CDVA Evaluation Framework

2.1

Generated by Doxygen 1.8.11

Contents

Chapter 1

Documentation

This is the documentation of the C++ classes implementing the MPEG CDVA Evaluation Framework.The software implements the recommendations contained in the following documents (at http://wg11.sc29.org/):

- N15338:"Evaluation Framework for Compact Descriptors for Video Analysis - Search and Retrieval", July 2015, Warsaw, Poland
- N15729:"Evaluation Framework for Compact Descriptors for Video Analysis - Search and Retrieval – Version 2.0", October 2015, Geneva, CH

Documentation on how to build and install the code is contained in the "CDVA_build_run_instructions" document which can be found in the "docs" directory.

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 mpeg7cdva Namespace Reference

Namespace used to encapsulate all MPEG-7 CDVA declarations.

Data Structures

- class Buffer

  A container class for a byte array, intended to replace all malloc() and new() instructions in the main code.
- class CdvaException

  Class defining a specific exception for CDVA.
- class CdvaImpl

  A CDVA implementation based on multiple CDVS descriptors.
- class ExtractData

  A class containing the results of an extraction operation.
- class FileManager

  Helper class to manage lists of file names.
- class LogManager

  Helper class to produce log files in various formats (csv, text, XML, etc.)
- class MatchData

  A class containing the results of a matching or retrieval operation.
- class MCdvsDescriptor

  Extends the CdvsDescriptor with time and size information.

Typedefs

- typedef std::vector< MCdvsDescriptor > MCdvsDescriptorList

Enumerations

- enum OPERATION { UNKNOWN, EXTRACT, MATCH, RETRIEVE }
- enum LogFormat { FORMAT_NONE = 0, FORMAT_CSV = 1, FORMAT_TEXT = 2, FORMAT_HTML = 4 }

  Format of output logs.

6.1.1 Detailed Description

Namespace used to encapsulate all MPEG-7 CDVA declarations.

6.1.2 Typedef Documentation

6.1.2.1 typedef std::vector<MCdvsDescriptor> mpeg7cdva::MCdvsDescriptorList

6.1.3 Enumeration Type Documentation

6.1.3.1 enum mpeg7cdva::LogFormat

Format of output logs.

Enumerator

FORMAT_NONE do not output any data

FORMAT_CSV output data in CSV format

FORMAT_TEXT output data as free text

FORMAT_HTML output data in HTML format

6.1.3.2 enum mpeg7cdva::OPERATION

Enumerator

UNKNOWN

EXTRACT

MATCH

RETRIEVE

Chapter 7

Data Structure Documentation

7.1 mpeg7cdva::Buffer Class Reference

A container class for a byte array, intended to replace all malloc() and new() instructions in the main code.

`#include <Buffer.h>`

Public Member Functions

- Buffer ()
- virtual ~Buffer ()
- Buffer (size_t size)

  create a buffer of the given size
- Buffer (unsigned char *data, size_t size)

  copy the given array into this Buffer
- Buffer (const Buffer &)

  copy the given Buffer into this Buffer
- Buffer & operator= (const Buffer &)

  assign a Buffer to another
- void swap (Buffer &x)

  swap the content of two Buffer(s)
- void fill (unsigned char value=0)

  fill a Buffer with the given value
- size_t size () const

  return the current size of the Buffer
- bool resize (size_t newsize)

  change buffer size; content is lost if newsize if less than the current size
- bool empty () const

  return true if the Buffer is empty
- void clear ()

  clear the Buffer
- bool assign (const unsigned char *data, size_t size)

  assign the given data to Buffer
- bool equals (Buffer &buffer)

  compare if two Buffer(s) are equal (i.e. if they have the same size and contain the same data)

- unsigned char * data ()

  access to Buffer's data as unsigned char (writable)
- const unsigned char * data () const

  access to Buffer's data as unsigned char (read only)
- char * sdata ()

  access to Buffer's data as signed char (writable)
- const char * sdata () const

  access to Buffer's data as signed char (read only)
- void read (const char *fname)

  read Buffer from a file
- void write (const char *fname) const

  write Buffer to file
- int compare (const Buffer &other) const

  Compare this buffer with another; return the number of different bytes.
- bool operator== (const Buffer &other) const

  compare if two Buffer(s) are equal (i.e. if they have the same size and contain the same data)

7.1.1 Detailed Description

A container class for a byte array, intended to replace all malloc() and new() instructions in the main code.

This class properly deallocates memory when an exception is thrown.

