volume-cartographer

Generated by Doxygen 1.8.12

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 volcart Namespace Reference

Typedefs

- using Dictionary = std::unordered_map< std::string, std::string >
- using Library = std::unordered_map< int, Dictionary >

Variables

- const Dictionary _1
- const Dictionary _2
- const Dictionary _3
- const Library VersionLibrary = {{1, _1}, {2, _2}, {3, _3}}

4.1.1 Typedef Documentation

4.1.1.1 Dictionary

```
using volcart::Dictionary = typedef std::unordered_map<std::string, std::string>
```

These constants represent the various versions of the Volume package Each version as a different library that may have varying information conatined or how the information is stored was changed between versions This type sets the dictionary to be a map which contains 2 strings for each entry. The first string tells the user what is being saved in that place The second string tells the user what type is saved.

4.1.1.2 Library

```
using volcart::Library = typedef std::unordered_map<int, Dictionary>
```

This type sets the Library to be a map which contains an integer and a dictonary type It essentially acts like a literal library, storing the dictionaries so they can be easily found. The integer acts as an index for each Dictonary The Dictionary is the type stored there

4.1.2 Variable Documentation

4.1.2.1 _1

```
const Dictionary volcart::_1
```

Initial value:

```
=
        {
        {"volumepkg name",   "string"},
        {"version",          "int"},
        {"width",            "int"},
        {"height",           "int"},
        {"number of slices", "int"},
        {"slice location",   "string"},
        {"min",              "double"},
        {"max",              "double"},
        {"voxelsize",        "double"}
        }
```

This is the dictionary that gives the data types for Volpkg 1

4.1.2.2 _2

```
const Dictionary volcart::_2
```

Initial value:

```
=
        {
        {"volumepkg name",   "string"},
        {"version",          "int"},
        {"width",            "int"},
        {"height",           "int"},
        {"number of slices", "int"},
        {"slice location",   "string"},
        {"min",              "double"},
        {"max",              "double"},
        {"voxelsize",        "double"},
        {"materialthickness","double"}
        }
```

This is the dictionary that gives the data types for Volkpg 2

4.1.2.3 _3

```
const Dictionary volcart::_3
```

Initial value:

```
=
        {
        {"volumepkg name",   "string"},
        {"version",          "int"},
        {"width",            "int"},
        {"height",           "int"},
        {"number of slices", "int"},
        {"slice location",   "string"},
        {"min",              "double"},
        {"max",              "double"},
        {"voxelsize",        "double"},
        {"materialthickness","double"}
        }
```

This is the dictionary that gives the data types for Volkpg 3

4.1.2.4 VersionLibrary

```
const Library volcart::VersionLibrary = {{1, _1}, {2, _2}, {3, _3}}
```

This library holds the Version Dictionaries and connects them to the possible versions that a user might enter when creating a Volume Package

Chapter 5

Class Documentation

5.1 VolumePkg Class Reference

```
#include <volumepkg.h>
```

Public Member Functions

- VolumePkg (const boost::filesystem::path &file_location, int version)
- VolumePkg (const boost::filesystem::path &file_location)
- int initialize ()
- const volcart::Volume & volume () const
- volcart::Volume & volume ()
- void printJSON () const
- void printDirs () const
- std::string getPkgName () const
- int getVersion () const
- int getNumberOfSlices () const
- int getSliceWidth () const
- int getSliceHeight () const
- double getVoxelSize () const
- double getMaterialThickness () const
- bool readOnly () const
- void readOnly (bool b)
- template< typename T >
  int setMetadata (const std::string &key, T value)
- void saveMetadata (const boost::filesystem::path &filePath)
- void saveMetadata ()
- bool setSliceData (size_t index, const cv::Mat &slice)
- std::string newSegmentation ()
- std::vector< std::string > getSegmentations () const
- void setActiveSegmentation (const std::string &name)
- std::string getActiveSegmentation ()
- boost::filesystem::path getActiveSegPath ()
- volcart::OrderedPointSet< volcart::Point3d > openCloud () const
- boost::filesystem::path getMeshPath () const
- cv::Mat getTextureData () const
- int saveCloud (const volcart::OrderedPointSet< volcart::Point3d > &segmentedCloud) const
- int saveMesh (const volcart::OrderedPointSet< volcart::Point3d > &segmentedCloud) const
- void saveMesh (const volcart::ITKMesh::Pointer mesh, const volcart::Texture &texture) const
- void saveTextureData (const cv::Mat &texture, const std::string &name="textured")
- void saveTextureData (volcart::Texture texture, int index=0)

Private Member Functions

- int _makeDirTree ()
- int getNumberOfSliceCharacters ()

Static Private Member Functions

- static volcart::Metadata _initConfig (const volcart::Dictionary &dict, int version)

Private Attributes

- bool _readOnly = true
- volcart::Metadata config
- volcart::Volume vol_
- boost::filesystem::path root_dir
- boost::filesystem::path segs_dir
- boost::filesystem::path slice_dir
- std::string activeSeg = ""
- std::vector< std::string > segmentations

