HPC-datastore-cpp

Generated by Doxygen 1.9.1

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

Class Documentation

2.1 ds::Connection Class Reference

Representation of connection to dataset.

`#include <hpc_ds_api.hpp>`

Public Member Functions

- Connection (std::string ip, int port, std::string uuid)

  Construct a new Connection object.
- ImageView get_view (int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version) const

  Get ImageView of specified image.
- dataset_props_ptr get_properties () const

  Get dataset properties.
- template<cnpts::Scalar T>
  i3d::Image3d< T > read_block (i3d::Vector3d< int > coord, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_props_ptr props=nullptr) const

  Read one block from server to image.
- template<cnpts::Scalar T>
  void read_block (i3d::Vector3d< int > coord, i3d::Image3d< T > &dest, i3d::Vector3d< int > dest_offset, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_props_ptr props=nullptr) const

  Read one block from server to image.
- template<cnpts::Scalar T>
  std::vector< i3d::Image3d< T > > read_blocks (const std::vector< i3d::Vector3d< int >> &coords, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_props_ptr props=nullptr) const

  Read blocks from server and return them.
- template<cnpts::Scalar T>
  void read_blocks (const std::vector< i3d::Vector3d< int >> &coords, i3d::Image3d< T > &dest, const std::vector< i3d::Vector3d< int >> &dest_offsets, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_props_ptr props=nullptr) const

  Read blocks from server and saves them into prealocated image.

- template<cnpts::Scalar T>
  i3d::Image3d< T > read_region (i3d::Vector3d< int > start_point, i3d::Vector3d< int > end_point, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_props_ptr props=nullptr) const

  Read region of interest from the server.
- template<cnpts::Scalar T>
  void read_region (i3d::Vector3d< int > start_point, i3d::Vector3d< int > end_point, i3d::Image3d< T > &dest, i3d::Vector3d< int > offset, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_props_ptr props=nullptr) const

  Read region of interest from the server.
- template<cnpts::Scalar T>
  i3d::Image3d< T > read_image (int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_props_ptr props=nullptr) const

  Read full image.
- template<cnpts::Scalar T>
  void write_block (const i3d::Image3d< T > &src, i3d::Vector3d< int > coord, i3d::Vector3d< int > src_offset, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_↩
  props_ptr props=nullptr) const

  Write block to server.
- template<cnpts::Scalar T>
  void write_blocks (const i3d::Image3d< T > &src, const std::vector< i3d::Vector3d< int >> &coords, const std::vector< i3d::Vector3d< int >> &src_offsets, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_props_ptr props=nullptr) const

  Write blocks to server.
- template<cnpts::Scalar T>
  void write_image (const i3d::Image3d< T > &img, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, const std::string &version, dataset_props_ptr props=nullptr) const

  Write image to server.
- template<cnpts::Scalar T>
  void write_with_pyramids (const i3d::Image3d< T > &img, int channel, int timepoint, int angle, const std↩
  ::string &version, SamplingMode m, dataset_props_ptr props=nullptr) const

2.1.1 Detailed Description

Representation of connection to dataset.

Class representing connection to specific dataset on the server. It provides basic methods for read/write operations necessary to tranfser images (in the dataset) from/to server. This class does not cache or precollect any data, so the first HTTP request will be send only when corresponding function is called.

All of the methods accepts arguments that uniquely identifies requested image. At the backend, this class tranfsers commands into ImageView objects.

2.1.2 Constructor & Destructor Documentation

2.1.2.1 Connection()

```
ds::Connection::Connection (
            std::string ip,
            int port,
            std::string uuid )
```

Construct a new Connection object.

Parameters

ip	IP address of server (`http://` at the beginning is not necessary)
port	Port, where the server is listening for requests
uuid	Unique identifier of dataset

2.1.3 Member Function Documentation

2.1.3.1 get_properties()

```
dataset_props_ptr ds::Connection::get_properties ( ) const
```

Get dataset properties.