Author

Massimo Balestri

Date

2013

7.1.2 Constructor & Destructor Documentation

7.1.2.1 mpeg7cdva::Buffer::Buffer ( )

7.1.2.2 virtual mpeg7cdva::Buffer::~Buffer ( ) [virtual]

7.1.2.3 mpeg7cdva::Buffer::Buffer (size_t *size*)

create a buffer of the given size

7.1.2.4 mpeg7cdva::Buffer::Buffer (unsigned char * *data*, size_t *size*)

copy the given array into this Buffer

7.1.2.5 mpeg7cdva::Buffer::Buffer (const Buffer &)

copy the given Buffer into this Buffer

7.1.3 Member Function Documentation

7.1.3.1 bool mpeg7cdva::Buffer::assign (const unsigned char * *data*, size_t *size*)

assign the given data to Buffer

7.1.3.2 void mpeg7cdva::Buffer::clear ( )

clear the Buffer

7.1.3.3 int mpeg7cdva::Buffer::compare (const Buffer & *other*) const

Compare this buffer with another; return the number of different bytes.

Parameters

other	the other Buffer

Returns

the number of differences; zero if no difference is found.

7.1.3.4 unsigned char* mpeg7cdva::Buffer::data ( )

access to Buffer's data as unsigned char (writable)

7.1.3.5 const unsigned char* mpeg7cdva::Buffer::data ( ) const

access to Buffer's data as unsigned char (read only)

7.1.3.6 bool mpeg7cdva::Buffer::empty ( ) const

return true if the Buffer is empty

7.1.3.7 bool mpeg7cdva::Buffer::equals (Buffer & *buffer*)

compare if two Buffer(s) are equal (i.e. if they have the same size and contain the same data)

7.1.3.8 void mpeg7cdva::Buffer::fill (unsigned char *value* = 0)

fill a Buffer with the given value

7.1.3.9 Buffer& mpeg7cdva::Buffer::operator= (const Buffer &)

assign a Buffer to another

7.1.3.10 bool mpeg7cdva::Buffer::operator== (const Buffer & *other*) const

compare if two Buffer(s) are equal (i.e. if they have the same size and contain the same data)

7.1.3.11 void mpeg7cdva::Buffer::read (const char * *fname*)

read Buffer from a file

7.1.3.12 bool mpeg7cdva::Buffer::resize (size_t *newsize*)

change buffer size; content is lost if newsize if less than the current size

7.1.3.13 char* mpeg7cdva::Buffer::sdata ( )

access to Buffer's data as signed char (writable)

7.1.3.14 const char* mpeg7cdva::Buffer::sdata ( ) const

access to Buffer's data as signed char (read only)

7.1.3.15 size_t mpeg7cdva::Buffer::size ( ) const

return the current size of the Buffer

7.1.3.16 void mpeg7cdva::Buffer::swap (Buffer & *x*)

swap the content of two Buffer(s)

7.1.3.17 void mpeg7cdva::Buffer::write (const char * *fname*) const

write Buffer to file

The documentation for this class was generated from the following file:

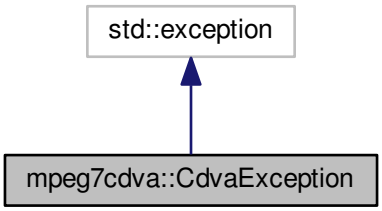
- Buffer.h

7.2 mpeg7cdva::CdvaException Class Reference

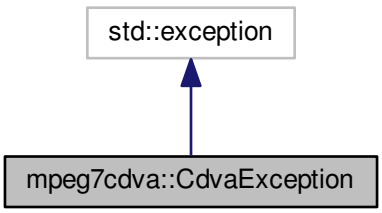
Class defining a specific exception for CDVA.

`#include <CdvaException.h>`

Inheritance diagram for mpeg7cdva::CdvaException:



Collaboration diagram for mpeg7cdva::CdvaException:



Public Member Functions

- CdvaException (std::string str)
  Create a new CDVA exception.
- virtual ~CdvaException () throw ()
- const char * what () const throw ()
  Get the exception message.

7.2.1 Detailed Description

Class defining a specific exception for CDVA.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 mpeg7cdva::CdvaException::CdvaException (std::string *str*) [inline]

Create a new CDVA exception.

Parameters

str	the exception message string.

7.2.2.2 virtual mpeg7cdva::CdvaException::~CdvaException ( ) throw) [inline], [virtual]

7.2.3 Member Function Documentation

7.2.3.1 const char* mpeg7cdva::CdvaException::what ( ) const throw) [inline]

Get the exception message.

The documentation for this class was generated from the following file:

- CdvaException.h

7.3 mpeg7cdva::CdvaImpl Class Reference

A CDVA implementation based on multiple CDVS descriptors.

