()
Constructs a new empty buffer.

byte [] getData()

int getLength()
Returns the length of the input.

int getPosition()
Returns the current position in the input.

void reset(byte[] input, int start, int length)
Resets the data that the buffer reads.

void reset(byte[] input, int length)

Resets the data that the buffer reads.

DataOutputBuffer Class Reference

A reusable DataOutput implementation that writes to an in-memory buffer.

Inherits DataOutputStream

Public Member Functions

DataOutputBuffer() Constructs
a new empty buffer.

DataOutputBuffer(int size)
byte [] getData()
Returns the current contents of the buffer.

int getLength()
Returns the length of the valid data currently in the buffer.

DataOutputBuffer reset()
Resets the buffer to empty.

void write(DataInput in, int length)
Writes bytes from a DataInput directly into the buffer.

void writeTo(OutputStream
out) Write to a file stream.

Detailed Description

A reusable DataOutput implementation that writes to an in-memory buffer.

This saves memory over creating a new DataOutputStream and ByteArrayOutputStream each time data is written.

Typical usage is something like the following:

```
DataOutputBuffer buffer = new DataOutputBuffer();
 while (... loop condition ...) {
   buffer.reset();
   ... write buffer using DataOutput methods ...
   byte[] data = buffer.getData();
   int dataLength = buffer.getLength();
   ... write data to its ultimate destination ...
 }
```

Public Member Function Documentation

DataOutputBuffer() Constructs
a new empty buffer.

DataOutputBuffer(int size)

byte [] getData()
Returns the current contents of the

buffer. Data is only valid to getLength .

int getLength()
Returns the length of the valid data currently in the buffer.

DataOutputBuffer reset()
Resets the buffer to empty.

Returns
DataOutputBuffer

void write(DataInput in, int length)
Writes bytes from a DataInput directly into the buffer.

**void writeTo(OutputStream
out)** Write to a file stream.

DBCombinerRecordReader< K, V > Class Reference

The ReducerRecordReader implementation for Combiner.

Inherits DBReducerRecordReader< K, V >

Protected Member Functions

void incInputGroupsCounter()
void setCounters(TaskAttemptContext context)
void setKeyValueClasses(TaskAttemptContext context)

Detailed Description

The ReducerRecordReader implementation for Combiner.

Protected Member Function Documentation

void incInputGroupsCounter()

void setCounters(TaskAttemptContext context)

void setKeyValueClasses(TaskAttemptContext context)

DBCombinerRecordWriter< K, V > Class Reference

The RecordWriter implementation for Combiner.

Inherits DBRecordWriter< K, V >

Protected Member Functions

void setCounters(TaskAttemptContext context)

Detailed Description

The RecordWriter implementation for Combiner.

Protected Member Function Documentation

void setCounters(TaskAttemptContext context)

DBMapperRecordReader< K, V > Class Reference

The MapperRecordReader implementation.

Inherits org::netezza::inza::mr::mapreduce::MapperRecordReader< K, V >

Public Member Functions

K getCurrentKey()
V getCurrentValue()
void initialize(Nzae ae, TaskAttemptContext context)
boolean nextKeyValue()

Detailed Description

The MapperRecordReader implementation.

Public Member Function Documentation

K getCurrentKey()

V getCurrentValue()

void initialize(Nzae ae, TaskAttemptContext context)

boolean nextKeyValue()

DBMapperRecordWriter< K, V > Class Reference

The RecordWriter implementation for Mapper .

Inherits DBRecordWriter< K, V >

Protected Member Functions

void setCounters(TaskAttemptContext context)

Detailed Description

The RecordWriter implementation for Mapper .

Protected Member Function Documentation

void setCounters(TaskAttemptContext context)

DBPartitionerRecordReader< K, V > Class Reference

The PartitionerRecordReader implementation.

Inherits org::netezza::inza::mr::mapreduce::PartitionerRecordReader< K, V >

Public Member Functions

K getCurrentKey()
V getCurrentValue()
void initialize(Nzae ae, JobContext context)
boolean nextKeyValue()

Detailed Description

The PartitionerRecordReader implementation.

Public Member Function Documentation

K getCurrentKey()

V getCurrentValue()

void initialize(Nzae ae, JobContext context)

boolean nextKeyValue()

DBRecordWriter< K, V > Class Reference

An abstract RecordWriter class.

Inherits org::netezza::inza::mr::mapreduce::RecordWriter< K, V >

Public Member Functions

void close(TaskAttemptContext context)
final void initialize(Nzae ae, TaskAttemptContext context)
void write(K key, V value)

Protected Member Functions

abstract void setCounters(TaskAttemptContext context)

Detailed Description

An abstract RecordWriter class.

Public Member Function Documentation

void close(TaskAttemptContext context)

final void initialize(Nzae ae, TaskAttemptContext context)

void write(K key, V value)

Protected Member Function Documentation

abstract void setCounters(TaskAttemptContext context)

DBReducerRecordReader< K, V > Class Reference

The ReducerRecordReader implementation for Reducer .

Inherits org::netezza::inza::mr::mapreduce::ReducerRecordReader< K, V >

Public Member Functions

K getCurrentKey()
V getCurrentValue()
boolean hasNextValue()
void initialize(Nzae ae, TaskAttemptContext context)
boolean nextKey()
boolean nextValue()

Protected Member Functions

void incInputGroupsCounter()
void incInputRecordsCounter()
void setCounters(TaskAttemptContext context)
void setKeyValueClasses(TaskAttemptContext context)

Detailed Description

The ReducerRecordReader implementation for Reducer .

Public Member Function Documentation

K getCurrentKey()

V getCurrentValue()

boolean hasNextValue()

void initialize(Nzae ae, TaskAttemptContext context)

boolean nextKey()

boolean nextValue()

Protected Member Function Documentation

void incInputGroupsCounter()

void incInputRecordsCounter()

void setCounters(TaskAttemptContext context)

void setKeyValueClasses(TaskAttemptContext context)

DBReducerRecordWriter< K, V > Class Reference

The RecordWriter implementation for Reducer .

Inherits DBRecordWriter< K, V >

Protected Member Functions

void setCounters(TaskAttemptContext context)

Detailed Description

The RecordWriter implementation for Reducer .

Protected Member Function Documentation

void setCounters(TaskAttemptContext context)

Deserializer< T > Interface Reference

Public Member Functions

void open(InputStream in)
void close()
T deserialize(T t)

Detailed Description

Provides a facility for deserializing objects of type <T> from an InputStream .

Deserializers are stateful, but must not buffer the input since other producers may read from the input between calls to deserialize(Object) .

Public Member Function Documentation

void open(InputStream in) Prepare the deserializer for reading.

void close()

Close the underlying input stream and clear up any resources.

T deserialize(T t)

Returns
the deserialized object

Deserialize the next object from the underlying input stream. If the object t is non-null then this deserializer *may* set its internal state to the next object read from the input stream. Other-wise, if the object t is null a new deserialized object will be created.

DoubleWritable Class Reference

A Writable for doubles.

Inherits Writable

Public Member Functions

DoubleWritable()
DoubleWritable(Double value)
boolean equals(Object o)
Returns true iff o is a DoubleWritable with the same value.

Double get()
Return the value of this DoubleWritable .

List<Class<?> > getStorageTypesList()
int hashCode()
void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

void set(Double value)
Set the value of this DoubleWritable .

String toString()
void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Detailed Description

A Writable for doubles.

Public Member Function Documentation

DoubleWritable()

DoubleWritable(Double value)

boolean equals(Object o)
Returns true iff o is a DoubleWritable with the same value.

Double get()
Return the value of this DoubleWritable .

List<Class<?> > getStorageTypesList()

Returns
list of classes of storage types. These classes are used by the framework for automatic conversion from database fields and for setting column types of output table.

int hashCode()

void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

Parameters
RecordInput in
RecordInput to read this object from.

Exceptions
IOException

void set(Double value)
Set the value of this DoubleWritable .

String toString()

void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Parameters
RecordOutput out
RecordOutput to write this object into.

Exceptions
IOException

ExitCodeException Class Reference

This is an IOException with exit code added.

Inherits IOException

Public Member Functions

ExitCodeException(int exitCode, String message)
int getExitCode()

Detailed Description

This is an IOException with exit code added.

Public Member Function Documentation

ExitCodeException(int exitCode, String message)

int getExitCode()

FloatWritable Class Reference

A Writable for floats.

Inherits Writable

Public Member Functions

boolean equals(Object o)
Returns true iff o is a FloatWritable with the same value.

FloatWritable()
FloatWritable(Float value)
Float get()
Return the value of this FloatWritable .

List<Class<?> > getStorageTypesList()
int hashCode()
void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

void set(Float value)
Set the value of this floatWritable.

String toString()
void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Detailed Description

A Writable for floats.

Public Member Function Documentation

boolean equals(Object o)
Returns true iff o is a FloatWritable with the same value.

FloatWritable()

FloatWritable(Float value)

Float get()
Return the value of this FloatWritable .

List<Class<?> > getStorageTypesList()

Returns
list of classes of storage types. These classes are used by the framework for automatic conversion from database fields and for setting column types of output table.

int hashCode()

void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

Parameters
RecordInput in
RecordInput to read this object from.

Exceptions
IOException

void set(Float value)
Set the value of this floatWritable.

String toString()

void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Parameters
RecordOutput out
RecordOutput to write this object into.

Exceptions
IOException

GenericOptionsParser Class Reference

GenericOptionsParser is a utility to parse command line arguments generic to the INZA MapRe-duce framework.

Public Member Functions

GenericOptionsParser(Options opts, String[] args)
Create an options parser with the given options to parse the args.

GenericOptionsParser(String[] args) Create
an options parser to parse the args.

GenericOptionsParser(Configuration conf, Options options, String[] args)
Create a GenericOptionsParser to parse given options as well as generic MapReduce options.

GenericOptionsParser(Configuration conf, String[] args)
Create a GenericOptionsParser to parse only the generic MapReduce arguments.

CommandLine getCommandLine()
Returns the commons-cli CommandLine object to process the parsed arguments.

Configuration getConfiguration()
Get the modified configuration.

String [] getRemainingArgs()
Returns an array of Strings containing only application-specific arguments.

Static Public Member Functions

static URL [] getArchives(Configuration conf)
static URL [] getFiles(Configuration conf)
static URL [] getLibJars(Configuration conf)
static URL [] getURLs(String tmpfiles)
static URL [] getURLs(ArrayList< String > files)
static void printGenericCommandUsage(PrintStream out)
Print the usage message for generic command-line options supported.

Detailed Description

GenericOptionsParser is a utility to parse command line arguments generic to the INZA MapRe-duce framework.

GenericOptionsParser recognizes several standarad command line arguments, enabling applica-tions to easily specify additional configuration resources etc.

Generic Options

The supported generic options are:

```
-conf <configuration file>   specify a configuration file
-D <property=value>          use value for given property
```

```
-files <comma separated list of files>  specify comma separated
                files to be copied to a job-unique shared directory,
                making them available to all job's tasks.
-libjars <comma separated list of jars>  specify comma separated
                jar files to be copied to a job-unique shared directory,
                files are added to the tasks' ClassLoaders. It allows
                the job to use any external library that the job depends on.
-archives <comma separated list of archives> specify comma
                separated archives to be unarchived in a job-unique
                shared directory, making them available to all job's tasks.
```

Examples to get access to directory with shared files:

```
public static class MyMapper extends org.netezza.inza.mr.mapreduce.Mapper {
        protected void setup(Context context) {
                File sharedFilesDir = new File (context.getRunDir(), "files");
                   reading files from sharedFilesDir directory
                   ...
               File sharedArchivesDir = new File (context.getRunDir(), "archives");
                   reading archives from sharedArchivesDir directory
                   ...
        }
           other methods
           ...
}
```

```
public class MyMapper implements org.netezza.inza.mr.mapred.Mapper {
        public void configure(JobConf job) {
                File sharedFilesDir = new File (job.getRunDir(), "files");
                   reading files from sharedFilesDir directory
                   ...
                File sharedArchivesDir = new File (job.getRunDir(), "archives");
                   reading archives from sharedArchivesDir directory
                   ...
        }
           other methods
           ...
}
```

The general command line syntax is:

```
bin/mapreduce command [genericOptions] [commandOptions]
```

Generic command line arguments **might** modify Configuration objects, given to constructors.

The functionality is implemented using Commons CLI.

Examples:

```
$ bin/mapreduce jar test.jar MyJob -conf conf.xml args
```
submit a job MyJob from a jar file test.jar with configuration from conf.xml file

```
bin/mapreduce jar test.jar MyJob -D mapred.job.test.value=paul args
```
submit a job with a property mapred.job.test.value set to value paul

```
bin/mapreduce jar test.jar MyJob -libjars testlib.jar -archives test.tgz -files file.txt
args
```
job submission with libjars, files and archives

See Also

Tool

ToolRunner

Public Member Function Documentation

GenericOptionsParser(Options opts, String[] args)

Create an options parser with the given options to parse the args.

Parameters

opts

the options

args

the command line arguments

Exceptions

IOException

GenericOptionsParser(String[] args) Create an options parser to parse the args.

Parameters

args

the command line arguments

Exceptions

IOException

GenericOptionsParser(Configuration conf, Options options, String[] args)

Create a GenericOptionsParser to parse given options as well as generic MapReduce options.

Parameters

Configuration conf
the configuration to modify

options
options built by the caller

args
User-specified arguments

Exceptions
IOException

The resulting CommandLine object can be obtained by getCommandLine .

GenericOptionsParser(Configuration conf, String[] args)
Create a GenericOptionsParser to parse only the generic MapReduce arguments.

Parameters

Configuration conf
the Configuration to modify.

args
command-line arguments.

Exceptions
IOException

The array of string arguments other than the generic arguments can be obtained by getRemainingArgs .

CommandLine getCommandLine()
Returns the commons-cli CommandLine object to process the parsed arguments.

Returns
CommandLine representing list of arguments parsed against Options descriptor.

Note: If the object is created with GenericOptionsParser(Configuration, String[]) , then returned object will only contain parsed generic options.

Configuration getConfiguration()
Get the modified configuration.

Returns
Configuration

the configuration that has the modified parameters.

String [] getRemainingArgs()
Returns an array of Strings containing only application-specific arguments.

Returns
array of Strings containing the un-parsed arguments or **empty array** if commandLine was not defined.

Static Public Member Function Documentation

static URL [] getArchives(Configuration conf)

static URL [] getFiles(Configuration conf)

static URL [] getLibJars(Configuration conf)

static URL [] getURLs(String tmpfiles)

static URL [] getURLs(ArrayList< String > files)

static void printGenericCommandUsage(PrintStream out)
Print the usage message for generic command-line options supported.

Parameters
out
stream to print the usage message to.

HashPartitioner< K2, V2 > Class Reference

Partition keys by their Object#hashCode() .

Inherits org::netezza::inza::mr::mapred::Partitioner< K2, V2 >

Public Member Functions

void configure(JobConf job)
int getPartition(K2 key, V2 value, int numReduceTasks)
Use Object#hashCode() to partition.

Detailed Description

Partition keys by their Object#hashCode() .

Public Member Function Documentation

void configure(JobConf job)

int getPartition(K2 key, V2 value, int numReduceTasks) Use Object#hashCode() to partition.

IdentityMapper< K, V > Class Reference

Implements the identity function, mapping inputs directly to outputs.

Inherits MapReduceBase

Public Member Functions

void map(K key, V val, OutputCollector< K, V > output, Reporter reporter) The identify function.

Detailed Description

Implements the identity function, mapping inputs directly to outputs.

Public Member Function Documentation

void map(K key, V val, OutputCollector< K, V > output, Reporter reporter) The identify function.

Input key/value pair is written directly to output.

IdentityReducer< K, V > Class Reference

Performs no reduction, writing all input values directly to the output.

Inherits MapReduceBase

Public Member Functions

void reduce(K key, Iterator< V > values, OutputCollector< K, V > output, Reporter reporter) Writes all keys and values directly to output.

Detailed Description

Performs no reduction, writing all input values directly to the output.

Public Member Function Documentation

void reduce(K key, Iterator< V > values, OutputCollector< K, V > output, Reporter reporter) Writes all keys and values directly to output.

IllegalJobConfigurationException Class Reference

An exception class used for signaling illegal job configuration.