Returns

DatasetProperties

2.1.3.2 get_view()

```
ImageView ds::Connection::get_view (
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version ) const
```

Get ImageView of specified image.

Parameters

channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")

Returns

ImageView

2.1.3.3 read_block() [1/2]

```
template<cnpts::Scalar T>
void ds::Connection::read_block (
            i3d::Vector3d< int > coord,
            i3d::Image3d< T > & dest,
            i3d::Vector3d< int > dest_offset,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Read one block from server to image.

Reads one block of image located at <coord> and saves it to <dest> with offset <dest_offset>.

If in DEBUG, function will check wheter given coordinate corresponds to valid block as well as wheter the block fits into the image (taking offset into account).

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

coord	Block coordinate
dest	Image to write data to
dest_offset	Offset by which the corresponding write should be moved
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

2.1.3.4 read_block() [2/2]

```
template<cnpts::Scalar T>
i3d::Image3d< T > ds::Connection::read_block (
            i3d::Vector3d< int > coord,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Read one block from server to image.

Reads one block of image located at <coord> and saves it to <dest> with offset <dest_offset>.

If in DEBUG, function will check wheter given coordinate corresponds to valid block as well as wheter the block fits into the image (taking offset into account).

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

coord	Block coordinate
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

Returns

Image containing selected block

2.1.3.5 read_blocks() [1/2]

```
template<cnpts::Scalar T>
void ds::Connection::read_blocks (
            const std::vector< i3d::Vector3d< int >> & coords,
            i3d::Image3d< T > & dest,
            const std::vector< i3d::Vector3d< int >> & dest_offsets,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Read blocks from server and saves them into prealocated image.

Read blocks specified in <coords> and saves them into locations given in <offsets>.

If in DEBUG, the function checks if coordinates given in <coords> points to a valid blocks, as well as wheter the offsets specified for each block are within image boundaries.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

coords	Block coordinates
dest	Prealocated destination image
dest_offsets	Offsets at wich the corresponding blocks should be saved
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

2.1.3.6 read_blocks() [2/2]

```
template<cnpts::Scalar T>
std::vector< i3d::Image3d< T > > ds::Connection::read_blocks (
            const std::vector< i3d::Vector3d< int >> & coords,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Read blocks from server and return them.

Reads blocks specified in <coords> and returns them. Corresponding sizes are collected from server and calculated specificaly for each block.

This function is not optimized, meaning that for each coord in <coord>, one HTTP request will be sent out to the server. This can heavily slow down speed of the application as communication via network is not cheap. If you do not have specific needs,most of the time it will be faster to collect blocks into prealocated image (second overload of read_blocks), however it will eat more RAM.

If in DEBUG, the fucntion checks if coordinates given in <coords> points to a valid blocks.

As there is no (meaningfull) way for C++ to choose correct underlying type in runtime, make sure to specify correct template type.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

coords	Block coordinates
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located

Parameters

resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

Returns

Vector of fetched blocks (the order is the same as given in <coords>)

2.1.3.7 read_image()

```
template<cnpts::Scalar T>
i3d::Image3d< T > ds::Connection::read_image (
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Read full image.

Read full image from the server and return it. The information about dimensions are fetched from the server.

As there is no (meaningfull) way for C++ to choose correct underlying type in runtime, make sure to specify correct template type.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

Returns

i3d::Image3d<T> etched image

2.1.3.8 read_region() [1/2]

```
template<cnpts::Scalar T>
void ds::Connection::read_region (
            i3d::Vector3d< int > start_point,
            i3d::Vector3d< int > end_point,
            i3d::Image3d< T > & dest,
            i3d::Vector3d< int > offset,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Read region of interest from the server.

Read all neccessary blocks intersecting with chosen region from the server and insert region into preallocated image <dest> at <offset>.

It is neccessary, that start_point < end_point (elem-wise)..