`#include <CdvaImpl.h>`

Public Member Functions

- CdvaImpl ()
- virtual ~CdvaImpl ()
- virtual void init (OPERATION op, bool verbose, size_t n_videos, int querybitrate, int refbitrate=0)

  initialization method - called once before processing videos.
- void extract (const std::string &descrname, const std::string &videopathname, int bitrate, ExtractData &outdata) const

  Video processing method - called once for each video in the list.
- virtual double match (MatchData &matchResults, const std::string &qdescrname, const std::string &rdescrname, int qbitrate, int rbitrate)

  Video matching method - called once for each pair of videos in the list.
- virtual void makeindex (const std::string &cdva_descriptor, const std::string &relativepathname)

  Video indexing method - builds a DB of reference video descriptors.
- virtual void retrieve (std::vector< MatchData > &retrievalResults, const std::string &qdescrname, int qbitrate)

  Video retrieval method - returns a list of reference videos matching the query video.
- virtual void close ()

  de-initialization method - called once at the end of processing.

Static Public Member Functions

- static bool checkBitrate (int bitrate)

  check if the given bitrate is one of the standard values defined in the CDVA evaluation framework.
- static const char * getDescriptorExt (int bitrate)

  get the file extension corresponding to the given bitrate.

Protected Member Functions

- virtual void parse (const std::string &descFile, MCdvsDescriptorList &descList)
- virtual double match (MatchData &matchResults, const MCdvsDescriptorList &qDescList, const MCdvsDescriptorList &rDescList)

Static Protected Member Functions

- static bool byDescendingScore (const MatchData &m1, const MatchData &m2)
- static std::string getExt (const std::string &imageName)

Protected Attributes

- mpeg7cdvs::CdvsConfiguration * cdvsconfig
- mpeg7cdvs::CdvsClient * cdvsclient
- mpeg7cdvs::CdvsServer * cdvsserver
- bool verboseMode

  verbose mode indicator
- OPERATION current_op

  the current operation
- int skip_before

  number of video frames to skip before decoding one
- int skip_after

  number of video frames to skip after decoding one
- double drop_frame_th

  drop frame threshold
- size_t max_retrieved

  maximum number of retrieved images

7.3.1 Detailed Description

A CDVA implementation based on multiple CDVS descriptors.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 mpeg7cdva::CdvaImpl::CdvaImpl ( )

7.3.2.2 virtual mpeg7cdva::CdvaImpl::~CdvaImpl ( ) [virtual]

7.3.3 Member Function Documentation

7.3.3.1 static bool mpeg7cdva::CdvaImpl::byDescendingScore (const MatchData & *m1*, const MatchData & *m2*) [static], [protected]

7.3.3.2 static bool mpeg7cdva::CdvaImpl::checkBitrate (int *bitrate*) [static]

check if the given bitrate is one of the standard values defined in the CDVA evaluation framework.

Parameters

bitrate	the bitrate in Kilo-byte per second (KB/s)

Returns

true if valid

7.3.3.3 virtual void mpeg7cdva::CdvaImpl::close ( ) [virtual]

de-initialization method - called once at the end of processing.

7.3.3.4 void mpeg7cdva::CdvaImpl::extract (const std::string & *descrname,* const std::string & *videopathname,* int *bitrate,* ExtractData & *outdata*) const

Video processing method - called once for each video in the list.

Parameters

descrname	output descriptor pathname
videopathname	input video stream pathname
bitrate	encoding bitrate (one of 0,16,64,256)
outdata	the container for output data

7.3.3.5 static const char* mpeg7cdva::CdvaImpl::getDescriptorExt (int *bitrate*) [static]

get the file extension corresponding to the given bitrate.

Parameters

bitrate	the bitrate in Kilo-byte per second (KB/s)

Returns

the file extension

7.3.3.6 static std::string mpeg7cdva::CdvaImpl::getExt (const std::string & *imageName*) [static], [protected]

7.3.3.7 virtual void mpeg7cdva::CdvaImpl::init (OPERATION *op,* bool *verbose,* size_t *n_videos,* int *querybitrate,* int *refbitrate* = 0) [virtual]

initialization method - called once before processing videos.

Parameters

op	one of EXTRACT, MATCH, RETRIEVE
verbose	when set, more information is provided
n_videos	the number of videos to be processed
querybitrate	the query encoding bitrate (one of 0,16,64,256)
refbitrate	the reference encoding bitrate (one of 0,16,64,256)

7.3.3.8 virtual void mpeg7cdva::CdvaImpl::makeindex (const std::string & *cdva_descriptor,* const std::string & *relativepathname*) [virtual]

Video indexing method - builds a DB of reference video descriptors.