Public Member Functions

IllegalJobConfigurationException()
Constructs an IllegalJobConfigurationException with null as its error detail message.

IllegalJobConfigurationException(Throwable cause)
Constructs an IllegalJobConfigurationException with the specified cause and a detail message of (cause==null ? null : cause.toString()) (which typically contains the class and detail message of cause).

IllegalJobConfigurationException(String message)
Constructs an IllegalJobConfigurationException with the specified detail message.

IllegalJobConfigurationException(String message, Throwable cause)
Constructs an IllegalJobConfigurationException with the specified detail message and cause.

Detailed Description

An exception class used for signaling illegal job configuration.

Public Member Function Documentation

IllegalJobConfigurationException()
Constructs an IllegalJobConfigurationException with null as its error detail message.

IllegalJobConfigurationException(Throwable cause)
Constructs an IllegalJobConfigurationException with the specified cause and a detail message of (cause==null ? null : cause.toString()) (which typically contains the class and detail message of cause).

IllegalJobConfigurationException(String message)
Constructs an IllegalJobConfigurationException with the specified detail message.

IllegalJobConfigurationException(String message, Throwable cause)
Constructs an IllegalJobConfigurationException with the specified detail message and cause.

IntSumReducer< Key > Class Reference

A Reducer that sums int values.

Inherits org::netezza::inza::mr::mapreduce::Reducer< Key, IntWritable, Key, IntWritable >

Public Member Functions

void reduce(Key key, Iterable< IntWritable > values, Context context) Sums all values and writes one pair: <key, sum>.

Detailed Description

A Reducer that sums int values.

Public Member Function Documentation

void reduce(Key key, Iterable< IntWritable > values, Context context) Sums all values and writes one pair: <key, sum>.

IntWritable Class Reference

A Writable for ints.

Inherits Writable

Public Member Functions

boolean equals(Object o)
Returns true iff o is a IntWritable with the same value.

Integer get()
Return the value of this IntWritable .

List<Class<?> > getStorageTypesList()
Class<?> getTypeClass()
int hashCode()
IntWritable(Integer value)
IntWritable()
void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

void set(Integer value)
Set the value of this IntWritable .

String toString()
void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Detailed Description

A Writable for ints.

Public Member Function Documentation

boolean equals(Object o)
Returns true iff o is a IntWritable with the same value.

Integer get()
Return the value of this IntWritable .

List<Class<?> > getStorageTypesList()

Returns

list of classes of storage types. These classes are used by the framework for automatic con-version from database fields and for setting column types of output table.

Class<?> getTypeClass()

int hashCode()

IntWritable(Integer value)

IntWritable()

void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

Parameters

RecordInput in
RecordInput to read this object from.

Exceptions

IOException

void set(Integer value)
Set the value of this IntWritable .

String toString()

void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Parameters

RecordOutput out
RecordOutput to write this object into.

Exceptions

IOException

InverseMapper< K, V > Class Reference

A Mapper that swaps keys and values.

Inherits org::netezza::inza::mr::mapreduce::Mapper< K, V, V, K >

Public Member Functions

void map(K key, V value, Context context) The inverse function.

Detailed Description

A Mapper that swaps keys and values.

Public Member Function Documentation

void map(K key, V value, Context context) The inverse function.

Input keys and values are swapped.

Job Class Reference

The job submitter's view of the Job .

Inherits JobContext

Public Member Functions

String getCombineStreamCommand()
Get the combiner streaming command.

String getDatabaseName()
Get the database name for the job.

String getDeployDir()
Get the path to the directory where all jobs are deployed.

String [] getInputKeyColumnNames() Get the names of the input key columns.

String getInputTableName() Get the name of the input table.

String [] getInputValueColumnNames() Get the names of the input value columns.

boolean getIsStreaming()

Returns true if the job uses streaming.

String getMapStreamCommand() Get
the mapper streaming command.

String [] getOutputKeyColumnNames() Get
the names of the output key columns.

String getOutputTableName() Get
the name of the output table.

String [] getOutputValueColumnNames() Get
the names of the output value columns.

String getReduceStreamCommand()
Get the reducer streaming command.

Job(Configuration conf)
Constructs a job with the specified Configuration.

Job()
Constructs a job with the new Configuration.

Job(Configuration conf, String jobName)
Constructs a job with the specified Configuration and the given name.

void setBadRecordsLimit(int value)
Set the maximum number of bad records that the framework should skip.

void setCombineOutputKeyClass(Class<?> cls)
Set the key class for the combine output data.

void setCombineOutputKeyColumnSize(int id, int size)
Set the size for the combine output key column with a given id.

void setCombineOutputValueClass(Class<?> cls)
Set the value class for the combine output data.

void setCombineOutputValueColumnSize(int id, int size)
Set the size for the combine output value column with a given id.

void setCombinerClass(Class<?extends Reducer >
cls) Set the combiner class for the job.

void setDatabaseName(String database)
Set the database name for the input and output tables.

void setInputKeyColumnNames(String...colNames)
Set the names of the input key columns.

void setInputTableName(String inputTable)
Set the name of the input table.

void setInputValueColumnNames(String...colNames)
Set the names of the input value columns.

void setIsStreaming(boolean value)

Set whether the job uses streaming tasks.

void setJarByClass(Class<?> cls)
Set the Jar by finding where a given class came from.

void setJobName(String name)
Set the user-specified job name.

void setMapInputKeyClass(Class<?> cls)
Set the key class for the map input data.

void setMapInputValueClass(Class<?> cls)
Set the value class for the map input data.

void setMapOutputKeyClass(Class<?> cls)
Set the key class for the map output data.

void setMapOutputKeyColumnSize(int id, int size)
Set the size for the map output key column with a given id.

void setMapOutputValueClass(Class<?> cls)
Set the value class for the map output data.

void setMapOutputValueColumnSize(int id, int size)
Set the size for the map output value column with a given id.

void setMapperClass(Class<?extends Mapper >
cls) Set the Mapper for the job.

void setOutputKeyColumnNames(String...colNames)
Set the names of the output key columns.

void setOutputTableName(String outputTable)
Set the name of the output table.

void setOutputValueColumnNames(String...colNames)
Set the names of the output value columns.

void setPartitionerClass(Class<?extends Partitioner >
cls) Set the Partitioner partitioner class for the job.

void setReduceOutputKeyClass(Class<?> cls)
Set the key class for the reduce output data.

void setReduceOutputKeyColumnSize(int id, int size)
Set the size for the reduce output key column with a given id.

void setReduceOutputValueClass(Class<?> cls)
Set the value class for the reduce output data.

void setReduceOutputValueColumnSize(int id, int size)
Set the size for the reduce output value column with a given id.

void setReducerClass(Class<?extends Reducer >
cls) Set the Reducer for the job.

void setRunDir(String runDir)
Set the path to a directory from which the job is run.

void setRunDirCleanup(boolean value)

Set whether the framework should clean the run dir after the job completion.

void setSkipBadRecords(boolean value)
Set whether the framework should skip bad records.

void setNumDataslices(int
value) Set number of dataslices.

Detailed Description

The job submitter's view of the Job .

It allows the user to configure the job and then run it via JobRunner .

Public Member Function Documentation

String getCombineStreamCommand()
Get the combiner streaming command.

Returns
the combiner streaming command, or null if the combiner is not run via streaming

String getDatabaseName()
Get the database name for the job.

Returns
the database name for the job

String getDeployDir()
Get the path to the directory where all jobs are deployed.

Returns
the path to the deployment directory

String [] getInputKeyColumnNames()
Get the names of the input key columns.

Returns
the array with the names of the input key columns

String getInputTableName() Get
the name of the input table.

Returns
the name of the input table

String [] getInputValueColumnNames()
Get the names of the input value columns.

Returns
the array with the names of the input value columns

boolean getIsStreaming()
Returns true if the job uses streaming.

Returns
true if the job uses streaming

String getMapStreamCommand() Get
the mapper streaming command.

Returns
the mapper streaming command, or null if the mapper is not run via streaming

String [] getOutputKeyColumnNames()
Get the names of the output key columns.

Returns
the array with the names of the output key columns

String getOutputTableName() Get
the name of the output table.

Returns
the name of the output table

String [] getOutputValueColumnNames()
Get the names of the output value columns.

Returns
the array with the names of the output value columns

String getReduceStreamCommand()
Get the reducer streaming command.

Returns
the reducer streaming command, or null if the reducer is not run via streaming

Job(Configuration conf)
Constructs a job with the specified Configuration.

Parameters
► **Configuration** conf

configuration

Job()

Constructs a job with the new Configuration.

Job(Configuration conf, String jobName)

Constructs a job with the specified Configuration and the given name.

Parameters

Configuration conf

configuration

jobName

job's name

void setBadRecordsLimit(int value)

Set the maximum number of bad records that the framework should skip.

Parameters

value

the maximum number of bad records

void setCombineOutputKeyClass(Class<?> cls)

Set the key class for the combine output data.

Parameters

cls

the combine output key class

void setCombineOutputKeyColumnSize(int id, int size)

Set the size for the combine output key column with a given id.

Parameters

id

the id of the output column

size

the size of the output column

This method should be invoked for each variable-sized column.

void setCombineOutputValueClass(Class<?> cls)

Set the value class for the combine output data.

▲ Parameters

► **cls**

the combine output value class

void setCombineOutputValueColumnSize(int id, int size)
Set the size for the combine output value column with a given id.

Parameters
id
the id of the output column

size
the size of the output column

This method should be invoked for each variable-sized column.

void setCombinerClass(Class<?extends Reducer >
cls) Set the combiner class for the job.

Parameters
cls
the combiner to use

void setDatabaseName(String database)
Set the database name for the input and output tables.

Parameters
database
the database name for the job

void setInputKeyColumnNames(String...colNames)
Set the names of the input key columns.

Parameters
colNames
the names of the input key columns

void setInputTableName(String inputTable)
Set the name of the input table.

Parameters
inputTable
the name of the input table

void setInputValueColumnNames(String...colNames)
Set the names of the input value columns.

Parameters
colNames
the names of the input value columns

void setIsStreaming(boolean value)

Set whether the job uses streaming tasks.

Parameters

value

true if the job uses streaming

void setJarByClass(Class<?> cls)

Set the Jar by finding where a given class came from.

Parameters

cls

the example class

void setJobName(String name)

Set the user-specified job name.

Parameters

name

the job's new name.

void setMapInputKeyClass(Class<?> cls)

Set the key class for the map input data.

Parameters

cls

the map input key class

void setMapInputValueClass(Class<?> cls)

Set the value class for the map input data.

Parameters

cls

the map input value class

void setMapOutputKeyClass(Class<?> cls)

Set the key class for the map output data.

Parameters

cls

the map output key class

void setMapOutputKeyColumnSize(int id, int size)

Set the size for the map output key column with a given id.

Parameters

id

the id of the output column

size

the size of the output column

This method should be invoked for each variable-sized column.

void setMapOutputValueClass(Class<?> cls)

Set the value class for the map output data.

Parameters

cls

the map output value class

void setMapOutputValueColumnSize(int id, int size)

Set the size for the map output value column with a given id.

Parameters

id

the id of the output column

size

the size of the output column

This method should be invoked for each variable-sized column.

void setMapperClass(Class<?extends Mapper > cls) Set the Mapper for the job.

Parameters

cls

the Mapper to use

void setOutputKeyColumnNames(String...colNames)

Set the names of the output key columns.

Parameters

colNames

the names of the output key columns

void setOutputTableName(String outputTable)

Set the name of the output table.

Parameters

outputTable

the name of the output table

void setOutputValueColumnNames(String...colNames)
Set the names of the output value columns.

Parameters
colNames
the names of the output value columns

**void setPartitionerClass(Class<?extends Partitioner >
cls)** Set the Partitioner partitioner class for the job.

Parameters
cls
the Partitioner to use

void setReduceOutputKeyClass(Class<?> cls)
Set the key class for the reduce output data.

Parameters
cls
the reduce output key class

void setReduceOutputKeyColumnSize(int id, int size)
Set the size for the reduce output key column with a given id.

Parameters
id
the id of the output column

size
the size of the output column

This method should be invoked for each variable-sized column.

void setReduceOutputValueClass(Class<?> cls)
Set the value class for the reduce output data.

Parameters
cls
the reduce output value class

void setReduceOutputValueColumnSize(int id, int size)
Set the size for the reduce output value column with a given id.

Parameters
id

the id of the output column

size
the size of the output column

This method should be invoked for each variable-sized column.

**void setReducerClass(Class<?extends Reducer >
cls)** Set the Reducer for the job.

Parameters
cls
the Reducer to use

void setRunDir(String runDir)
Set the path to a directory from which the job is run.

Parameters
runDir
the path

void setRunDirCleanup(boolean value)
Set whether the framework should clean the run dir after the job completion.

Parameters
value
true if framework should clean the run dir, false otherwise.

void setSkipBadRecords(boolean value)
Set whether the framework should skip bad records.

Parameters
value
true if framework should skip bad records, false otherwise

**void setNumDataslices(int
value)** Set number of dataslices.

Parameters
value
number of dataslices

JobConf Class Reference

Inherits Configuration

Public Member Functions

int getBadRecordsLimit()
Get the maximum number of bad records that the framework should skip.

Class<?> getCombineInputKeyClass()
Get the key class for the combine input data.

Class<?> getCombineInputValueClass()
Get the value class for the combine input data.

Class<?> getCombineOutputKeyClass()
Get the key class for the combine output data.

List<Integer> getCombineOutputKeyColumnSizes()
Get the list of combine output key column sizes.

Class<?> getCombineOutputValueClass()
Get the value class for the combine output data.

List<Integer> getCombineOutputValueColumnSizes()
Get the list of combine output value column sizes.

Class<? extends Reducer> getCombinerClass()
Get the user-defined *combiner* class used to combine map-outputs before being sent to the re-ducers.

String getCombineStreamCommand()
Get the combiner streaming command.

String getDatabaseName()
Get the database name for the job.

String getDeployDir()
Get the path to the directory where all jobs are deployed.

String [] getInputKeyColumnNames() Get
the names of the input key columns.

String getInputTableName() Get
the name of the input table.

String [] getInputValueColumnNames() Get
the names of the input value columns.

boolean getIsStreaming()
Returns true if the job uses streaming.

String getJar()
Get the user jar for the map-reduce job.

String getJobName()
Get the user-specified job name.

Class<?> getMapInputKeyClass()
Get the key class for the map input data.

Class<?> getMapInputValueClass()
Get the value class for the map input data.

Class<?> getMapOutputKeyClass()
Get the key class for the map output data.

List<Integer> getMapOutputKeyColumnSizes()
Get the list of map output key column sizes.

Class<?> getMapOutputValueClass()
Get the value class for the map output data.

List<Integer> getMapOutputValueColumnSizes()
Get the list of map output value column sizes.

Class<? extends Mapper> getMapperClass()
Get the Mapper class for the job.

String getMapStreamCommand() Get
the mapper streaming command.

int getNumDataslices()
Get the number of dataslices.

String [] getOutputKeyColumnNames() Get
the names of the output key columns.

String getOutputTableName() Get
the name of the output table.

String [] getOutputValueColumnNames() Get
the names of the output value columns.

Class<? extends Partitioner> getPartitionerClass()
Get the Partitioner used to partition Mapper -outputs to be sent to the Reducer s.

Class<?> getReduceInputKeyClass()
Get the key class for the reduce input data.

Class<?> getReduceInputValueClass()
Get the value class for the reduce input data.

Class<?> getReduceOutputKeyClass()
Get the key class for the reduce output data.

List<Integer> getReduceOutputKeyColumnSizes()
Get the list of reduce output key column sizes.

Class<?> getReduceOutputValueClass()
Get the value class for the reduce output data.

List<Integer> getReduceOutputValueColumnSizes()
Get the list of reduce output value column sizes.

Class<? extends Reducer> getReducerClass()
Get the Reducer class for the job.

String getReduceStreamCommnad()
Get the reducer streaming command.

String getRunDir()
Get the path to the directory from which the job is run.

boolean getRunDirCleanup()
Get the value of the property indicating whether the framework should clean the run dir after the job completion.

boolean getSkipBadRecords()
Get the value of the property indicating whether the framework should skip bad records.

JobConf(Configuration conf, Class exampleClass)
Construct a map/reduce job configuration.

