# sosicon

Version v 0.1 prerelease
16.08.14 12:10

# Table of Contents

# Module Index

## Modules

Here is a list of all modules:

# Namespace Index

## Namespace List

Here is a list of all namespaces with brief descriptions:

# Hierarchical Index

## Class Hierarchy

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

# Class Index

## Class List

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

# File Index

## File List

Here is a list of all files with brief descriptions:

# Module Documentation

## Converters

### Classes

- class **sosicon::ConverterSosi2shp**
- *SOSI to ESRI Shape converter.* class **sosicon::ConverterSosi2tsv**
- *SOSI to TSV converter.* class **sosicon::ConverterSosi2xml**
- *SOSI to ESRI Shape converter.* class **sosicon::ConverterSosiStat**

*SOSI to ESRI Shape converter.*

### Detailed Description

Collection of **sosicon::IConverter** implementations for handling and generating file output. Different command-line arguments will employ different converters. The factory class **sosicon::Factory** is responsible for creating correct **IConverter** instance. The following arguments are currently interpreted:

- -2shp: **sosicon::ConverterSosi2shp** Shapefile conversion
- -2tsv: **sosicon::ConverterSosi2tsv** TSV file conversion
- -2xml: **sosicon::ConverterSosi2xml** Shape file conversion
- -stat: **sosicon::ConverterSosiStat** SOSI statistics (printout)

Collection of **sosicon::IConverter** implementations for handling and generating file output. Different command-line arguments will employ different converters. The factory class **sosicon::Factory** is responsible for creating correct **IConverter** instance. The following arguments are currently interpreted:

- -2shp: **sosicon::ConverterSosi2shp** Shapefile conversion
- -2tsv: **sosicon::ConverterSosi2tsv** TSV file conversion
- -stat: sosicon::ConverterSosi2Xml XML conversion
- -stat: **sosicon::ConverterSosiStat** SOSI statistics (printout)

## Interfaces

### Classes

- class **sosicon::IBinaryStreamable**
- *Interface: Binary streamable object.* class **sosicon::IConverter**
- *Interface: Converter.* class **sosicon::ICoordinate**
- *Interface:* ***Coordinate****.* class **sosicon::ILookupTable**
- *Interface: Lookup table.* class **sosicon::IRectangle**
- *Interface: Rectangle.* class **sosicon::IShapeElement**
- *Interface: Shape element.* class **sosicon::IShapeElementHeader**
- *Interface: Shape element header.* class **sosicon::IShapeHeader**
- *Interface: Shape element.* class **sosicon::IShapefile**
- *Interface: Shapefile.* class **sosicon::IShapefileDbfPart**
- *Interface: ShapefileDbfPart.* class **sosicon::IShapefilePrjPart**
- *Interface: ShapefilePrjPart.* class **sosicon::IShapefileShpPart**
- *Interface: ShapefileShpPart.* class **sosicon::IShapefileShxPart**
- *Interface: ShapefileShxPart.* class **sosicon::ISosiElement**
- *Interface: SOSI element.* class **sosicon::ISosiHeadMember**

## *Interface: SOSI header element.* Functions

- std::ostream & **sosicon::operator<<** (std::ostream &os, IBinaryStreamable &binaryStreamable)
  *Stream output operator.*

---

## Detailed Description

This is a listing of generic interfaces used within sosicon.

---

## Function Documentation

### std::ostream& sosicon::operator<< (std::ostream & *os*, IBinaryStreamable & *binaryStreamable*)`[inline]`

Stream output operator.

Placed outside any class definitions. Invokes WriteBinary() on the source **IBinaryStreamable** object.

**Parameters:**

| | |
|---|---|
| *os* | target stream. |
| *binaryStreamable* | target object. |

**Returns:**

The stream object is returned to allow for chaining.

Definition at line 58 of file i_binary_streamable.h.

# SOSI Elements

## Classes

- class **sosicon::sosi::SosiElement**
- *Basic SOSI element.* class **sosicon::sosi::SosiElementSearch**
- class **sosicon::sosi::SosiJunctionPoint**
- *SOSI Junction point.* class **sosicon::sosi::SosiNorthEast**
- *SOSI North-east element.* class **sosicon::sosi::SosiOrigoNE**
- *SOSI Junction point.* class **sosicon::sosi::SosiRefList**
- *SOSI REF list.* class **sosicon::sosi::SosiTranslationTable**
- struct **sosicon::sosi::ReferenceData**
- *SOSI reference number.* class **sosicon::sosi::CoordSys**
- *SOSI coordinate system.* class **sosicon::sosi::SosiUnit**

## *SOSI Unit.* Typedefs

- typedef std::map< std::string,
- ISosiElement * > **sosicon::sosi::SosiElementMap**
  *Element index type.*
- typedef std::vector
- < ISosiElement * > **sosicon::sosi::SosiChildrenList**
- typedef SosiChildrenList::iterator **sosicon::sosi::SosiChildrenIterator**

- typedef std::vector
- < ReferenceData * > **sosicon::sosi::GeometryRef**
  *List of SOSI references.*
- typedef std::vector
- < GeometryRef * > **sosicon::sosi::GeometryCollection**
  *Collection of SOSI reference lists.*

## Enumerations

- enum **sosicon::sosi::ElementType** { **sosicon::sosi::sosi_element_unknown**, **sosicon::sosi::sosi_element_airport_roads**, **sosicon::sosi::sosi_element_airport_type**, **sosicon::sosi::sosi_element_area**, **sosicon::sosi::sosi_element_charset**, **sosicon::sosi::sosi_element_coordsys**, **sosicon::sosi::sosi_element_curve**, **sosicon::sosi::sosi_element_eof**, **sosicon::sosi::sosi_element_head**, **sosicon::sosi::sosi_element_height**, **sosicon::sosi::sosi_element_iata_code**, **sosicon::sosi::sosi_element_icao_code**, **sosicon::sosi::sosi_element_kp**, **sosicon::sosi::sosi_element_level**, **sosicon::sosi::sosi_element_max_ne**, **sosicon::sosi::sosi_element_min_ne**, **sosicon::sosi::sosi_element_municipality**, **sosicon::sosi::sosi_element_name**, **sosicon::sosi::sosi_element_ne**, **sosicon::sosi::sosi_element_neh**, **sosicon::sosi::sosi_element_objtype**, **sosicon::sosi::sosi_element_origo_ne**, **sosicon::sosi::sosi_element_owner**, **sosicon::sosi::sosi_element_point**, **sosicon::sosi::sosi_element_quality**, **sosicon::sosi::sosi_element_ref**, **sosicon::sosi::sosi_element_surface**, **sosicon::sosi::sosi_element_text**, **sosicon::sosi::sosi_element_traffic_type**, **sosicon::sosi::sosi_element_transpar**, **sosicon::sosi::sosi_element_unit**, **sosicon::sosi::sosi_element_updatedate**, **sosicon::sosi::sosi_element_water_width**, **sosicon::sosi::sosi_element_vendor**, **sosicon::sosi::sosi_element_version** }
- *List of SOSI element types.* enum **sosicon::sosi::ObjType** { **sosicon::sosi::sosi_objtype_unknown**, **sosicon::sosi::sosi_objtype_airport**, **sosicon::sosi::sosi_objtype_airport_type**, **sosicon::sosi::sosi_objtype_baseline**, **sosicon::sosi::sosi_objtype_carriageway**, **sosicon::sosi::sosi_objtype_cadastral_address**, **sosicon::sosi::sosi_objtype_coastline**, **sosicon::sosi::sosi_objtype_county_boundary**, **sosicon::sosi::sosi_objtype_data_delineation**, **sosicon::sosi::sosi_objtype_edge_view**, **sosicon::sosi::sosi_objtype_fictious_dividing_line**, **sosicon::sosi::sosi_objtype_forest**, **sosicon::sosi::sosi_objtype_developed_area**, **sosicon::sosi::sosi_objtype_golf_course**, **sosicon::sosi::sosi_objtype_industrial_area**, **sosicon::sosi::sosi_objtype_lake**, **sosicon::sosi::sosi_objtype_lane**, **sosicon::sosi::sosi_objtype_lake_edge**, **sosicon::sosi::sosi_objtype_lake_river_barrier**, **sosicon::sosi::sosi_objtype_land_use_boundary**, **sosicon::sosi::sosi_objtype_level_crossing**, **sosicon::sosi::sosi_objtype_municipal_divide**, **sosicon::sosi::sosi_objtype_municipality**, **sosicon::sosi::sosi_objtype_municipality_boundary**, **sosicon::sosi::sosi_objtype_marsh**, **sosicon::sosi::sosi_objtype_national_border**, **sosicon::sosi::sosi_objtype_pedestrian_bicycle_road_centre_line**, **sosicon::sosi::sosi_objtype_sea_river_delineation**, **sosicon::sosi::sosi_objtype_snow_field**, **sosicon::sosi::sosi_objtype_open_land**, **sosicon::sosi::sosi_objtype_river_brook**, **sosicon::sosi::sosi_objtype_river_brook_edge**, **sosicon::sosi::sosi_objtype_road_block**, **sosicon::sosi::sosi_objtype_road_centre_line**, **sosicon::sosi::sosi_objtype_road_under_railway**, **sosicon::sosi::sosi_objtype_sea_surface**, **sosicon::sosi::sosi_objtype_sidewalk**, **sosicon::sosi::sosi_objtype_spelling**, **sosicon::sosi::sosi_objtype_stone_quarry**, **sosicon::sosi::sosi_objtype_street_address**, **sosicon::sosi::sosi_objtype_territorial_boundary**, **sosicon::sosi::sosi_objtype_turn_connecting_segment** }
- *List of SOSI OBJTYPEs.* enum **sosicon::sosi::JunctionPoint** { **sosicon::sosi::sosi_junction_node**, **sosicon::sosi::sosi_junction_connection**, **sosicon::sosi::sosi_junction_open_end** }

## *Default SOSI junction point layer types.* Functions

- CoordSys **sosicon::sosi::sysCodeToCoordSys** (int sysCode)
  *Convert SOSI SYSKODE value to coordinate system data.*
- ElementType **sosicon::sosi::sosiNameToType** (std::string sosiElementName)

*Convert SOSI element names to ElementType enum value.*

- ObjType **sosicon::sosi::sosiObjNameToType** (std::string sosiObjTypeName)
  *Convert SOSI objtype names to ObjType enum value.*

## Detailed Description

Implemented representation of SOSI file elements.

## Typedef Documentation

### typedef std::vector<GeometryRef*> sosicon::sosi::GeometryCollection

Collection of SOSI reference lists.

Definition at line 163 of file sosi_types.h.

### typedef std::vector<ReferenceData*> sosicon::sosi::GeometryRef

List of SOSI references.

Definition at line 160 of file sosi_types.h.

### typedef SosiChildrenList::iterator sosicon::sosi::SosiChildrenIterator

Definition at line 44 of file sosi_element_search.h.

### typedef std::vector<ISosiElement*> sosicon::sosi::SosiChildrenList

Definition at line 42 of file sosi_element_search.h.

### typedef std::map<std::string,ISosiElement*> sosicon::sosi::SosiElementMap

Element index type.

Definition at line 40 of file sosi_element_search.h.

## Enumeration Type Documentation

### enum sosicon::sosi::ElementType

List of SOSI element types.

**Enumerator**

***sosi_element_unknown*** Unknown element.

***sosi_element_airport_roads*** Airport roads.

***sosi_element_airport_type*** Airport type.

***sosi_element_area*** Area.

***sosi_element_charset*** Character set.

***sosi_element_coordsys*** Grid type.

***sosi_element_curve*** Curve.

***sosi_element_eof*** End of file.

***sosi_element_head*** Header.

***sosi_element_height*** Height.

***sosi_element_iata_code*** IATA code (aviation)

***sosi_element_icao_code*** ICAO code (aviation)

***sosi_element_kp*** Junction point.

***sosi_element_level*** SOSI level.

***sosi_element_max_ne*** Maximum north-east (bbox)

***sosi_element_min_ne*** Minimum north-east (bbox)

***sosi_element_municipality*** Municipality.

***sosi_element_name*** Name.

***sosi_element_ne*** North-east coordinate.

***sosi_element_neh*** North-east/height coordinate.

***sosi_element_objtype*** Object type.

***sosi_element_origo_ne*** Origo north-east.

***sosi_element_owner*** Dataset owner.

***sosi_element_point*** Point.

***sosi_element_quality*** Quality of data.

***sosi_element_ref*** Element reference.

***sosi_element_surface*** Surface.

***sosi_element_text*** Text.

***sosi_element_traffic_type*** Traffic type.

***sosi_element_transpar*** Datum/projection/coordinate system.

***sosi_element_unit*** Resolution (fraction of a metre)

***sosi_element_updatedate*** Update date.

***sosi_element_water_width*** Water width.

***sosi_element_vendor*** Data vendor.

***sosi_element_version*** SOSI version.

Definition at line 38 of file sosi_types.h.

### enum sosicon::sosi::JunctionPoint

Default SOSI junction point layer types.

**Enumerator**

***sosi_junction_node*** Node point (KP 1)

***sosi_junction_connection*** Connection point (KP 900)

***sosi_junction_open_end*** Valid open-ended point (KP 999)

Definition at line 123 of file sosi_types.h.

### enum sosicon::sosi::ObjType

List of SOSI OBJTYPEs.

**Enumerator**

***sosi_objtype_unknown*** Unknown or no feature.

***sosi_objtype_airport*** Airport.

***sosi_objtype_airport_type*** Airport type.

***sosi_objtype_baseline*** Baseline.

***sosi_objtype_carriageway*** Carriageway.

***sosi_objtype_cadastral_address*** Cadastral address.

***sosi_objtype_coastline*** Coast line.

***sosi_objtype_county_boundary*** County boundary.

***sosi_objtype_data_delineation*** Clipping path.

***sosi_objtype_edge_view*** Edge view.

***sosi_objtype_fictious_dividing_line*** Line splitting large surfeces.

***sosi_objtype_forest*** Forest.

***sosi_objtype_developed_area*** Built up area.

***sosi_objtype_golf_course*** Golf course.

***sosi_objtype_industrial_area*** Industrial area.

***sosi_objtype_lake*** Lake.

***sosi_objtype_lane*** Driving lane.

***sosi_objtype_lake_edge*** Lake edge.

***sosi_objtype_lake_river_barrier*** Lake-to-river delimitation.

***sosi_objtype_land_use_boundary*** Land use border.

***sosi_objtype_level_crossing*** Track level crossing.

***sosi_objtype_municipal_divide*** Municipal boundary crossing.

***sosi_objtype_municipality*** Municipality.

***sosi_objtype_municipality_boundary*** Municipality boundary.

***sosi_objtype_marsh*** Marsh.

***sosi_objtype_national_border*** National border.

***sosi_objtype_pedestrian_bicycle_road_centre_line*** mid-way line

***sosi_objtype_sea_river_delineation*** Sea or river delineation.

***sosi_objtype_snow_field*** Snow/glacier.

***sosi_objtype_open_land*** Open land.

***sosi_objtype_river_brook*** River or stream.

***sosi_objtype_river_brook_edge*** River or stream bank.

***sosi_objtype_road_block*** Road block.

***sosi_objtype_road_centre_line*** Road centre line.

***sosi_objtype_road_under_railway*** Road under railway.

***sosi_objtype_sea_surface*** Sea surface.

***sosi_objtype_sidewalk*** Sidewalk.

***sosi_objtype_spelling*** Spelling of place name.

***sosi_objtype_stone_quarry*** Area for stone quarry.

***sosi_objtype_street_address*** Street address.

***sosi_objtype_territorial_boundary*** Territorial boundary (nautical)

***sosi_objtype_turn_connecting_segment*** Turn connection segment (artificial)

Definition at line 77 of file sosi_types.h.

## Function Documentation

### ElementType sosicon::sosi::sosiNameToType (std::string *sosiElementName*)

Convert SOSI element names to ElementType enum value.

The enum member names are translations of the Norwegian element names.

**Parameters:**

| | |
|---|---|
| *std::string* | sosiElementName The standard SOSI element name in Norwegian. |

**Returns:**

ElementType enumeration value representing current element type.

### ObjType sosicon::sosi::sosiObjNameToType (std::string *sosiObjTypeName*)

Convert SOSI objtype names to ObjType enum value.

The enum member names are translations of the Norwegian geograpic features.

**Parameters:**

| | |
|---|---|
| *std::string* | sosiObjtypeName The standard SOSI objtype name in Norwegian. |

**Returns:**

ObjType enumeration value representing current element objtype.

### CoordSys sosicon::sosi::sysCodeToCoordSys (int *sysCode*)

Convert SOSI SYSKODE value to coordinate system data.

**Parameters:**

| | |
|---|---|
| *int* | The SOSI SYSKODE value. |

**Returns:**

**CoordSys** structure with information about the requested coordinate system.

# Namespace Documentation

## sosicon Namespace Reference

Application root.