Parameters

cdva_descriptor	the descriptor to add to the DB
relativepathname	the relative pathname of the video file to be used as unique identifier

7.3.3.9 virtual double mpeg7cdva::CdvaImpl::match (MatchData & *matchResults,* const MCdvsDescriptorList & *qDescList,* const MCdvsDescriptorList & *rDescList*) [protected], [virtual]

7.3.3.10 virtual double mpeg7cdva::CdvaImpl::match (MatchData & *matchResults,* const std::string & *qdescrname,* const std::string & *rdescrname,* int *qbitrate,* int *rbitrate*) [virtual]

Video matching method - called once for each pair of videos in the list.

Parameters

matchResults	container for the results of matching
qdescrname	input query descriptor name
rdescrname	input reference descriptor name
qbitrate	query bitrate (one of 16,64,256)
rbitrate	reference bitrate (one of 16,64,256)

Returns

the matching score (normalized in the [0..1] range)

7.3.3.11 virtual void mpeg7cdva::CdvaImpl::parse (const std::string & *descFile,* MCdvsDescriptorList & *descList*) [protected], [virtual]

7.3.3.12 virtual void mpeg7cdva::CdvaImpl::retrieve (std::vector< MatchData > & *retrievalResults,* const std::string & *qdescrname,* int *qbitrate*) [virtual]

Video retrieval method - returns a list of reference videos matching the query video.

Parameters

retrievalResults	the output vector containing an ordered list of matching reference videos
qdescrname	the video query descriptor
qbitrate	query bitrate (one of 16,64,256)

7.3.4 Field Documentation

7.3.4.1 mpeg7cdvs::CdvsClient* mpeg7cdva::CdvaImpl::cdvsclient [protected]

7.3.4.2 mpeg7cdvs::CdvsConfiguration* mpeg7cdva::CdvaImpl::cdvsconfig [protected]

7.3.4.3 mpeg7cdvs::CdvsServer* mpeg7cdva::CdvaImpl::cdvsserver [protected]

7.3.4.4 OPERATION mpeg7cdva::CdvaImpl::current_op [protected]

the current operation

7.3.4.5 double mpeg7cdva::CdvaImpl::drop_frame_th [protected]

drop frame threshold

7.3.4.6 size_t mpeg7cdva::CdvaImpl::max_retrieved [protected]

maximum number of retrieved images

7.3.4.7 int mpeg7cdva::CdvaImpl::skip_after [protected]

number of video frames to skip after decoding one

7.3.4.8 int mpeg7cdva::CdvaImpl::skip_before [protected]

number of video frames to skip before decoding one

7.3.4.9 bool mpeg7cdva::CdvaImpl::verboseMode [protected]

verbose mode indicator

The documentation for this class was generated from the following file:

- CdvaImpl.h

7.4 mpeg7cdva::ExtractData Class Reference

A class containing the results of an extraction operation.

`#include <cdva.h>`

Public Member Functions

- ExtractData ()
- void setVideoDuration (double time)

  set the video duration in seconds.
- void setNumFrames (double nframes)

  set the number of frames of the video clip.
- void setDescriptorLength (double length)

  set the actual descriptor length (in bytes).

Data Fields

- double descriptorlength
- double clip_duration
- double numframes
- int header_bit_count
- int local_bit_count
- int global_bit_count
- int coordinate_bit_count
- int n_keyframes

7.4.1 Detailed Description

A class containing the results of an extraction operation.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 mpeg7cdva::ExtractData::ExtractData ( ) [inline]

7.4.3 Member Function Documentation

7.4.3.1 void mpeg7cdva::ExtractData::setDescriptorLength (double *length*) [inline]

set the actual descriptor length (in bytes).

Parameters

length	the size in bytes of the encoded descriptor.

7.4.3.2 void mpeg7cdva::ExtractData::setNumFrames (double *nframes*) [inline]

set the number of frames of the video clip.

Parameters

nframes	number of frames of the video clip.

7.4.3.3 void mpeg7cdva::ExtractData::setVideoDuration (double *time*) [inline]

set the video duration in seconds.

Parameters

time	the time in seconds

7.4.4 Field Documentation

7.4.4.1 double mpeg7cdva::ExtractData::clip_duration

7.4.4.2 int mpeg7cdva::ExtractData::coordinate_bit_count

7.4.4.3 double mpeg7cdva::ExtractData::descriptorlength

7.4.4.4 int mpeg7cdva::ExtractData::global_bit_count

7.4.4.5 int mpeg7cdva::ExtractData::header_bit_count

7.4.4.6 int mpeg7cdva::ExtractData::local_bit_count

7.4.4.7 int mpeg7cdva::ExtractData::n_keyframes

7.4.4.8 double mpeg7cdva::ExtractData::numframes

The documentation for this class was generated from the following file:

- cdva.h

7.5 mpeg7cdva::FileManager Class Reference

Helper class to manage lists of file names.