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

start_point	smallest point of the region
end_point	largest point of the region
dest	destination image
offset	offset to destination image
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

2.1.3.9 read_region() [2/2]

```
template<cnpts::Scalar T>
i3d::Image3d< T > ds::Connection::read_region (
            i3d::Vector3d< int > start_point,
            i3d::Vector3d< int > end_point,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
```

```
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Read region of interest from the server.

Read all neccessary blocks intersecting with chosen region from the server. It is neccessary, that start_point < end_point (elem-wise)..

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

start_point	smallest point of the region
end_point	largest point of the region
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

Returns

Image containing selected block

2.1.3.10 write_block()

```
template<cnpts::Scalar T>
void ds::Connection::write_block (
            const i3d::Image3d< T > & src,
            i3d::Vector3d< int > coord,
            i3d::Vector3d< int > src_offset,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Write block to server.

Write block from source image to server. The information about dimensions are fetched from the server.

If in DEBUG, the function checks if coordinate given in <coord> points to a valid block, as well as wheter the offset specified for block is within image boundaries.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

src	Source image to collect block from
coord	Block coordinates
src_offset	Offset of given block in source image
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

2.1.3.11 write_blocks()

```
template<cnpts::Scalar T>
void ds::Connection::write_blocks (
            const i3d::Image3d< T > & src,
            const std::vector< i3d::Vector3d< int >> & coords,
            const std::vector< i3d::Vector3d< int >> & src_offsets,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Write blocks to server.

Write blocks from source image to server. The information about dimensions are fetched from the server.

If in DEBUG, the function checks if coordinates given in <coords> points to a valid block, as well as wheter the offsets specified for each block is within image boundaries.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

src	Source image to collect blocks from
coords	Vector of block coordinates
src_offsets	Offsets of corresponding blocks in source image
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located

Parameters

angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

2.1.3.12 write_image()

```
template<cnpts::Scalar T>
void ds::Connection::write_image (
            const i3d::Image3d< T > & img,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            const std::string & version,
            dataset_props_ptr props = nullptr ) const
```

Write image to server.

Write full image to server.

It is recommended to make sure that the dimension of the source image is the same as the dimension of image at server side.

Mostly, given smaller source image will emit error and fail to upload. Given larger source image will result in cropping.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

img	Source image
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

2.1.3.13 write_with_pyramids()

```
template<cnpts::Scalar T>
void ds::Connection::write_with_pyramids (
```

```
            const i3d::Image3d< T > & img,
            int channel,
            int timepoint,
            int angle,
            const std::string & version,
            SamplingMode m,
            dataset_props_ptr props = nullptr ) const