### Namespaces

- **byteOrder**
- *Big/low-endian conversions.* **shape**
- *ESRI Shape.* **sosi**
- *SOSI.* **utils**

### *String manipulation routines.* Classes

- class **CommandLine**
- *Command-line parser.* class **ConverterSosi2shp**
- *SOSI to ESRI Shape converter.* class **ConverterSosi2tsv**
- *SOSI to TSV converter.* class **ConverterSosi2xml**
- *SOSI to ESRI Shape converter.* class **ConverterSosiStat**
- *SOSI to ESRI Shape converter.* class **Coordinate**
- ***Coordinate** container.* class **CoordinateCollection**
- ***Coordinate** container.* class **Factory**
- ***Factory** class.* class **IBinaryStreamable**
- *Interface: Binary streamable object.* class **IConverter**
- *Interface: Converter.* class **ICoordinate**
- *Interface: **Coordinate**.* class **ILookupTable**
- *Interface: Lookup table.* class **IRectangle**
- *Interface: Rectangle.* class **IShapeElement**
- *Interface: Shape element.* class **IShapeElementHeader**
- *Interface: Shape element header.* class **IShapefile**
- *Interface: Shapefile.* class **IShapefileDbfPart**
- *Interface: ShapefileDbfPart.* class **IShapefilePrjPart**
- *Interface: ShapefilePrjPart.* class **IShapefileShpPart**
- *Interface: ShapefileShpPart.* class **IShapefileShxPart**
- *Interface: ShapefileShxPart.* class **IShapeHeader**
- *Interface: Shape element.* class **ISosiElement**
- *Interface: SOSI element.* class **ISosiHeadMember**
- *Interface: SOSI header element.* class **Parser**

### *SOSI file parser.* Typedefs

- typedef std::vector
- < **ICoordinate** * > **CoordinateList**

### Functions

- bool **getNext** (**ICoordinate** *&coord, **sosi::NorthEastList** &list, sosi::NorthEastList::iterator &i)
  *Get next coordinate in list.*
- bool **getNextOffset** (int &offset, std::vector< int > &offsets, std::vector< int >::iterator &iterator)
  *Get next offset in part offsets list.*
- std::ostream & **operator<<** (std::ostream &os, **IBinaryStreamable** &binaryStreamable)

*Stream output operator.*

## Detailed Description

Application root.

## Typedef Documentation

### typedef std::vector< ICoordinate * > sosicon::CoordinateList

Definition at line 26 of file common_types.h.

## Function Documentation

### bool sosicon::getNext (ICoordinate *& *coord*, sosi::NorthEastList & *list*, sosi::NorthEastList::iterator & *i*)

Get next coordinate in list.

Definition at line 21 of file coordinate_collection.cpp.

### bool sosicon::getNextOffset (int & *offset*, std::vector< int > & *offsets*, std::vector< int >::iterator & *iterator*)

Get next offset in part offsets list.

Definition at line 42 of file coordinate_collection.cpp.

# sosicon::byteOrder Namespace Reference

Big/low-endian conversions.

## Enumerations

- enum **Endianness** { **not_set**, **big**, **little** }

## *Big/little flag.* Functions

- **Endianness determine** ()
  *Determines system endianness.*
- void **doubleToLittleEndian** (double from, char *to)
  *Writes little endian representation of double.*
- void **toBigEndian** (const char *from, char *to, size_t bufSize)
  *Reverses buffer to big endian if required.*
- void **toLittleEndian** (const char *from, char *to, size_t bufSize)
  *Reverses buffer to little endian if required.*

## Variables

- enum **sosicon::byteOrder::Endianness endianness**
  *Stores system endianness.*

## Detailed Description

Big/low-endian conversions.

## Enumeration Type Documentation

### enum sosicon::byteOrder::Endianness

Big/little flag.

**Enumerator**

***not_set***
***big***
***little***

Definition at line 39 of file byte_order.h.

## Function Documentation

### sosicon::byteOrder::Endianness sosicon::byteOrder::determine ()

Determines system endianness.

Tests byte-order to see if the program runs on a big endian or a little endian architecture. Flags the byteOrder::endian variable.

**Returns:**

System endianness.

**Return values:**

| | |
|---|---|
| *Endianness::big* | Big endian system. |
| *Endianness::little* | Little endian system. |

Definition at line 24 of file byte_order.cpp.

### void sosicon::byteOrder::doubleToLittleEndian (double *from*, char * *to*)

Writes little endian representation of double.

Serializes double-precision floating point value to IEEE little endian representation for binary embedding in files.

**Parameters:**

| | |
|---|---|
| *from* | The double value to parse. |
| *to* | pointer to destination buffer. The buffer must be at least 8 bytes wide, as this is the size of the IEEE 754 format. |

Definition at line 56 of file byte_order.cpp.

### void sosicon::byteOrder::toBigEndian (const char * *from*, char * *to*, size_t *bufSize*)

Reverses buffer to big endian if required.

Copies source buffer to destination buffer. If the program runs on a little-endian system, the byte order will be reversed.

**Parameters:**

| | |
|---|---|
| *from* | pointer to source buffer. |
| *to* | pointer to destination buffer. The buffer must be at least as big as the source buffer. |
| *bufSize* | The number of bytes to copy. |

Definition at line 36 of file byte_order.cpp.

### void sosicon::byteOrder::toLittleEndian (const char * *from*, char * *to*, size_t *bufSize*)

Reverses buffer to little endian if required.

Copies source buffer to destination buffer. If the program runs on a big-endian system, the byte order will be reversed.

**Parameters:**

| | |
|---|---|
| *from* | pointer to source buffer. |
| *to* | pointer to destination buffer. The buffer must be at least as big as the source buffer. |
| *bufSize* | The number of bytes to copy. |

Definition at line 46 of file byte_order.cpp.

## Variable Documentation

### enum sosicon::byteOrder::Endianness sosicon::byteOrder::endianness

Stores system endianness.

# sosicon::shape Namespace Reference

ESRI Shape.

## Classes

- union **DoubleField**
- *32 bit double / byte field* union **Int16Field**
- *16 bit integer / byte field* union **Int32Field**
- *32 bit integer / byte field* union **Int32TField**
- *32 bit integer / byte / geom::ShapeType field* union **Int8Field**
- *8 bit integer / byte field* class **Shapefile**
- ***Shapefile** implementation.* struct **ShxIndex**

## Typedefs

- typedef std::map< std::string,
- std::string > **DbfRecord**
- typedef std::vector< **DbfRecord** > **DbfRecordSet**
- typedef std::map< std::string,
- int > **DbfFieldLengths**
- typedef std::vector< **ShxIndex** > **ShxOffsets**

## Enumerations

- enum **ShapeType** { **shape_type_none**, **shape_type_nullShape**, **shape_type_point**, **shape_type_polyLine**, **shape_type_polygon**, **shape_type_multipoint**, **shape_type_pointZ**, **shape_type_polyLineZ**, **shape_type_polygonZ**, **shape_type_multipointZ**, **shape_type_pointM**, **shape_type_polyLineM**, **shape_type_polygonM**, **shape_type_multiPointM**, **shape_type_multiPatch** }

## *Geometry types.* Functions

- **ShapeType getShapeEquivalent** (**sosi::ElementType** sosiType)
  *Resolve geometry type.*

## Detailed Description

ESRI Shape.

## Typedef Documentation

### typedef std::map<std::string, int> sosicon::shape::DbfFieldLengths

Definition at line 90 of file shapefile_types.h.

### typedef std::map<std::string, std::string> sosicon::shape::DbfRecord

Definition at line 88 of file shapefile_types.h.

### typedef std::vector<DbfRecord> sosicon::shape::DbfRecordSet

Definition at line 89 of file shapefile_types.h.

### typedef std::vector<ShxIndex> sosicon::shape::ShxOffsets

Definition at line 91 of file shapefile_types.h.

## Enumeration Type Documentation

### enum sosicon::shape::ShapeType

Geometry types.

The numeric values are in accordance with the shapefile specification.

**Enumerator**

***shape_type_none***
***shape_type_nullShape***
***shape_type_point***
***shape_type_polyLine***
***shape_type_polygon***
***shape_type_multipoint***
***shape_type_pointZ***
***shape_type_polyLineZ***
***shape_type_polygonZ***
***shape_type_multipointZ***
***shape_type_pointM***
***shape_type_polyLineM***
***shape_type_polygonM***
***shape_type_multiPointM***
***shape_type_multiPatch***

Definition at line 34 of file shapefile_types.h.

## Function Documentation

### sosicon::shape::ShapeType sosicon::shape::getShapeEquivalent (sosi::ElementType *sosiType*)

Resolve geometry type.

Translate SOSI geomtry type to corresponding shape geometry, if applicable

Definition at line 21 of file shapefile.cpp.

# sosicon::sosi Namespace Reference

SOSI.

## Classes

- class **CoordSys**
- *SOSI coordinate system.* struct **ReferenceData**
- *SOSI reference number.* class **SosiElement**
- *Basic SOSI element.* class **SosiElementSearch**
- class **SosiJunctionPoint**
- *SOSI Junction point.* class **SosiNorthEast**
- *SOSI North-east element.* class **SosiOrigoNE**
- *SOSI Junction point.* class **SosiRefList**
- *SOSI REF list.* class **SosiTranslationTable**
- class **SosiUnit**

## *SOSI Unit.* Typedefs

- typedef std::map< std::string,
- **ISosiElement** * > **SosiElementMap**
  *Element index type.*
- typedef std::vector
- < **ISosiElement** * > **SosiChildrenList**
- typedef SosiChildrenList::iterator **SosiChildrenIterator**
- typedef std::vector
- < **SosiNorthEast** * > **NorthEastList**
  *List of SosiSNorthEast elements.*
- typedef std::vector
- < **ReferenceData** * > **GeometryRef**
  *List of SOSI references.*
- typedef std::vector
- < **GeometryRef** * > **GeometryCollection**
  *Collection of SOSI reference lists.*

## Enumerations

- enum **ElementType** { **sosi_element_unknown**, **sosi_element_airport_roads**, **sosi_element_airport_type**, **sosi_element_area**, **sosi_element_charset**, **sosi_element_coordsys**, **sosi_element_curve**, **sosi_element_eof**, **sosi_element_head**, **sosi_element_height**, **sosi_element_iata_code**, **sosi_element_icao_code**, **sosi_element_kp**, **sosi_element_level**, **sosi_element_max_ne**, **sosi_element_min_ne**, **sosi_element_municipality**, **sosi_element_name**, **sosi_element_ne**, **sosi_element_neh**, **sosi_element_objtype**, **sosi_element_origo_ne**, **sosi_element_owner**, **sosi_element_point**, **sosi_element_quality**, **sosi_element_ref**, **sosi_element_surface**, **sosi_element_text**, **sosi_element_traffic_type**, **sosi_element_transpar**, **sosi_element_unit**, **sosi_element_updatedate**, **sosi_element_water_width**, **sosi_element_vendor**, **sosi_element_version** }
- *List of SOSI element types.* enum **ObjType** { **sosi_objtype_unknown**, **sosi_objtype_airport**, **sosi_objtype_airport_type**, **sosi_objtype_baseline**, **sosi_objtype_carriageway**, **sosi_objtype_cadastral_address**, **sosi_objtype_coastline**, **sosi_objtype_county_boundary**, **sosi_objtype_data_delineation**, **sosi_objtype_edge_view**, **sosi_objtype_fictious_dividing_line**, **sosi_objtype_forest**, **sosi_objtype_developed_area**, **sosi_objtype_golf_course**, **sosi_objtype_industrial_area**, **sosi_objtype_lake**, **sosi_objtype_lane**, **sosi_objtype_lake_edge**, **sosi_objtype_lake_river_barrier**, **sosi_objtype_land_use_boundary**, **sosi_objtype_level_crossing**,

**sosi_objtype_municipal_divide**, **sosi_objtype_municipality**, **sosi_objtype_municipality_boundary**, **sosi_objtype_marsh**, **sosi_objtype_national_border**, **sosi_objtype_pedestrian_bicycle_road_centre_line**, **sosi_objtype_sea_river_delineation**, **sosi_objtype_snow_field**, **sosi_objtype_open_land**, **sosi_objtype_river_brook**, **sosi_objtype_river_brook_edge**, **sosi_objtype_road_block**, **sosi_objtype_road_centre_line**, **sosi_objtype_road_under_railway**, **sosi_objtype_sea_surface**, **sosi_objtype_sidewalk**, **sosi_objtype_spelling**, **sosi_objtype_stone_quarry**, **sosi_objtype_street_address**, **sosi_objtype_territorial_boundary**, **sosi_objtype_turn_connecting_segment** }

- *List of SOSI OBJTYPEs.* enum **JunctionPoint** { **sosi_junction_node**, **sosi_junction_connection**, **sosi_junction_open_end** }

## *Default SOSI junction point layer types.* Functions

- **CoordSys sysCodeToCoordSys** (int sysCode)
  *Convert SOSI SYSKODE value to coordinate system data.*
- **ElementType sosiNameToType** (std::string sosiElementName)
  *Convert SOSI element names to ElementType enum value.*
- **ObjType sosiObjNameToType** (std::string sosiObjTypeName)
  *Convert SOSI objtype names to ObjType enum value.*
- void **deleteNorthEasts** (**NorthEastList** &lst)
  *Deletes **SosiNorthEast** elements of NorthEastList.*

## Detailed Description

SOSI.

## Typedef Documentation

### typedef std::vector<SosiNorthEast*> sosicon::sosi::NorthEastList

List of SosiSNorthEast elements.

Definition at line 115 of file sosi_north_east.h.

## Function Documentation

### void sosicon::sosi::deleteNorthEasts (NorthEastList & *lst*)

Deletes **SosiNorthEast** elements of NorthEastList.

Definition at line 21 of file sosi_north_east.cpp.

# sosicon::utils Namespace Reference

String manipulation routines.

## Functions

- std::string **className2FileName** (const std::string &className)
  *Converts Class name to file name string.*
- bool **fileExists** (const std::string &name)
  *Test if file exists.*
- std::string **normalizeAppClassName** (const std::string &className)
  *Asserts correct name of application classes.*
- std::string **repeat** (const std::string &seq, unsigned int count)
  *Repeat string N times.*
- std::string **replaceAll** (const std::string &from, const std::string &to, const std::string &subject)
  *Replace all occurences of one string witn another.*
- std::string **trim** (const std::string &str)
  *Removes leading and trailing space characters.*
- std::string **trimLeft** (const std::string &str)
- std::string **trimRight** (const std::string &str)
- std::string **toLower** (const std::string &from)
- std::string **ucFirst** (const std::string &str)
- void **getPathInfo** (std::string path, std::string &dir, std::string &tit, std::string &ext)

## Detailed Description

String manipulation routines.

## Function Documentation

### string sosicon::utils::className2FileName (const std::string & *className*)

Converts Class name to file name string.

Class names are written in pascal case (i.e. 'CarmineEntity', 'XMLParser'). This method constructs a file name string for a given class name (i.e. 'carmine_entity', 'xml_parser'). The file names are always written in lower case, with underscores separating the words.

**Parameters:**

| | |
|---|---|
| *className* | The pascal-cased class name to convert to a file name. |

**Returns:**

The file name string without extension.

Definition at line 23 of file utils.cpp.

### bool sosicon::utils::fileExists (const std::string & *name*)`[inline]`

Test if file exists.

Definition at line 44 of file utils.h.

### void sosicon::utils::getPathInfo (std::string *path*, std::string & *dir*, std::string & *tit*, std::string & *ext*)

Definition at line 147 of file utils.cpp.

### string sosicon::utils::normalizeAppClassName (const std::string & *className*)

Asserts correct name of application classes.

Application classes should always begin with the 'App' prefix. This method adds the prefix to the provided class name if it is missing.

**Parameters:**

| | |
|---|---|
| *className* | The class name string to be resolved and normalized. |

**Returns:**

Normalized and corrected class name string.

Definition at line 44 of file utils.cpp.

### string sosicon::utils::repeat (const std::string & *seq*, unsigned int *count*)

Repeat string N times.

Creates a new string containing the provided string sequence for a predetermined number of repetitions.

**Parameters:**

| | |
|---|---|
| *seq* | Reference to the string to be repeated. |
| *count* | The numner of times to repeat the string sequence. |

**Returns:**

The result string.

Definition at line 64 of file utils.cpp.

### string sosicon::utils::replaceAll (const std::string & *from*, const std::string & *to*, const std::string & *subject*)

Replace all occurences of one string witn another.

Searches for a given string sequence, replacing all occurences by th provided substitution string.

**Parameters:**

| | |
|---|---|
| *from* | The string sequence to be changed. |
| *to* | The string to replace the 'from' sequence with. |
| *subject* | The string to perform the search on. |

**Returns:**

The new string, a copy of 'subject' where all occurences of 'from' are replaced with 'to'.

Definition at line 75 of file utils.cpp.

### string sosicon::utils::toLower (const std::string & *from*)

Definition at line 89 of file utils.cpp.

### string sosicon::utils::trim (const std::string & *str*)

Removes leading and trailing space characters.

Space characters in the beginning and at the end of the source string are trimmed.

**Parameters:**

| | |
|---|---|
| *str* | The target string. |

**Returns:**

A copy of the target string, without leading and/or trailing space characters.

Definition at line 106 of file utils.cpp.

### string sosicon::utils::trimLeft (const std::string & *str*)

Definition at line 112 of file utils.cpp.

### string sosicon::utils::trimRight (const std::string & *str*)

Definition at line 119 of file utils.cpp.