`#include <FileManager.h>`

Public Member Functions

- FileManager (const char *annotationpathname, int level=0)
  Read the list of images from the given annotation file.
- virtual ~FileManager ()
- std::string getDatasetPath () const
  Get the dataset base directory.
- std::string getDatasetName () const
  Get the dataset name.
- std::string getDatasetPathName () const
  Get the dataset full pathname.
- void setWorkspaceDir (const char *workdir)
  Set the workspace directory.
- std::string getWorkspaceDir () const
  Get the workspace directory.
- size_t getDatasetSize () const
  Get the dataset size.
- std::string replaceExt (const std::string &imageName, const char *ext) const
  Convert a pathname into a pathname with the given extension.
- std::string getQueryName (size_t i, bool absolutePathname=true) const
  Get the first image name found at the i-th position in the annotation file.
- std::string getReferenceName (size_t i, bool absolutePathname=true) const
  Get the second image name found at the i-th position in the annotation file.
- int countNames (size_t i) const
  Count how many pathnames are contained in the i-th line.

Static Public Member Functions

- static std::string replacePath (const std::string &imageName, const char *newpath)
  Convert a pathname into a pathname with the given new path.

7.5.1 Detailed Description

Helper class to manage lists of file names.

Author

Massimo Balestri

Date

2012

7.5.2 Constructor & Destructor Documentation

7.5.2.1 mpeg7cdva::FileManager::FileManager (const char * *annotationpathname*, int *level* = 0)

Read the list of images from the given annotation file.

Parameters

annotationpathname	the pathname of the annotation text file containing the list of images.
level	the recursion level (to avoid infinite loops); must be zero when called the first time.

7.5.2.2 virtual mpeg7cdva::FileManager::~FileManager ( ) [virtual]

7.5.3 Member Function Documentation

7.5.3.1 int mpeg7cdva::FileManager::countNames (size_t *i*) const

Count how many pathnames are contained in the i-th line.

Parameters

i	the index of the image in the annotation file.

Returns

the number of pathnames found.

7.5.3.2 std::string mpeg7cdva::FileManager::getDatasetName ( ) const

Get the dataset name.

Returns

the dataset name

7.5.3.3 std::string mpeg7cdva::FileManager::getDatasetPath ( ) const

Get the dataset base directory.

Returns

the dataset path

7.5.3.4 std::string mpeg7cdva::FileManager::getDatasetPathName ( ) const

Get the dataset full pathname.

Returns

the dataset pathname

7.5.3.5 size_t mpeg7cdva::FileManager::getDatasetSize ( ) const

Get the dataset size.

Returns

the number of lines read from the filename.

7.5.3.6 std::string mpeg7cdva::FileManager::getQueryName (size_t *i,* bool *absolutePathname* = `true`) const

Get the first image name found at the i-th position in the annotation file.

The image name is provided as an absolute pathname.

Parameters

i	the index of the image in the annotation file.
absolutePathname	if true, the absolute pathname of the file is returned; otherwise, the relative pathname is returned.

Returns

the relative or absolute pathname of the image.

7.5.3.7 std::string mpeg7cdva::FileManager::getReferenceName (size_t *i,* bool *absolutePathname* = `true`) const

Get the second image name found at the i-th position in the annotation file.

The image name is provided as an absolute pathname.

Parameters

i	the index of the image in the annotation file.
absolutePathname	if true, the absolute pathname of the file is returned; otherwise, the relative pathname is returned.

Returns

the relative or absolute pathname of the image.

7.5.3.8 std::string mpeg7cdva::FileManager::getWorkspaceDir ( ) const

Get the workspace directory.

This is the directory where output files will be stored.

Returns

the workspace directory

7.5.3.9 std::string mpeg7cdva::FileManager::replaceExt (const std::string & *imageName,* const char * *ext*) const

Convert a pathname into a pathname with the given extension.

Parameters

imageName	the original image/video name;
ext	new extension;

Returns

the modified pathname.

7.5.3.10 static std::string mpeg7cdva::FileManager::replacePath (const std::string & *imageName,* const char * *newpath*) [static]

Convert a pathname into a pathname with the given new path.

Parameters

imageName	the original image name;
newpath	the new path;

Returns

the modified pathname.

7.5.3.11 void mpeg7cdva::FileManager::setWorkspaceDir (const char * *workdir*)

Set the workspace directory.

This is the directory where output files will be stored.

Parameters

workdir	the workspace directory

The documentation for this class was generated from the following file:

- FileManager.h

7.6 mpeg7cdva::LogManager Class Reference

Helper class to produce log files in various formats (csv, text, XML, etc.)