  @brief Write full image and generate pyramids

  Creates (several if needed) HTTP requests and sends whole image to
```

datastore. Input image is considered to be full-resolution (that is: {1, 1, 1}). All other resolutions will be generated with selected <ResamplingMode> and uploaded to server as well.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

img	Input image in original resolution
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
m	Sampling mode used for image resampling
props	[Optional] cached dataset properties

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

- /home/somik/CBIA/hpc-datastore-cpp/src/hpc_ds_api.hpp

2.2 ds::DatasetProperties Class Reference

Class representing dataset properties.

`#include <hpc_ds_structs.hpp>`

Public Member Functions

- i3d::Vector3d< int > **get_block_dimensions** (i3d::Vector3d< int > resolution) const
- i3d::Vector3d< int > **get_block_size** (i3d::Vector3d< int > coord, i3d::Vector3d< int > resolution) const
- i3d::Vector3d< int > **get_block_count** (i3d::Vector3d< int > resolution) const
- i3d::Vector3d< int > **get_img_dimensions** (i3d::Vector3d< int > resolution) const
- std::vector< i3d::Vector3d< int > > **get_all_resolutions** () const
- std::string **str** () const

Public Attributes

- std::string **uuid**
- std::string **voxel_type**
- i3d::Vector3d< int > **dimensions**
- int **channels**
- int **angles**
- std::optional< std::string > **transformations**
- std::string **voxel_unit**
- std::optional< i3d::Vector3d< double > > **voxel_resolution**
- std::optional< ResolutionUnit > **timepoint_resolution**
- std::optional< ResolutionUnit > **channel_resolution**
- std::optional< ResolutionUnit > **angle_resolution**
- std::string **compression**
- std::vector< std::map< std::string, i3d::Vector3d< int > > > **resolution_levels**
- std::vector< int > **versions**
- std::string **label**
- std::optional< std::string > **view_registrations**
- std::set< int > **timepoint_ids**

Friends

- std::ostream & **operator**<< (std::ostream &stream, const DatasetProperties &ds)

2.2.1 Detailed Description

Class representing dataset properties.

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

- /home/somik/CBIA/hpc-datastore-cpp/src/hpc_ds_structs.hpp

2.3 ds::ImageView Class Reference

Representation of connection to specific image.

`#include <hpc_ds_api.hpp>`

Public Member Functions

- ImageView (std::string ip, int port, std::string uuid, int channel, int timepoint, int angle, i3d::Vector3d< int > resolution, std::string version)

  Construct a new Image View object.
- dataset_props_ptr get_properties () const

  Get dataset properties.
- template<cnpts::Scalar T>
  i3d::Image3d< T > read_block (i3d::Vector3d< int > coord, dataset_props_ptr props=nullptr) const

  Read one block from server.
- template<cnpts::Scalar T>
  void read_block (i3d::Vector3d< int > coord, i3d::Image3d< T > &dest, i3d::Vector3d< int > dest_offset={0, 0, 0}, dataset_props_ptr props=nullptr) const

  Read one block from server to image.
- template<cnpts::Scalar T>
  std::vector< i3d::Image3d< T > > read_blocks (const std::vector< i3d::Vector3d< int >> &coords, dataset_props_ptr props=nullptr) const

  Read blocks from server and return them.
- template<cnpts::Scalar T>
  void read_blocks (const std::vector< i3d::Vector3d< int >> &coords, i3d::Image3d< T > &dest, const std::vector< i3d::Vector3d< int >> &offsets, dataset_props_ptr props=nullptr) const

  Read blocks from server and saves them into prealocated image.
- template<cnpts::Scalar T>
  i3d::Image3d< T > read_region (i3d::Vector3d< int > start_point, i3d::Vector3d< int > end_point, dataset_props_ptr props=nullptr) const

  Read region of interest from the server.
- template<cnpts::Scalar T>
  void read_region (i3d::Vector3d< int > start_point, i3d::Vector3d< int > end_point, i3d::Image3d< T > &dest, i3d::Vector3d< int > offset={0, 0, 0}, dataset_props_ptr props=nullptr) const

  Read region of interest from the server.
- template<cnpts::Scalar T>
  i3d::Image3d< T > read_image (dataset_props_ptr props=nullptr) const

  Read full image.
- template<cnpts::Scalar T>
  void write_block (const i3d::Image3d< T > &src, i3d::Vector3d< int > coord, i3d::Vector3d< int > src_offset={0, 0, 0}, dataset_props_ptr props=nullptr) const

  Write block to server.
- template<cnpts::Scalar T>
  void write_blocks (const i3d::Image3d< T > &src, const std::vector< i3d::Vector3d< int >> &coords, const std::vector< i3d::Vector3d< int >> &src_offsets, dataset_props_ptr props=nullptr) const

  Write blocks to server.
- template<cnpts::Scalar T>
  void write_image (const i3d::Image3d< T > &img, dataset_props_ptr props=nullptr) const

  Write image to server.

2.3.1 Detailed Description

Representation of connection to specific image.

Class representing connection to specific image on the server. This class provides basic methods for read/write operations necessary to transfer images from/to server. This class does not cache or precollect any data, so the first HTTP request will be send only when corresponding function is called.

2.3.2 Constructor & Destructor Documentation

2.3.2.1 ImageView()

