volume-cartographer

Generated by Doxygen 1.8.12

Contents

Chapter 1

Todo List

Member VolumePkg::openCloud () const

Error if activeSeg not set

Member VolumePkg::setActiveSegmentation (const std::string &)

Check that this seg actually exists in the volume

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 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}}

5.1.1 Typedef Documentation

5.1.1.1 Dictionary

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

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.

5.1.1.2 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

5.1.2 Variable Documentation

5.1.2.1 _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

5.1.2.2 _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

5.1.2.3 _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

5.1.2.4 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 6

Class Documentation

6.1 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 &)
- 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

6.1.1 Detailed Description

This class exists to be a container for all of the data about a particular set of data. It holds the slices, segmentations, mesh and texture data.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 VolumePkg() [1/2]

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

These are the Constructors, this first one is used to create a new volume package and the second is for opening existing ones

Parameters

file_location	This is where you want to store the base directory of the volume package
version	This is the version of Volpkg you want to use, the current version is 3 and is the only one that will work

6.1.2.2 VolumePkg() [2/2]

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

6.1.3 Member Function Documentation

6.1.3.1 _initConfig()

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

Sets up which version of the Volume Package you're using and associates all the keys with information from the dictonary

Parameters

dict	Which set of data types you want to use to create the VolumePkg, corresponds to the version
version	which version of the Volume Package you want to use, current is 3

Returns

Inital metadata

See also

common/types/Metadata.h

< Populate the config file with keys from the dictionary

6.1.3.2 _makeDirTree()

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

Makes the subdirectories for the Volume Package

Returns

integer indicating success

< Directories we need to make

< Make directories that don't exist

6.1.3.3 getActiveSegmentation()

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

Get the name of the segmentation currently active

Returns

string containing the name of the active segmentation

6.1.3.4 getActiveSegPath()

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

Returns the file path of the segmentation that is currently active

Returns

file path where the active segmentation is

6.1.3.5 getMaterialThickness()

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

Returns the thickness of the material that was scanned

Returns

Thickness of material scan

6.1.3.6 getMeshPath()

```
fs::path VolumePkg::getMeshPath ( ) const
```

Gets the file path where the mesh for the currently active segmentation is stored

Returns

Boost File path

6.1.3.7 getNumberOfSliceCharacters()

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

6.1.3.8 getNumberOfSlices()

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

Returns how many slices there are in this set of data

Returns

integer representing the number of slices

6.1.3.9 getPkgName()

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

Gets the name of the VolumePkg you are currently working on

Returns

Name of the Volume package

< Gets the Volume name from the configuration file

6.1.3.10 getSegmentations()

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

Returns a list of the current segmentations for that VolumePkg

Returns

a vector of strings that contains the names of all the segmentations for the VolumePkg

6.1.3.11 getSliceHeight()

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

Returns the height of the slices in the data, this is the same for all slices in a VolumePkg

Returns

Integer represents the height of the slices

6.1.3.12 getSliceWidth()

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

Returns the width of the slices, this is the same for all slices in a VolumePkg

Returns

integer that represents the slice width

6.1.3.13 getTextureData()

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

Returns the file that contains the Texture Data for the Volume

Returns

a Mat with the texture data

6.1.3.14 getVersion()

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

Gets the version that this VolumePkg is, current version is 3

Returns

integer that represents the version of VolumePkg

6.1.3.15 getVoxelSize()

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

Returns the size of the voxels in the data, this is the same for all voxels in a VolumePkg

Returns

Double that represents the size of the voxels

6.1.3.16 initialize()

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

This function writes the Volpkg out to the disk when you create it initially, it saves the metadata and builds the directory tree

Returns

An integer signalling success or failure

< A check to see if the file can be written to disk

< Save the JSON to disk

6.1.3.17 newSegmentation()

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

Creates a new segmentation

Returns

name of the segmentation created

< make a new dir based off the current date and time

< If the directory is successfully created, adds the name of the segementation to the list

6.1.3.18 openCloud()

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

This opens the file containing the information for the points that make up the Volume

Returns

An OrderedPointSet which contains all of the points on the Volume

See also

common/types/OrderedPointSet.h
common/types/PointSet.h

<

Todo Error if activeSeg not set

6.1.3.19 printDirs()

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

Prints the locations of the directories, mainly used for Debug

6.1.3.20 printJSON()