`#include <LogManager.h>`

Public Member Functions

- LogManager ()

  constructor
- virtual ~LogManager ()

  destructor
- void init (int formats, const std::string &datasetpath, const std::string &datasetname, size_t n_videos, int mode, int refmode=0)

  initialization method used by the Evaluation Framework; do not change.
- void printExtractHeader ()

  method used by the Evaluation Framework to produce log files; do not change.
- void printExtractData (int index, const std::string &videoname, const ExtractData &data)

  method used by the Evaluation Framework to produce log files; do not change.
- void printMatchHeader ()

  method used by the Evaluation Framework to produce log files; do not change.
- void printMatchData (int index, const std::string &queryvideoname, const std::string &refvideoname, const MatchData &matchData)

  method used by the Evaluation Framework to produce log files; do not change.
- void printRetrievalHeader ()

  method used by the Evaluation Framework to produce log files; do not change.
- void printRetrievalData (int index, const std::string &queryvideoname, const std::vector< MatchData > &retrievalResults)

  method used by the Evaluation Framework to produce log files; do not change.
- void close ()

7.6.1 Detailed Description

Helper class to produce log files in various formats (csv, text, XML, etc.)

Author

Massimo Balestri

Date

2015

7.6.2 Constructor & Destructor Documentation

7.6.2.1 mpeg7cdva::LogManager::LogManager ( )

constructor

7.6.2.2 virtual mpeg7cdva::LogManager::~LogManager ( ) [virtual]

destructor

7.6.3 Member Function Documentation

7.6.3.1 void mpeg7cdva::LogManager::close ( )

7.6.3.2 void mpeg7cdva::LogManager::init (int *formats,* const std::string & *datasetpath,* const std::string & *datasetname,* size_t *n_videos,* int *mode,* int *refmode* = 0)

initialization method used by the Evaluation Framework; do not change.

7.6.3.3 void mpeg7cdva::LogManager::printExtractData (int *index,* const std::string & *videoname,* const **ExtractData** & *data*)

method used by the Evaluation Framework to produce log files; do not change.

7.6.3.4 void mpeg7cdva::LogManager::printExtractHeader ( )

method used by the Evaluation Framework to produce log files; do not change.

7.6.3.5 void mpeg7cdva::LogManager::printMatchData (int *index,* const std::string & *queryvideoname,* const std::string & *refvideoname,* const **MatchData** & *matchData*)

method used by the Evaluation Framework to produce log files; do not change.

7.6.3.6 void mpeg7cdva::LogManager::printMatchHeader ( )

method used by the Evaluation Framework to produce log files; do not change.

7.6.3.7 void mpeg7cdva::LogManager::printRetrievalData (int *index,* const std::string & *queryvideoname,* const std::vector< **MatchData** > & *retrievalResults*)

method used by the Evaluation Framework to produce log files; do not change.

7.6.3.8 void mpeg7cdva::LogManager::printRetrievalHeader ( )

method used by the Evaluation Framework to produce log files; do not change.

The documentation for this class was generated from the following file:

- LogManager.h

7.7 mpeg7cdva::MatchData Class Reference

A class containing the results of a matching or retrieval operation.

`#include <cdva.h>`

Public Member Functions

- MatchData ()
- virtual ~MatchData ()
- void setMatchingScore (double myscore)

  set the score of matching the query image with the reference image.
- void setMatchingTime (double time_s)

  set the time of each frame matching (only the first and the last will be saved).
- void setReferenceID (const std::string reference)

  set the string that identifies the matching reference video clip.
- double getScore () const

  Get the matching score.
- double getFirstMatchingTime () const

  get the time in seconds indicating the fist matching frame of the query clip.
- double getLastMatchingTime () const

  get the time in seconds indicating the last matching frame of the query clip.
- std::string getReferenceId () const

  get the string that identifies the matching reference video clip.

7.7.1 Detailed Description

A class containing the results of a matching or retrieval operation.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 mpeg7cdva::MatchData::MatchData ( ) [inline]

7.7.2.2 virtual mpeg7cdva::MatchData::~MatchData ( ) [inline], [virtual]

7.7.3 Member Function Documentation

7.7.3.1 double mpeg7cdva::MatchData::getFirstMatchingTime ( ) const [inline]

get the time in seconds indicating the fist matching frame of the query clip.

Returns

the time in seconds from the start of the video clip

7.7.3.2 double mpeg7cdva::MatchData::getLastMatchingTime ( ) const [inline]

get the time in seconds indicating the last matching frame of the query clip.

Returns

the time in seconds from the start of the video clip

7.7.3.3 std::string mpeg7cdva::MatchData::getReferenceId ( ) const [inline]

get the string that identifies the matching reference video clip.

Returns

the video clip relative pathname

7.7.3.4 double mpeg7cdva::MatchData::getScore ( ) const [inline]

Get the matching score.