JobConf(Class exampleClass)
Construct a map/reduce job configuration.

JobConf(boolean loadDefaults)
A new map/reduce configuration where the behavior of reading from the default resources can be turned off.

JobConf(Configuration conf)
Construct a map/reduce job configuration.

JobConf()
Construct a map/reduce job configuration.

void setBadRecordsLimit(int value)
Set the maximum number of bad records that the framework should skip.

void setCombineOutputKeyClass(Class<?> cls)
Set the key class for the combine output data.

void setCombineOutputKeyColumnSize(int id, int size)
Set the size for the combine output key column with a given id.

void setCombineOutputValueClass(Class<?> cls)
Set the value class for the combine output data.

void setCombineOutputValueColumnSize(int id, int size)
Set the size for the combine output value column with a given id.

void setCombinerClass(Class<?extends Reducer > theClass)
Set the user-defined *combiner* class used to combine map-outputs before being sent to the re-ducers.

void setDatabaseName(String database)
Set the database name for the job.

void setInputKeyColumnNames(String...colNames)
Set the names of the input key columns.

void setInputTableName(String inputTable)
Set the name of the input table.

void setInputValueColumnNames(String...colNames)
Set the names of the input value columns.

void setIsStreaming(boolean value)
Set whether the job uses streaming tasks.

void setJar(String jar)
Set the user jar for the map-reduce job.

void setJarByClass(Class cls)
Set the job's jar file by finding an example class location.

void setJobName(String name)
Set the user-specified job name.

void setMapInputKeyClass(Class<?> cls)
Set the key class for the map input data.

void setMapInputValueClass(Class<?> cls)
Set the value class for the map input data.

void setMapOutputKeyClass(Class<?> cls)
Set the key class for the map output data.

void setMapOutputKeyColumnSize(int id, int size)
Set the size for the map output key column with a given id.

void setMapOutputValueClass(Class<?> cls)
Set the value class for the map output data.

void setMapOutputValueColumnSize(int id, int size)
Set the size for the map output value column with a given id.

void setMapperClass(Class<?extends Mapper >
theClass) Set the Mapper class for the job.

void setOutputKeyColumnNames(String...colNames)
Set the names of the output key columns.

void setOutputTableName(String outputTable)
Set the name of the output table.

void setOutputValueColumnNames(String...colNames)
Set the names of the output value columns.

void setPartitionerClass(Class<?extends Partitioner > theClass)
Set the Partitioner class used to partition intermediate data to be sent to the Reducer s.

void setReduceOutputKeyClass(Class<?> cls)
Set the key class for the reduce output data.

void setReduceOutputKeyColumnSize(int id, int size)
Set the size for the reduce output key column with a given id.

void setReduceOutputValueClass(Class<?> cls)
Set the value class for the reduce output data.

void setReduceOutputValueColumnSize(int id, int size)
Set the size for the reduce output value column with a given id.

void setReducerClass(Class<?extends Reducer >
theClass) Set the Reducer class for the job.

void setRunDir(String runDir)
Set the path to a directory from which the job is run.

void setRunDirCleanup(boolean value)
Set whether the framework should clean the run dir after the job completion.

void setSkipBadRecords(boolean value)
Set whether the framework should skip bad records.

Static Public Attributes

DEFAULT_JOB_NAME

Public Member Function Documentation

int getBadRecordsLimit()
Get the maximum number of bad records that the framework should skip.

Returns
the maximum number of bad records

Class<?> getCombineInputKeyClass()
Get the key class for the combine input data.

Returns
the combine input key class

Class<?> getCombineInputValueClass()
Get the value class for the combine input data.

Returns
the combine input value class

Class<?> getCombineOutputKeyClass()
Get the key class for the combine output data.

Returns
the combine output key class

List<Integer> getCombineOutputKeyColumnSizes()
Get the list of combine output key column sizes.

Returns
the list of combine output key column sizes

Class<?> getCombineOutputValueClass()

Get the value class for the combine output data.

Returns
the combine output value class

List<Integer> getCombineOutputValueColumnSizes()
Get the list of combine output value column sizes.

Returns
the list of combine output value column sizes

Class<? extends Reducer> getCombinerClass()
Get the user-defined *combiner* class used to combine map-outputs before being sent to the reducers.

Returns
the user-defined combiner class used to combine map-outputs.

Typically the combiner is same as the the Reducer for the job i.e. getReducerClass .

String getCombineStreamCommand()
Get the combiner streaming command.

Returns
the combiner streaming command, or null if the combiner is not run via streaming

String getDatabaseName()
Get the database name for the job.

Returns
the database name for the job

String getDeployDir()
Get the path to the directory where all jobs are deployed.

Returns
the path to the deployment directory

String [] getInputKeyColumnNames()
Get the names of the input key columns.

Returns
the array with the names of the input key columns

String getInputTableName() Get
the name of the input table.

Returns
the name of the input table

String [] getInputValueColumnNames()

Get the names of the input value columns.

Returns
the array with the names of the input value columns

boolean getIsStreaming()

Returns true if the job uses streaming.

Returns
true if the job uses streaming

String getJar()

Get the user jar for the map-reduce job.

Returns
the user jar for the map-reduce job.

String getJobName()

Get the user-specified job name.

Returns
the job's name, defaulting to "NONAME".

This is only used to identify the job to the user.

Class<?> getMapInputKeyClass()

Get the key class for the map input data.

Returns
the map input key class

Class<?> getMapInputValueClass()

Get the value class for the map input data.

Returns
the map input value class

Class<?> getMapOutputKeyClass()

Get the key class for the map output data.

Returns
the map output key class

List<Integer> getMapOutputKeyColumnSizes()
Get the list of map output key column sizes.

Returns
the list of map output key column sizes

Class<?> getMapOutputValueClass()
Get the value class for the map output data.

Returns
the map output value class

List<Integer> getMapOutputValueColumnSizes()
Get the list of map output value column sizes.

Returns
the list of map output value column sizes

Class<? extends Mapper> getMapperClass()
Get the Mapper class for the job.

Returns
the Mapper class for the job.

String getMapStreamCommand() Get
the mapper streaming command.

Returns
the mapper streaming command, or null if the mapper is not run via streaming

int getNumDataslices()
Get the number of dataslices.

Returns
the number of dataslices

String [] getOutputKeyColumnNames()
Get the names of the output key columns.

Returns
the array with the names of the output key columns

String getOutputTableName() Get
the name of the output table.

Returns
the name of the output table

String [] getOutputValueColumnNames()

Get the names of the output value columns.

Returns
the array with the names of the output value columns

Class<? extends Partitioner> getPartitionerClass()

Get the Partitioner used to partition Mapper -outputs to be sent to the Reducer s.

Returns
the Partitioner used to partition map-outputs.

Class<?> getReduceInputKeyClass()

Get the key class for the reduce input data.

Returns
the reduce input key class

Class<?> getReduceInputValueClass()

Get the value class for the reduce input data.

Returns
the reduce input value class

Class<?> getReduceOutputKeyClass()

Get the key class for the reduce output data.

Returns
the reduce output key class

List<Integer> getReduceOutputKeyColumnSizes()

Get the list of reduce output key column sizes.

Returns
the list of reduce output key column sizes

Class<?> getReduceOutputValueClass()

Get the value class for the reduce output data.

Returns
the reduce output value class

List<Integer> getReduceOutputValueColumnSizes()

Get the list of reduce output value column sizes.

Returns
the list of reduce output value column sizes

Class<? extends Reducer> getReducerClass()
Get the Reducer class for the job.

Returns
the Reducer class for the job.

String getReduceStreamCommnad()
Get the reducer streaming command.

Returns
the reducer streaming command, or null if the reducer is not run via streaming

String getRunDir()
Get the path to the directory from which the job is run.

Returns
the path to the run directory

boolean getRunDirCleanup()
Get the value of the property indicating whether the framework should clean the run dir after the job completion.

Returns
true if the framework should clean the run dir

boolean getSkipBadRecords()
Get the value of the property indicating whether the framework should skip bad records.

Returns
true if the framework should skip bad records

JobConf(Configuration conf, Class exampleClass)
Construct a map/reduce job configuration.

Parameters

Configuration conf
a Configuration whose settings will be inherited.

exampleClass
a class whose containing jar is used as the job's jar.

JobConf(Class exampleClass)

Construct a map/reduce job configuration.

Parameters

exampleClass

a class whose containing jar is used as the job's jar.

JobConf(boolean loadDefaults)

A new map/reduce configuration where the behavior of reading from the default resources can be turned off.

Parameters

loadDefaults

specifies whether to load from the default files

If the parameter loadDefaults is false, the new instance will not load resources from the de-fault files.

JobConf(Configuration conf)

Construct a map/reduce job configuration.

Parameters

Configuration conf

a Configuration whose settings will be inherited.

JobConf()

Construct a map/reduce job configuration.

void setBadRecordsLimit(int value)

Set the maximum number of bad records that the framework should skip.

Parameters

value

the maximum number of bad records

void setCombineOutputKeyClass(Class<?> cls)

Set the key class for the combine output data.

Parameters

cls

the combine output key class

void setCombineOutputKeyColumnSize(int id, int size)

Set the size for the combine output key column with a given id.

Parameters

id
the id of the output column

size
the size of the output column

This method should be invoked for each variable-sized column.

void setCombineOutputValueClass(Class<?> cls)
Set the value class for the combine output data.

Parameters
cls
the combine output value class

void setCombineOutputValueColumnSize(int id, int size)
Set the size for the combine output value column with a given id.

Parameters
id
the id of the output column

size
the size of the output column

This method should be invoked for each variable-sized column.

void setCombinerClass(Class<?extends Reducer > theClass)
Set the user-defined *combiner* class used to combine map-outputs before being sent to the reducers.

Parameters
theClass
the user-defined combiner class used to combine map-outputs.

The combiner is an application-specified aggregation operation, which can help cut down the amount of data transferred between the Mapper and the Reducer , leading to better performance.

The framework always invokes the combiner on all map-outputs

Typically the combiner is same as the Reducer for the job i.e. setReducerClass(Class) .

void setDatabaseName(String database)
Set the database name for the job.

Parameters
database
the database name for the job

void setInputKeyColumnNames(String...colNames)
Set the names of the input key columns.

Parameters

colNames

the names of the input key columns

void setInputTableName(String inputTable)

Set the name of the input table.

Parameters

inputTable

the name of the input table

void setInputValueColumnNames(String...colNames)

Set the names of the input value columns.

Parameters

colNames

the names of the input value columns

void setIsStreaming(boolean value)

Set whether the job uses streaming tasks.

Parameters

value

true if the job uses streaming

void setJar(String jar)

Set the user jar for the map-reduce job.

Parameters

jar

the user jar for the map-reduce job.

void setJarByClass(Class cls)

Set the job's jar file by finding an example class location.

Parameters

cls

the example class.

void setJobName(String name)

Set the user-specified job name.

Parameters

name

the job's new name.

void setMapInputKeyClass(Class<?> cls)
Set the key class for the map input data.

Parameters
cls
the map input key class

void setMapInputValueClass(Class<?> cls)
Set the value class for the map input data.

Parameters
cls
the map input value class

void setMapOutputKeyClass(Class<?> cls)
Set the key class for the map output data.

Parameters
cls
the map output key class

void setMapOutputKeyColumnSize(int id, int size)
Set the size for the map output key column with a given id.

Parameters
id
the id of the output column

size
the size of the output column

This method should be invoked for each variable-sized column.

void setMapOutputValueClass(Class<?> cls)
Set the value class for the map output data.

Parameters
cls
the map output value class

void setMapOutputValueColumnSize(int id, int size)
Set the size for the map output value column with a given id.

Parameters
id
the id of the output column

size

the size of the output column

This method should be invoked for each variable-sized column.

void setMapperClass(Class<?extends Mapper > theClass) Set the Mapper class for the job.

Parameters

theClass

the Mapper class for the job.

void setOutputKeyColumnNames(String...colNames)

Set the names of the output key columns.

Parameters

colNames

the names of the output key columns

void setOutputTableName(String outputTable) Set the name of the output table.

Parameters

outputTable

the name of the output table

void setOutputValueColumnNames(String...colNames)

Set the names of the output value columns.

Parameters

colNames

the names of the output value columns

void setPartitionerClass(Class<?extends Partitioner > theClass)

Set the Partitioner class used to partition intermediate data to be sent to the Reducer s.

Parameters

theClass

the Partitioner used to partition intermediate data.

void setReduceOutputKeyClass(Class<?> cls)

Set the key class for the reduce output data.

Parameters

cls

the reduce output key class

void setReduceOutputKeyColumnSize(int id, int size)
Set the size for the reduce output key column with a given id.

Parameters
id
the id of the output column

size
the size of the output column

This method should be invoked for each variable-sized column.

void setReduceOutputValueClass(Class<?> cls)
Set the value class for the reduce output data.

Parameters
cls
the reduce output value class

void setReduceOutputValueColumnSize(int id, int size)
Set the size for the reduce output value column with a given id.

Parameters
id
the id of the output column

size
the size of the output column

This method should be invoked for each variable-sized column.

void setReducerClass(Class<?extends Reducer > theClass) Set the Reducer class for the job.

Parameters
theClass
the Reducer class for the job.

void setRunDir(String runDir)
Set the path to a directory from which the job is run.

Parameters
runDir
the path

void setRunDirCleanup(boolean value)
Set whether the framework should clean the run dir after the job completion.

Parameters

value
true if framework should clean the run dir, false otherwise.

void setSkipBadRecords(boolean value)
Set whether the framework should skip bad records.

Parameters
value
true if framework should skip bad records, false otherwise

Static Member Data Documentation

final String DEFAULT_JOB_NAME="NONAME"

JobConfigurable Interface Reference

That what may be configured.

Public Member Functions

void configure(JobConf job)
Initializes a new instance from a JobConf .

Detailed Description

That what may be configured.

Public Member Function Documentation

void configure(JobConf job)
Initializes a new instance from a JobConf .

Parameters
JobConf job
the configuration

JobContext Class Reference

A read-only view of the job that is provided to the tasks while they are running.

Inherits MRJobConfig

Public Member Functions

int getBadRecordsLimit()
Get the maximum number of bad records that the framework should skip.

Class<?> getCombineInputKeyClass()
Get the key class for the combine input data.

Class<?> getCombineInputValueClass()
Get the value class for the combine input data.

Class<?> getCombineOutputKeyClass()
Get the key class for the combine output data.

List<Integer> getCombineOutputKeyColumnSizes()
Get the list of combine output key column sizes.

Class<?> getCombineOutputValueClass()
Get the value class for the combine output data.

List<Integer> getCombineOutputValueColumnSizes()
Get the list of combine output value column sizes.

Class<? extends Reducer<?, ?, ?, ?> >
getCombinerClass() Get the combiner class for the job.

Configuration getConfiguration()
Return the configuration for the job.

String getJar()
Get the user jar for the map-reduce job.

String getJobName()
Get the user-specified job name.

Class<?> getMapInputKeyClass()
Get the key class for the map input data.

Class<?> getMapInputValueClass()
Get the value class for the map input data.

Class<?> getMapOutputKeyClass()
Get the key class for the map output data.

List<Integer> getMapOutputKeyColumnSizes()
Get the list of map output key column sizes.

Class<?> getMapOutputValueClass()
Get the value class for the map output data.

List<Integer> getMapOutputValueColumnSizes()
Get the list of map output value column sizes.

Class<? extends Mapper<?, ?, ?, ?> >
getMapperClass() Get the Mapper class for the job.

int getNumDataslices()
Get the number of dataslices.

Class<? extends Partitioner<?, ?> > getPartitionerClass()

Get the Partitioner class for the job.

Class<?> getPartitionKeyClass()
Get the key class for the partitioner input and output data.

Class<?> getPartitionValueClass()
Get the value class for the partitioner input and output data.

Class<?> getReduceInputKeyClass()
Get the key class for the reduce input data.

Class<?> getReduceInputValueClass()
Get the value class for the reduce input data.

Class<?> getReduceOutputKeyClass()
Get the key class for the reduce output data.

List<Integer>   getReduceOutputKeyColumnSizes()
Get column sizes for reduce output key columns.

Class<?> getReduceOutputValueClass()
Get the value class for the reduce output data.

List<Integer> getReduceOutputValueColumnSizes()
Get the list of reduce output value column sizes.