```
ds::ImageView::ImageView (
            std::string ip,
            int port,
            std::string uuid,
            int channel,
            int timepoint,
            int angle,
            i3d::Vector3d< int > resolution,
            std::string version )
```

Construct a new Image View object.

Parameters

ip	IP address of server (`http://` at the beginning is not necessary)
port	Port, where the server is listening for requests
uuid	Unique identifier of dataset
channel	Channel, at which the image is located
timepoint	Timepoint, at which the image is located
angle	Angle, at which the image is located
resolution	Resolution, at which the image is located
version	Version, at which the image is located (integer identifier or "latest")
props	[Optional] cached dataset properties

2.3.3 Member Function Documentation

2.3.3.1 get_properties()

```
dataset_props_ptr ds::ImageView::get_properties ( ) const
```

Get dataset properties.

Returns

DatasetProperties

2.3.3.2 read_block() [1/2]

```
template<cnpts::Scalar T>
i3d::Image3d< T > ds::ImageView::read_block (
            i3d::Vector3d< int > coord,
            dataset_props_ptr props = nullptr ) const
```

Read one block from server.

Reads one block of image located at <coord> and returns it. The information about size of the image is collected from the server.

If in DEBUG, function will check wheter given coordinate corresponds to valid block.

As there is no (meaningfull) way for C++ to choose correct underlying type in runtime, make sure to specify correct template type.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

coord	Block coordinate
props	[Optional] cached dataset properties

Returns

Image containing selected block

2.3.3.3 read_block() [2/2]

```
template<cnpts::Scalar T>
void ds::ImageView::read_block (
            i3d::Vector3d< int > coord,
            i3d::Image3d< T > & dest,
            i3d::Vector3d< int > dest_offset = {0, 0, 0},
            dataset_props_ptr props = nullptr ) const
```

Read one block from server to image.

Reads one block of image located at <coord> and saves it to <dest> with offset <dest_offset>.

If in DEBUG, function will check wheter given coordinate corresponds to valid block as well as wheter the block fits into the image (taking offset into account).

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

coord	Block coordinate
dest	Image to write data to
dest_offset	Offset by which the corresponding write should be moved
props	[Optional] cached dataset properties

2.3.3.4 read_blocks() [1/2]

```
template<cnpts::Scalar T>
std::vector< i3d::Image3d< T > > ds::ImageView::read_blocks (
            const std::vector< i3d::Vector3d< int >> & coords,
            dataset_props_ptr props = nullptr ) const
```

Read blocks from server and return them.

Reads blocks specified in <coords> and returns them. Corresponding sizes are collected from server and calculated specificaly for each block.

This function is not optimized, meaning that for each coord in <coord>, one HTTP request will be sent out to the server. This can heavily slow down speed of the application as communication via network is not cheap. If you do not have specific needs,most of the time it will be faster to collect blocks into prealocated image (second overload of read_blocks), however it will eat more RAM.

If in DEBUG, the fucntion checks if coordinates given in <coords> points to a valid blocks.

As there is no (meaningfull) way for C++ to choose correct underlying type in runtime, make sure to specify correct template type.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

coords	Block coordinates
props	[Optional] cached dataset properties

Returns

Vector of fetched blocks (the order is the same as given in <coords>)

2.3.3.5 read_blocks() [2/2]

```
template<cnpts::Scalar T>
void ds::ImageView::read_blocks (
```

```
const std::vector< i3d::Vector3d< int >> & coords,
i3d::Image3d< T > & dest,
const std::vector< i3d::Vector3d< int >> & offsets,
dataset_props_ptr props = nullptr ) const
```

Read blocks from server and saves them into prealocated image.

Read blocks specified in <coords> and saves them into locations given in <offsets>.

If in DEBUG, the function checks if coordinates given in <coords> points to a valid blocks.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

coords	Block coordinates
dest	Prealocated destination image
offsets	Offsets at wich the corresponding blocks should be saved
props	[Optional] cached dataset properties

2.3.3.6 read_image()

```
template<cnpts::Scalar T>
i3d::Image3d< T > ds::ImageView::read_image (
            dataset_props_ptr props = nullptr ) const
```

Read full image.

Read full image from the server and return it. The information about dimensions are fetched from the server.

As there is no (meaningfull) way for C++ to choose correct underlying type in runtime, make sure to specify correct template type.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

props	[Optional] cached dataset properties

Returns

i3d::Image3d<T> Fetched image

2.3.3.7 read_region() [1/2]