Returns

the score

7.7.3.5 void mpeg7cdva::MatchData::setMatchingScore (double *myscore*) [inline]

set the score of matching the query image with the reference image.

Parameters

myscore	the overall matching score

7.7.3.6 void mpeg7cdva::MatchData::setMatchingTime (double *time_s*) [inline]

set the time of each frame matching (only the first and the last will be saved).

Parameters

time_s	the time in seconds from the start of the query video sequence

7.7.3.7 void mpeg7cdva::MatchData::setReferenceID (const std::string *reference*) [inline]

set the string that identifies the matching reference video clip.

Parameters

reference	the identifier (usually the relative pathname) of the matching reference video.

The documentation for this class was generated from the following file:

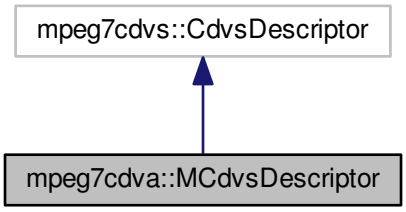
- cdva.h

7.8 mpeg7cdva::MCdvsDescriptor Class Reference

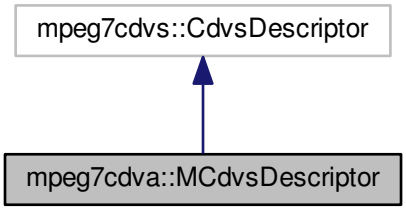
Extends the CdvsDescriptor with time and size information.

`#include <CdvaImpl.h>`

Inheritance diagram for mpeg7cdva::MCdvsDescriptor:



Collaboration diagram for mpeg7cdva::MCdvsDescriptor:



Public Member Functions

- void setStartTimeMs (unsigned long position_ms)
  Set the start time in milliseconds.
- void setEndTimeMs (unsigned long position_ms)
  Set the end time in milliseconds.
- unsigned long getStartTimeMs () const
  Get the start time in milliseconds.
- unsigned long getEndTimeMs () const
  Get the end time in milliseconds.
- size_t write (std::ofstream &fout) const
  write (appending) this descriptor to a file
- size_t read (std::ifstream &fin)
  read (from the current position) this descriptor from a file

7.8.1 Detailed Description

Extends the CdvsDescriptor with time and size information.

7.8.2 Member Function Documentation

7.8.2.1 unsigned long mpeg7cdva::MCdvsDescriptor::getEndTimeMs ( ) const

Get the end time in milliseconds.

7.8.2.2 unsigned long mpeg7cdva::MCdvsDescriptor::getStartTimeMs ( ) const

Get the start time in milliseconds.

7.8.2.3 size_t mpeg7cdva::MCdvsDescriptor::read (std::ifstream & *fin*)

read (from the current position) this descriptor from a file

7.8.2.4 void mpeg7cdva::MCdvsDescriptor::setEndTimeMs (unsigned long *position_ms*)

Set the end time in milliseconds.

7.8.2.5 void mpeg7cdva::MCdvsDescriptor::setStartTimeMs (unsigned long *position_ms*)

Set the start time in milliseconds.

7.8.2.6 size_t mpeg7cdva::MCdvsDescriptor::write (std::ofstream & *fout*) const

write (appending) this descriptor to a file

The documentation for this class was generated from the following file:

- CdvaImpl.h

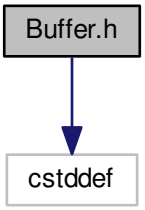
Chapter 8

File Documentation

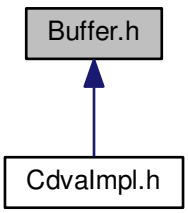
8.1 Buffer.h File Reference

```
#include <cstddef>
```

Include dependency graph for Buffer.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- class mpeg7cdva::Buffer

  A container class for a byte array, intended to replace all malloc() and new() instructions in the main code.

Namespaces

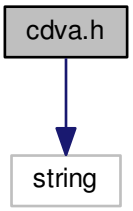
- mpeg7cdva

  Namespace used to encapsulate all MPEG-7 CDVA declarations.

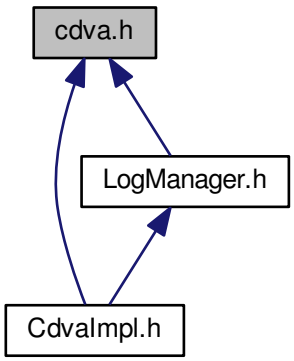
8.2 cdva.h File Reference

```
#include <string>
```

Include dependency graph for cdva.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- class mpeg7cdva::ExtractData

  A class containing the results of an extraction operation.
- class mpeg7cdva::MatchData

  A class containing the results of a matching or retrieval operation.