5.1.1 Detailed Description

This class creates the container that holds the entire volume. It includes the slices, segmentations, and 3D models of that volume.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 VolumePkg() [1/2]

```
VolumePkg::VolumePkg (
              const boost::filesystem::path & file_location,
              int version )
```

Constructors

Parameters

file_location	the location where you want the Volume Package to be stored
version	which version of Volume Package would you like to create, current is 3Create a new Volume Package with a certain version

5.1.2.2 VolumePkg() [2/2]

```
VolumePkg::VolumePkg (
              const boost::filesystem::path & file_location )
```

Open an existing Volume Package

5.1.3 Member Function Documentation

5.1.3.1 _initConfig()

```
static volcart::Metadata VolumePkg::_initConfig (
             const volcart::Dictionary & dict,
             int version )  [static], [private]
```

Initalizes the metadata of the volume package

Parameters

dict	
version	which version of the Volume package you want to use, current is 3

Returns

returns a metadata type that has the metadata for the Volume Package

See also

common/types/Metadata.h

5.1.3.2 _makeDirTree()

```
int VolumePkg::_makeDirTree ( )  [private]
```

The Directory tree

5.1.3.3 getActiveSegmentation()

```
std::string VolumePkg::getActiveSegmentation ( )
```

Returns

the ID of the segmentation currently active

5.1.3.4 getActiveSegPath()

```
boost::filesystem::path VolumePkg::getActiveSegPath ( )
```

Returns

the file path of the segmentation that is currently active

5.1.3.5 getMaterialThickness()

```
double VolumePkg::getMaterialThickness ( ) const
```

Returns

the thickness of the material which was scanned

5.1.3.6 getMeshPath()

```
boost::filesystem::path VolumePkg::getMeshPath ( ) const
```

Returns

the path to the mesh of the active segmentation

5.1.3.7 getNumberOfSliceCharacters()

```
int VolumePkg::getNumberOfSliceCharacters ( )  [private]
```

5.1.3.8 getNumberOfSlices()

```
int VolumePkg::getNumberOfSlices ( ) const
```

Returns

How many slices are in this Volume Package

5.1.3.9 getPkgName()

```
std::string VolumePkg::getPkgName ( ) const
```

This set of functions is used to access the metadata of the Volume Package from the configuration file Note: The height and width of one slice is the same for all slices and the voxel size is constant throughout

Returns

the name of the Volume Package

5.1.3.10 getSegmentations()

```
std::vector<std::string> VolumePkg::getSegmentations ( ) const
```

Return a vector of strings representing the names of segmentations in the volpkg

Returns

vector of strings containing the names of the segmentations

5.1.3.11 getSliceHeight()

```
int VolumePkg::getSliceHeight ( ) const
```

Returns

The height of the slices in the Volume Package

5.1.3.12 getSliceWidth()

```
int VolumePkg::getSliceWidth ( ) const
```

Returns

The width of the slices in the Volume Package

5.1.3.13 getTextureData()

```
cv::Mat VolumePkg::getTextureData ( ) const
```

Returns

the texture image of active segmentation as a CV::Mat

5.1.3.14 getVersion()

```
int VolumePkg::getVersion ( ) const
```

Returns

Which Version this Volume Package is

5.1.3.15 getVoxelSize()

```
double VolumePkg::getVoxelSize ( ) const
```

Returns

The size of the voxels in the Volume Package

5.1.3.16 initialize()

```
int VolumePkg::initialize ( )
```

Writes the configuration file to disk

5.1.3.17 newSegmentation()

```
std::string VolumePkg::newSegmentation ( )
```

/brief Segmentation funtions Make a new folder inside the volume package to house everything for this segmentation and push back the new segmentation into our vector of segmentations

Returns

the name of the new segmentation

5.1.3.18 openCloud()

```
volcart::OrderedPointSet<volcart::Point3d> VolumePkg::openCloud ( ) const
```

5.1.3.19 printDirs()

```
void VolumePkg::printDirs ( ) const  [inline]
```

Print the configuration file < Print the directories in the Volume Package

5.1.3.20 printJSON()

```
void VolumePkg::printJSON ( ) const  [inline]
```

Prints information to help debug issues with creating Volume Packages

5.1.3.21 readOnly() [1/2]

```
bool VolumePkg::readOnly ( ) const  [inline]
```

Checks to see if the metadata can be overwritten

Returns

bool that tells if it's read only or not

5.1.3.22 readOnly() [2/2]

```
void VolumePkg::readOnly (
             bool b )  [inline]
```

Checks to see if the metadata can be overwritten and saves it to a variable

Parameters

b	variable where the result is saved

5.1.3.23 saveCloud()

```
int VolumePkg::saveCloud (
            const volcart::OrderedPointSet< volcart::Point3d > & segmentedCloud ) const
```

Save the Point Cloud of the active segmentation back to the volume package

Parameters

segmentedCloud	The PointCloud that has been segmented or otherwise changed

Returns

An integer indicating success or failure

5.1.3.24 saveMesh() [1/2]

```
int VolumePkg::saveMesh (
            const volcart::OrderedPointSet< volcart::Point3d > & segmentedCloud ) const
```