Class<? extends Reducer<?, ?, ?, ?> >
getReducerClass() Get the Reducer class for the job.

String getRunDir()
Get the path to the directory from which the job is run.

boolean getRunDirCleanup()
Get the value of the property indicating whether the framework should clean the run dir after the job completion.

boolean getSkipBadRecords()
Get the value of the property indicating whether the framework should skip bad records.

JobContext(Configuration conf)

Detailed Description

A read-only view of the job that is provided to the tasks while they are running.

Public Member Function Documentation

int getBadRecordsLimit()
Get the maximum number of bad records that the framework should skip.

Returns
the maximum number of bad records

Class<?> getCombineInputKeyClass()
Get the key class for the combine input data.

Returns
the combine input key class

Class<?> getCombineInputValueClass()
Get the value class for the combine input data.

Returns
the combine input value class

Class<?> getCombineOutputKeyClass()
Get the key class for the combine output data.

Returns
the combine output key class

List<Integer> getCombineOutputKeyColumnSizes()
Get the list of combine output key column sizes.

Returns
the list of combine output key column sizes

Class<?> getCombineOutputValueClass()
Get the value class for the combine output data.

Returns
the combine output value class

List<Integer> getCombineOutputValueColumnSizes()
Get the list of combine output value column sizes.

Returns
the list of combine output value column sizes

Class<? extends Reducer<?, ?, ?, ?> >
getCombinerClass() Get the combiner class for the job.

Returns
the combiner class for the job.

Configuration getConfiguration()
Return the configuration for the job.

Returns
Configuration

the shared configuration object

String getJar()
Get the user jar for the map-reduce job.

Returns
Get the user jar for the map-reduce job

String getJobName()
Get the user-specified job name.

Returns
the job's name, defaulting to "NONAME".

This is only used to identify the job to the user.

Class<?> getMapInputKeyClass()
Get the key class for the map input data.

Returns
the map input key class

Class<?> getMapInputValueClass()
Get the value class for the map input data.

Returns
the map input value class

Class<?> getMapOutputKeyClass()
Get the key class for the map output data.

Returns
the map output key class

List<Integer> getMapOutputKeyColumnSizes()
Get the list of map output key column sizes.

Returns
the list of map output key column sizes

Class<?> getMapOutputValueClass()
Get the value class for the map output data.

Returns
the map output value class.

List<Integer> getMapOutputValueColumnSizes()
Get the list of map output value column sizes.

Returns
the list of map output value column sizes

Class<? extends Mapper<?, ?, ?, ?> >
getMapperClass() Get the Mapper class for the job.

Returns
the Mapper class for the job.

int getNumDataslices()
Get the number of dataslices.

Returns
the number of dataslices

Class<? extends Partitioner<?, ?> > getPartitionerClass()
Get the Partitioner class for the job.

Returns
the Partitioner class for the job.

Class<?> getPartitionKeyClass()
Get the key class for the partitioner input and output data.

Returns
the partitioner input/output key class.

Class<?> getPartitionValueClass()
Get the value class for the partitioner input and output data.

Returns
the partitioner input/output value class.

Class<?> getReduceInputKeyClass()
Get the key class for the reduce input data.

Returns
the reduce input key class

Class<?> getReduceInputValueClass()
Get the value class for the reduce input data.

Returns

the reduce input value class

Class<?> getReduceOutputKeyClass()
Get the key class for the reduce output data.

Returns
the reduce output key class

List<Integer> getReduceOutputKeyColumnSizes()
Get column sizes for reduce output key columns.

Returns
the list of reduce output key column sizes

Class<?> getReduceOutputValueClass()
Get the value class for the reduce output data.

Returns
the reduce output value class

List<Integer> getReduceOutputValueColumnSizes()
Get the list of reduce output value column sizes.

Returns
the list of reduce output value column sizes

Class<? extends Reducer<?, ?, ?, ?> >
getReducerClass() Get the Reducer class for the job.

Returns
the Reducer class for the job.

String getRunDir()
Get the path to the directory from which the job is run.

Returns
the path to the run directory

boolean getRunDirCleanup()
Get the value of the property indicating whether the framework should clean the run dir after the job completion.

Returns
true if the framework should clean the run dir

boolean getSkipBadRecords()
Get the value of the property indicating whether the framework should skip bad records.

Returns
true if the framework should skip bad records

JobContext(Configuration conf)

JobDeployException Class Reference

An exception class used for signaling failures of job deployment.

Public Member Functions

JobDeployException()
Constructs an JobDeployException with null as its error detail message.

JobDeployException(String message, Throwable cause)
Constructs an JobDeployException with the specified detail message and cause.

JobDeployException(Throwable cause)
Constructs an JobDeployException with the specified cause and a detail message of (cause==null ? null : cause.toString()) (which typically contains the class and detail message of cause).

JobDeployException(String message)
Constructs an JobDeployException with the specified detail message.

Detailed Description

An exception class used for signaling failures of job deployment.

Public Member Function Documentation

JobDeployException()
Constructs an JobDeployException with null as its error detail message.

JobDeployException(String message, Throwable cause)
Constructs an JobDeployException with the specified detail message and cause.

JobDeployException(Throwable cause)
Constructs an JobDeployException with the specified cause and a detail message of (cause==null ? null : cause.toString()) (which typically contains the class and detail message of cause).

JobDeployException(String message)

Constructs an JobDeployException with the specified detail message.

JobRunner Class Reference

This class allows to run jobs specified either in a Job or JobConf object.

Public Member Functions

JobRunner(Job job)
void validateJob(boolean nps)

Static Public Member Functions

static boolean runJob(Job job)
Utility that runs the job specified in the given Job object.

static boolean runJob(JobConf conf)
Utility that runs the job specified in the given JobConf object.

Detailed Description

This class allows to run jobs specified either in a Job or JobConf object.

Normally the user creates the application, describes various facets of the job via Job or JobConf and then uses the JobRunner to run the job.

Public Member Function Documentation

JobRunner(Job job)

void validateJob(boolean nps)

Static Public Member Function Documentation

static boolean runJob(Job job)
Utility that runs the job specified in the given Job object.

Parameters
Job job
the job to run

Returns
true if the job succeeded

Exceptions
Exception

static boolean runJob(JobConf conf)
Utility that runs the job specified in the given JobConf object.

Parameters
JobConf conf
the job's configuration

Returns
true if the job succeeded

Exceptions
Exception

LongSumReducer< K > Class Reference

A Reducer that sums long values.

Inherits MapReduceBase

Public Member Functions

void reduce(K key, Iterator< LongWritable > values, OutputCollector< K, LongWritable > output, Report-er reporter)
Sums all values and writes one pair: <key, sum>.

Detailed Description

A Reducer that sums long values.

Public Member Function Documentation

void reduce(K key, Iterator< LongWritable > values, OutputCollector< K, LongWritable > output, Re-porter reporter)
Sums all values and writes one pair: <key, sum>.

LongSumReducer< KEY > Class Reference

A Reducer that sums long values.

Inherits org::netezza::inza::mr::mapreduce::Reducer< KEY, LongWritable, KEY, LongWritable >

Public Member Functions

void reduce(KEY key, Iterable< LongWritable > values, Context context) Sums all values and writes one pair: <key, sum>.

Detailed Description

A Reducer that sums long values.

Public Member Function Documentation

void reduce(KEY key, Iterable< LongWritable > values, Context context) Sums all values and writes one pair: <key, sum>.

LongWritable Class Reference

A Writable for longs.

Inherits Writable

Public Member Functions

boolean equals(Object o)
Returns true iff o is a LongWritable with the same value.

Long get()
Return the value of this LongWritable .

List<Class<?> > getStorageTypesList()
int hashCode()
LongWritable(long value)
LongWritable()
void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

void set(Long value)
Set the value of this LongWritable .

String toString()
void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Detailed Description

A Writable for longs.

Public Member Function Documentation

boolean equals(Object o)
Returns true iff o is a LongWritable with the same value.

Long get()
Return the value of this LongWritable .

List<Class<?> > getStorageTypesList()

Returns
list of classes of storage types. These classes are used by the framework for automatic conversion from database fields and for setting column types of output table.

int hashCode()

LongWritable(long value)

LongWritable()

void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

Parameters
RecordInput in
RecordInput to read this object from.

Exceptions
IOException

void set(Long value)
Set the value of this LongWritable .

String toString()

void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Parameters
RecordOutput out
RecordOutput to write this object into.

Exceptions
IOException

MainCounters Interface Reference

This interface contains Counters constants which are used by the framework for built-in counters.

Public Attributes

COMBINE_COUNTER_GROUP
COMBINE_INPUT_RECORDS
COMBINE_OUTPUT_RECORDS
JOB_COMBINE_TASKS
JOB_MAP_TASKS
JOB_REDUCE_TASKS
MAP_COUNTER_GROUP
MAP_INPUT_BAD_RECORDS
MAP_INPUT_RECORDS
MAP_OUTPUT_RECORDS
REDUCE_COUNTER_GROUP
REDUCE_INPUT_GROUPS
REDUCE_INPUT_RECORDS
REDUCE_OUTPUT_RECORDS

Detailed Description

This interface contains Counters constants which are used by the framework for built-in counters.

Member Data Documentation

String COMBINE_COUNTER_GROUP

String COMBINE_INPUT_RECORDS

String COMBINE_OUTPUT_RECORDS

String JOB_COMBINE_TASKS

String JOB_MAP_TASKS

String JOB_REDUCE_TASKS

String MAP_COUNTER_GROUP

String MAP_INPUT_BAD_RECORDS

String MAP_INPUT_RECORDS

String MAP_OUTPUT_RECORDS

String REDUCE_COUNTER_GROUP

String REDUCE_INPUT_GROUPS

String REDUCE_INPUT_RECORDS

String REDUCE_OUTPUT_RECORDS

MapContext< KEYIN, VALUEIN, KEYOUT, VALUEOUT > Class Reference

The context that is given to the Mapper .

Inherits TaskInputOutputContext< KEYIN, VALUEIN, KEYOUT, VALUEOUT >

Public Member Functions

KEYIN getCurrentKey()
Get the current key.

VALUEIN getCurrentValue()
Get the current value.

MapContext(Configuration conf, MapperRecordReader< KEYIN, VALUEIN > reader, RecordWriter< KEY-OUT, VALUEOUT > writer, StatusReporter reporter)
boolean nextKeyValue()
Advance to the next key, value pair.

Detailed Description

The context that is given to the Mapper .

Public Member Function Documentation

KEYIN getCurrentKey()
Get the current key.

Returns
the current key object or null if there isn't one

VALUEIN getCurrentValue()
Get the current value.

Returns
the value object that was read into

MapContext(Configuration conf, MapperRecordReader< KEYIN, VALUEIN > reader, Record-Writer< KEYOUT, VALUEOUT > writer, StatusReporter reporter)

boolean nextKeyValue()
Advance to the next key, value pair.

Returns
true if a key/value pair was read

Mapper< K1, V1, K2, V2 > Interface Reference

Maps input key/value pairs to a set of intermediate key/value pairs.

Inherits JobConfigurable

Public Member Functions

void map(K1 key, V1 value, OutputCollector< K2, V2 > output, Reporter reporter)
Maps a single input key/value pair into an intermediate key/value pair.

Detailed Description

Maps input key/value pairs to a set of intermediate key/value pairs.

Maps are the individual tasks which transform input records into a intermediate records. The transformed intermediate records need not be of the same type as the input records. A given in-put pair may map to zero or many output pairs.

The map/reduce framework spawns one map task for each dataslice. Mapper implementations can access the JobConf for the job via the configure and initialize themselves. Similarly they can use the Closeable#close() method for de-initialization.

The framework then calls map(Object, Object, OutputCollector, Reporter) for each key/value pair from the input.

All intermediate values associated with a given output key are subsequently grouped by the frame-work, and passed to a Reducer to determine the final output.

The grouped Mapper outputs are partitioned per Reducer. Users can control which keys (and hence records) go to which Reducer by implementing a custom Partitioner .

Users can optionally specify a combiner, via JobConf#setCombinerClass(Class) , to perform local aggregation of the intermediate outputs, which helps to cut down the amount of data transferred from the Mapper to the Reducer.

If the job has no reducer nor combiner specified then the output of the Mapper is directly written to the output table without grouping by keys.

Example:

```
public class MyMapper<K extends Writable, V extends Writable>
extends MapReduceBase implements Mapper<K, V, K, V> {
 static enum MyCounters { NUM_RECORDS }
 private String mapInputText;
  private int mapInputValue;
 public void configure(JobConf job) {
       Get the values of some properties mapInputText
   = job.get("map.input.text"); mapInputValue =
   job.getInt("map.input.value", -1);
  }
 public void map(K key, V val,
               OutputCollector<K, V> output, Reporter reporter)
  throws IOException {
      Process the <key, value> pair
       ...
       ...
      Increment counters
   reporter.incrCounter(MyCounters.NUM_RECORDS, 1);
      Output the result
   output.collect(key, val);
  }
}
```

See Also

JobConf

MapReduceBase

Public Member Function Documentation

void map(K1 key, V1 value, OutputCollector< K2, V2 > output, Reporter reporter) Maps a single input key/value pair into an intermediate key/value pair.

Parameters

key
the input key.

value
the input value.

output
collects mapped keys and values.

Reporter reporter
facility to update counters.

Output pairs need not be of the same types as input pairs. A given input pair may map to zero or many output pairs. Output pairs are collected with calls to OutputCollector#collect(Object,Object) .

Mapper< KEYIN, VALUEIN, KEYOUT, VALUEOUT > Class Reference

Maps input key/value pairs to a set of intermediate key/value pairs.

Public Member Functions

void run(Context context)
Expert users can override this method for more complete control over the execution of the Mapper.

Protected Member Functions

void cleanup(Context context)
Called once at the end of the task.

void map(KEYIN key, VALUEIN value, Context
context) Called once for each input key/value pair.

void setup(Context context)
Called once at the beginning of the task.

Detailed Description

Maps input key/value pairs to a set of intermediate key/value pairs.

Maps are the individual tasks which transform input records into intermediate records. The trans-formed intermediate records need not be of the same type as the input records. A given input pair may map to zero or many output pairs.

Mapper implementations can access the Configuration for the job via the getConfiguration .

The framework first calls setup(org.netezza.inza.mr.mapreduce.Mapper.Context) , followed by map(Object, Object, Context) for each key/value pair. Finally cleanup is called.

All intermediate values associated with a given output key are subsequently grouped by the frame-work, and passed to a Reducer to determine the final output.

The Mapper outputs are partitioned per Reducer. Users can control which keys (and hence re-cords) go to which Reducer by implementing a custom Partitioner .

Users can optionally specify a combiner, via Job#setCombinerClass(Class) , to perform local aggreg-ation of the intermediate outputs, which helps to cut down the amount of data transferred from the Mapper to the Reducer.

If the job has zero reducers then the output of the Mapper is directly written to the output table.

Example:

```
public class TokenCounterMapper
   extends Mapper<LongWritable, Text, Text, IntWritable>{
```

```
 private final static IntWritable one = new IntWritable(1);
  private Text word = new Text();

 public void map(LongWritable key, Text value, Context context) throws
   IOException { StringTokenizer itr = new StringTokenizer(value.toString());
   while (itr.hasMoreTokens()) {
     word.set(itr.nextToken());
     context.write(word, one);
   }
  }
}
```

See Also

- JobContext

Public Member Function Documentation

void run(Context context)
Expert users can override this method for more complete control over the execution of the Mapper.

Parameters
context
Exceptions
IOException

Protected Member Function Documentation

void cleanup(Context context)
Called once at the end of the task.

void map(KEYIN key, VALUEIN value, Context context) Called once for each input key/value pair.

Most applications should override this, but the default is the identity function.

void setup(Context context)
Called once at the beginning of the task.

MapperRecordReader< KEYIN, VALUEIN > Class Reference

The record reader breaks the data into key/value pairs for input to the Mapper .

Public Member Functions

abstract KEYIN getCurrentKey()

Get the current key.

abstract VALUEIN getCurrentValue()
Get the current value.

abstract void initialize(Nzae ae, TaskAttemptContext context) Called once at initialization.

abstract boolean nextKeyValue()
Read the next key, value pair.