### string sosicon::utils::ucFirst (const std::string & *str*)

Definition at line 127 of file utils.cpp.

# Class Documentation

## sosicon::CommandLine Class Reference

Command-line parser.

```
#include <command_line.h>
```

### Public Member Functions

- void **outputHelpText** ()
  *Display help text.*
- void **parse** (int argc, char *argv[])
  *Read command-line arguments.*
- **CommandLine** ()
  *Constructor.*
- virtual **~CommandLine** ()
  *Destructor.*

### Public Attributes

- std::string **mCommand**
  *Conversion command.*
- std::vector< std::string > **mSourceFiles**
  *List of input files.*
- std::vector< std::string > **mObjTypes**
  *List of object types to output.*
- std::vector< std::string > **mGeomTypes**
  *List of geometry types to output.*
- std::vector< std::string > **mFieldSelection**
  *List of selected fields.*
- std::string **mDestinationDirectory**
  *Destination directory.*
- std::string **mOutputFile**
  *Destination file.*
- bool **mIsTtyIn**
  *TTY in flag.*
- bool **mIsTtyOut**
  *TTY out flag.*
- bool **mAppend**
  *Append flag.*
- bool **mIncludeHeader**
  *Include column headers.*
- int **mVerbose**
  *Verbose output.*

## Detailed Description

Command-line parser.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Takes the arguments from the command-line and parses them into the class member variables. On Linux systems, this class also reads piped content (file list to be processed) from stdin, using it as input parameters.

Definition at line 46 of file command_line.h.

## Constructor & Destructor Documentation

### sosicon::CommandLine::CommandLine ()

Constructor.

Definition at line 21 of file command_line.cpp.

### sosicon::CommandLine::~CommandLine () `[virtual]`

Destructor.

Definition at line 34 of file command_line.cpp.

## Member Function Documentation

### void sosicon::CommandLine::outputHelpText ()

Display help text.

Outputs simple help text to the command-line.

Definition at line 153 of file command_line.cpp.

### void sosicon::CommandLine::parse (int *argc*, char * *argv*[])

Read command-line arguments.

Parses the command-line arguments and loads the settings into the member variables. This function will also read piped content (file name list) from stdin on linux systems, adding it to the **CommandLine::mSourceFiles** list of files to be processed.

**Parameters:**

| | |
|---|---|
| *argc* | Number of arguments present. Passed on from **main()** function. |
| *argv* | Array of string pointers to each argument. Passed on from **main()** function. |

Definition at line 43 of file command_line.cpp.

## Member Data Documentation

### bool sosicon::CommandLine::mAppend

Append flag.

If the destination file (-o ...) is specified together with the -a argument, this flag will be true to signal that data from several source files should be merged into one destination file.

Definition at line 121 of file command_line.h.

### std::string sosicon::CommandLine::mCommand

Conversion command.

Specifies what type of conversion to perform. If this string is -2tsv, the SOSI file will be exported as tab separated values. The factory class uses this parameter to determine which **IConverter** implementation to employ upon initialization.

Definition at line 55 of file command_line.h.

### std::string sosicon::CommandLine::mDestinationDirectory

Destination directory.

Path to the target directory where the output files will be written. Specified by the -d argument.

Definition at line 95 of file command_line.h.

### std::vector<std::string> sosicon::CommandLine::mFieldSelection

List of selected fields.

String vector containing the identifiers for the SOSI fields to be included in the export. Specified as a comma-separated list of strings following the -f argument.

Definition at line 88 of file command_line.h.

### std::vector<std::string> sosicon::CommandLine::mGeomTypes

List of geometry types to output.

String vector containing the geometry types for the elements to be included in the export. Relevant for shapefile exports, since shapefiles can only contain one geometry type at a time. Specified as a comma-separated list of strings following the -g argument. The converter will output one shapefile for each selected geometry.

Definition at line 81 of file command_line.h.

### bool sosicon::CommandLine::mIncludeHeader

Include column headers.

For some output formats, such as tsv, this flag governs whether a line with the column header names should be included in the target file.

Definition at line 128 of file command_line.h.

### bool sosicon::CommandLine::mIsTtyIn

TTY in flag.

This flag is false if input is redirected (not a terminal window).

Definition at line 107 of file command_line.h.

### bool sosicon::CommandLine::mIsTtyOut

TTY out flag.

This flag is false if output is redirected (not a terminal window).

Definition at line 113 of file command_line.h.

### std::vector<std::string> sosicon::CommandLine::mObjTypes

List of object types to output.

String vector containing the SOSI OBJTYPE identifiers for the elements to be included in the export. Specified as a comma-separated list of strings following the -t argument.

Definition at line 72 of file command_line.h.

### std::string sosicon::CommandLine::mOutputFile

Destination file.

Specified by the -o argument. The target file name.

Definition at line 101 of file command_line.h.

### std::vector<std::string> sosicon::CommandLine::mSourceFiles

List of input files.

String vector containing the list of SOSI input files to be converted. This list is populated either by the file names specified directly on the command-line, or by the content of stdin as piped in from other commands (such as ls *.sos | ...) on Linux based systems.

Definition at line 64 of file command_line.h.

### int sosicon::CommandLine::mVerbose

Verbose output.

Verbose level. If this value is 0, no informative output will be emitted during file parsing. If the value is 1 (-v), limited output will be written to stdout - mostly file header information from each SOSI file to be converted. If the value i 2 (-V), a more comprehensive summary of every SOSI element in all source files will be output.

Definition at line 137 of file command_line.h.

---

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

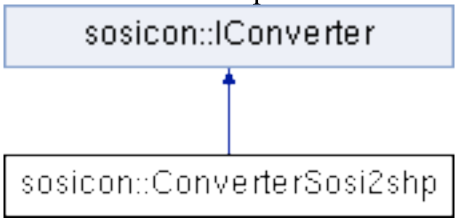
- /prosjekter/sosicon/src/**command_line.h**
- /prosjekter/sosicon/src/**command_line.cpp**

## sosicon::ConverterSosi2shp Class Reference

SOSI to ESRI Shape converter.

`#include <converter_sosi2shp.h>`

Inheritance diagram for sosicon::ConverterSosi2shp:



### Public Member Functions

- **ConverterSosi2shp** ()
  *Constructor.*
- virtual **~ConverterSosi2shp** ()
  *Destructor.*
- virtual void **init** (**CommandLine** *cmd)
  *Initialize converter.*
- virtual void **run** ()
  *Start conversion.*

### Private Member Functions

- template<typename T > void **writeFile** (**shape::Shapefile** &shp, std::string basePath, std::string extension)
  *Save specific shapefile part.*
- void **makeShp** (**ISosiElement** *sosiTree)
- std::string **makeBasePath** (std::string objTypeName)
  *Make base file path for destination files.*

### Private Attributes

- **CommandLine** * **mCmd**
  *Command line wrapper.*
- std::string **mCurrentSourcefile**
  *Souce file currently in process.*

### Detailed Description

SOSI to ESRI Shape converter.

If command-line parameter -2shp is specified, this converter will handle the output generation. Produces an ESRI Shape-file from SOSI source.

Definition at line 53 of file converter_sosi2shp.h.

## Constructor & Destructor Documentation

### sosicon::ConverterSosi2shp::ConverterSosi2shp () `[inline]`

Constructor.

Definition at line 110 of file converter_sosi2shp.h.

### virtual sosicon::ConverterSosi2shp::~ConverterSosi2shp () `[inline]`, `[virtual]`

Destructor.

Definition at line 113 of file converter_sosi2shp.h.

## Member Function Documentation

### virtual void sosicon::ConverterSosi2shp::init (CommandLine * *cmd*) `[inline]`, `[virtual]`

Initialize converter.

Implementation details in **sosicon::IConverter::init()**

**See also:**

**sosicon::IConverter::init()**

Implements **sosicon::IConverter** (*p.63*).

Definition at line 120 of file converter_sosi2shp.h.

### std::string sosicon::ConverterSosi2shp::makeBasePath (std::string *objTypeName*) `[private]`

Make base file path for destination files.

If the user specified an output file name, it will be used as a candidate for a base name to create shp, shx and dbf files for the shape export. Otherwise, the name of the first source file will be used by default.

This function checks if there are any name collisions, incrementing a postfixed number to the base name until a unique name is found.

**Returns:**

Modified, unique destination base name with directory (if provided), without file name extension.

Definition at line 101 of file converter_sosi2shp.cpp.

### void sosicon::ConverterSosi2shp::makeShp (ISosiElement * *sosiTree*) `[private]`

Definition at line 21 of file converter_sosi2shp.cpp.

### void sosicon::ConverterSosi2shp::run () `[virtual]`

Start conversion.

Implementation details in **sosicon::IConverter::run()**

**See also:**

**sosicon::IConverter::run()**

Implements **sosicon::IConverter** (*p.64*).

Definition at line 124 of file converter_sosi2shp.cpp.

### template<typename T > void sosicon::ConverterSosi2shp::writeFile (shape::Shapefile & *shp*, std::string *basePath*, std::string *extension*)`[inline]`, `[private]`

Save specific shapefile part.

The shapefile format consists of several files. Use corresponding interface to cast a instance of **IShapefile** to the correct file part for writing. The shapefile parts interfaces are:

- **IShapefileShpPart**
- **IShapefileShxPart**
- **IShapefileDbfPart**
- **IShapefilePrjPart**

**Parameters:**

| | |
|---|---|
| *shp* | Reference to the source ShapeFile instance. |
| *basePath* | Path and file title for the file to be written, without extension. |
| *extension* | additional file extensions to be appended before the main extension, which is one of the following:<br>• shp (shapefile part)<br>• shx (index part)<br>• dbf (attributes part)<br>• prj (projection part) |

Definition at line 76 of file converter_sosi2shp.h.

## Member Data Documentation

### CommandLine* sosicon::ConverterSosi2shp::mCmd`[private]`

Command line wrapper.

Definition at line 86 of file converter_sosi2shp.h.

### std::string sosicon::ConverterSosi2shp::mCurrentSourcefile`[private]`

Souce file currently in process.

Definition at line 89 of file converter_sosi2shp.h.

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

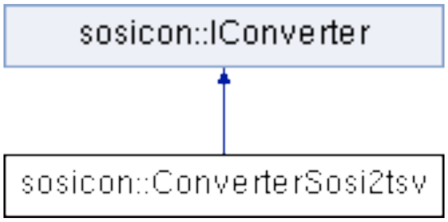
- /prosjekter/sosicon/src/**converter_sosi2shp.h**
- /prosjekter/sosicon/src/**converter_sosi2shp.cpp**

# sosicon::ConverterSosi2tsv Class Reference

SOSI to TSV converter.

`#include <converter_sosi2tsv.h>`

Inheritance diagram for sosicon::ConverterSosi2tsv:



## Public Member Functions

- **ConverterSosi2tsv** ()
  *Constructor.*
- virtual void **init** (**CommandLine** *cmd)
  *Initialize converter.*
- virtual void **run** ()
  *Start conversion.*

## Private Member Functions

- virtual **~ConverterSosi2tsv** ()
  *Destructor.*

## Private Attributes

- **CommandLine * mCmd**
  *Command line wrapper.*

## Detailed Description

SOSI to TSV converter.

If command-line parameter -2tsv is specified, this converter will handle the output generation. Produces a TSV file (tab separated values) SOSI source.

Definition at line 39 of file converter_sosi2tsv.h.

## Constructor & Destructor Documentation

### virtual sosicon::ConverterSosi2tsv::~ConverterSosi2tsv () `[inline]`, `[private]`, `[virtual]`

Destructor.

Definition at line 45 of file converter_sosi2tsv.h.

### sosicon::ConverterSosi2tsv::ConverterSosi2tsv ()`[inline]`

Constructor.

Definition at line 49 of file converter_sosi2tsv.h.

## Member Function Documentation

### virtual void sosicon::ConverterSosi2tsv::init (CommandLine * *cmd*)`[inline]`, `[virtual]`

Initialize converter.

Implementation details in **sosicon::IConverter::init()**

**See also:**

**sosicon::IConverter::init()**

Implements **sosicon::IConverter** (*p.63*).

Definition at line 56 of file converter_sosi2tsv.h.

### void sosicon::ConverterSosi2tsv::run ()`[virtual]`

Start conversion.

Implementation details in **sosicon::IConverter::run()**

**See also:**

**sosicon::IConverter::run()**

Implements **sosicon::IConverter** (*p.64*).

Definition at line 21 of file converter_sosi2tsv.cpp.

## Member Data Documentation

### CommandLine* sosicon::ConverterSosi2tsv::mCmd`[private]`

Command line wrapper.

Definition at line 42 of file converter_sosi2tsv.h.

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

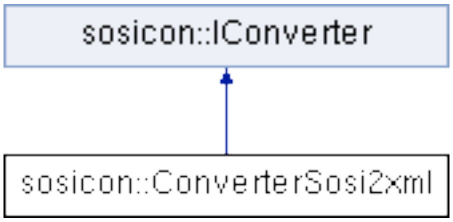
- /prosjekter/sosicon/src/**converter_sosi2tsv.h**
- /prosjekter/sosicon/src/**converter_sosi2tsv.cpp**

# sosicon::ConverterSosi2xml Class Reference

SOSI to ESRI Shape converter.

```
#include <converter_sosi2xml.h>
```

Inheritance diagram for sosicon::ConverterSosi2xml:



## Public Member Functions

- **ConverterSosi2xml** ()
  *Constructor.*
- virtual **~ConverterSosi2xml** ()
  *Destructor.*
- virtual void **init** (**CommandLine** *cmd)
  *Initialize converter.*
- virtual void **run** ()
  *Start conversion.*

## Private Member Functions

- void **makeXML** (**ISosiElement** *parent)

## Private Attributes

- **CommandLine** * **mCmd**
  *Command line wrapper.*

## Detailed Description

SOSI to ESRI Shape converter.

If command-line parameter -2xml is specified, this converter will handle the output generation. Produces an ESRI Shape-file from SOSI source.

Definition at line 49 of file converter_sosi2xml.h.

## Constructor & Destructor Documentation

### sosicon::ConverterSosi2xml::ConverterSosi2xml () `[inline]`

Constructor.

Definition at line 59 of file converter_sosi2xml.h.

### virtual sosicon::ConverterSosi2xml::~ConverterSosi2xml () `[inline]`, `[virtual]`

Destructor.

Definition at line 62 of file converter_sosi2xml.h.

## Member Function Documentation

### virtual void sosicon::ConverterSosi2xml::init (CommandLine * *cmd*) `[inline]`, `[virtual]`

Initialize converter.

Implementation details in **sosicon::IConverter::init()**

**See also:**

**sosicon::IConverter::init()**

Implements **sosicon::IConverter** (*p.63*).

Definition at line 69 of file converter_sosi2xml.h.

### void sosicon::ConverterSosi2xml::makeXML (ISosiElement * *parent*) `[private]`

Definition at line 21 of file converter_sosi2xml.cpp.

### void sosicon::ConverterSosi2xml::run () `[virtual]`

Start conversion.

Implementation details in **sosicon::IConverter::run()**

**See also:**

**sosicon::IConverter::run()**

Implements **sosicon::IConverter** (*p.64*).

Definition at line 26 of file converter_sosi2xml.cpp.

## Member Data Documentation

### CommandLine* sosicon::ConverterSosi2xml::mCmd `[private]`

Command line wrapper.

Definition at line 52 of file converter_sosi2xml.h.

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

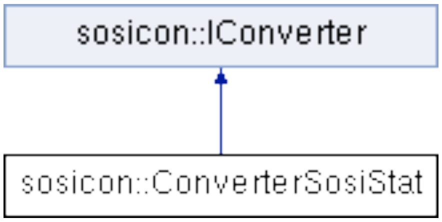
- /prosjekter/sosicon/src/**converter_sosi2xml.h**
- /prosjekter/sosicon/src/**converter_sosi2xml.cpp**

# sosicon::ConverterSosiStat Class Reference

SOSI to ESRI Shape converter.

```
#include <converter_sosi_stat.h>
```

Inheritance diagram for sosicon::ConverterSosiStat:



## Public Member Functions

- **ConverterSosiStat** ()
  *Constructor.*
- virtual **~ConverterSosiStat** ()
  *Destructor.*
- virtual void **init** (**CommandLine** *cmd)
  *Initialize converter.*
- virtual void **run** ()
  *Start conversion.*

## Private Member Functions

- void **printElementData** (**ISosiElement** *e, **sosi::SosiElementSearch** src, int padding)
  *Output simple element attributes.*
- void **printListContent** (std::map< std::string, int > list, int padding)
  *Output content of map<string,int>*
- void **printTableHeader** (std::string col1, std::string col2, int padding)
  *Output simple element attributes.*
- void **makeStat** (**ISosiElement** *parent)
  *Output table header with column titles.*

## Private Attributes

- **CommandLine * mCmd**
  *Command line wrapper.*
- std::map< std::string, int > **mObjTypes**
  *Map keeping count of objtypes.*
- std::map< std::string, int > **mGeoTypes**
  *Map keeping count of geometry.*

## Detailed Description

SOSI to ESRI Shape converter.

If command-line parameter -stat is specified, this converter will handle the output generation. Produces an ESRI Shape-file from SOSI source.