Namespaces

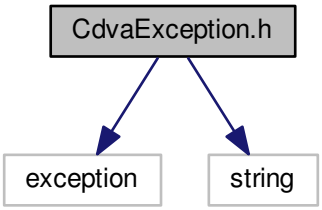
- mpeg7cdva

  Namespace used to encapsulate all MPEG-7 CDVA declarations.

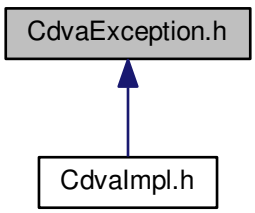
8.3 CdvaException.h File Reference

```
#include <exception>
#include <string>
```

Include dependency graph for CdvaException.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- class mpeg7cdva::CdvaException

  Class defining a specific exception for CDVA.

Namespaces

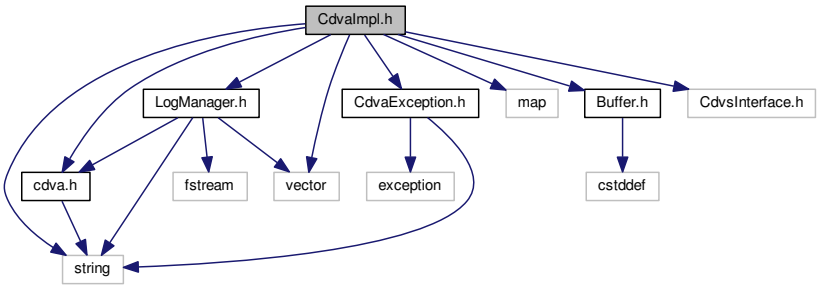
- mpeg7cdva

  Namespace used to encapsulate all MPEG-7 CDVA declarations.

8.4 CdvaImpl.h File Reference

```
#include <string>
#include <vector>
#include <map>
#include "cdva.h"
#include "CdvaException.h"
#include "LogManager.h"
#include "Buffer.h"
#include "CdvsInterface.h"
```

Include dependency graph for CdvaImpl.h:



Data Structures

- class mpeg7cdva::MCdvsDescriptor

  Extends the CdvsDescriptor with time and size information.
- class mpeg7cdva::CdvaImpl

  A CDVA implementation based on multiple CDVS descriptors.

Namespaces

- mpeg7cdva

  Namespace used to encapsulate all MPEG-7 CDVA declarations.

Typedefs

- typedef std::vector< MCdvsDescriptor > mpeg7cdva::MCdvsDescriptorList

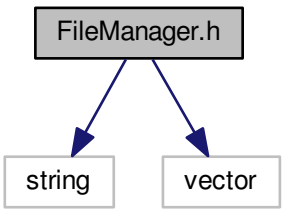
Enumerations

- enum mpeg7cdva::OPERATION { mpeg7cdva::UNKNOWN, mpeg7cdva::EXTRACT, mpeg7cdva::MATCH, mpeg7cdva::RETRIEVE }

8.5 FileManager.h File Reference

```
#include <string>
#include <vector>
```

Include dependency graph for FileManager.h:



Data Structures

- class mpeg7cdva::FileManager

  Helper class to manage lists of file names.

Namespaces

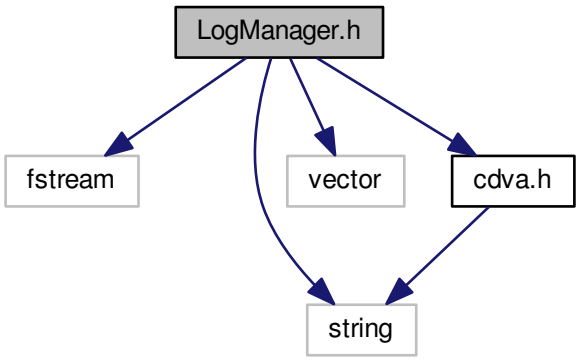
- mpeg7cdva

  Namespace used to encapsulate all MPEG-7 CDVA declarations.

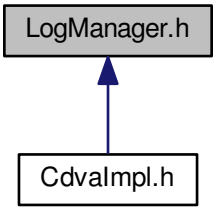
8.6 LogManager.h File Reference

```
#include <fstream>
#include <string>
#include <vector>
#include "cdva.h"
```

Include dependency graph for LogManager.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- class mpeg7cdva::LogManager

  Helper class to produce log files in various formats (csv, text, XML, etc.)

Namespaces

- mpeg7cdva

  Namespace used to encapsulate all MPEG-7 CDVA declarations.

Enumerations

- enum mpeg7cdva::LogFormat { mpeg7cdva::FORMAT_NONE = 0, mpeg7cdva::FORMAT_CSV = 1, mpeg7cdva::FORMAT_TEXT = 2, mpeg7cdva::FORMAT_HTML = 4 }

  Format of output logs.