Detailed Description

The record reader breaks the data into key/value pairs for input to the Mapper .

Public Member Function Documentation

abstract KEYIN getCurrentKey()
Get the current key.

Returns
the current key or null if there is no current key

abstract VALUEIN getCurrentValue()
Get the current value.

Returns
the object that was read

abstract void initialize(Nzae ae, TaskAttemptContext context) Called once at initialization.

Parameters

ae
ae handler

TaskAttemptContext context
the information about the task

abstract boolean nextKeyValue()
Read the next key, value pair.

Returns
true if a key/value pair was read

MapReduceBase Class Reference

Base class for Mapper and Reducer implementations.

Inherits Closeable

Public Member Functions

void close()
Default implementation that does nothing.

void configure(JobConf job)
Default implementation that does nothing.

Detailed Description

Base class for Mapper and Reducer implementations.

Provides default no-op implementations for a few methods, most non-trivial applications need to override some of them.

Public Member Function Documentation

void close()
Default implementation that does nothing.

void configure(JobConf job)
Default implementation that does nothing.

MissingConfigurationPropertyException Class Reference

Signals that some property is not set in a configuration.

Inherits IllegalJobConfigurationException

Public Member Functions

MissingConfigurationPropertyException(String propertyName)
Constructs a MissingConfigurationPropertyException for the specified propertyName.

Detailed Description

Signals that some property is not set in a configuration.

Public Member Function Documentation

MissingConfigurationPropertyException(String propertyName)

Constructs a MissingConfigurationPropertyException for the specified propertyName.

MissingEnvironmentVariableException Class Reference

Signals that some environment variable is not set.

Public Member Functions

MissingEnvironmentVariableException(String name)
Constructs a MissingEnvironmentVariableException with the specified missing environment variable name.

MissingEnvironmentVariableException(String name, Throwable cause)
Constructs a MissingEnvironmentVariableException with the specified missing environment variable name and cause.

Detailed Description

Signals that some environment variable is not set.

Public Member Function Documentation

MissingEnvironmentVariableException(String name)
Constructs a MissingEnvironmentVariableException with the specified missing environment variable name.

MissingEnvironmentVariableException(String name, Throwable cause)
Constructs a MissingEnvironmentVariableException with the specified missing environment variable name and cause.

MRJobConfig Interface Reference

Static Public Attributes

BAD_RECORDS_LIMIT
COMBINE_CLASS_ATTR
COMBINE_OUTPUT_KEY_CLASS
COMBINE_OUTPUT_KEY_COLUMN_SIZE
COMBINE_OUTPUT_VALUE_CLASS
COMBINE_OUTPUT_VALUE_COLUMN_SIZE
COMBINER_NEW_API

DATABASE_NAME
DEPLOY_DIR
INPUT_KEY_COLUMNS
INPUT_TABLE
INPUT_VALUE_COLUMNS
JAR
JOB_IS_STREAMING
JOB_NAME
JOB_RUN_DIR
JOB_RUN_DIR_CLEANUP
MAP_CLASS_ATTR
MAP_INPUT_KEY_CLASS
MAP_INPUT_VALUE_CLASS
MAP_OUTPUT_KEY_CLASS
MAP_OUTPUT_KEY_COLUMN_SIZE
MAP_OUTPUT_VALUE_CLASS
MAP_OUTPUT_VALUE_COLUMN_SIZE
MAPPER_NEW_API
NUM_DATASLICES
OUTPUT_KEY_COLUMNS
OUTPUT_TABLE
OUTPUT_VALUE_COLUMNS
PARTITION_CLASS_ATTR
PARTITIONER_NEW_API
REDUCE_CLASS_ATTR
REDUCE_OUTPUT_KEY_CLASS
REDUCE_OUTPUT_KEY_COLUMN_SIZE
REDUCE_OUTPUT_VALUE_CLASS
REDUCE_OUTPUT_VALUE_COLUMN_SIZE
REDUCER_NEW_API
SKIP_BAD_RECORDS
STREAM_COMBINE_CMD
STREAM_MAP_CMD
STREAM_REDUCE_CMD
TASK_DATASLICE_ID

Static Member Data Documentation

final String BAD_RECORDS_LIMIT="mapreduce.map.bad.records.limit"

final String COMBINE_CLASS_ATTR="mapreduce.job.combine.class"

final String COMBINE_OUTPUT_KEY_CLASS="mapreduce.combine.output.key.class"

final String COMBINE_OUTPUT_KEY_COLUMN_SIZE="mapreduce.combine.output.key.columns.sizes" final String COMBINE_OUTPUT_VALUE_CLASS="mapreduce.combine.output.value.class"

final String COMBINE_OUTPUT_VALUE_COLUMN_SIZE="mapreduce.combine.output.value.-columns.sizes"

final String COMBINER_NEW_API="mapreduce.combiner.new-api"

final String DATABASE_NAME="mapreduce.job.database"

final String DEPLOY_DIR="mapreduce.deploy.dir"

final String INPUT_KEY_COLUMNS="mapreduce.job.input.key.columns"

final String INPUT_TABLE="mapreduce.job.input.table"

final String INPUT_VALUE_COLUMNS="mapreduce.job.input.values.columns"

final String JAR="mapreduce.job.jar"

final String JOB_IS_STREAMING="mapreduce.job.is.streaming"

final String JOB_NAME="mapreduce.job.name"

final String JOB_RUN_DIR="mapreduce.job.run.dir"

final String JOB_RUN_DIR_CLEANUP="mapreduce.job.run.dir.cleanup"

final String MAP_CLASS_ATTR="mapreduce.job.map.class"

final String MAP_INPUT_KEY_CLASS="mapreduce.map.input.key.class"

final String MAP_INPUT_VALUE_CLASS="mapreduce.map.input.value.class"

final String MAP_OUTPUT_KEY_CLASS="mapreduce.map.output.key.class"

final String MAP_OUTPUT_KEY_COLUMN_SIZE="mapreduce.map.output.key.columns.sizes"

final String MAP_OUTPUT_VALUE_CLASS="mapreduce.map.output.value.class" final String MAP_OUTPUT_VALUE_COLUMN_SIZE="mapreduce.map.output.value.columns.sizes"

final String MAPPER_NEW_API="mapreduce.mapper.new-api"

final String NUM_DATASLICES="mapreduce.dataslice.num"

final String OUTPUT_KEY_COLUMNS="mapreduce.job.output.key.columns"

final String OUTPUT_TABLE="mapreduce.job.output.table"

final String OUTPUT_VALUE_COLUMNS="mapreduce.job.output.value.columns"

final String PARTITION_CLASS_ATTR="mapreduce.job.partition.class"

final String PARTITIONER_NEW_API="mapreduce.partitioner.new-api"

final String REDUCE_CLASS_ATTR="mapreduce.job.reduce.class"

final String REDUCE_OUTPUT_KEY_CLASS="mapreduce.reduce.output.key.class"

final String REDUCE_OUTPUT_KEY_COLUMN_SIZE="mapreduce.reduce.output.key.columns.sizes"

final String REDUCE_OUTPUT_VALUE_CLASS="mapreduce.reduce.output.value.class"

final String REDUCE_OUTPUT_VALUE_COLUMN_SIZE="mapreduce.reduce.output.value.columns.sizes"

final String REDUCER_NEW_API="mapreduce.reducer.new-api"

final String SKIP_BAD_RECORDS="mapreduce.map.bad.records.ignore"

final String STREAM_COMBINE_CMD="mapreduce.streaming.combine.command"

final String STREAM_MAP_CMD="mapreduce.streaming.map.command"

final String STREAM_REDUCE_CMD="mapreduce.streaming.reduce.command"

final String TASK_DATASLICE_ID="mapreduce.task.dataslice.id"

NString Class Reference

A Class used in storage types list of NText .

Detailed Description

A Class used in storage types list of NText .

See Also

▲ getStorageTypesList

NText Class Reference

A Text with national characters.

Inherits Text

Public Member Functions

List<Class<?> > getStorageTypesList()
NText()
NText(String value)

Detailed Description

A Text with national characters.

Public Member Function Documentation

List<Class<?> > getStorageTypesList()

Returns

list of classes of storage types. These classes are used by the framework for automatic con-version from database fields and for setting column types of output table.

NText()

NText(String value)

NullWritable Class Reference

Singleton Writable with no data.

Inherits Writable

Public Member Functions

boolean equals(Object other)

List<Class<?> > getStorageTypesList()
int hashCode()
void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

String toString()
void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Static Public Member Functions

static NullWritable get()
Returns the single instance of this class.

Detailed Description

Singleton Writable with no data.

Public Member Function Documentation

boolean equals(Object other)

List<Class<?> > getStorageTypesList()

Returns

list of classes of storage types. These classes are used by the framework for automatic conversion from database fields and for setting column types of output table.

int hashCode()

void readFields(RecordInput in)

Read the fields of this object from in, based on a database record.

Parameters

RecordInput in

RecordInput to read this object from.

Exceptions

IOException

String toString()

void write(RecordOutput out)

Write the fields of this object to out, based on a database record.

Parameters

RecordOutput out

RecordOutput to write this object into.

Exceptions

IOException

Static Public Member Function Documentation

static NullWritable get()
Returns the single instance of this class.

Returns
NullWritable

OutputCollector< K, V > Interface Reference

Collects the <key, value> pairs output by Mapper s and Reducer s.

Public Member Functions

void collect(K key, V value)
Adds a key/value pair to the output.

Detailed Description

Collects the <key, value> pairs output by Mapper s and Reducer s.

OutputCollector is the generalization of the facility provided by the Map-Reduce framework to col-lect data output by either the Mapper or the Reducer i.e. intermediate outputs or the output of the job.

Public Member Function Documentation

void collect(K key, V value)
Adds a key/value pair to the output.

Parameters
key
the key to collect.

value
to value to collect.

Exceptions
IOException

Partitioner< K2, V2 > Interface Reference

Partitions the key space.

Inherits JobConfigurable

Public Member Functions

int getPartition(K2 key, V2 value, int numPartitions)
Get the partition number for a given key (hence record) given the total number of partitions.

Detailed Description

Partitions the key space.

Partitioner controls the partitioning of the keys of the intermediate map-outputs. The key (or a subset of the key) is used to derive the partition, typically by a hash function. The total number of partitions is the same as the number of reduce tasks for the job. Hence this controls which of the m reduce tasks the intermediate key (and hence the record) is sent for reduction.

See Also
▲ Reducer

Public Member Function Documentation

int getPartition(K2 key, V2 value, int numPartitions)
Get the partition number for a given key (hence record) given the total number of partitions.

Parameters
key
the key to be partitioned.

value
the entry value.

numPartitions
the total number of partitions.

Returns
the partition number for the key.

Typically a hash function on a all or a subset of the key.

Partitioner< KEY, VALUE > Class Reference

Partitions the key space.

Public Member Functions

abstract int getPartition(KEY key, VALUE value, int numPartitions)
Get the partition number for a given key (hence record) given the total number of partitions.

Detailed Description

Partitions the key space.

Partitioner controls the partitioning of the keys of the intermediate map-outputs. The key (or a subset of the key) is used to derive the partition, typically by a hash function. The total number of partitions is the same as the number of reduce tasks for the job. Hence this controls which of the m reduce tasks the intermediate key (and hence the record) is sent for reduction.

See Also

▲ Reducer

Public Member Function Documentation

abstract int getPartition(KEY key, VALUE value, int numPartitions)

Get the partition number for a given key (hence record) given the total number of partitions.

Parameters

key

the key to be partitioned.

value

the entry value.

numPartitions

the total number of partitions.

Returns

the partition number for the key.

Typically a hash function on a all or a subset of the key.

PartitionerRecordReader< KEYIN, VALUEIN > Class Reference

The record reader breaks the data into key/value pairs for input to the Partitioner .

Public Member Functions

abstract KEYIN getCurrentKey()
Get the current key.

abstract VALUEIN getCurrentValue()
Get the current value.

abstract void initialize(Nzae ae, JobContext
context) Called once at initialization.

abstract boolean nextKeyValue()
Read the next key, value pair.

Detailed Description

The record reader breaks the data into key/value pairs for input to the Partitioner .

Public Member Function Documentation

abstract KEYIN getCurrentKey()
Get the current key.

Returns
the current key or null if there is no current key

abstract VALUEIN getCurrentValue()
Get the current value.

Returns
the object that was read

abstract void initialize(Nzae ae, JobContext context) Called once at initialization.

Parameters

ae
ae handler

JobContext context
the information about the job

abstract boolean nextKeyValue()
Read the next key, value pair.

Returns
true if a key/value pair was read

ProgramDriver Class Reference

A driver that is used to run programs added to it.

Public Member Functions

void addClass(String name, Class mainClass, String description)
This is the method that adds the classed to the repository.

void driver(String[] args)
This is a driver for the example programs.

ProgramDriver()

Detailed Description

A driver that is used to run programs added to it.

Public Member Function Documentation

void addClass(String name, Class mainClass, String description)

This is the method that adds the classed to the repository.

Parameters

name

The name of the string you want the class instance to be called with

mainClass

The class that you want to add to the repository

description

The description of the class

Exceptions

NoSuchMethodException
SecurityException

void driver(String[] args)

This is a driver for the example programs.

Parameters

args

The argument from the user. args[0] is the command to run.

Exceptions

NoSuchMethodException
SecurityException
IllegalAccessException
IllegalArgumentException
Throwable

It looks at the first command line argument and tries to find an example program with that name. If it is found, it calls the main method in that class with the rest of the command line ar-guments.

ProgramDriver()

RecordConversionUnsupported Class Reference

Public Member Functions

String getMessage()
RecordConversionUnsupported(int nzaeType, Class<?> to)

Public Member Function Documentation

String getMessage()

RecordConversionUnsupported(int nzaeType, Class<?> to)

RecordConverter< FROM, TO > Class Reference

Public Member Functions

abstract TO convert(NzaeRecord inputRow, int index)
TypeConverter<FROM, TO> getTypeConverter()
RecordConverter(TypeConverter< FROM, TO > typeConverter)

Public Member Function Documentation

abstract TO convert(NzaeRecord inputRow, int index)

TypeConverter<FROM, TO> getTypeConverter()

RecordConverter(TypeConverter< FROM, TO > typeConverter)

RecordConverterFactory Class Reference

Static Public Member Functions

static <FROM,TO> RecordConverter<FROM, TO> getConverter(int colType, Class< TO > toClass)

Static Public Member Function Documentation

static <FROM,TO> RecordConverter<FROM, TO> getConverter(int colType, Class< TO > toClass)

RecordFieldsConverter Class Reference

Public Member Functions

Object getConvertedField(NzaeRecord record, int index)
RecordFieldsConverter(NzaeRecord record, List< Class<?>> types)

Public Member Function Documentation

Object getConvertedField(NzaeRecord record, int index)

Parameters

record

index

Returns

Not null converted value of field

Exceptions

ConversionException

RecordFieldsConverter(NzaeRecord record, List< Class<?>> types)

RecordInput Class Reference

The RecordInput class provides methods for reading fields from NzaeRecord (database record) and converting them to java primitive types.

Public Member Functions

boolean readBoolean()
double readDouble()
float readFloat()
int readInt()
long readLong()
String readString()
RecordInput(RecordFieldsConverter converter)
Constructs RecordInput with the given database fields converter.

void setRecord(NzaeRecord record)
Set the database input record for reading.

Detailed Description

The RecordInput class provides methods for reading fields from NzaeRecord (database record) and converting them to java primitive types.

There is also a facility for reading a String. Each read operation reads the value from the next data-base field of NzaeRecord.

Public Member Function Documentation

boolean readBoolean()

double readDouble()

float readFloat()

int readInt()

long readLong()

String readString()

RecordInput(RecordFieldsConverter converter)
Constructs RecordInput with the given database fields converter.

Parameters
RecordFieldsConverter converter
the database fields converter

void setRecord(NzaeRecord record)
Set the database input record for reading.

Parameters
record input
record

Reading will be performed starting from the first field of this record.

RecordOutput Class Reference

The RecordOutput class provides methods for writing java primitives to NzaeRecord (database record).

Public Member Functions

void setRecord(NzaeRecord record)
Set the database output record for writing.

void writeBoolean(boolean v)
void writeByte(byte v)
void writeDouble(double v)
void writeFloat(float v)
void writeInt(int v)
void writeLong(long v)
void writeShort(short v)
void writeString(String v)
NzaeRecord getRecord()

Detailed Description

The RecordOutput class provides methods for writing java primitives to NzaeRecord (database re-cord).