Definition at line 51 of file converter_sosi_stat.h.

## Constructor & Destructor Documentation

### sosicon::ConverterSosiStat::ConverterSosiStat () `[inline]`

Constructor.

Definition at line 77 of file converter_sosi_stat.h.

### virtual sosicon::ConverterSosiStat::~ConverterSosiStat () `[inline]`, `[virtual]`

Destructor.

Definition at line 80 of file converter_sosi_stat.h.

## Member Function Documentation

### virtual void sosicon::ConverterSosiStat::init (CommandLine * *cmd*) `[inline]`, `[virtual]`

Initialize converter.

Implementation details in **sosicon::IConverter::init()**

**See also:**

**sosicon::IConverter::init()**

Implements **sosicon::IConverter** (*p.63*).

Definition at line 87 of file converter_sosi_stat.h.

### void sosicon::ConverterSosiStat::makeStat (ISosiElement * *parent*) `[private]`

Output table header with column titles.

Definition at line 21 of file converter_sosi_stat.cpp.

### void sosicon::ConverterSosiStat::printElementData (ISosiElement * *e*, sosi::SosiElementSearch *src*, int *padding*) `[private]`

Output simple element attributes.

Definition at line 46 of file converter_sosi_stat.cpp.

### void sosicon::ConverterSosiStat::printListContent (std::map< std::string, int > *list*, int *padding*) `[private]`

Output content of map<string,int>

Definition at line 58 of file converter_sosi_stat.cpp.

### void sosicon::ConverterSosiStat::printTableHeader (std::string *col1*, std::string *col2*, int *padding*) `[private]`

Output simple element attributes.

Definition at line 70 of file converter_sosi_stat.cpp.

### void sosicon::ConverterSosiStat::run () `[virtual]`

Start conversion.

Implementation details in **sosicon::IConverter::run()**

**See also:**

**sosicon::IConverter::run()**

Implements **sosicon::IConverter** (*p.64*).

Definition at line 77 of file converter_sosi_stat.cpp.

## Member Data Documentation

### CommandLine* sosicon::ConverterSosiStat::mCmd `[private]`

Command line wrapper.

Definition at line 54 of file converter_sosi_stat.h.

### std::map<std::string, int> sosicon::ConverterSosiStat::mGeoTypes `[private]`

Map keeping count of geometry.

Definition at line 60 of file converter_sosi_stat.h.

### std::map<std::string, int> sosicon::ConverterSosiStat::mObjTypes `[private]`

Map keeping count of objtypes.

Definition at line 57 of file converter_sosi_stat.h.

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

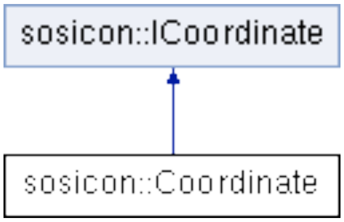
- /prosjekter/sosicon/src/**converter_sosi_stat.h**
- /prosjekter/sosicon/src/**converter_sosi_stat.cpp**

# sosicon::Coordinate Class Reference

**Coordinate** container.

```
#include <coordinate.h>
```

Inheritance diagram for sosicon::Coordinate:



## Public Member Functions

- virtual **~Coordinate** ()
- **Coordinate** ()
- virtual double **getE** ()
  *Get east coordinate.*
- virtual double **getN** ()
  *Get north coordinate.*
- virtual bool **leftOf** (**ICoordinate** *c)
  *Test if this coordinate is to the left of another.*
- virtual bool **rightOf** (**ICoordinate** *c)
  *Test if this coordinate is to the right of another.*
- virtual void **setE** (double coordEast)
  *Set east coordinate.*
- virtual void **setN** (double coordNorth)
  *Set north coordinate.*
- virtual void **setH** (double altitude)
  *Set altitude.*
- virtual void **shift** (int offsetN, int offsetE)
  *Shift coordinate by specified offset.*
- virtual void **divide** (int divisor)
  *Divide coordinate by specified divisor.*
- virtual std::string **toString** ()
  *Make string representation.*

## Private Attributes

- double **mEast**
- double **mNorth**
- double **mAltitude**

---

## Detailed Description

**Coordinate** container.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Stores a geographical position with some additional information.

Definition at line 36 of file coordinate.h.

## Constructor & Destructor Documentation

### virtual sosicon::Coordinate::~Coordinate ()`[inline]`, `[virtual]`

Definition at line 44 of file coordinate.h.

### sosicon::Coordinate::Coordinate ()`[inline]`

Definition at line 45 of file coordinate.h.

## Member Function Documentation

### virtual void sosicon::Coordinate::divide (int *divisor*)`[inline]`, `[virtual]`

Divide coordinate by specified divisor.

Implements **sosicon::ICoordinate** (*p.66*).

Definition at line 54 of file coordinate.h.

### virtual double sosicon::Coordinate::getE ()`[inline]`, `[virtual]`

Get east coordinate.

Implements **sosicon::ICoordinate** (*p.66*).

Definition at line 46 of file coordinate.h.

### virtual double sosicon::Coordinate::getN ()`[inline]`, `[virtual]`

Get north coordinate.

Implements **sosicon::ICoordinate** (*p.66*).

Definition at line 47 of file coordinate.h.

### virtual bool sosicon::Coordinate::leftOf (ICoordinate * *c*)`[inline]`, `[virtual]`

Test if this coordinate is to the left of another.

Implements **sosicon::ICoordinate** (*p.66*).

Definition at line 48 of file coordinate.h.

### virtual bool sosicon::Coordinate::rightOf (ICoordinate * *c*)`[inline]`, `[virtual]`

Test if this coordinate is to the right of another.

Implements **sosicon::ICoordinate** (*p.66*).

Definition at line 49 of file coordinate.h.

### virtual void sosicon::Coordinate::setE (double *coordEast*)`[inline]`, `[virtual]`

Set east coordinate.

Implements **sosicon::ICoordinate** (*p.66*).

Definition at line 50 of file coordinate.h.

### virtual void sosicon::Coordinate::setH (double *altitude*)`[inline]`, `[virtual]`

Set altitude.

Implements **sosicon::ICoordinate** (*p.67*).

Definition at line 52 of file coordinate.h.

### virtual void sosicon::Coordinate::setN (double *coordNorth*)`[inline]`, `[virtual]`

Set north coordinate.

Implements **sosicon::ICoordinate** (*p.67*).

Definition at line 51 of file coordinate.h.

### virtual void sosicon::Coordinate::shift (int *offsetN*, int *offsetE*)`[inline]`, `[virtual]`

Shift coordinate by specified offset.

Implements **sosicon::ICoordinate** (*p.67*).

Definition at line 53 of file coordinate.h.

### virtual std::string sosicon::Coordinate::toString ()`[inline]`, `[virtual]`

Make string representation.

Implements **sosicon::ICoordinate** (*p.67*).

Definition at line 55 of file coordinate.h.

## Member Data Documentation

### double sosicon::Coordinate::mAltitude`[private]`

Definition at line 40 of file coordinate.h.

### double sosicon::Coordinate::mEast`[private]`

Definition at line 38 of file coordinate.h.

### double sosicon::Coordinate::mNorth`[private]`

Definition at line 39 of file coordinate.h.

---

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

- /prosjekter/sosicon/src/**coordinate.h**

# sosicon::CoordinateCollection Class Reference

**Coordinate** container.

`#include <coordinate_collection.h>`

## Public Member Functions

- virtual **~CoordinateCollection** ()
  *Destructor.*
- **CoordinateCollection** ()
  *Constructor.*
- void **free** ()
  *Free allocated memory.*
- void **discoverCoords** (**ISosiElement** *sosi)
- bool **getNextInGeom** (**ICoordinate** *&coord)
- std::vector< **ICoordinate** * > & **getGeom** ()
- std::vector< int > & **getGeomSizes** ()
- int **getNumPointsGeom** ()
- int **getNumPartsGeom** ()
- std::vector< **ICoordinate** * > & **getHoles** ()
- std::vector< int > & **getHoleSizes** ()
- int **getNumPointsHoles** ()
- int **getNumPartsHoles** ()
- double **getXmin** ()
- double **getYmin** ()
- double **getXmax** ()
- double **getYmax** ()

## Private Member Functions

- void **extractPath** (**ISosiElement** *referencedElement, bool reverse, int &numPoints, **sosi::NorthEastList** &target)
  *Get coordinate values from SOSI element.*
- bool **isClockwise** (std::vector< **ICoordinate** * >::iterator &begin, std::vector< **ICoordinate** * >::iterator &end)
- bool **isCounterClockwise** (std::vector< **ICoordinate** * >::iterator begin, std::vector< **ICoordinate** * >::iterator end)

## Private Attributes

- **sosi::NorthEastList mGeom**
  *Stores collection of pointers to coordinates for geometries.*
- std::vector< **ICoordinate** * > **mGeomNormalized**
- **sosi::NorthEastList mHoles**
- std::vector< **ICoordinate** * > **mHolesNormalized**
- int **mNumPartsGeom**
- int **mNumPartsHoles**
- int **mNumPointsGeom**
- int **mNumPointsHoles**
- std::vector< int > **mGeomSizes**
- std::vector< int > **mHoleSizes**
- sosi::NorthEastList::iterator **mGeomIndex**

- double **mXmin**
- double **mYmin**
- double **mXmax**
- double **mYmax**

## Detailed Description

**Coordinate** container.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Stores a collection of geographical positions.

Definition at line 50 of file coordinate_collection.h.

## Constructor & Destructor Documentation

### sosicon::CoordinateCollection::~CoordinateCollection () `[virtual]`

Destructor.

Definition at line 55 of file coordinate_collection.cpp.

### sosicon::CoordinateCollection::CoordinateCollection () `[inline]`

Constructor.

Definition at line 90 of file coordinate_collection.h.

## Member Function Documentation

### void sosicon::CoordinateCollection::discoverCoords (ISosiElement * *sosi*)

Definition at line 66 of file coordinate_collection.cpp.

### void sosicon::CoordinateCollection::extractPath (ISosiElement * *referencedElement*, bool *reverse*, int & *numPoints*, sosi::NorthEastList & *target*) `[private]`

Get coordinate values from SOSI element.

Definition at line 132 of file coordinate_collection.cpp.

### void sosicon::CoordinateCollection::free ()

Free allocated memory.

Definition at line 60 of file coordinate_collection.cpp.

### std::vector< sosicon::ICoordinate * > & sosicon::CoordinateCollection::getGeom ()

Definition at line 163 of file coordinate_collection.cpp.

### std::vector<int>& sosicon::CoordinateCollection::getGeomSizes () `[inline]`

Definition at line 107 of file coordinate_collection.h.

### std::vector< sosicon::ICoordinate * > & sosicon::CoordinateCollection::getHoles ()

Definition at line 181 of file coordinate_collection.cpp.

### std::vector<int>& sosicon::CoordinateCollection::getHoleSizes () `[inline]`

Definition at line 112 of file coordinate_collection.h.

### bool sosicon::CoordinateCollection::getNextInGeom (ICoordinate *& *coord*)

Definition at line 210 of file coordinate_collection.cpp.

### int sosicon::CoordinateCollection::getNumPartsGeom () `[inline]`

Definition at line 109 of file coordinate_collection.h.

### int sosicon::CoordinateCollection::getNumPartsHoles () `[inline]`

Definition at line 114 of file coordinate_collection.h.

### int sosicon::CoordinateCollection::getNumPointsGeom () `[inline]`

Definition at line 108 of file coordinate_collection.h.

### int sosicon::CoordinateCollection::getNumPointsHoles () `[inline]`

Definition at line 113 of file coordinate_collection.h.

### double sosicon::CoordinateCollection::getXmax () `[inline]`

Definition at line 120 of file coordinate_collection.h.

### double sosicon::CoordinateCollection::getXmin () `[inline]`

Definition at line 116 of file coordinate_collection.h.

### double sosicon::CoordinateCollection::getYmax () `[inline]`

Definition at line 122 of file coordinate_collection.h.

### double sosicon::CoordinateCollection::getYmin () `[inline]`

Definition at line 118 of file coordinate_collection.h.

### bool sosicon::CoordinateCollection::isClockwise (std::vector< ICoordinate * >::iterator & *begin*, std::vector< ICoordinate * >::iterator & *end*) `[private]`

Definition at line 215 of file coordinate_collection.cpp.

### bool sosicon::CoordinateCollection::isCounterClockwise (std::vector< ICoordinate * >::iterator *begin*, std::vector< ICoordinate * >::iterator *end*) `[inline]`, `[private]`

Definition at line 82 of file coordinate_collection.h.

## Member Data Documentation

### sosi::NorthEastList sosicon::CoordinateCollection::mGeom `[private]`

Stores collection of pointers to coordinates for geometries.

Definition at line 53 of file coordinate_collection.h.

### sosi::NorthEastList::iterator sosicon::CoordinateCollection::mGeomIndex `[private]`

Definition at line 68 of file coordinate_collection.h.

### std::vector<ICoordinate*> sosicon::CoordinateCollection::mGeomNormalized `[private]`

Definition at line 54 of file coordinate_collection.h.

### std::vector<int> sosicon::CoordinateCollection::mGeomSizes `[private]`

Definition at line 65 of file coordinate_collection.h.

### sosi::NorthEastList sosicon::CoordinateCollection::mHoles `[private]`

Definition at line 56 of file coordinate_collection.h.

### std::vector<int> sosicon::CoordinateCollection::mHoleSizes `[private]`

Definition at line 66 of file coordinate_collection.h.

### std::vector<ICoordinate*> sosicon::CoordinateCollection::mHolesNormalized `[private]`

Definition at line 57 of file coordinate_collection.h.

### int sosicon::CoordinateCollection::mNumPartsGeom `[private]`

Definition at line 59 of file coordinate_collection.h.

### int sosicon::CoordinateCollection::mNumPartsHoles `[private]`

Definition at line 60 of file coordinate_collection.h.

### int sosicon::CoordinateCollection::mNumPointsGeom `[private]`

Definition at line 62 of file coordinate_collection.h.

### int sosicon::CoordinateCollection::mNumPointsHoles `[private]`

Definition at line 63 of file coordinate_collection.h.

### double sosicon::CoordinateCollection::mXmax `[private]`

Definition at line 72 of file coordinate_collection.h.

### double sosicon::CoordinateCollection::mXmin `[private]`

Definition at line 70 of file coordinate_collection.h.

### double sosicon::CoordinateCollection::mYmax `[private]`

Definition at line 73 of file coordinate_collection.h.

### double sosicon::CoordinateCollection::mYmin `[private]`

Definition at line 71 of file coordinate_collection.h.

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

- /prosjekter/sosicon/src/**coordinate_collection.h**
- /prosjekter/sosicon/src/**coordinate_collection.cpp**

# sosicon::sosi::CoordSys Class Reference

SOSI coordinate system.

```
#include <sosi_types.h>
```

## Public Member Functions

- **CoordSys** ()
- **CoordSys** (int sysCode, std::string displayString, std::string **prjString**)
- std::string **prjString** ()
- bool **valid** ()

## Private Attributes

- int **mSysCode**
  *SOSI SYSKODE.*
- std::string **mPrjString**
  *Projection string.*
- std::string **mDisplayString**
  *Display string.*

## Detailed Description

SOSI coordinate system.

Definition at line 137 of file sosi_types.h.

## Constructor & Destructor Documentation

### sosicon::sosi::CoordSys::CoordSys () `[inline]`

Definition at line 145 of file sosi_types.h.

### sosicon::sosi::CoordSys::CoordSys (int *sysCode*, std::string *displayString*, std::string *prjString*) `[inline]`

Definition at line 147 of file sosi_types.h.

## Member Function Documentation

### std::string sosicon::sosi::CoordSys::prjString () `[inline]`

Definition at line 153 of file sosi_types.h.

### bool sosicon::sosi::CoordSys::valid () `[inline]`

Definition at line 155 of file sosi_types.h.

## Member Data Documentation

### std::string sosicon::sosi::CoordSys::mDisplayString `[private]`

Display string.

Definition at line 141 of file sosi_types.h.

### std::string sosicon::sosi::CoordSys::mPrjString `[private]`

Projection string.

Definition at line 140 of file sosi_types.h.

### int sosicon::sosi::CoordSys::mSysCode `[private]`

SOSI SYSKODE.

Definition at line 139 of file sosi_types.h.

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

- /prosjekter/sosicon/src/sosi/**sosi_types.h**

# sosicon::shape::DoubleField Union Reference

32 bit double / byte field

```
#include <shapefile_types.h>
```

## Public Attributes

- double **d**
- char **b** [sizeof(double)]

## Detailed Description

32 bit double / byte field

Definition at line 78 of file shapefile_types.h.

## Member Data Documentation

### char sosicon::shape::DoubleField::b[sizeof(double)]

Definition at line 80 of file shapefile_types.h.

### double sosicon::shape::DoubleField::d

Definition at line 79 of file shapefile_types.h.

**The documentation for this union was generated from the following file:**

- /prosjekter/sosicon/src/shape/**shapefile_types.h**

## sosicon::Factory Class Reference

**Factory** class.

`#include <factory.h>`

### Static Public Member Functions

- static void **get** (**IConverter** *&converter, **CommandLine** *cmd)
  *Retrieve converter.*
- static void **release** (**IConverter** *&converter)
  *Releases converter.*

---

### Detailed Description

**Factory** class.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Responsible for creating an appropriate **IConverter** implementation instance. Uses the command-line arguments to decide what type of converter is required.