```
template<cnpts::Scalar T>
i3d::Image3d< T > ds::ImageView::read_region (
             i3d::Vector3d< int > start_point,
             i3d::Vector3d< int > end_point,
             dataset_props_ptr props = nullptr ) const
```

Read region of interest from the server.

Read all neccessary blocks intersecting with chosen region from the server. It is neccessary, that start_point < end_point (elem-wise)..

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

start_point	smallest point of the region
end_point	largest point of the region
props	[Optional] cached dataset properties

Returns

i3d::Image3d<T> Selected region

2.3.3.8 read_region() [2/2]

```
template<cnpts::Scalar T>
void ds::ImageView::read_region (
             i3d::Vector3d< int > start_point,
             i3d::Vector3d< int > end_point,
             i3d::Image3d< T > & dest,
             i3d::Vector3d< int > offset = {0, 0, 0},
             dataset_props_ptr props = nullptr ) const
```

Read region of interest from the server.

Read all neccessary blocks intersecting with chosen region from the server and insert region into preallocated image <dest> at <offset>.

It is neccessary, that start_point < end_point (elem-wise)..

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

start_point	smallest point of the region
end_point	largest point of the region
dest	destination image
offset	offset to destination image
props	[Optional] cached dataset properties

2.3.3.9 write_block()

```
template<cnpts::Scalar T>
void ds::ImageView::write_block (
            const i3d::Image3d< T > & src,
            i3d::Vector3d< int > coord,
            i3d::Vector3d< int > src_offset = {0, 0, 0},
            dataset_props_ptr props = nullptr ) const
```

Write block to server.

Write block from source image to server. The information about dimensions are fetched from the server.

If in DEBUG, the function checks if coordinate given in <coord> points to a valid block, as well as wheter the offset specified for block is within image boundaries.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

src	Source image to collect block from
coord	Block coordinates
src_offset	Offset of given block in source image
props	[Optional] cached dataset properties

2.3.3.10 write_blocks()

```
template<cnpts::Scalar T>
void ds::ImageView::write_blocks (
            const i3d::Image3d< T > & src,
            const std::vector< i3d::Vector3d< int >> & coords,
            const std::vector< i3d::Vector3d< int >> & src_offsets,
            dataset_props_ptr props = nullptr ) const
```

Write blocks to server.

Write blocks from source image to server. The information about dimensions are fetched from the server.

If in DEBUG, the function checks if coordinates given in <coords> points to a valid block, as well as wheter the offsets specified for each block is within image boundaries.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

src	Source image to collect blocks from
coords	Vector of block coordinates
src_offsets	Offsets of corresponding blocks in source image
props	[Optional] cached dataset properties

2.3.3.11 write_image()

```
template<cnpts::Scalar T>
void ds::ImageView::write_image (
             const i3d::Image3d< T > & img,
             dataset_props_ptr props = nullptr ) const
```

Write image to server.

Write full image to server.

It is recommended to make sure that the dimension of the source image is the same as the dimension of image at server side.

Mostly, given smaller source image will emit error and fail to upload. Given larger source image will result in cropping.

Template Parameters

T	Scalar used as underlying type for image representation

Parameters

img	Source image
props	[Optional] cached dataset properties

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

- /home/somik/CBIA/hpc-datastore-cpp/src/hpc_ds_api.hpp

2.4 ds::ResolutionUnit Class Reference

Class representing resolution unit (in DatasetProperties)

`#include <hpc_ds_structs.hpp>`

Public Attributes

- double **value** = 0.0
- std::string **unit** = ""

Friends

- std::ostream & **operator**<< (std::ostream &stream, const ResolutionUnit &res)

2.4.1 Detailed Description

Class representing resolution unit (in DatasetProperties)

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

- /home/somik/CBIA/hpc-datastore-cpp/src/hpc_ds_structs.hpp

Index