There is also a facility for writing a String. Each write operation writes the given value to the next database field of NzaeRecord.

Public Member Function Documentation

void setRecord(NzaeRecord record)
Set the database output record for writing.

Parameters
record output
record

Writing will be performed starting from the first field of this record.

void writeBoolean(boolean v)

void writeByte(byte v)

void writeDouble(double v)

void writeFloat(float v)

void writeInt(int v)

void writeLong(long v)

void writeShort(short v)

void writeString(String v)

NzaeRecord getRecord()

RecordWriter< K, V > Class Reference

RecordWriter writes the output <key, value> pairs to an output table.

Public Member Functions

abstract void close(TaskAttemptContext context)
Close this RecordWriter to future operations.

abstract void initialize(Nzae ae, TaskAttemptContext context)
abstract void write(K key, V
value) Writes a key/value pair.

Detailed Description

RecordWriter writes the output <key, value> pairs to an output table.

Public Member Function Documentation

abstract void close(TaskAttemptContext context)
Close this RecordWriter to future operations.

Parameters
TaskAttemptContext context
the context of the task

Exceptions
IOException

abstract void initialize(Nzae ae, TaskAttemptContext context)

abstract void write(K key, V
value) Writes a key/value pair.

Parameters
key
the key to write.

value
the value to write.

ReduceContext< KEYIN, VALUEIN, KEYOUT, VALUEOUT > Class Refer-ence

The context passed to the Reducer .

Inherits TaskInputOutputContext< KEYIN, VALUEIN, KEYOUT, VALUEOUT >

Public Member Functions

KEYIN getCurrentKey()
Get the current key.

VALUEIN getCurrentValue()
Get the current value.

Iterable<VALUEIN> getValues()
Iterate through the values for the current key, reusing the same value object, which is stored in the context.

boolean nextKey()
Start processing the next group with unique key.

ReduceContext(Configuration conf, ReducerRecordReader< KEYIN, VALUEIN > reader, Record-Writer< KEYOUT, VALUEOUT > writer, StatusReporter reporter)

Detailed Description

The context passed to the Reducer .

Public Member Function Documentation

KEYIN getCurrentKey()
Get the current key.

Returns
the current key object or null if there isn't one

VALUEIN getCurrentValue()
Get the current value.

Returns
the value object that was read into

Iterable<VALUEIN> getValues()
Iterate through the values for the current key, reusing the same value object, which is stored in the context.

Returns
the series of values associated with the current key. All of the objects returned directly and indirectly from this method are reused.

boolean nextKey()
Start processing the next group with unique key.

Returns
true if the next group with unique key exists

ReduceContext(Configuration conf, ReducerRecordReader< KEYIN, VALUEIN > reader, Re-cordWriter< KEYOUT, VALUEOUT > writer, StatusReporter reporter)

Reducer< K2, V2, K3, V3 > Interface Reference

Reduces a set of intermediate values which share a key to a smaller set of values.

Inherits JobConfigurable

Public Member Functions

void reduce(K2 key, Iterator< V2 > values, OutputCollector< K3, V3 > output, Reporter reporter) *Reduces* values for a given key.

Detailed Description

Reduces a set of intermediate values which share a key to a smaller set of values.

The number of Reducers for the job is determined by the number of intermediate data partitions. Reducer implementations can access the JobConf for the job via the configure method and initialize themselves. Sim-ilarly they can use the Closeable#close() method for de-initialization.

The output of the Reducer is **not re-sorted**.

Example:

```
public class MyReducer<K extends Writable, V extends Writable>
extends MapReduceBase implements Reducer<K, V, K, V> {

 static enum MyCounters { NUM_VALUES }

 private String reduceInputText;
  private int reduceInputValue;

 public void configure(JobConf job) {
   reduceInputText = job.get("reduce.input.text");
   reduceInputValue = job.getInt("reduce.input.value", -1);
  }

 public void reduce(K key, Iterator<V> values,
                   OutputCollector<K, V> output, Reporter reporter)
  throws IOException {
   // Process
   while (values.hasNext()) {
     V value = values.next();

        Process the <key, value> pair (assume this takes a while)
         ...
         ...

        Increment counters
     reporter.incrCounter(MyCounters.NUM_VALUES, 1);

        Output the <key, value>
     output.collect(key, value);
   }
  }
}
```

See Also
Reporter
MapReduceBase

Public Member Function Documentation

void reduce(K2 key, Iterator< V2 > values, OutputCollector< K3, V3 > output, Reporter re-porter)

Reduces values for a given key.

Parameters

key the
key.

values
the list of values to reduce.

output
to collect keys and combined values.

Reporter reporter
facility to update counters.

The framework calls this method for each <key, (list of values)> pair in the grouped inputs. The framework will **reuse** the key and value objects that are passed into the reduce, therefore the application should clone the objects they want to keep a copy of. In many cases, all values are combined into zero or one value.

Output pairs are collected with calls to OutputCollector#collect(Object,Object) .

Applications can use the Reporter provided to update counters.

Reducer< KEYIN, VALUEIN, KEYOUT, VALUEOUT > Class Reference

Reduces a set of intermediate values which share a key to a smaller set of values.

Public Member Functions

void run(Context context)
Advanced application writers can use the run(org.netezza.inza.mr.mapreduce.Reducer.Context) method to control how the reduce task works.

Protected Member Functions

void cleanup(Context context)

Called once at the end of the task.

void reduce(KEYIN key, Iterable< VALUEIN > values, Context context) This method is called once for each key.

void setup(Context context) Called once at the start of the task.

Detailed Description

Reduces a set of intermediate values which share a key to a smaller set of values.

Reducer implementations can access the Configuration for the job via the getConfiguration method.

The output of the Reducer is **not re-sorted**.

Example:

```
public class IntSumReducer<Key> extends Reducer<Key,IntWritable,
                                   Key,IntWritable> {
  private IntWritable result = new IntWritable();

 public void reduce(Key key, Iterable<IntWritable> values,
                 Context context) throws IOException {
   int sum = 0;
   for (IntWritable val : values) {
     sum += val.get();
   }
   result.set(sum);
   context.write(key, result);
  }
}
```

See Also

- Mapper

Public Member Function Documentation

void run(Context context)
Advanced application writers can use the run(org.netezza.inza.mr.mapreduce.Reducer.Context) method to control how the reduce task works.

Protected Member Function Documentation

void cleanup(Context context)
Called once at the end of the task.

void reduce(KEYIN key, Iterable< VALUEIN > values, Context context) This method is called once for each key.

Most applications will define their reduce class by overriding this method. The default imple-mentation is an identity function.

void setup(Context context) Called
once at the start of the task.

ReducerRecordReader< KEYIN, VALUEIN > Class Reference

The record reader breaks the data into key/value pairs for input to the Reducer .

Public Member Functions

abstract KEYIN getCurrentKey()
Get the current key.

abstract VALUEIN getCurrentValue()
Get the current value.

abstract boolean hasNextValue()
abstract void initialize(Nzae ae, TaskAttemptContext
context) Called once at initialization.

abstract boolean nextKey()
Jump to the next key group.

abstract boolean nextValue()
Jump to the next value.

Detailed Description

The record reader breaks the data into key/value pairs for input to the Reducer .

Public Member Function Documentation

abstract KEYIN getCurrentKey()
Get the current key.

Returns
the current key or null if there is no current key

abstract VALUEIN getCurrentValue()
Get the current value.

Returns
the object that was read

abstract boolean hasNextValue()

abstract void initialize(Nzae ae, TaskAttemptContext context) Called once at initialization.

Parameters

ae
ae handler

TaskAttemptContext context
the information about the task

abstract boolean nextKey()
Jump to the next key group.

Returns
true if the first key/value pair from the new group was read

abstract boolean nextValue()
Jump to the next value.

Returns
true if the next value was read

ReflectionUtils Class Reference

General reflection utils.

Static Public Member Functions

static void cloneWritableInto(CoreWritable dst, CoreWritable src)
static <T> T copy(Configuration conf, T src, T dst)
Make a copy of the writable object using serialization to a buffer.

static <T> Class<T> getClass(T o)
Return the correctly-typed Class of the given object.

static void logThreadInfo(Log log, String title, long minInterval) Log the current thread stacks at INFO level.

static <T> T newInstance(Class< T > theClass, Configuration conf)
Create an object for the given class and initialize it from conf.

static void printThreadInfo(PrintWriter stream, String title)
Print all of the thread's information and stack traces.

static void setConf(Object theObject, Configuration conf) Check and set 'configuration' if necessary.

static void setContentionTracing(boolean val)

Detailed Description

General reflection utils.

Static Public Member Function Documentation

static void cloneWritableInto(CoreWritable dst, CoreWritable src)

static <T> T copy(Configuration conf, T src, T dst)
Make a copy of the writable object using serialization to a buffer.

Parameters
dst
the object to copy from

src
the object to copy into, which is destroyed

Exceptions
IOException

static <T> Class<T> getClass(T o)
Return the correctly-typed Class of the given object.

Parameters
o
object whose correctly-typed Class is to be obtained

Returns
the correctly typed Class of the given object.

static void logThreadInfo(Log log, String title, long minInterval) Log the current thread stacks at INFO level.

Parameters
log
the logger that logs the stack trace

title
a descriptive title for the call stacks

minInterval
the minimum time from the last

static <T> T newInstance(Class< T > theClass, Configuration conf)
Create an object for the given class and initialize it from conf.

Parameters

theClass
class of which an object is created

Configuration
conf Configuration

Returns
a new object

static void printThreadInfo(PrintWriter stream, String title) Print all of the thread's information and stack traces.

Parameters

stream
the stream to

title
a string title for the stack trace

static void setConf(Object theObject, Configuration conf) Check and set 'configuration' if necessary.

Parameters

theObject
object for which to set configuration

Configuration
conf Configuration

static void setContentionTracing(boolean val)

RegexMapper< K > Class Reference

A Mapper that extracts text matching a regular expression.

Inherits org::netezza::inza::mr::mapreduce::Mapper< K, Text, Text, LongWritable >

Public Member Functions

void map(K key, Text value, Context context)
void setup(Context context)

Static Public Attributes

GROUP
PATTERN

Detailed Description

A Mapper that extracts text matching a regular expression.

Public Member Function Documentation

void map(K key, Text value, Context context)

void setup(Context context)

Static Member Data Documentation

String GROUP="mapreduce.mapper.regexmapper.group"

String PATTERN="mapreduce.mapper.regex"

Reporter Interface Reference

A facility for Map-Reduce applications to update Counters .

Public Member Functions

abstract Counter getCounter(Enum<?> key)
Get the Counter identified by the given Enum type.

abstract Counter getCounter(String group, String name)
Get the Counter of the given group with the given name.

abstract void incrCounter(Enum<?> key, long amount)
Increments the counter identified by the key, which can be of any Enum type, by the specified amount.

abstract void incrCounter(String group, String counter, long amount)
Increments the counter identified by the group and counter name by the specified amount.

Detailed Description

A facility for Map-Reduce applications to update Counters .

See Also

▲ Counters

Public Member Function Documentation

abstract Counter getCounter(Enum<?> key)

Get the Counter identified by the given Enum type.

Parameters

key
key to identify the counter

Returns
Counter

the Counter identified by the given key

abstract Counter getCounter(String group, String name)
Get the Counter of the given group with the given name.

Parameters

group counter
group

name counter
name

Returns
Counter

the Counter of the given group/name.

abstract void incrCounter(Enum<?> key, long amount)
Increments the counter identified by the key, which can be of any Enum type, by the specified amount.

Parameters

key
key to identify the counter to be incremented. The key can be be any Enum.

amount
A non-negative amount by which the counter is to be incremented.

abstract void incrCounter(String group, String counter, long amount)
Increments the counter identified by the group and counter name by the specified amount.

Parameters

group
name to identify the group of the counter to be incremented.

counter
name to identify the counter within the group.

amount
A non-negative amount by which the counter is to be incremented.

RunJar Class Reference

Run a map/reduce job jar.

Static Public Member Functions

static void main(String[] args)
Run a map/reduce job jar.

static void unJar(File jarFile, File toDir)
Unpack a jar file into a directory.

Detailed Description

Run a map/reduce job jar.

Static Public Member Function Documentation

static void main(String[] args)
Run a map/reduce job jar.

If the main class is not in the jar's manifest, then it must be provided on the command line.

static void unJar(File jarFile, File toDir)
Unpack a jar file into a directory.

Serialization< T > Interface Reference

Public Member Functions

boolean accept(Class<?> c)
Allows clients to test whether this Serialization supports the given class.

Deserializer<T> getDeserializer(Class< T > c)
Serializer<T> getSerializer(Class< T > c)

Detailed Description

Encapsulates a Serializer / Deserializer pair.

Public Member Function Documentation

boolean accept(Class<?> c)
Allows clients to test whether this Serialization supports the given class.

Deserializer<T> getDeserializer(Class< T > c)

Returns
a Deserializer for the given class.

Serializer<T> getSerializer(Class< T > c)
Returns
a Serializer for the given class.

SerializationFactory Class Reference

Inherits Configured

Public Member Functions

SerializationFactory(Configuration conf)
public<T> Deserializer<T> getDeserializer(Class< T > c)
public<T> Serialization<T> getSerialization(Class< T > c)
public<T> Serializer<T> getSerializer(Class< T > c)

Detailed Description

A factory for Serialization s.

Public Member Function Documentation

SerializationFactory(Configuration conf)
Serializations are found by reading the io.serializations property from conf, which is a comma-delimited list of classnames.

public<T> Deserializer<T> getDeserializer(Class< T > c)

public<T> Serialization<T> getSerialization(Class< T > c)

public<T> Serializer<T> getSerializer(Class< T > c)

Serializer< T > Interface Reference

Public Member Functions

void open(OutputStream out)
void close()
void serialize(T t)

Detailed Description

Provides a facility for serializing objects of type <T> to an OutputStream .

Serializers are stateful, but must not buffer the output since other producers may write to the out-put between calls to serialize(Object) .

Public Member Function Documentation

void open(OutputStream out)
Prepare the serializer for writing.

void close()
Close the underlying output stream and clear up any resources.

void serialize(T t)
Serialize t to the underlying output stream.

Shell Class Reference

A base class for running a Unix command.

Public Member Functions

int getExitCode()
get the exit code

Process getProcess()
get the current sub-process executing the given command

boolean isTimedOut()
To check if the passed script to shell command executor timed out or not.

Shell()
Shell(long interval)

Protected Member Functions

abstract String [] getExecString()
return an array containing the command name & its parameters

abstract void parseExecResult(BufferedReader
lines) Parse the execution result.

void run()
check to see if a command needs to be executed and execute if needed

void setEnvironment(Map< String, String >
env) set the environment for the command

void setWorkingDirectory(File
dir) set the working directory

Static Public Attributes

LOG
SET_GROUP_COMMAND
SET_OWNER_COMMAND
a Unix command to set owner

SET_PERMISSION_COMMAND
a Unix command to set permission

USER_NAME_COMMAND
a Unix command to get the current user's name

WINDOWS
Set to true on Windows platforms.

Static Public Member Functions

static String execCommand(String...cmd)
Static method to execute a shell command.

static String execCommand(Map< String, String > env,
String...cmd) Static method to execute a shell command.

static String execCommand(Map< String, String > env, String[] cmd, long
timeout) Static method to execute a shell command.

static String [] getGET_PERMISSION_COMMAND()
Return a Unix command to get permission information.

static String [] getGroupsCommand()
a Unix command to get the current user's groups list

static String [] getGroupsForUserCommand(final String
user) a Unix command to get a given user's groups list

static String [] getUsersForNetgroupCommand(final String
netgroup) a Unix command to get a given netgroup's user list

Detailed Description

A base class for running a Unix command.

Shell can be used to run unix commands like du or df. It also offers facilities to gate commands by time-in-tervals.

Public Member Function Documentation

int getExitCode()

get the exit code

Returns
the exit code of the process

Process getProcess()
get the current sub-process executing the given command

Returns
process executing the command

boolean isTimedOut()
To check if the passed script to shell command executor timed out or not.

Returns
if the script timed out.