Definition at line 37 of file factory.h.

---

### Member Function Documentation

#### void sosicon::Factory::get (sosicon::IConverter *& *converter*, sosicon::CommandLine * *cmd*)`[static]`

Retrieve converter.

Selects an **IConverter** implementation and creates an object of that class based on the user's command-line arguments.

**Note:**

Any object delivered with **Factory::get()** must be freed by calling **Factory::release()**

**See also:**

**Factory::release()**

**Parameters:**

| | |
|---|---|
| *converter* | Reference to the pointer to receive the new **IConverter**. |
| *cmd* | **CommandLine** object with current command-line arguments. |

Definition at line 21 of file factory.cpp.

#### void sosicon::Factory::release (sosicon::IConverter *& *converter*)`[static]`

Releases converter.

Frees allocated resources and releases **IConverter** object. Any object retrieved from **Factory::get()** must be disposed of through this function.

**Parameters:**

| | |
|---|---|
| *converter* | Reference to a pointer containing the **IConverter** instance to be released. The pointer will be reset to 0 after object deletion. |

Definition at line 41 of file factory.cpp.

---

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

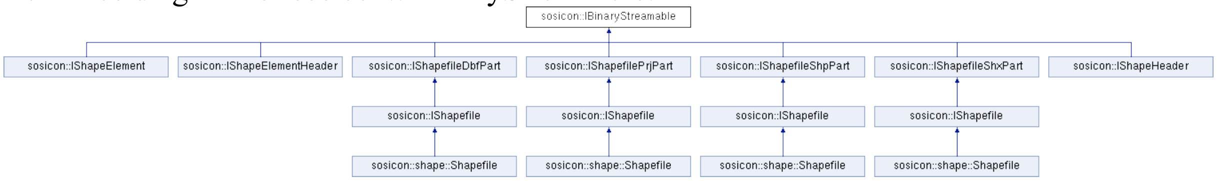
- /prosjekter/sosicon/src/**factory.h**
- /prosjekter/sosicon/src/**factory.cpp**

# sosicon::IBinaryStreamable Class Reference

Interface: Binary streamable object.

`#include <i_binary_streamable.h>`

Inheritance diagram for sosicon::IBinaryStreamable:


sosicon::IBinaryStreamable
sosicon::IShapeElement
sosicon::IShapeElementHeader
sosicon::IShapefileDbfPart
sosicon::IShapefilePrjPart
sosicon::IShapefileShpPart
sosicon::IShapefileShxPart
sosicon::IShapeHeader
sosicon::IShapefile
sosicon::IShapefile
sosicon::IShapefile
sosicon::IShapefile
sosicon::shape::Shapefile
sosicon::shape::Shapefile
sosicon::shape::Shapefile
sosicon::shape::Shapefile

## Public Member Functions

- virtual **~IBinaryStreamable** ()
  *Destructor.*
- virtual void **writeBinary** (std::ostream &os)=0
  *Writes binary data to output stream.*

---

## Detailed Description

Interface: Binary streamable object.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 34 of file i_binary_streamable.h.

---

## Constructor & Destructor Documentation

### virtual sosicon::IBinaryStreamable::~IBinaryStreamable () `[inline]`, `[virtual]`

Destructor.

Definition at line 38 of file i_binary_streamable.h.

---

## Member Function Documentation

### virtual void sosicon::IBinaryStreamable::writeBinary (std::ostream & *os*) `[pure virtual]`

Writes binary data to output stream.

Invoked when the operator<< is used to strem to an std::ostream object. Binary write operation is performed for current impementation here.

**Parameters:**

| | |
|---|---|
| *os* | Target stream object. |

Implemented in **sosicon::IShapefileDbfPart** (*p.84*), **sosicon::IShapefilePrjPart** (*p.86*), **sosicon::IShapefileShpPart** (*p.88*), and **sosicon::IShapefileShxPart** (*p.90*).

---

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

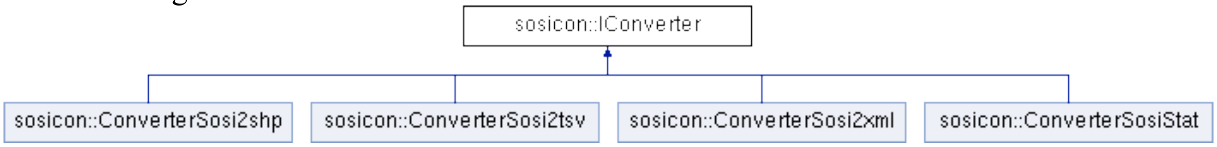
- /prosjekter/sosicon/src/interface/**i_binary_streamable.h**

# sosicon::IConverter Class Reference

Interface: Converter.

`#include <i_converter.h>`

Inheritance diagram for sosicon::IConverter:


sosicon::IConverter
sosicon::ConverterSosi2shp
sosicon::ConverterSosi2tsv
sosicon::ConverterSosi2xml
sosicon::ConverterSosiStat

## Public Member Functions

- virtual **~IConverter** ()
  *Destructor.*
- virtual void **init** (**CommandLine** *cmd)=0
  *Initialize converter.*
- virtual void **run** ()=0
  *Start conversion.*

## Detailed Description

Interface: Converter.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Represents the generic form of a converter. The factory class is responsible for creating a converter based upon input parameters. The returned object is then interacted on through this interface.

Definition at line 38 of file i_converter.h.

## Constructor & Destructor Documentation

### virtual sosicon::IConverter::~IConverter () `[inline]`, `[virtual]`

Destructor.

Definition at line 43 of file i_converter.h.

## Member Function Documentation

### virtual void sosicon::IConverter::init (CommandLine * *cmd*) `[pure virtual]`

Initialize converter.

Conversion setup. Uses the **CommandLine** to determine what operations to perform.

**Parameters:**

| | |
|---|---|
| *cmd* | Arguments from the command-line parsed and ready within a **sosicon::CommandLine** object. |

Implemented in **sosicon::ConverterSosi2shp** (*p.36*), **sosicon::ConverterSosiStat** (*p.44*), **sosicon::ConverterSosi2xml** (*p.42*), and **sosicon::ConverterSosi2tsv** (*p.40*).

### virtual void sosicon::IConverter::run ()`[pure virtual]`

Start conversion.

Run the conversion routine. Outputs the destination file accrding to the preferences given from the command-line.

Implemented in **sosicon::ConverterSosi2shp** (*p.36*), **sosicon::ConverterSosiStat** (*p.45*), **sosicon::ConverterSosi2xml** (*p.42*), and **sosicon::ConverterSosi2tsv** (*p.40*).

---

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

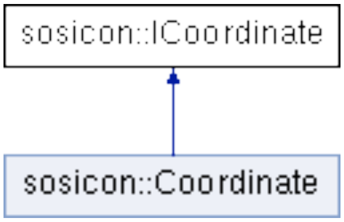
- /prosjekter/sosicon/src/interface/**i_converter.h**

# sosicon::ICoordinate Class Reference

Interface: **Coordinate**.

```
#include <i_coordinate.h>
```

Inheritance diagram for sosicon::ICoordinate:



## Public Member Functions

- virtual **~ICoordinate** ()
  *Destructor.*
- virtual double **getE** ()=0
  *Get east coordinate.*
- virtual double **getN** ()=0
  *Get north coordinate.*
- virtual bool **leftOf** (**ICoordinate** *c)=0
  *Test if this coordinate is to the left of another.*
- virtual bool **rightOf** (**ICoordinate** *c)=0
  *Test if this coordinate is to the right of another.*
- virtual void **setE** (double coordEast)=0
  *Set east coordinate.*
- virtual void **setN** (double coordNorth)=0
  *Set north coordinate.*
- virtual void **setH** (double altitude)=0
  *Set altitude.*
- virtual void **shift** (int offsetN, int offsetE)=0
  *Shift coordinate by specified offset.*
- virtual void **divide** (int divisor)=0
  *Divide coordinate by specified divisor.*
- virtual std::string **toString** ()=0
  *Make string representation.*

---

## Detailed Description

Interface: **Coordinate**.

### Author:

Espen Andersen

### Copyright:

GNU General Public License

Definition at line 35 of file i_coordinate.h.

## Constructor & Destructor Documentation

### virtual sosicon::ICoordinate::~ICoordinate () `[inline]`, `[virtual]`

Destructor.

Definition at line 39 of file i_coordinate.h.

## Member Function Documentation

### virtual void sosicon::ICoordinate::divide (int *divisor*) `[pure virtual]`

Divide coordinate by specified divisor.

Implemented in **sosicon::Coordinate** (*p.47*).

### virtual double sosicon::ICoordinate::getE () `[pure virtual]`

Get east coordinate.

Implemented in **sosicon::Coordinate** (*p.47*).

### virtual double sosicon::ICoordinate::getN () `[pure virtual]`

Get north coordinate.

Implemented in **sosicon::Coordinate** (*p.47*).

### virtual bool sosicon::ICoordinate::leftOf (ICoordinate * *c*) `[pure virtual]`

Test if this coordinate is to the left of another.

Implemented in **sosicon::Coordinate** (*p.47*).

### virtual bool sosicon::ICoordinate::rightOf (ICoordinate * *c*) `[pure virtual]`

Test if this coordinate is to the right of another.

Implemented in **sosicon::Coordinate** (*p.48*).

### virtual void sosicon::ICoordinate::setE (double *coordEast*) `[pure virtual]`

Set east coordinate.

Implemented in **sosicon::Coordinate** (*p.48*).

### virtual void sosicon::ICoordinate::setH (double *altitude*)`[pure virtual]`

Set altitude.

Implemented in **sosicon::Coordinate** (*p.48*).

### virtual void sosicon::ICoordinate::setN (double *coordNorth*)`[pure virtual]`

Set north coordinate.

Implemented in **sosicon::Coordinate** (*p.48*).

### virtual void sosicon::ICoordinate::shift (int *offsetN*, int *offsetE*)`[pure virtual]`

Shift coordinate by specified offset.

Implemented in **sosicon::Coordinate** (*p.48*).

### virtual std::string sosicon::ICoordinate::toString ()`[pure virtual]`

Make string representation.

Implemented in **sosicon::Coordinate** (*p.48*).

---

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

- /prosjekter/sosicon/src/interface/**i_coordinate.h**

# sosicon::ILookupTable Class Reference

Interface: Lookup table.

`#include <i_lookup_table.h>`

## Public Member Functions

- virtual **~ILookupTable** ()
  *Destructor.*
- virtual std::string **toString** ()=0
  *Print string representation of current table.*
- virtual **ISosiElement** * **get** (std::string key)=0
  *Get SOSI element by ref ID.*

## Detailed Description

Interface: Lookup table.

**Author:**

Espen Andersen

**Copyright:**

Interface to a lookup table, implemented by ReferenceLookup class.

Definition at line 40 of file i_lookup_table.h.

## Constructor & Destructor Documentation

### virtual sosicon::ILookupTable::~ILookupTable ()`[inline]`, `[virtual]`

Destructor.

Definition at line 44 of file i_lookup_table.h.

## Member Function Documentation

### virtual ISosiElement* sosicon::ILookupTable::get (std::string *key*)`[pure virtual]`

Get SOSI element by ref ID.

### virtual std::string sosicon::ILookupTable::toString ()`[pure virtual]`

Print string representation of current table.

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

- /prosjekter/sosicon/src/interface/**i_lookup_table.h**

# imaxdiv_t Struct Reference

```
#include <inttypes.h>
```

## Public Attributes

- intmax_t **quot**
- intmax_t **rem**

## Detailed Description

Definition at line 47 of file inttypes.h.

## Member Data Documentation

### intmax_t imaxdiv_t::quot

Definition at line 48 of file inttypes.h.

### intmax_t imaxdiv_t::rem

Definition at line 49 of file inttypes.h.

### The documentation for this struct was generated from the following file:

- /prosjekter/sosicon/src/**inttypes.h**

# sosicon::shape::Int16Field Union Reference

16 bit integer / byte field

```
#include <shapefile_types.h>
```

## Public Attributes

- uint16_t **i**
- char **b** [sizeof(uint16_t)]

## Detailed Description

16 bit integer / byte field

Definition at line 59 of file shapefile_types.h.

## Member Data Documentation

### char sosicon::shape::Int16Field::b[sizeof(uint16_t)]

Definition at line 61 of file shapefile_types.h.

### uint16_t sosicon::shape::Int16Field::i

Definition at line 60 of file shapefile_types.h.

**The documentation for this union was generated from the following file:**

- /prosjekter/sosicon/src/shape/**shapefile_types.h**

# sosicon::shape::Int32Field Union Reference

32 bit integer / byte field

```
#include <shapefile_types.h>
```

## Public Attributes

- uint32_t **i**
- char **b** [sizeof(uint32_t)]

## Detailed Description

32 bit integer / byte field

Definition at line 65 of file shapefile_types.h.

## Member Data Documentation

### char sosicon::shape::Int32Field::b[sizeof(uint32_t)]

Definition at line 67 of file shapefile_types.h.

### uint32_t sosicon::shape::Int32Field::i

Definition at line 66 of file shapefile_types.h.

**The documentation for this union was generated from the following file:**

- /prosjekter/sosicon/src/shape/**shapefile_types.h**

# sosicon::shape::Int32TField Union Reference

32 bit integer / byte / geom::ShapeType field

`#include <shapefile_types.h>`

## Public Attributes

- uint32_t **i**
- char **b** [sizeof(uint32_t)]
- **ShapeType t**

## Detailed Description

32 bit integer / byte / geom::ShapeType field

Definition at line 71 of file shapefile_types.h.

## Member Data Documentation

### char sosicon::shape::Int32TField::b[sizeof(uint32_t)]

Definition at line 73 of file shapefile_types.h.

### uint32_t sosicon::shape::Int32TField::i

Definition at line 72 of file shapefile_types.h.

### ShapeType sosicon::shape::Int32TField::t

Definition at line 74 of file shapefile_types.h.

**The documentation for this union was generated from the following file:**

- /prosjekter/sosicon/src/shape/**shapefile_types.h**

# sosicon::shape::Int8Field Union Reference

8 bit integer / byte field

```
#include <shapefile_types.h>
```

## Public Attributes

- uint8_t **i**
- char **b** [sizeof(uint8_t)]

## Detailed Description

8 bit integer / byte field

Definition at line 53 of file shapefile_types.h.

## Member Data Documentation

### char sosicon::shape::Int8Field::b[sizeof(uint8_t)]

Definition at line 55 of file shapefile_types.h.

### uint8_t sosicon::shape::Int8Field::i

Definition at line 54 of file shapefile_types.h.

**The documentation for this union was generated from the following file:**

- /prosjekter/sosicon/src/shape/**shapefile_types.h**

# sosicon::IRectangle Class Reference

Interface: Rectangle.

`#include <i_rectangle.h>`

## Public Member Functions

- virtual **~IRectangle** ()
  *Destructor.*
- virtual double **left** ()=0
  *Get left position.*
- virtual void **left** (double val)=0
  *Set left position.*
- virtual double **top** ()=0
  *Get top position.*
- virtual void **top** (double val)=0
  *Set top position.*
- virtual double **right** ()=0
  *Get right position.*
- virtual void **right** (double val)=0
  *Set right position.*
- virtual double **bottom** ()=0
  *Get bottom position.*
- virtual void **bottom** (double val)=0
  *Set bottom position.*

## Detailed Description

Interface: Rectangle.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 33 of file i_rectangle.h.

## Constructor & Destructor Documentation

### virtual sosicon::IRectangle::~IRectangle () `[inline]`, `[virtual]`

Destructor.

Definition at line 37 of file i_rectangle.h.

## Member Function Documentation

### virtual double sosicon::IRectangle::bottom ()`[pure virtual]`

Get bottom position.

Returns the bottom (y1) coordinate of current rectangle.

**Returns:**

Bottom/y1 position.

### virtual void sosicon::IRectangle::bottom (double *val*)`[pure virtual]`

Set bottom position.

Sets the bottom (y1) coordinate of current rectangle.

**Parameters:**

| | |
|---|---|
| *val* | The new bottom/y1 position. |

### virtual double sosicon::IRectangle::left ()`[pure virtual]`

Get left position.

Returns the left (x0) coordinate of current rectangle.

**Returns:**

Left/x0 position.

### virtual void sosicon::IRectangle::left (double *val*)`[pure virtual]`

Set left position.

Sets the left (x0) coordinate of current rectangle.

**Parameters:**

| | |
|---|---|
| *val* | The new left/x0 position. |

### virtual double sosicon::IRectangle::right ()`[pure virtual]`

Get right position.

Returns the right (x1) coordinate of current rectangle.

**Returns:**

Right/x1 position.

### virtual void sosicon::IRectangle::right (double *val*)`[pure virtual]`

Set right position.

Sets the right (x1) coordinate of current rectangle.

**Parameters:**

| | |
|---|---|
| *val* | The new right/x1 position. |

### virtual double sosicon::IRectangle::top () `[pure virtual]`

Get top position.