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

Prints the contents of the JSON file where the metadata is stored, mainly used for Debug

6.1.3.21 readOnly() [1/2]

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

Checks to see if the VolumePkg is read only

Returns

Bool that states if the data is read only

6.1.3.22 readOnly() [2/2]

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

Checks to see if the VolumePkg is read only and stores it in a variable

Parameters

variable	where the value of _readOnly is stored

6.1.3.23 saveCloud()

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

Saves the points of the Volume that may have been altered due to segmentation

Parameters

segmentedCloud	The set of points returned by the segmentation algorithm that need to be saved

Returns

Integer indicating success

6.1.3.24 saveMesh() [1/2]

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

Generates a mesh from the Points provided and saves it to the Segmentation folder of the active Segmentation

Parameters

segmentedCloud	The set of points returned by the segmentation algorithm that need to be meshed and saved

Returns

an Integer indicating success

< Creates a OrderedPointSetMesher type than uses the compute function to generate a mesh

See also

meshing/include/OrderedPointSetMesher.h

< Creates a PLY writer type and then writes the mesh out to the file

See also

common/io/plyWriter.h

6.1.3.25 saveMesh() [2/2]

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

Saves a generated mesh along with the Texture information that is provided

Parameters

mesh	Mesh that was generated from the points in the cloud of the current segmentation
texture	Texture information for a mesh

See also

common/types/Texture.h

< Creates an OBJ writer type and then writes the mesh and the texture out to the file

See also

common/io/objWriter.h

6.1.3.26 saveMetadata() [1/2]

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

Saves the metadata to a file

Parameters

filePath	File path where you want the metadata to be stored

6.1.3.27 saveMetadata() [2/2]

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

Saves the metadata to a file determined by the program

6.1.3.28 saveTextureData() [1/2]

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

Saves the texture data for the current segmentation

Parameters

texture	Texture information as a Mat
name	automatcially set to be textured and represents the name of the file where this is stored

6.1.3.29 saveTextureData() [2/2]

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

Saves the texture data for the current segmentation

Parameters

texture	Texture information

See also

common/types/Texture.h

Parameters

index	Tells the function which slice to use the texture data from, automatically set to 0

6.1.3.30 setActiveSegmentation()

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

Set the active segmentation to be a particular segmentation

Parameters

name	of the segmentation you want to be the active one

<

Todo Check that this seg actually exists in the volume

6.1.3.31 setMetadata()

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

Sets a particular metadata value to a key so that it can be quickly found later

Parameters

key	what the metadata is set to
value	metadata that you want to store

Returns

Integer indicating success

6.1.3.32 setSliceData()

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

Allows you to set the slice height and width

Parameters

index	Slice number that you want to store data for
slice	Slice that contains the information to set height and width

< Performs a read only check and then sets the data

6.1.3.33 volume() [1/2]

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

Returns the Volume information stored as a Volume type

Returns

VolumeType

See also

common/types/Volume.h

6.1.3.34 volume() [2/2]

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

6.1.4 Member Data Documentation

6.1.4.1 _readOnly

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

Bool that tells if the Volume Package is read only

6.1.4.2 activeSeg

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

This is the segmentation that is currently being worked on

6.1.4.3 config

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

Contains the Metadata for the Volume package

See also

common/types/Metadata.h

6.1.4.4 root_dir

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

The root directory of the Volume package, stores the other directories

6.1.4.5 segmentations

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

The list of all the segmentations for a specific VolumePkg

6.1.4.6 segs_dir

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

The directory containing the Segmentations that have been made

6.1.4.7 slice_dir

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

The directory containing the slices that the volume represents

6.1.4.8 vol_

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

Contains the information stored in the Volume

See also

common/types/Volume.h

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

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

6.2 VolumePkg_Version Class Reference

```
#include <volumepkg_version.h>
```

6.2.1 Detailed Description

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

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

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

Chapter 7

File Documentation

7.1 /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

7.2 /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}}

7.3 /Volumes/VC-Hannah/VC-Source-Code/volume-cartographer/volumepkg/src/volumepkg.cpp File Reference

```
#include "volumepkg/volumepkg.h"
#include "common/io/PointSetIO.h"
#include "common/io/objWriter.h"
#include "common/io/plyWriter.h"
#include "common/types/OrderedPointSet.h"
#include "common/types/Point.h"
#include "meshing/OrderedPointSetMesher.h"
```

Index