Saves the mesh of the active segmentation to the Volume Package

Parameters

segmentedCloud	The Point Cloud that contains the points in the mesh

Returns

An integer indicating success or failure

5.1.3.25 saveMesh() [2/2]

```
void VolumePkg::saveMesh (
            const volcart::ITKMesh::Pointer mesh,
            const volcart::Texture & texture ) const
```

Saves the mesh and the texture of the active segmentation to a mesh

Parameters

mesh	Mesh you want to save, should represent the points in that segmentation
texture	Texture of the segmentation that is to be saved with the mesh

5.1.3.26 saveMetadata() [1/2]

```
void VolumePkg::saveMetadata (
            const boost::filesystem::path & filePath )
```

Saves the data to any config file

Parameters

filePath	where you want the file to be stored

5.1.3.27 saveMetadata() [2/2]

```
void VolumePkg::saveMetadata ( )
```

5.1.3.28 saveTextureData() [1/2]

```
void VolumePkg::saveTextureData (
            const cv::Mat & texture,
            const std::string & name = "textured" )
```

Saves the texture data of the active segmentation to textured.png

Parameters

texture	the texture that you want to save as a cv::Mat
name	a constant that acts as the file name of the saved texture

5.1.3.29 saveTextureData() [2/2]

```
void VolumePkg::saveTextureData (
            volcart::Texture texture,
            int index = 0 )  [inline]
```

Saves the texture data that was generated by our texturing algorithms to textured.png

See also

common/types/Texture.h

Parameters

texture	Stores the texture information generate by our algorithms
index	Constant that tells the function to get the first image

5.1.3.30 setActiveSegmentation()

```
void VolumePkg::setActiveSegmentation (
             const std::string & name )
```

Set the private variable activeSeg to the seg we want to work with

Parameters

name	the name of the segementation you want to switch to

5.1.3.31 setMetadata()

```
template<typename T >
int VolumePkg::setMetadata (
             const std::string & key,
             T value )  [inline]
```

Sets a metadata member to have a specific key so it can be easily found later

Parameters

key	what is used to quickly find the data
value	value that you want to store in the key

Returns

whether or not it successfully saved that data to the key

5.1.3.32 setSliceData()

```
bool VolumePkg::setSliceData (
             size_t index,
             const cv::Mat & slice )
```

Sets the Slice Data

Parameters

index	which slice you're setting the data for
slice	the data you want to save to that index

5.1.3.33 volume() [1/2]

```
const volcart::Volume& VolumePkg::volume ( ) const  [inline]
```

2 ways of accessing the Volume type,

See also

common/types/Volume.h

5.1.3.34 volume() [2/2]

```
volcart::Volume& VolumePkg::volume ( ) [inline]
```

5.1.4 Member Data Documentation

5.1.4.1 _readOnly

```
bool VolumePkg::_readOnly = true [private]
```

Bool that tells if the data is read only

5.1.4.2 activeSeg

```
std::string VolumePkg::activeSeg = "" [private]
```

The segmentation currently being worked on

5.1.4.3 config

```
volcart::Metadata VolumePkg::config [private]
```

The metadata of the Volume Package

5.1.4.4 root_dir

```
boost::filesystem::path VolumePkg::root_dir [private]
```

The root directory of the Volume Package

5.1.4.5 segmentations

```
std::vector<std::string> VolumePkg::segmentations [private]
```

All of the segmentations that exist for this Volume Package

5.1.4.6 segs_dir

```
boost::filesystem::path VolumePkg::segs_dir [private]
```

The directory containing the segmentatiosn

5.1.4.7 slice_dir

```
boost::filesystem::path VolumePkg::slice_dir  [private]
```

The director containing the slices

5.1.4.8 vol_

```
volcart::Volume VolumePkg::vol_  [private]
```

The actual data of the Volume

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

- /Volumes/VC-Hannah/VC-Source-Code/volume-cartographer/volumepkg/include/volumepkg/volumepkg.h

Chapter 6

File Documentation

6.1 /Volumes/VC-Hannah/VC-Source-Code/volume-cartographer/volumepkg/include/volumepkg/volume File Reference

```
#include <cstdlib>
#include <iostream>
#include <boost/foreach.hpp>
#include <boost/lexical_cast.hpp>
#include <boost/filesystem.hpp>
#include "common/types/OrderedPointSet.h"
#include "common/types/Point.h"
#include "common/types/Texture.h"
#include "common/types/Volume.h"
#include "common/vc_defines.h"
#include "external/json.hpp"
#include "volumepkg/volumepkg_version.h"
```

Classes

- class VolumePkg

6.2 /Volumes/VC-Hannah/VC-Source-Code/volume-cartographer/volumepkg/include/volumepkg/volume _version.h File Reference

```
#include <string>
#include <unordered_map>
```

Namespaces

- volcart

Typedefs

- using volcart::Dictionary = std::unordered_map< std::string, std::string >
- using volcart::Library = std::unordered_map< int, Dictionary >

Variables

- const Dictionary volcart::_1
- const Dictionary volcart::_2
- const Dictionary volcart::_3
- const Library volcart::VersionLibrary = {{1, _1}, {2, _2}, {3, _3}}

Index