Returns the top (y0) coordinate of current rectangle.

**Returns:**

Top/y0 position.

### virtual void sosicon::IRectangle::top (double *val*) `[pure virtual]`

Set top position.

Sets the top (y0) coordinate of current rectangle.

**Parameters:**

| | |
|---|---|
| *val* | The new top/y0 position. |

---

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

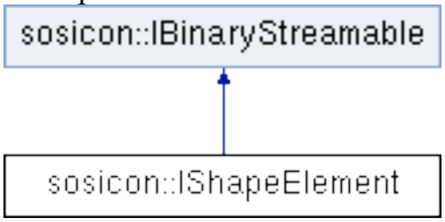
- /prosjekter/sosicon/src/interface/**i_rectangle.h**

# sosicon::IShapeElement Class Reference

Interface: Shape element.

`#include <i_shape_element.h>`

Inheritance diagram for sosicon::IShapeElement:



## Public Member Functions

- virtual **~IShapeElement** ()
  *Destructor.*
- virtual bool **populate** (**ISosiElement** *sosiElement)=0
  *Create from SOSI element.*
- virtual **ISosiElement** * **getSosiElement** ()=0
  *Get original SOSI element.*
- virtual void **getMBR** (**IRectangle** &rect)=0
  *Get minimum bounding rectangle.*
- virtual int **getWordSize** ()=0
  *Get element size in 16-bit words.*
- virtual int **getByteSize** ()=0
  *Get element size in bytes.*

## Detailed Description

Interface: Shape element.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 37 of file i_shape_element.h.

## Constructor & Destructor Documentation

### virtual sosicon::IShapeElement::~IShapeElement () `[inline]`, `[virtual]`

Destructor.

Definition at line 41 of file i_shape_element.h.

## Member Function Documentation

### virtual int sosicon::IShapeElement::getByteSize () `[pure virtual]`

Get element size in bytes.

Size of current element, in bytes.

**See also:**

**IShapeElement::getWordSize()**

**Returns:**

The vinary size of current element in bytes.

### virtual void sosicon::IShapeElement::getMBR (IRectangle & *rect*) `[pure virtual]`

Get minimum bounding rectangle.

Populates the referenced **IRectangle** implementation with the coordinates for the minimum bounding rectangle (MBR) of current element.

**Parameters:**

| | |
|---|---|
| *Reference* | to the **IRectangle** implementation to receive the coordinates of the minium bounding rectangle. |

### virtual ISosiElement* sosicon::IShapeElement::getSosiElement () `[pure virtual]`

Get original SOSI element.

Delivers a pointer to the original SOSI element populating current **IShapeElement**. The pointer is usually stored with current object when **IShapeElement::populate()** is called.

**See also:**

**IShapeElement::populate()**

**Returns:**

Pointer to the source SOSI element.

### virtual int sosicon::IShapeElement::getWordSize () `[pure virtual]`

Get element size in 16-bit words.

Size of current element, in 16-bit words. Some parts of the SOSI standard requires size notation on 16-bit words. This is the byte size divided by two.

**See also:**

**IShapeElement::getByteSize()**

**Returns:**

The binary size of current element in 16-bit words. Equals its byte size divided by two.

### virtual bool sosicon::IShapeElement::populate (ISosiElement * *sosiElement*) `[pure virtual]`

Create from SOSI element.

Takes a SOSI element and creates a shape element from it.

**Parameters:**

| | |
|---|---|
| *sosiElement* | Pointer to the SOSI element to be converted to a shape element. |

**Returns:**

The result of the operation.

**Return values:**

| | |
|---|---|
| *true* | on success. |
| *false* | on failure. |

---

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

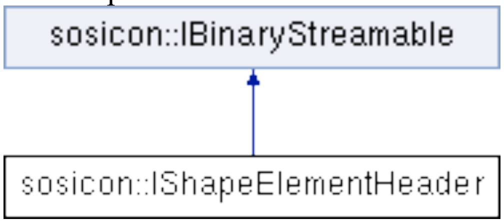
- /prosjekter/sosicon/src/interface/**i_shape_element.h**

## sosicon::IShapeElementHeader Class Reference

Interface: Shape element header.

`#include <i_shape_element_header.h>`

Inheritance diagram for sosicon::IShapeElementHeader:



### Public Member Functions

- virtual **~IShapeElementHeader** ()
  *Destructor.*

### Detailed Description

Interface: Shape element header.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 35 of file i_shape_element_header.h.

### Constructor & Destructor Documentation

**virtual sosicon::IShapeElementHeader::~IShapeElementHeader () `[inline]`, `[virtual]`**

Destructor.

Definition at line 39 of file i_shape_element_header.h.

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

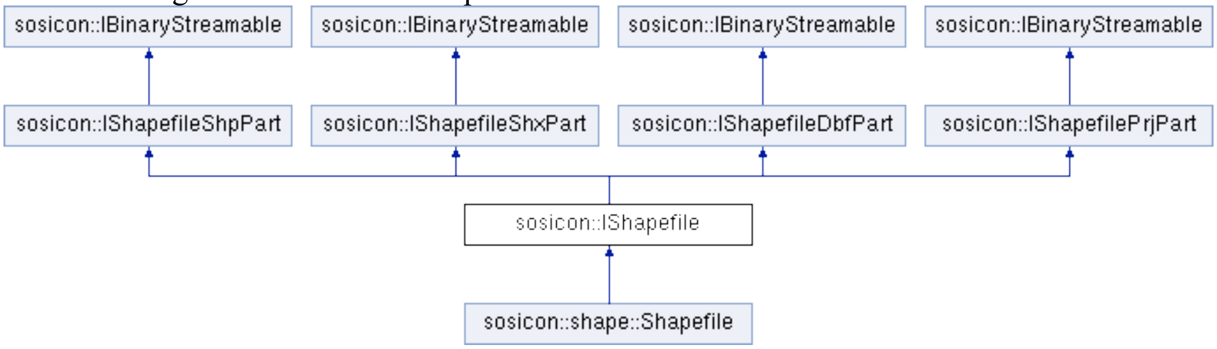
- /prosjekter/sosicon/src/interface/**i_shape_element_header.h**

## sosicon::IShapefile Class Reference

Interface: Shapefile.

`#include <i_shapefile.h>`

Inheritance diagram for sosicon::IShapefile:


sosicon::IBinaryStreamable
sosicon::IBinaryStreamable
sosicon::IBinaryStreamable
sosicon::IBinaryStreamable
sosicon::IShapefileShpPart
sosicon::IShapefileShxPart
sosicon::IShapefileDbfPart
sosicon::IShapefilePrjPart
sosicon::IShapefile
sosicon::shape::Shapefile

### Public Member Functions

- virtual ~**IShapefile** ()
- virtual int **build** (**ISosiElement** *sosiTree, std::string selection, **sosi::ElementType** geomType)=0
  *Build shapefile from SOSI data.*

---

### Detailed Description

Interface: Shapefile.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 39 of file i_shapefile.h.

---

### Constructor & Destructor Documentation

#### virtual sosicon::IShapefile::~IShapefile () `[inline]`, `[virtual]`

Definition at line 46 of file i_shapefile.h.

---

### Member Function Documentation

#### virtual int sosicon::IShapefile::build (ISosiElement * *sosiTree*, std::string *selection*, sosi::ElementType *geomType*) `[pure virtual]`

Build shapefile from SOSI data.

**Parameters:**

| | |
|---|---|
| *sosiTree* | Root SOSI element. The first-level children of this element will be examined and exported if they are compatible. |
| *selection* | SOSI OBJTYPE scheduled for shapefile conversion. |
| *geomType* | SOSI element type scheduled for shapefile conversion. Since a shapefile may contain only one geometry type at a time, one must select what element type to extract from the SOSI file. |

**Returns:**

Number of elements exported.

Implemented in **sosicon::shape::Shapefile** (*p.107*).

---

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

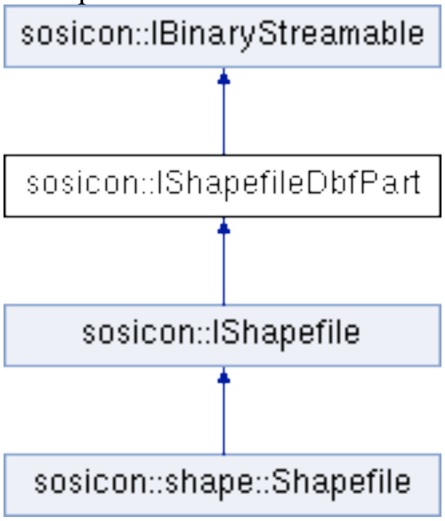
- /prosjekter/sosicon/src/interface/**i_shapefile.h**

## sosicon::IShapefileDbfPart Class Reference

Interface: ShapefileDbfPart.

`#include <i_shapefile_dbf_part.h>`

Inheritance diagram for sosicon::IShapefileDbfPart:


sosicon::IBinaryStreamable
sosicon::IShapefileDbfPart
sosicon::IShapefile
sosicon::shape::Shapefile

### Public Member Functions

- virtual void **writeBinary** (std::ostream &os)
  *Writes binary data to output stream.*
- virtual void **writeDbf** (std::ostream &os)=0

### Detailed Description

Interface: ShapefileDbfPart.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 35 of file i_shapefile_dbf_part.h.

### Member Function Documentation

#### virtual void sosicon::IShapefileDbfPart::writeBinary (std::ostream & *os*)`[inline]`, `[virtual]`

Writes binary data to output stream.

Invoked when the operator<< is used to strem to an std::ostream object. Binary write operation is performed for current impementation here.

**Parameters:**

| | |
|---|---|
| *os* | Target stream object. |

Implements **sosicon::IBinaryStreamable** (*p.61*).

Definition at line 39 of file i_shapefile_dbf_part.h.

### virtual void sosicon::IShapefileDbfPart::writeDbf (std::ostream & *os*)`[pure virtual]`

Implemented in **sosicon::shape::Shapefile** (*p.110*).

---

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

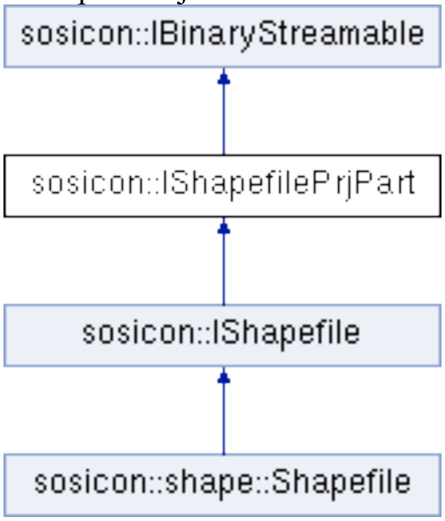
- /prosjekter/sosicon/src/interface/**i_shapefile_dbf_part.h**

# sosicon::IShapefilePrjPart Class Reference

Interface: ShapefilePrjPart.

`#include <i_shapefile_prj_part.h>`

Inheritance diagram for sosicon::IShapefilePrjPart:


sosicon::IBinaryStreamable
sosicon::IShapefilePrjPart
sosicon::IShapefile
sosicon::shape::Shapefile

## Public Member Functions

- virtual void **writeBinary** (std::ostream &os)
  *Writes binary data to output stream.*
- virtual void **writePrj** (std::ostream &os)=0

## Detailed Description

Interface: ShapefilePrjPart.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 35 of file i_shapefile_prj_part.h.

## Member Function Documentation

### virtual void sosicon::IShapefilePrjPart::writeBinary (std::ostream & *os*) `[inline]`, `[virtual]`

Writes binary data to output stream.

Invoked when the operator<< is used to strem to an std::ostream object. Binary write operation is performed for current impementation here.

**Parameters:**

| | |
|---|---|
| *os* | Target stream object. |

Implements **sosicon::IBinaryStreamable** (*p.61*).

Definition at line 39 of file i_shapefile_prj_part.h.

### virtual void sosicon::IShapefilePrjPart::writePrj (std::ostream & *os*)`[pure virtual]`

Implemented in **sosicon::shape::Shapefile** (*p.110*).

---

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

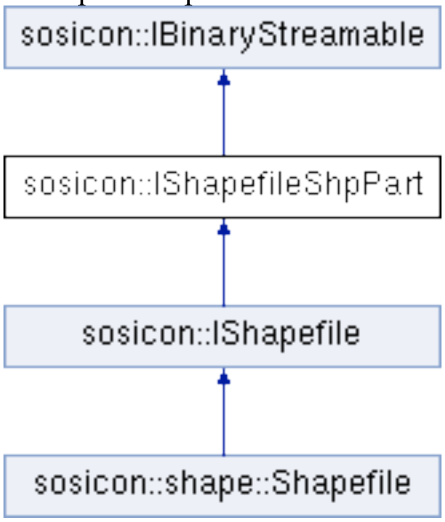
- /prosjekter/sosicon/src/interface/**i_shapefile_prj_part.h**

## sosicon::IShapefileShpPart Class Reference

Interface: ShapefileShpPart.

`#include <i_shapefile_shp_part.h>`

Inheritance diagram for sosicon::IShapefileShpPart:


sosicon::IBinaryStreamable
sosicon::IShapefileShpPart
sosicon::IShapefile
sosicon::shape::Shapefile

### Public Member Functions

- virtual void **writeBinary** (std::ostream &os)
  *Writes binary data to output stream.*
- virtual void **writeShp** (std::ostream &os)=0

### Detailed Description

Interface: ShapefileShpPart.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 35 of file i_shapefile_shp_part.h.

### Member Function Documentation

#### virtual void sosicon::IShapefileShpPart::writeBinary (std::ostream & *os*)`[inline]`, `[virtual]`

Writes binary data to output stream.

Invoked when the operator<< is used to strem to an std::ostream object. Binary write operation is performed for current impementation here.

**Parameters:**

| | |
|---|---|
| *os* | Target stream object. |

Implements **sosicon::IBinaryStreamable** (*p.61*).

Definition at line 39 of file i_shapefile_shp_part.h.

### virtual void sosicon::IShapefileShpPart::writeShp (std::ostream & *os*)`[pure virtual]`

Implemented in **sosicon::shape::Shapefile** (*p.110*).

---

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

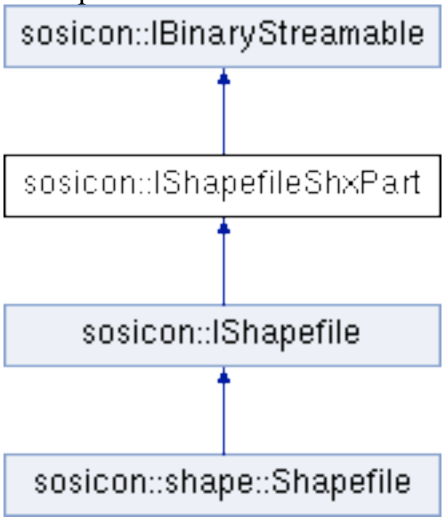
- /prosjekter/sosicon/src/interface/**i_shapefile_shp_part.h**

# sosicon::IShapefileShxPart Class Reference

Interface: ShapefileShxPart.

```
#include <i_shapefile_shx_part.h>
```

Inheritance diagram for sosicon::IShapefileShxPart:


sosicon::IBinaryStreamable
sosicon::IShapefileShxPart
sosicon::IShapefile
sosicon::shape::Shapefile

## Public Member Functions

- virtual void **writeBinary** (std::ostream &os)
  *Writes binary data to output stream.*
- virtual void **writeShx** (std::ostream &os)=0

## Detailed Description

Interface: ShapefileShxPart.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 35 of file i_shapefile_shx_part.h.

## Member Function Documentation

### virtual void sosicon::IShapefileShxPart::writeBinary (std::ostream & *os*) `[inline]`, `[virtual]`

Writes binary data to output stream.

Invoked when the operator<< is used to strem to an std::ostream object. Binary write operation is performed for current impementation here.

**Parameters:**

| | |
|---|---|
| *os* | Target stream object. |

Implements **sosicon::IBinaryStreamable** (*p.61*).

Definition at line 39 of file i_shapefile_shx_part.h.

### virtual void sosicon::IShapefileShxPart::writeShx (std::ostream & *os*)`[pure virtual]`

Implemented in **sosicon::shape::Shapefile** (*p.110*).

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

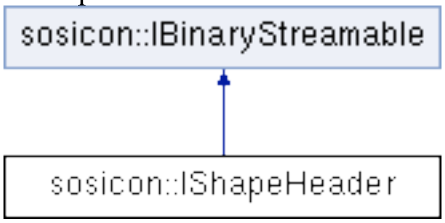
- /prosjekter/sosicon/src/interface/**i_shapefile_shx_part.h**

# sosicon::IShapeHeader Class Reference

Interface: Shape element.

`#include <i_shape_header.h>`

Inheritance diagram for sosicon::IShapeHeader:



## Public Member Functions

- virtual **~IShapeHeader** ()
  *Destructor.*
- virtual shape::geom::ShapeType **getShapeType** ()=0
  *Return mShapeType.*
- virtual void **setShapeType** (shape::geom::ShapeType shapeType)=0
  *Set mShapeType.*
- virtual void **setFileLength** (int fileLength)=0
- virtual int **getFileLength** ()=0
- virtual void **setBoundingBox** (shape::BoundingBox boundingBox)=0
- virtual shape::BoundingBox **getBoundingBox** ()=0
- virtual int **getByteSize** ()=0
  *Size of header in bytes.*
- virtual int **getWordSize** ()=0
  *Size of header in 16-bit words.*

---

## Detailed Description

Interface: Shape element.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 37 of file i_shape_header.h.