Shell()

Shell(long interval)
Parameters
interval
the minimum duration to wait before re-executing the command.

Protected Member Function Documentation

abstract String [] getExecString()
return an array containing the command name & its parameters

abstract void parseExecResult(BufferedReader lines) Parse the execution result.

void run()
check to see if a command needs to be executed and execute if needed

void setEnvironment(Map< String, String > env) set the environment for the command

Parameters
env
Mapping of environment variables

void setWorkingDirectory(File dir) set the working directory

Parameters

dir

The directory where the command would be executed

Static Member Data Documentation

final Log LOG= LogFactory.getLog(Shell.class)

final String SET_GROUP_COMMAND="chgrp"

final String SET_OWNER_COMMAND="chown"
a Unix command to set owner

final String SET_PERMISSION_COMMAND="chmod"
a Unix command to set permission

final String USER_NAME_COMMAND="whoami"
a Unix command to get the current user's name

final boolean WINDOWS= System.getProperty("os.name").startsWith("Windows")
Set to true on Windows platforms.

Static Public Member Function Documentation

static String execCommand(String...cmd)
Static method to execute a shell command.

Parameters

cmd

shell command to execute.

Returns

the output of the executed command.

Covers most of the simple cases without requiring the user to implement the Shell interface.

static String execCommand(Map< String, String > env, String...cmd) Static method to execute a shell command.

Parameters

env

the map of environment key=value

cmd
shell command to execute.

Returns
the output of the executed command.

Covers most of the simple cases without requiring the user to implement the Shell interface.

static String execCommand(Map< String, String > env, String[] cmd, long timeout) Static method to execute a shell command.

Parameters

env
the map of environment key=value

cmd
shell command to execute.

timeout
time in milliseconds after which script should be marked timeout

Returns
the output of the executed command.o

Covers most of the simple cases without requiring the user to implement the Shell interface.

static String [] getGET_PERMISSION_COMMAND()
Return a Unix command to get permission information.

static String [] getGroupsCommand()
a Unix command to get the current user's groups list

static String [] getGroupsForUserCommand(final String user) a Unix command to get a given user's groups list

static String [] getUsersForNetgroupCommand(final String netgroup) a Unix command to get a given netgroup's user list

ShellCommandExecutor Class Reference

A simple shell command executor.

Inherits Shell

Public Member Functions

void execute()
Execute the shell command.

String getOutput()
Get the output of the shell command.

ShellCommandExecutor(String[] execString)
ShellCommandExecutor(String[] execString, File dir)
ShellCommandExecutor(String[] execString, File dir, Map< String, String > env, long timeout) Create a new instance of the ShellCommandExecutor to execute a command.

ShellCommandExecutor(String[] execString, File dir, Map< String, String > env)
String toString()
Returns the commands of this instance.

Protected Member Functions

String [] getExecString()
void parseExecResult(BufferedReader lines)

Detailed Description

A simple shell command executor.

ShellCommandExecutorshould be used in cases where the output of the command needs no explicit parsing and where the command, working directory and the environment remains unchanged. The output of the command is stored as-is and is expected to be small.

Public Member Function Documentation

void execute()
Execute the shell command.

String getOutput()
Get the output of the shell command.

ShellCommandExecutor(String[] execString)

ShellCommandExecutor(String[] execString, File dir)

ShellCommandExecutor(String[] execString, File dir, Map< String, String > env, long timeout) Create a new instance of the ShellCommandExecutor to execute a command.

Parameters

execString
The command to execute with arguments

dir

If not-null, specifies the directory which should be set as the current working directory for the command. If null, the current working directory is not modified.

env
If not-null, environment of the command will include the key-value pairs specified in the map. If null, the current environment is not modified.

timeout
Specifies the time in milliseconds, after which the command will be killed and the status marked as timedout. If 0, the command will not be timed out.

ShellCommandExecutor(String[] execString, File dir, Map< String, String > env)

String toString()
Returns the commands of this instance.

Returns
a string representation of the object.

Arguments with spaces in are presented with quotes round; other arguments are presented raw

Protected Member Function Documentation

String [] getExecString()

void parseExecResult(BufferedReader lines)

StatusReporter Class Reference

Public Member Functions

abstract Counter getCounter(Enum<?> key)
Get the Counter identified by the given Enum type.

abstract Counter getCounter(String group, String name)
Get the Counter of the given group with the given name.

Public Member Function Documentation

abstract Counter getCounter(Enum<?> key)
Get the Counter identified by the given Enum

type. ▲ Parameters

key
key to identify the counter

Returns
Counter

the Counter identified by the given key

abstract Counter getCounter(String group, String name)
Get the Counter of the given group with the given name.

Parameters
group counter
group

name counter
name

Returns
Counter

the Counter of the given group/name.

StringUtils Class Reference

General string utils.

Public Types

enum TraditionalBinaryPrefix {
KILO=(1024), MEGA=(KILO.value << 10), GIGA=(MEGA.value << 10), TERA=(GIGA.value << 10), PETA=(TERA.value << 10), EXA=(PETA.value << 10) }

The traditional binary prefixes, kilo, mega, ..., exa, which can be represented by a 64-bit integer.

Static Public Attributes

COMMA
COMMA_STR
ESCAPE_CHAR

Static Public Member Functions

static String arrayToString(String[] strs)
Given an array of strings, return a comma-separated list of its elements.

static String byteDesc(long len)
Return an abbreviated English-language desc of the byte length.

static String byteToHexString(byte[] bytes, int start, int end)
Given an array of bytes it will convert the bytes to a hex string representation of the bytes.

static String byteToHexString(byte bytes[])

Same as byteToHexString(bytes, 0, bytes.length).

static String camelize(String s)
Convert SOME_STUFF to SomeStuff.

static String capitalize(String
s) Capitalize a word.

static String escapeHTML(String string)
Escapes HTML Special characters present in the string.

static String escapeString(String str)
Escape commas in the string using the default escape char.

static String escapeString(String str, char escapeChar, char charToEscape)
Escape charToEscape in the string with the escape char escapeChar

static String escapeString(String str, char escapeChar, char[] charsToEscape)
static int findNext(String str, char separator, char escapeChar, int start, StringBuilder split)
Finds the first occurrence of the separator character ignoring the escaped separators starting from the index.

static String formatPercent(double done, int digits)
Format a percentage for presentation to the user.

static String formatTime(long timeDiff)
Given the time in long milliseconds, returns a String in the format Xhrs, Ymins, Z sec.

static String formatTimeDiff(long finishTime, long startTime)
Given a finish and start time in long milliseconds, returns a String in the format Xhrs, Ymins, Z sec, for the time difference between two times.

static String getFormattedTimeWithDiff(DateFormat dateFormat, long finishTime, long start-Time)
Formats time in ms and appends difference (finishTime - startTime) as returned by format-TimeDiff .

static String getHostname()
Return hostname without throwing exception.

static Collection<String> getStringCollection(String
str) Returns a collection of strings.

static String [] getStrings(String
str) Returns an arraylist of strings.

static byte [] hexStringToByte(String hex)
Given a hexstring this will return the byte array corresponding to the string.

static String humanReadableInt(long number)
Given an integer, return a string that is in an approximate, but human readable format.

static String join(CharSequence separator, Iterable< String >
strings) Concatenates strings, using a separator.

static String join(CharSequence separator, String[] strings)

Concatenates strings, using a separator.

static synchronized String limitDecimalTo2(double d)
static String simpleHostname(String fullHostname) Given a full hostname, return the word upto the first dot.

static String [] split(String str)
Split a string using the default separator.

static String [] split(String str, char escapeChar, char separator) Split a string using the given separator.

static String stringifyException(Throwable e)
Make a string representation of the exception.

static URI [] stringToURI(String[] str)
static String unEscapeString(String str, char escapeChar, char charToEscape)
Unescape charToEscape in the string with the escape char escapeChar

static String unEscapeString(String str, char escapeChar, char[] charsToEscape)
static String unEscapeString(String str)
Unescape commas in the string using the default escape char.

static String uriToString(URI[] uris)

Detailed Description

General string utils.

Enumeration Type Documentation

enum TraditionalBinaryPrefix
The traditional binary prefixes, kilo, mega, ..., exa, which can be represented by a 64-bit integer.

KILO

MEGA

GIGA

TERA

PETA

EXA

Static Member Data Documentation

final static char COMMA= ','

final static String COMMA_STR=","

final static char ESCAPE_CHAR= '\\'

Static Public Member Function Documentation

static String arrayToString(String[] strs)
Given an array of strings, return a comma-separated list of its elements.

Parameters

strs
Array of strings

Returns
Empty string if strs.length is 0, comma separated list of strings otherwise

static String byteDesc(long len)
Return an abbreviated English-language desc of the byte length.

static String byteToHexString(byte[] bytes, int start, int end)
Given an array of bytes it will convert the bytes to a hex string representation of the bytes.

Parameters

bytes

start
start index, inclusively

end
end index, exclusively

Returns
hex string representation of the byte array

static String byteToHexString(byte bytes[]) Same
as byteToHexString(bytes, 0, bytes.length).

static String camelize(String s)
Convert SOME_STUFF to SomeStuff.

Parameters

s
input string

Returns
camelized string

static String capitalize(String
s) Capitalize a word.

Parameters

s
the input string

Returns
capitalized string

static String escapeHTML(String string)
Escapes HTML Special characters present in the string.

Parameters
string
Returns
HTML Escaped String representation

static String escapeString(String str)
Escape commas in the string using the default escape char.

Parameters
str
a string

Returns
an escaped string

static String escapeString(String str, char escapeChar, char charToEscape)
Escape charToEscape in the string with the escape char escapeChar

Parameters
str
string

escapeChar
escape char

charToEscape
the char to be escaped

Returns
an escaped string

static String escapeString(String str, char escapeChar, char[] charsToEscape)
Parameters
charsToEscape
array of characters to be escaped

static int findNext(String str, char separator, char escapeChar, int start, StringBuilder split)
Finds the first occurrence of the separator character ignoring the escaped separators starting from the index.

Parameters

str
the source string

separator
the character to find

escapeChar
character used to escape

start
from where to search

split
used to pass back the extracted string

Note the substring between the index and the position of the separator is passed.

static String formatPercent(double done, int digits)
Format a percentage for presentation to the user.

Parameters

done
the percentage to format (0.0 to 1.0)

digits
the number of digits past the decimal point

Returns
a string representation of the percentage

static String formatTime(long timeDiff)
Given the time in long milliseconds, returns a String in the format Xhrs, Ymins, Z sec.

Parameters

timeDiff
The time difference to format

static String formatTimeDiff(long finishTime, long startTime)
Given a finish and start time in long milliseconds, returns a String in the format Xhrs, Ymins, Z sec, for the time difference between two times.

Parameters

finishTime
finish time

startTime
start time

If finish time comes before start time then negative valeus of X, Y and Z wil return.

static String getFormattedTimeWithDiff(DateFormat dateFormat, long finishTime, long startTime)
Formats time in ms and appends difference (finishTime - startTime) as returned by formatTimeDiff .

Parameters

dateFormat
date format to use

finishTime
fnish time

startTime
start time

Returns
formatted value.

If finish time is 0, empty string is returned, if start time is 0 then difference is not appended to return value.

static String getHostname()
Return hostname without throwing exception.

Returns
hostname

static Collection<String> getStringCollection(String str) Returns a collection of strings.

Parameters

str
comma seperated string values

Returns
an ArrayList of string values

static String [] getStrings(String str) Returns an arraylist of strings.

Parameters

str
the comma seperated string values

Returns
the arraylist of the comma seperated string values

static byte [] hexStringToByte(String hex)
Given a hexstring this will return the byte array corresponding to the string.

Parameters

hex
the hex String array

Returns
a byte array that is a hex string representation of the given string. The size of the byte ar-ray is therefore hex.length/2

static String humanReadableInt(long number)
Given an integer, return a string that is in an approximate, but human readable format.

Parameters
number
the number to format

Returns
a human readable form of the integer

It uses the bases 'k', 'm', and 'g' for 1024, 1024**2, and 1024**3.

static String join(CharSequence separator, Iterable< String > strings) Concatenates strings, using a separator.

Parameters
separator
Separator to join with.

strings Strings
to join.

Returns
the joined string

static String join(CharSequence separator, String[] strings) Concatenates strings, using a separator.

Parameters
separator
to join with

strings
to join

Returns
the joined string

static synchronized String limitDecimalTo2(double d)

static String simpleHostname(String fullHostname) Given a full hostname, return the word upto the first dot.

Parameters

fullHostname the
full hostname

Returns
the hostname to the first dot

static String [] split(String str)
Split a string using the default separator.

Parameters
str
a string that may have escaped separator

Returns
an array of strings

static String [] split(String str, char escapeChar, char separator) Split a string using the given separator.

Parameters
str
a string that may have escaped separator

escapeChar
a char that be used to escape the separator

separator
a separator char

Returns
an array of strings

static String stringifyException(Throwable e)
Make a string representation of the exception.

Parameters
e
The exception to stringify

Returns
A string with exception name and call stack.

static URI [] stringToURI(String[] str)
Parameters
str

static String unEscapeString(String str, char escapeChar, char charToEscape)
Unescape charToEscape in the string with the escape char escapeChar

Parameters
str

string

escapeChar
escape char

charToEscape
the escaped char

Returns
an unescaped string

static String unEscapeString(String str, char escapeChar, char[] charsToEscape)
Parameters
charsToEscape
array of characters to unescape

static String unEscapeString(String str)
Unescape commas in the string using the default escape char.

Parameters
str
a string

Returns
an unescaped string

static String uriToString(URI[] uris)
Parameters
uris

TaskAttemptContext Class Reference

The context for task attempts.

Inherits JobContext

Public Member Functions

Counter getCounter(Enum<?> counterName)
Get the Counter for the given counterName.

Counter getCounter(String groupName, String counterName)
Get the Counter for the given groupName and counterName.

int getTaskDatasliceID()
Get the datasliceID for the running task.

TaskAttemptContext(Configuration conf, StatusReporter reporter)

Detailed Description

The context for task attempts.

Public Member Function Documentation

Counter getCounter(Enum<?> counterName)

Get the Counter for the given counterName.

Parameters

counterName

counter name

Returns

Counter

the Counter for the given counterName

Counter getCounter(String groupName, String counterName)

Get the Counter for the given groupName and counterName.

Parameters

counterName

counter name

Returns

Counter

the Counter for the given groupName and counterName

int getTaskDatasliceID()

Get the datasliceID for the running task.

Returns

datasliceID

TaskAttemptContext(Configuration conf, StatusReporter reporter)

TaskInputOutputContext< KEYIN, VALUEIN, KEYOUT, VALUEOUT > Class Reference

A context object that allows input and output from the task.

Inherits TaskAttemptContext

Public Member Functions

abstract KEYIN getCurrentKey()

Get the current key.

abstract VALUEIN getCurrentValue()
Get the current value.

TaskInputOutputContext(Configuration conf, RecordWriter< KEYOUT, VALUEOUT > output, StatusReporter reporter)
void write(KEYOUT key, VALUEOUT value)
Generate an output key/value pair.

Detailed Description

A context object that allows input and output from the task.

It is only supplied to the Mapper or Reducer .

Public Member Function Documentation

abstract KEYIN getCurrentKey()
Get the current key.

Returns
the current key object or null if there isn't one

abstract VALUEIN getCurrentValue()
Get the current value.

Returns
the value object that was read into

TaskInputOutputContext(Configuration conf, RecordWriter< KEYOUT, VALUEOUT > output, StatusReporter reporter)

void write(KEYOUT key, VALUEOUT value) Generate an output key/value pair.

Text Class Reference

A Writable for strings.

Inherits Writable

Public Member Functions

boolean equals(Object obj)

Returns true iff o is a Text with the same contents.

String get()
Return the contents of this Text .

List<Class<?> > getStorageTypesList()
int hashCode()
void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

void set(String value)
Set to contain the contents of a string.

Text(String value)
Text()
String toString()
void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Detailed Description

A Writable for strings.

Public Member Function Documentation

boolean equals(Object obj)
Returns true iff o is a Text with the same contents.

String get()
Return the contents of this Text .

List<Class<?> > getStorageTypesList()

Returns
list of classes of storage types. These classes are used by the framework for automatic conversion from database fields and for setting column types of output table.

int hashCode()

void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

Parameters

RecordInput in
RecordInput to read this object from.