---

## Constructor & Destructor Documentation

### virtual sosicon::IShapeHeader::~IShapeHeader () `[inline]`, `[virtual]`

Destructor.

Definition at line 41 of file i_shape_header.h.

## Member Function Documentation

### virtual shape::BoundingBox sosicon::IShapeHeader::getBoundingBox () `[pure virtual]`

### virtual int sosicon::IShapeHeader::getByteSize () `[pure virtual]`

Size of header in bytes.

### virtual int sosicon::IShapeHeader::getFileLength () `[pure virtual]`

### virtual shape::geom::ShapeType sosicon::IShapeHeader::getShapeType () `[pure virtual]`

Return mShapeType.

**See also:**

ShapeHeader::mShapeType

**Returns:**

The shape type in current file.

### virtual int sosicon::IShapeHeader::getWordSize () `[pure virtual]`

Size of header in 16-bit words.

### virtual void sosicon::IShapeHeader::setBoundingBox (shape::BoundingBox *boundingBox*) `[pure virtual]`

### virtual void sosicon::IShapeHeader::setFileLength (int *fileLength*) `[pure virtual]`

### virtual void sosicon::IShapeHeader::setShapeType (shape::geom::ShapeType *shapeType*) `[pure virtual]`

Set mShapeType.

**See also:**

ShapeHeader::mShapeType

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

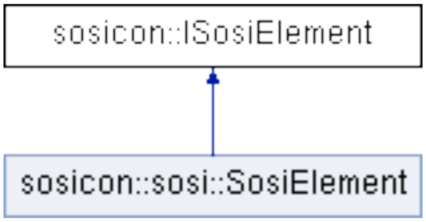
- /prosjekter/sosicon/src/interface/**i_shape_header.h**

## sosicon::ISosiElement Class Reference

Interface: SOSI element.

```
#include <i_sosi_element.h>
```

Inheritance diagram for sosicon::ISosiElement:



### Public Member Functions

- virtual **~ISosiElement** ()
  *Destructor.*
- virtual std::vector
- < **ISosiElement** * > & **children** ()=0
- virtual void **addChild** (**ISosiElement** *child)=0
- virtual std::string **getName** ()=0
- virtual bool **getChild** (**sosi::SosiElementSearch** &src)=0
- virtual std::string **getData** ()=0
- virtual int **getLevel** ()=0
- virtual **sosi::ElementType getType** ()=0
- virtual std::string **getObjType** ()=0
- virtual **ISosiElement** * **getRoot** ()=0
- virtual std::string **getSerial** ()=0
- virtual void **deleteChildren** ()=0
- virtual void **dump** (int indent=0)=0
- virtual **ISosiElement** * **find** (std::string ref)=0

---

### Detailed Description

Interface: SOSI element.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Represents the generic form of a SOSI element. All SOSI elements must implement this interface. It provides functionality for setting and retrieveing field values. The parser writes field values to current SOSI element through this interface.

**See also:**

sosicon::Parser::parseSosiLine()

Data retrieval is provided by one of the **getData()** overloads. Single string data fields are associated with their unique keys, while collections of other data objects, such as address units and cadastral units, are retrieved one-by-one by sequential calls to **getData()**.

Definition at line 51 of file i_sosi_element.h.

## Constructor & Destructor Documentation

### virtual sosicon::ISosiElement::~ISosiElement () `[inline]`, `[virtual]`

Destructor.

Definition at line 55 of file i_sosi_element.h.

## Member Function Documentation

### virtual void sosicon::ISosiElement::addChild (ISosiElement * *child*) `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.116*).

### virtual std::vector<ISosiElement*>& sosicon::ISosiElement::children () `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.117*).

### virtual void sosicon::ISosiElement::deleteChildren () `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.117*).

### virtual void sosicon::ISosiElement::dump (int *indent* = 0) `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.117*).

### virtual ISosiElement* sosicon::ISosiElement::find (std::string *ref*) `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.117*).

### virtual bool sosicon::ISosiElement::getChild (sosi::SosiElementSearch & *src*) `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.117*).

### virtual std::string sosicon::ISosiElement::getData () `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.117*).

### virtual int sosicon::ISosiElement::getLevel () `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.117*).

### virtual std::string sosicon::ISosiElement::getName () `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.118*).

### virtual std::string sosicon::ISosiElement::getObjType () `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.118*).

### virtual ISosiElement* sosicon::ISosiElement::getRoot () `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.118*).

### virtual std::string sosicon::ISosiElement::getSerial () `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.118*).

### virtual sosi::ElementType sosicon::ISosiElement::getType () `[pure virtual]`

Implemented in **sosicon::sosi::SosiElement** (*p.118*).

---

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

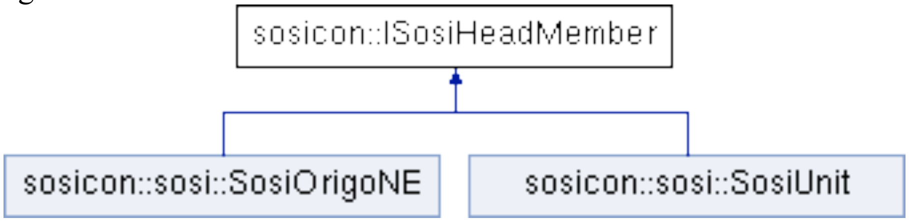
- /prosjekter/sosicon/src/interface/**i_sosi_element.h**

# sosicon::ISosiHeadMember Class Reference

Interface: SOSI header element.

`#include <i_sosi_head_member.h>`

Inheritance diagram for sosicon::ISosiHeadMember:



## Public Member Functions

- virtual **~ISosiHeadMember** ()
  *Destructor.*
- virtual void **init** (**ISosiElement** *e)=0
- virtual bool **initialized** ()=0

## Detailed Description

Interface: SOSI header element.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 36 of file i_sosi_head_member.h.

## Constructor & Destructor Documentation

### virtual sosicon::ISosiHeadMember::~ISosiHeadMember () `[inline]`, `[virtual]`

Destructor.

Definition at line 40 of file i_sosi_head_member.h.

## Member Function Documentation

### virtual void sosicon::ISosiHeadMember::init (ISosiElement * *e*) `[pure virtual]`

Implemented in **sosicon::sosi::SosiOrigoNE** (*p.131*), and **sosicon::sosi::SosiUnit** (*p.139*).

### virtual bool sosicon::ISosiHeadMember::initialized ()`[pure virtual]`

Implemented in **sosicon::sosi::SosiOrigoNE** (*p.131*), and **sosicon::sosi::SosiUnit** (*p.139*).

---

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

- /prosjekter/sosicon/src/interface/**i_sosi_head_member.h**

# sosicon::Parser Class Reference

SOSI file parser.

`#include <parser.h>`

## Public Member Functions

- **Parser** ()  
  *Constructor.*
- **~Parser** ()  
  *Destructor.*
- void **complete** ()  
  *Flush parsed data.*
- void **dump** ()  
  *Debug output.*
- **ISosiElement** * **getRootElement** ()  
  *Retrieve pointer to root element.*
- void **ragelParseSosiLine** (std::string sosiLine)  
  *Main parser routine.*

## Private Member Functions

- void **digestPendingElement** ()  
  *Save current SOSI element.*

## Private Attributes

- std::vector< **ISosiElement** * > **mElementStack**  
  *Working stack.*
- **sosi::SosiElementMap** **mElementIndex**  
  *Index.*
- int **mPendingElementLevel**  
  *SOSI level of element currently in parser.*
- std::string **mPendingElementName**  
  *Name of element currently in parser.*
- std::string **mPendingElementSerial**  
  *Serial number of element currently in parser.*
- std::string **mPendingElementAttributes**  
  *Attribute data of element currently in parser.*

---

## Detailed Description

SOSI file parser.

### Author:

Espen Andersen

**Copyright:**

GNU General Public License

The file parser. Reads and organizes SOSI file input, preparing the data for conversion and output. This class wraps a Ragel-generated state machine set up to parse SOSI content line-by-line. For more information about the Ragel state machine compiler, visit `http://www.complang.org/ragel/`

Specifically, the function parseSosiLine() is implemented in Ragel. The implementation script is located in parser/parser_sosi_line.rl. The file parser_sosi_line.cpp is generated on the basis of parser/parser_sosi_line.rl during pre-build processing.

**Note:**

Since parser_sosi_line.cpp is automatically re-generated before each compile, no redacting mat take place here. Any changes will be lost upon compile. Instead, its source script parser/parser_sosi_line.rl have to be edited.

Definition at line 52 of file parser.h.

## Constructor & Destructor Documentation

### sosicon::Parser::Parser ()

Constructor.

Initializes internal iterators and pointers.

Definition at line 21 of file parser.cpp.

### sosicon::Parser::~Parser ()

Destructor.

Calls Parser::reset() for final clean-up.

Definition at line 27 of file parser.cpp.

## Member Function Documentation

### void sosicon::Parser::complete () `[inline]`

Flush parsed data.

Definition at line 115 of file parser.h.

### void sosicon::Parser::digestPendingElement () `[private]`

Save current SOSI element.

The parser stores intermediate data in the mPendingElementXXX member variables. When a SOSI element had been fully parsed, this function is called to move the data into the element tree structure before carrying on.

Definition at line 33 of file parser.cpp.

#### void sosicon::Parser::dump ()

Debug output.

Definition at line 62 of file parser.cpp.

#### sosicon::ISosiElement * sosicon::Parser::getRootElement ()

Retrieve pointer to root element.

Definition at line 67 of file parser.cpp.

#### void sosicon::Parser::ragelParseSosiLine (std::string *sosiLine*)

Main parser routine.

Processes one line from the SOSI file. This function is called repeatedly, consuming the input file line-by-line until EOF.

**Note:**

This function is implemented in the ragel script at parser/parser_sosi_line.rl, the c++ file parser_sosi_line.cpp is merely generated from the ragel script. Thus, any changes to the implementation must be done in the ragel script, since the c++ file will be automatically overwritten during the pre-build process.

**Parameters:**

| | |
|---|---|
| *sosiLine* | Current line from the SOSI input file. |

Definition at line 139 of file parser_ragel.cpp.

## Member Data Documentation

#### sosi::SosiElementMap sosicon::Parser::mElementIndex `[private]`

Index.

Index elements by serial number. Lookup table to resolve SOSI references (REF element).

Definition at line 66 of file parser.h.

#### std::vector<ISosiElement*> sosicon::Parser::mElementStack `[private]`

Working stack.

Keeps track of element relationship while parsing the SOSI file. The root element is inserted at the front of the stack. When the parser has completed, the stack should contain the root element only.

Definition at line 60 of file parser.h.

#### std::string sosicon::Parser::mPendingElementAttributes `[private]`

Attribute data of element currently in parser.

Intermediate storage member.

Definition at line 90 of file parser.h.

### int sosicon::Parser::mPendingElementLevel `[private]`

SOSI level of element currently in parser.

Intermediate storage member.

Definition at line 72 of file parser.h.

### std::string sosicon::Parser::mPendingElementName `[private]`

Name of element currently in parser.

Intermediate storage member.

Definition at line 78 of file parser.h.

### std::string sosicon::Parser::mPendingElementSerial `[private]`

Serial number of element currently in parser.

Intermediate storage member.

Definition at line 84 of file parser.h.

---

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

- /prosjekter/sosicon/src/**parser.h**
- /prosjekter/sosicon/src/**parser.cpp**
- /prosjekter/sosicon/src/**parser_ragel.cpp**
- /prosjekter/sosicon/src/ragel/**parser.rl**

## sosicon::sosi::ReferenceData Struct Reference

SOSI reference number.

```
#include <sosi_types.h>
```

### Public Attributes

- std::string **serial**  
  *The element ID.*
- bool **reverse**  
  *Minus sign = reverse coordinate sequence.*
- bool **subtract**  
  *Parenthesis = subtract shape.*

### Detailed Description

SOSI reference number.

Definition at line 130 of file sosi_types.h.

### Member Data Documentation

#### bool sosicon::sosi::ReferenceData::reverse

Minus sign = reverse coordinate sequence.

Definition at line 132 of file sosi_types.h.

#### std::string sosicon::sosi::ReferenceData::serial

The element ID.

Definition at line 131 of file sosi_types.h.

#### bool sosicon::sosi::ReferenceData::subtract

Parenthesis = subtract shape.

Definition at line 133 of file sosi_types.h.

#### The documentation for this struct was generated from the following file:

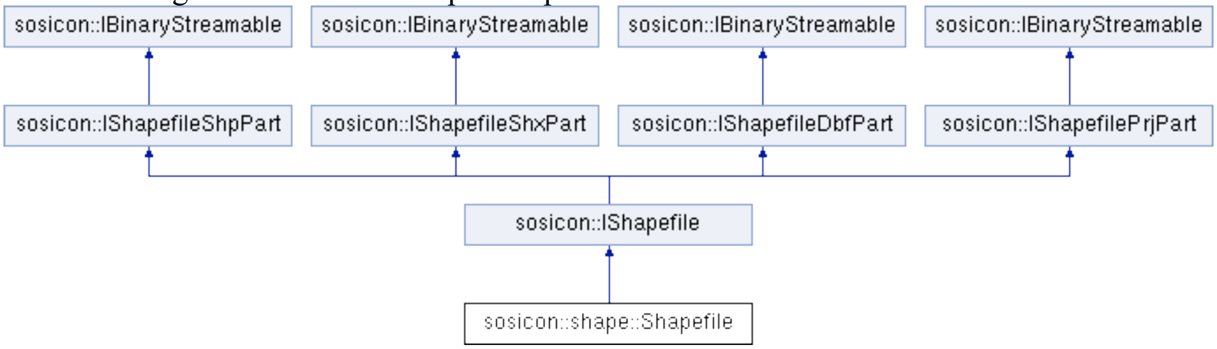
- /prosjekter/sosicon/src/sosi/**sosi_types.h**

## sosicon::shape::Shapefile Class Reference

**Shapefile** implementation.

```
#include <shapefile.h>
```

Inheritance diagram for sosicon::shape::Shapefile:


sosicon::IBinaryStreamable
sosicon::IBinaryStreamable
sosicon::IBinaryStreamable
sosicon::IBinaryStreamable
sosicon::IShapefileShpPart
sosicon::IShapefileShxPart
sosicon::IShapefileDbfPart
sosicon::IShapefilePrjPart
sosicon::IShapefile
sosicon::shape::Shapefile

### Public Member Functions

- **Shapefile** ()
  *Constructor.*
- virtual ~**Shapefile** ()
  *Destructor.*
- int **build** (**ISosiElement** *sosiTree, std::string objType, **sosi::ElementType** geomType)
  *Described in **IShapefile**.*
- virtual void **writeDbf** (std::ostream &os)
  *Described in **IShapefileDbfPart**.*
- virtual void **writeShp** (std::ostream &os)
  *Described in **IShapefileShpPart**.*
- virtual void **writeShx** (std::ostream &os)
  *Described in **IShapefileShxPart**.*
- virtual void **writePrj** (std::ostream &os)
  *Described in **IShapefilePrjPart**.*

### Private Member Functions

- void **adjustMasterMbr** (double xMin, double yMin, double xMax, double yMax)
  *Expand MBR to contain **Coordinate** collection.*
- void **buildShpElement** (**ISosiElement** *sosi, **ShapeType** type)
  *Create SHP element.*
- void **buildShpHeader** (**ShapeType** type)
  *Populate shape header struct.*
- void **buildShpPoint** (**CoordinateCollection** &cc)
  *Build shape element: Point.*
- void **buildShpPolygon** (**CoordinateCollection** &cc)
  *Build shape element: Polygon.*
- void **buildShpPolyLine** (**CoordinateCollection** &cc)
  *Build shape element: PolyLine.*
- void **buildShpRecCoordinate** (int &o, **CoordinateCollection** &cc)

*Write first coordinate pair in collection to shapefile buffer.*

- void **buildShpRecCoordinate** (int &o, **ICoordinate** *c)
  *Write coordinate pair to shapefile buffer.*
- void **buildShpRecCoordinates** (int &o, **CoordinateCollection** &cc)
  *Write multiple coordinate pairs to shapefile buffer.*
- void **buildShpRecHeaderCommonPart** (int &o, int contentLength, **ShapeType** type)
  *Create shapefile record header, common part.*
- void **buildShpRecHeaderExtended** (int &o, **CoordinateCollection** &cc)
  *Create shapefile record header, multipoint/polyLine/polygon part.*
- void **buildShpRecHeaderOffsets** (int &o, **CoordinateCollection** &cc)
  *Create shapefile record header, offsets.*
- void **buildDbf** ()
  *Create DBF file content.*
- void **buildDbfFieldDescriptor** (int &pos)
  *Create DBF field descriptor.*
- void **buildDbfHeader** (**Int16Field** recordLength)
  *Create DBF header.*
- void **buildDbfRecordSection** (int &pos, int recordLength)
  *Create DBF records.*
- void **buildShx** ()
  *Create SHX file content.*
- int **expandShpBuffer** (int byteLength)
  *Expand shp payload buffer.*
- void **extractDbfFields** (**ISosiElement** *sosi, **DbfRecord** &rec)
  *Recursive func to extract SOSI field data.*
- void **insertDbfRecord** (**ISosiElement** *sosi)
  *Create and insert DBF record.*
- void **insertShxOffset** (int contentLength)
  *Append offset value to SHX (index)*
- std::vector< **ICoordinate** * > **getNormalized** (**sosi::NorthEastList** &neLst)
  ***Shapefile*** *polys must have clockwise-ordered vertices.*
- void **saveToDbf** (**DbfRecord** &rec, std::string field, std::string data)
  *Update or insert new DBF field.*

## Private Attributes

- **ISosiElement * mSosiTree**
  *SOSI source.*
- char **mShpHeader** [100]
  *Main SHP file header.*
- char * **mShpBuffer**
  *SHP file payload.*
- int **mShpSize**
  *Data length of SHP file buffer.*
- int **mShpBufferSize**
  *Allocated buffer length.*
- char **mShxHeader** [100]

*Index file header.*

- char * **mShxBuffer**
  *Index file payload.*
- int **mShxBufferSize**
  *Length of SHX file buffer.*
- char **mDbfHeader** [32]
  *dBase file header*
- char * **mDbfBuffer**
  *dBase file payload*
- int **mDbfBufferSize**
  *Length of dBase file buffer.*
- int **mRecordNumber**
  *Number of current record in process.*
- double **mXmin**
  *Minimum bounding rectangle, min X.*
- double **mYmin**
  *Minimum bounding rectangle, min Y.*
- double **mXmax**
  *Minimum bounding rectangle, max X.*
- double **mYmax**
  *Minimum bounding rectangle, max Y.*
- **DbfFieldLengths mDbfFieldLengths**
  *Accumulation of DBF fields and their lenghts.*
- **DbfRecordSet mDbfRecordSet**
  *All DBF records.*
- **ShxOffsets mShxOffsets**
  *Index file offsets.*

## Static Private Attributes

- static const int **BUFFER_CHUNK_SIZE**
  *Buffer allocation size.*

---

## Detailed Description

**Shapefile** implementation.

Wraps all ESRI Shape output files (shp, shx, dbf, prj...) in one class.