Exceptions

IOException

void set(String value)
Set to contain the contents of a string.

Text(String value)

Text()

String toString()

void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Parameters
RecordOutput out
RecordOutput to write this object into.

Exceptions
IOException

TokenCounterMapper Class Reference

Tokenize the input values and emit each word with a count of 1.

Inherits org::netezza::inza::mr::mapreduce::Mapper< Object, Text, Text, IntWritable >

Public Member Functions

void map(Object key, Text value, Context context)

Detailed Description

Tokenize the input values and emit each word with a count of 1.

Public Member Function Documentation

void map(Object key, Text value, Context context)

TokenCountMapper< K > Class Reference

A Mapper that maps text values into <token,freq> pairs.

Inherits MapReduceBase

Public Member Functions

void map(K key, Text value, OutputCollector< Text, LongWritable > output, Reporter reporter) Writes <token, 1> pair for each token found in the given input Text.

Detailed Description

A Mapper that maps text values into <token,freq> pairs.

Uses StringTokenizer to break text into tokens.

Public Member Function Documentation

void map(K key, Text value, OutputCollector< Text, LongWritable > output, Reporter reporter) Writes <token, 1> pair for each token found in the given input Text.

Tool Interface Reference

A tool interface that supports handling of generic command-line options.

Inherits Configurable

Public Member Functions

int run(String[] args)
Execute the command with the given arguments.

Detailed Description

A tool interface that supports handling of generic command-line options.

Tool , is the standard for any Map-Reduce tool/application. The tool/application should delegate the hand-ling of

to ToolRunner#run(Tool, String[]) and only handle its custom arguments.

Here is how a typical Tool is implemented:

```
public class MyApp extends Configured implements Tool {
  public int run(String[] args) throws Exception {
      Configuration processed by ToolRunner
    Configuration conf = getConf();
      Create a Job using the processed
    conf Job job = new Job(conf);
    job.setJarByClass(this.getClass());
       Process custom command-line options, e.g. database name, input and output table
    names job.setDatabaseName(args[0]);
    job.setInputTableName(args[1]);
    job.setInputKeyColumnNames(args[2]);
    job.setInputValueColumnNames(args[3]);

    job.setOutputTableName(args[4]);
    job.setOutputKeyColumnNames(args[5]);
    job.setOutputValueColumnNames(args[6]);

      Specify various job-specific parameters
    job.setJobName("my-app");
    job.setMapperClass(MyApp.MyMapper.class);
    job.setReducerClass(MyApp.MyReducer.class);

      Specify input and output types of data that will be passed to
    mappers/reducers job.setMapInputKeyClass(LongWritable.class);
    job.setMapInputValueClass(Text.class); job.setMapOutputKeyClass(Text.class);
    job.setMapOutputKeyColumnSize(0, MAX_WORD_LENGTH);
    job.setMapOutputValueClass(IntWritable.class);
    job.setReduceOutputKeyClass(Text.class);
    job.setReduceOutputKeyColumnSize(0, MAX_WORD_LENGTH);
    job.setReduceOutputValueClass(IntWritable.class);


      Submit the job, then poll for progress until the job is complete
    boolean success = JobRunner.runJob(job);
    if (!success)
          return -1;

    return 0;
  }

  public static void main(String[] args) throws Exception {
      Let ToolRunner handle generic command-line
    options int res = ToolRunner.run(new MyApp(), args);

    System.exit(res);
  }
}
```

See Also

GenericOptionsParser

ToolRunner

Public Member Function Documentation

int run(String[] args)

Execute the command with the given arguments.

Parameters

args

command specific arguments

Returns
exit code

Exceptions ►
Exception

ToolRunner Class Reference

A utility to help run Tool s.

Static Public Member Functions

static void printGenericCommandUsage(PrintStream out)
Prints generic command-line argurments and usage information.

static int run(Configuration conf, Tool tool, String[] args)
Runs the given Tool by Tool#run(String[]) , after parsing with the given generic arguments.

static int run(Tool tool, String[] args)
Runs the Tool with its Configuration.

Detailed Description

A utility to help run Tool s.

ToolRunner can be used to run classes implementing Tool interface. It works in conjunction with Generi-cOptionsParser to parse the

and modifies the Configuration of the Tool . The application-specific options are passed along without being modified.

See Also
Tool
GenericOptionsParser

Static Public Member Function Documentation

static void printGenericCommandUsage(PrintStream out) Prints generic command-line argurments and usage information.

Parameters
out
stream to write usage information to.

static int run(Configuration conf, Tool tool, String[] args)
Runs the given Tool by Tool#run(String[]) , after parsing with the given generic arguments.

Parameters
Configuration conf
Configuration for the Tool .

Tool tool
Tool to run.

args
command-line arguments to the tool.

Returns
exit code of the Tool#run(String[]) method.

Uses the given Configuration, or builds one if null.

Sets the Tool 's configuration with the possibly modified version of the conf.

static int run(Tool tool, String[] args)
Runs the Tool with its Configuration.

Parameters

Tool tool
Tool to run.

args
command-line arguments to the tool.

Returns
exit code of the Tool#run(String[]) method.

Equivalent to run(tool.getConf(), tool, args).

TypeConversionUnsupported Class Reference

Public Member Functions

String getMessage()
TypeConversionUnsupported(Class<?> from, Class<?> to)

Public Member Function Documentation

String getMessage()

TypeConversionUnsupported(Class<?> from, Class<?> to)

TypeConverter< FROM, TO > Interface Reference

Public Member Functions

TO convert(FROM from)

Public Member Function Documentation

TO convert(FROM from)

TypeConverterFactory Class Reference

Static Public Member Functions

static <FROM,TO> TypeConverter<FROM, TO> getConverter(Class< FROM > fromClass, Class< TO > to-Class)

Static Public Member Function Documentation

static <FROM,TO> TypeConverter<FROM, TO> getConverter(Class< FROM > fromClass, Class< TO > toClass)

Writable Interface Reference

A serializable object which implements a simple, efficient, serialization protocol, based on RecordInput and RecordOutput .

Public Member Functions

void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

List<Class<?> > getStorageTypesList()
void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

Detailed Description

A serializable object which implements a simple, efficient, serialization protocol, based on RecordInput and RecordOutput .

Any key or value type in the INZA MapReduce framework implements this interface.

Example:

```
public class MyWritable implements Writable {
    Some data
 private int counter;
 private long timestamp;
public void write(RecordOutput out) throws IOException {
  out.writeInt(counter);
  out.writeLong(timestamp);
 }
public void readFields(RecordInput in) throws
  IOException { counter = in.readInt();
  timestamp = in.readLong();
 }
public List<Class<?>> getStorageTypesList() {
  List<Class<?>> ret = new ArrayList<Class<?>>();
  ret.add(Integer.class);
  ret.add(Long.class);
  return ret;
 }
}
```

Public Member Function Documentation

void write(RecordOutput out)
Write the fields of this object to out, based on a database record.

Parameters
RecordOutput out
RecordOutput to write this object into.

Exceptions
IOException

List<Class<?> > getStorageTypesList()
Returns
list of classes of storage types. These classes are used by the framework for automatic con-version from database fields and for setting column types of output table.

void readFields(RecordInput in)
Read the fields of this object from in, based on a database record.

Parameters
RecordInput in
RecordInput to read this object from.

Exceptions

IOException

WritableUtils Class Reference

Static Public Member Functions

static <TextendsCoreWritable> T clone(T orig, Configuration conf)
Make a copy of a writable object using serialization to a buffer.

static void cloneInto(CoreWritable dst, CoreWritable src)
Make a copy of the writable object using serialiation to a buffer.

static int decodeVIntSize(byte value)
Parse the first byte of a vint/vlong to determine the number of bytes.

static void displayByteArray(byte[] record)
static int getVIntSize(long i)
Get the encoded length if an integer is stored in a variable-length format.

static boolean isNegativeVInt(byte value)
Given the first byte of a vint/vlong, determine the sign.

static byte [] readCompressedByteArray(DataInput in)
static String readCompressedString(DataInput in)
static String [] readCompressedStringArray(DataInput in)
static <TextendsEnum<T> T readEnum(DataInput in, Class< T > enumType)
Read an Enum value from DataInput, Enums are read and written using String values.

static String readString(DataInput in)
static String [] readStringArray(DataInput in)
static int readVInt(DataInput stream)
Reads a zero-compressed encoded integer from input stream and returns it.

static long readVLong(DataInput stream)
Reads a zero-compressed encoded long from input stream and returns it.

static void skipCompressedByteArray(DataInput in)
static void skipFully(DataInput in, int len)
Skip *len* number of bytes in input stream *in*

static byte [] toByteArray(CoreWritable...writables)
Convert writables to a byte array.

static int writeCompressedByteArray(DataOutput out, byte[] bytes)
static int writeCompressedString(DataOutput out, String s)
static void writeCompressedStringArray(DataOutput out, String[] s)
static void writeEnum(DataOutput out, Enum<?>
enumVal) writes String value of enum to DataOutput.

static void writeString(DataOutput out, String s)
static void writeStringArray(DataOutput out, String[] s)
static void writeVInt(DataOutput stream, int i)
Serializes an integer to a binary stream with zero-compressed encoding.

static void writeVLong(DataOutput stream, long i)
Serializes a long to a binary stream with zero-compressed encoding.

Static Public Member Function Documentation

static <TextendsCoreWritable> T clone(T orig, Configuration conf)
Make a copy of a writable object using serialization to a buffer.

Parameters
orig
The object to copy

Returns
The copied object

static void cloneInto(CoreWritable dst, CoreWritable src) Make a copy of the writable object using serialiation to a buffer.

Parameters
CoreWritable dst
the object to copy from

CoreWritable src
the object to copy into, which is destroyed

Exceptions
IOException
use ReflectionUtils.cloneInto instead.

static int decodeVIntSize(byte value)
Parse the first byte of a vint/vlong to determine the number of bytes.

Parameters
value
the first byte of the vint/vlong

Returns
the total number of bytes (1 to 9)

static void displayByteArray(byte[] record)

static int getVIntSize(long i)
Get the encoded length if an integer is stored in a variable-length format.

Returns
the encoded length

static boolean isNegativeVInt(byte value)
Given the first byte of a vint/vlong, determine the sign.

Parameters
value
the first byte

Returns
is the value negative

static byte [] readCompressedByteArray(DataInput in)

static String readCompressedString(DataInput in)

static String [] readCompressedStringArray(DataInput in)

static <TextendsEnum<T> T readEnum(DataInput in, Class< T > enumType)
Read an Enum value from DataInput, Enums are read and written using String values.

Parameters
<T>
Enum type

in
DataInput to read from

enumType
Class type of Enum

Returns
Enum represented by String read from DataInput

Exceptions
IOException

static String readString(DataInput in)

static String [] readStringArray(DataInput in)

static int readVInt(DataInput stream)
Reads a zero-compressed encoded integer from input stream and returns it.

Parameters
stream
Binary input stream

Returns
deserialized integer from stream.

Exceptions
java.io.IOException

static long readVLong(DataInput stream)
Reads a zero-compressed encoded long from input stream and returns it.

Parameters
stream
Binary input stream

Returns
deserialized long from stream.

Exceptions
java.io.IOException

static void skipCompressedByteArray(DataInput in)

static void skipFully(DataInput in, int len)
Skip *len* number of bytes in input stream *in*

Parameters
in
input stream

len
number of bytes to skip

Exceptions
IOException

static byte [] toByteArray(CoreWritable...writables)
Convert writables to a byte array.

static int writeCompressedByteArray(DataOutput out, byte[] bytes)

static int writeCompressedString(DataOutput out, String s)

static void writeCompressedStringArray(DataOutput out, String[] s)

**static void writeEnum(DataOutput out, Enum<?>
enumVal)** writes String value of enum to DataOutput.

Parameters
out
Dataoutput stream

enumVal
enum value

Exceptions

IOException

static void writeString(DataOutput out, String s)

static void writeStringArray(DataOutput out, String[] s)

static void writeVInt(DataOutput stream, int i)

Serializes an integer to a binary stream with zero-compressed encoding.

Parameters

stream

Binary output stream

i

Integer to be serialized

Exceptions

java.io.IOException

For -120 <= i <= 127, only one byte is used with the actual value. For other values of i, the first byte value indicates whether the integer is positive or negative, and the number of bytes that follow. If the first byte value v is between -121 and -124, the following integer is positive, with number of bytes that follow are -(v+120). If the first byte value v is between -125 and -128, the following integer is negative, with number of bytes that follow are -(v+124). Bytes are stored in the high-non-zero-byte-first order.

static void writeVLong(DataOutput stream, long i)

Serializes a long to a binary stream with zero-compressed encoding.

Parameters

stream

Binary output stream

i

Long to be serialized

Exceptions

java.io.IOException

For -112 <= i <= 127, only one byte is used with the actual value. For other values of i, the first byte value indicates whether the long is positive or negative, and the number of bytes that follow. If the first byte value v is between -113 and -120, the following long is positive, with number of bytes that follow are -(v+112). If the first byte value v is between -121 and -128, the following long is negative, with num-ber of bytes that follow are -(v+120). Bytes are stored in the high-non-zero-byte-first order.

Notices and Trademarks

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this docu-ment. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellec-tual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IM-PLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publica-tion. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:
IBM Corporation
26 Forest Street
Marlborough, MA 01752 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been es-timated through extrapolation. Actual results may vary. Users of this document should verify the ap-plicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only. This information is for planning purposes only. The in-formation herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. (enter the year or years). All rights reserved.

Trademarks

IBM, the IBM logo, ibm.com and Netezza are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™),these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trade-mark information" at ibm.com/legal/copytrade.shtml.

The following terms are trademarks or registered trademarks of other companies:

Adobe is a registered trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

NEC is a registered trademark of NEC Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Red Hat is a trademark or registered trademark of Red Hat, Inc. in the United States and/or other countries.

D-CC, D-C++, Diab+, FastJ, pSOS+, SingleStep, Tornado, VxWorks, Wind River, and the Wind River logo are trademarks, registered trademarks, or service marks of Wind River Systems, Inc. Tornado patent pending.

APC and the APC logo are trademarks or registered trademarks of American Power Conversion Corporation.

Other company, product or service names may be trademarks or service marks of others.

Regulatory and Compliance

Regulatory Notices

Install the NPS system in a restricted-access location. Ensure that only those trained to operate or service the equipment have physical access to it. Install each AC power outlet near the NPS rack that plugs into it, and keep it freely accessible. Provide approved 30A circuit breakers on all power sources.

Product may be powered by redundant power sources. Disconnect ALL power sources before servi-cing. High leakage current. Earth connection essential before connecting supply. Courant de fuite élevé. Raccordement à la terre indispensable avant le raccordement au réseau.

Homologation Statement

This product may not be certified in your country for connection by any means whatsoever to interfaces of public telecommunications networks. Further certification may be required by law prior to making any such connection. Contact an IBM representative or reseller for any questions.

FCC - Industry Canada Statement

This equipment has been tested and found to comply with the limits for a Class A digital device, pursu-ant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

This Class A digital apparatus meets all requirements of the Canadian Interference-Causing Equipment Regulations.

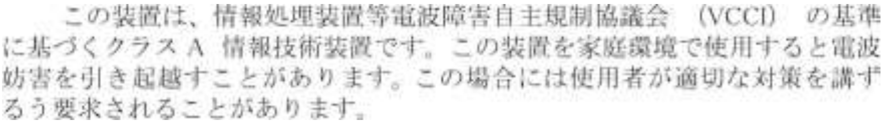
Cet appareil numérique de la classe A respecte toutes les exigences du Règlement sur le matériel brouilleur du Canada.

CE Statement (Europe)

This product complies with the European Low Voltage Directive 73/23/EEC and EMC Directive 89/336/EEC as amended by European Directive 93/68/EEC.

Warning: This is a class A product. In a domestic environment this product may cause radio interfer-ence in which case the user may be required to take adequate measures.

VCCI Statement



この装置は、情報処理装置等電波障害自主規制協議会（VCCI）の基準に基づくクラスA情報技術装置です。この装置を家庭環境で使用すると電波妨害を引き起越すことがあります。この場合には使用者が適切な対策を講ずるう要求されることがあります。

Index