**Author:**

Espen Andersen

**Copyright:**

GNU General Public License

Definition at line 56 of file shapefile.h.

---

## Constructor & Destructor Documentation

### sosicon::shape::Shapefile::Shapefile ()`[inline]`

Constructor.

Definition at line 164 of file shapefile.h.

### sosicon::shape::Shapefile::~Shapefile ()`[virtual]`

Destructor.

Definition at line 40 of file shapefile.cpp.

## Member Function Documentation

### void sosicon::shape::Shapefile::adjustMasterMbr (double *xMin*, double *yMin*, double *xMax*, double *yMax*)`[private]`

Expand MBR to contain **Coordinate** collection.

Definition at line 47 of file shapefile.cpp.

### int sosicon::shape::Shapefile::build (ISosiElement * *sosiTree*, std::string *objType*, sosi::ElementType *geomType*)`[virtual]`

Described in **IShapefile**.

Implements **sosicon::IShapefile** (*p.82*).

Definition at line 55 of file shapefile.cpp.

### void sosicon::shape::Shapefile::buildDbf ()`[private]`

Create DBF file content.

Definition at line 283 of file shapefile.cpp.

### void sosicon::shape::Shapefile::buildDbfFieldDescriptor (int & *pos*)`[private]`

Create DBF field descriptor.

Definition at line 317 of file shapefile.cpp.

### void sosicon::shape::Shapefile::buildDbfHeader (Int16Field *recordLength*)`[private]`

Create DBF header.

Definition at line 348 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildDbfRecordSection (int & *pos*, int *recordLength*) `[private]`

Create DBF records.

Definition at line 379 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpElement (ISosiElement * *sosi*, ShapeType *type*) `[private]`

Create SHP element.

Definition at line 127 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpHeader (ShapeType *type*) `[private]`

Populate shape header struct.

Definition at line 92 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpPoint (CoordinateCollection & *cc*) `[private]`

Build shape element: Point.

Definition at line 152 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpPolygon (CoordinateCollection & *cc*) `[private]`

Build shape element: Polygon.

Definition at line 174 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpPolyLine (CoordinateCollection & *cc*) `[private]`

Build shape element: PolyLine.

Definition at line 162 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpRecCoordinate (int & *o*, CoordinateCollection & *cc*) `[private]`

Write first coordinate pair in collection to shapefile buffer.

Definition at line 186 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpRecCoordinate (int & *o*, ICoordinate * *c*) `[private]`

Write coordinate pair to shapefile buffer.

Definition at line 197 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpRecCoordinates (int & *o*, CoordinateCollection & *cc*) `[private]`

Write multiple coordinate pairs to shapefile buffer.

Definition at line 205 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpRecHeaderCommonPart (int & *o*, int *contentLength*, ShapeType *type*) `[private]`

Create shapefile record header, common part.

Definition at line 264 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpRecHeaderExtended (int & *o*, CoordinateCollection & *cc*) `[private]`

Create shapefile record header, multipoint/polyLine/polygon part.

Definition at line 217 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShpRecHeaderOffsets (int & *o*, CoordinateCollection & *cc*) `[private]`

Create shapefile record header, offsets.

Definition at line 243 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::buildShx () `[private]`

Create SHX file content.

Definition at line 414 of file shapefile.cpp.

#### int sosicon::shape::Shapefile::expandShpBuffer (int *byteLength*) `[private]`

Expand shp payload buffer.

Definition at line 439 of file shapefile.cpp.

#### void sosicon::shape::Shapefile::extractDbfFields (ISosiElement * *sosi*, DbfRecord & *rec*) `[private]`

Recursive func to extract SOSI field data.

Definition at line 479 of file shapefile.cpp.

#### std::vector<ICoordinate*> sosicon::shape::Shapefile::getNormalized (sosi::NorthEastList & *neLst*) `[private]`

**Shapefile** polys must have clockwise-ordered vertices.

### void sosicon::shape::Shapefile::insertDbfRecord (ISosiElement * *sosi*)[private]

Create and insert DBF record.

Definition at line 498 of file shapefile.cpp.

### void sosicon::shape::Shapefile::insertShxOffset (int *contentLength*)[private]

Append offset value to SHX (index)

Definition at line 509 of file shapefile.cpp.

### void sosicon::shape::Shapefile::saveToDbf (DbfRecord & *rec*, std::string *field*, std::string *data*)[private]

Update or insert new DBF field.

Definition at line 519 of file shapefile.cpp.

### void sosicon::shape::Shapefile::writeDbf (std::ostream & *os*)[virtual]

Described in **IShapefileDbfPart**.

Implements **sosicon::IShapefileDbfPart** (*p.85*).

Definition at line 547 of file shapefile.cpp.

### void sosicon::shape::Shapefile::writePrj (std::ostream & *os*)[virtual]

Described in **IShapefilePrjPart**.

Implements **sosicon::IShapefilePrjPart** (*p.87*).

Definition at line 553 of file shapefile.cpp.

### void sosicon::shape::Shapefile::writeShp (std::ostream & *os*)[virtual]

Described in **IShapefileShpPart**.

Implements **sosicon::IShapefileShpPart** (*p.89*).

Definition at line 535 of file shapefile.cpp.

### void sosicon::shape::Shapefile::writeShx (std::ostream & *os*)[virtual]

Described in **IShapefileShxPart**.

Implements **sosicon::IShapefileShxPart** (*p.91*).

Definition at line 541 of file shapefile.cpp.

## Member Data Documentation

### const int sosicon::shape::Shapefile::BUFFER_CHUNK_SIZE `[static]`, `[private]`

Buffer allocation size.

To speed things up, heap allocations are done in large chunks. This parameters defines how many bytes to request for each buffer expansion.

Definition at line 64 of file shapefile.h.

### char* sosicon::shape::Shapefile::mDbfBuffer `[private]`

dBase file payload

Definition at line 78 of file shapefile.h.

### int sosicon::shape::Shapefile::mDbfBufferSize `[private]`

Length of dBase file buffer.

Definition at line 79 of file shapefile.h.

### DbfFieldLengths sosicon::shape::Shapefile::mDbfFieldLengths `[private]`

Accumulation of DBF fields and their lenghts.

Definition at line 88 of file shapefile.h.

### char sosicon::shape::Shapefile::mDbfHeader[32] `[private]`

dBase file header

Definition at line 77 of file shapefile.h.

### DbfRecordSet sosicon::shape::Shapefile::mDbfRecordSet `[private]`

All DBF records.

Definition at line 89 of file shapefile.h.

### int sosicon::shape::Shapefile::mRecordNumber `[private]`

Number of current record in process.

Definition at line 81 of file shapefile.h.

### char* sosicon::shape::Shapefile::mShpBuffer `[private]`

SHP file payload.

Definition at line 69 of file shapefile.h.

### int sosicon::shape::Shapefile::mShpBufferSize [private]

Allocated buffer length.

Definition at line 71 of file shapefile.h.

### char sosicon::shape::Shapefile::mShpHeader[100] [private]

Main SHP file header.

Definition at line 68 of file shapefile.h.

### int sosicon::shape::Shapefile::mShpSize [private]

Data length of SHP file buffer.

Definition at line 70 of file shapefile.h.

### char* sosicon::shape::Shapefile::mShxBuffer [private]

Index file payload.

Definition at line 74 of file shapefile.h.

### int sosicon::shape::Shapefile::mShxBufferSize [private]

Length of SHX file buffer.

Definition at line 75 of file shapefile.h.

### char sosicon::shape::Shapefile::mShxHeader[100] [private]

Index file header.

Definition at line 73 of file shapefile.h.

### ShxOffsets sosicon::shape::Shapefile::mShxOffsets [private]

Index file offsets.

Definition at line 90 of file shapefile.h.

### ISosiElement* sosicon::shape::Shapefile::mSosiTree [private]

SOSI source.

Definition at line 66 of file shapefile.h.

### double sosicon::shape::Shapefile::mXmax [private]

Minimum bounding rectangle, max X.

Definition at line 85 of file shapefile.h.

### double sosicon::shape::Shapefile::mXmin `[private]`

Minimum bounding rectangle, min X.

Definition at line 83 of file shapefile.h.

### double sosicon::shape::Shapefile::mYmax `[private]`

Minimum bounding rectangle, max Y.

Definition at line 86 of file shapefile.h.

### double sosicon::shape::Shapefile::mYmin `[private]`

Minimum bounding rectangle, min Y.

Definition at line 84 of file shapefile.h.

---

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

- /prosjekter/sosicon/src/shape/**shapefile.h**
- /prosjekter/sosicon/src/shape/**shapefile.cpp**

# sosicon::shape::ShxIndex Struct Reference

```
#include <shapefile_types.h>
```

## Public Attributes

- **Int32Field offset**
- **Int32Field length**

## Detailed Description

Definition at line 83 of file shapefile_types.h.

## Member Data Documentation

### Int32Field sosicon::shape::ShxIndex::length

Definition at line 85 of file shapefile_types.h.

### Int32Field sosicon::shape::ShxIndex::offset

Definition at line 84 of file shapefile_types.h.

### The documentation for this struct was generated from the following file:

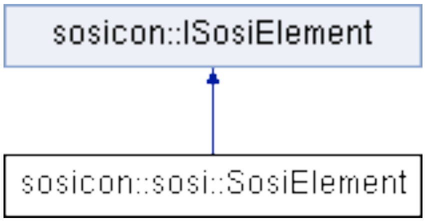
- /prosjekter/sosicon/src/shape/**shapefile_types.h**

# sosicon::sosi::SosiElement Class Reference

Basic SOSI element.

```
#include <sosi_element.h>
```

Inheritance diagram for sosicon::sosi::SosiElement:



## Public Member Functions

- **SosiElement** (std::string name, std::string serial, std::string data, int level, **ISosiElement** *root, **SosiElementMap** &index)
  *Construct new SOSI element.*
- virtual void **addChild** (**ISosiElement** *child)
  *Insert children element.*
- virtual void **deleteChildren** ()
  *Recursively deletes all children.*
- virtual void **dump** (int indent=0)
  *Debug function.*
- virtual **ISosiElement** * **find** (std::string ref)
  *Find element by reference.*
- std::vector< **ISosiElement** * > & **children** ()
- virtual bool **getChild** (**SosiElementSearch** &src)
  *Get next child in list.*
- virtual std::string **getData** ()
  *Get unparsed element data.*
- virtual int **getLevel** ()
  *Get nesting level of current element.*
- virtual std::string **getObjType** ()
  *Get ObjType of current element.*
- virtual std::string **getName** ()
  *Get name of current element.*
- virtual **ISosiElement** * **getRoot** ()
  *Get root element.*
- virtual std::string **getSerial** ()
  *Get serial number (ID) of current element.*
- virtual **ElementType** **getType** ()
  *Get ElementType of current element.*

## Private Member Functions

- virtual bool **nextChild** (**SosiElementSearch** &src)
  *Increment to next child in list.*

## Private Attributes

- **SosiTranslationTable mTranslation**
  *SOSI string translations.*
- std::string **mData**
  *Current element's data content.*
- **SosiChildrenList mChildren**
  *List of children elements.*
- int **mLevel**
  *Current element's nesting level.*
- std::string **mName**
  *Current element's name.*
- **ElementType mType**
  *Current element's geometric type.*
- **ObjType mObjType**
  *Current element's objtype.*
- std::string **mObjTypeStr**
  *Current element's objtype.*
- std::string **mSerial**
  *Current element's serial number if provided.*
- **ISosiElement** * **mRoot**
  *Pointer to root element.*
- **SosiElementMap** & **mIndex**
  *Reference to parser's lookup table.*

## Detailed Description

Basic SOSI element.

Implements basic characteristics of a SOSI element.

Definition at line 71 of file sosi_element.h.

## Constructor & Destructor Documentation

### sosicon::sosi::SosiElement::SosiElement (std::string *name*, std::string *serial*, std::string *data*, int *level*, ISosiElement * *root*, SosiElementMap & *index*)

Construct new SOSI element.

Definition at line 21 of file sosi_element.cpp.

## Member Function Documentation

### void sosicon::sosi::SosiElement::addChild (ISosiElement * *child*)`[virtual]`

Insert children element.

Implements **sosicon::ISosiElement** (*p.95*).

Definition at line 35 of file sosi_element.cpp.

### std::vector<ISosiElement*>& sosicon::sosi::SosiElement::children ()[inline], [virtual]

Implements **sosicon::ISosiElement** (*p.95*).

Definition at line 126 of file sosi_element.h.

### void sosicon::sosi::SosiElement::deleteChildren ()[virtual]

Recursively deletes all children.

Implements **sosicon::ISosiElement** (*p.95*).

Definition at line 44 of file sosi_element.cpp.

### void sosicon::sosi::SosiElement::dump (int *indent* = 0)[virtual]

Debug function.

Implements **sosicon::ISosiElement** (*p.95*).

Definition at line 52 of file sosi_element.cpp.

### sosicon::ISosiElement * sosicon::sosi::SosiElement::find (std::string *ref*)[virtual]

Find element by reference.

Implements **sosicon::ISosiElement** (*p.95*).

Definition at line 62 of file sosi_element.cpp.

### bool sosicon::sosi::SosiElement::getChild (SosiElementSearch & *src*)[virtual]

Get next child in list.

Always pass a null pointer to start iterating through the children list. The referenced pointer will point to the next child in list when the function returns. If the end of the list is reached, the function returns false.

Implements **sosicon::ISosiElement** (*p.95*).

Definition at line 94 of file sosi_element.cpp.

### virtual std::string sosicon::sosi::SosiElement::getData ()[inline], [virtual]

Get unparsed element data.

Implements **sosicon::ISosiElement** (*p.95*).

Definition at line 137 of file sosi_element.h.

### virtual int sosicon::sosi::SosiElement::getLevel ()[inline], [virtual]

Get nesting level of current element.

Implements **sosicon::ISosiElement** (*p.95*).

Definition at line 140 of file sosi_element.h.

### virtual std::string sosicon::sosi::SosiElement::getName ()`[inline]`, `[virtual]`

Get name of current element.

Implements **sosicon::ISosiElement** (*p.96*).

Definition at line 146 of file sosi_element.h.

### virtual std::string sosicon::sosi::SosiElement::getObjType ()`[inline]`, `[virtual]`

Get ObjType of current element.

Implements **sosicon::ISosiElement** (*p.96*).

Definition at line 143 of file sosi_element.h.

### virtual ISosiElement* sosicon::sosi::SosiElement::getRoot ()`[inline]`, `[virtual]`

Get root element.

Implements **sosicon::ISosiElement** (*p.96*).

Definition at line 149 of file sosi_element.h.

### virtual std::string sosicon::sosi::SosiElement::getSerial ()`[inline]`, `[virtual]`

Get serial number (ID) of current element.

Implements **sosicon::ISosiElement** (*p.96*).

Definition at line 152 of file sosi_element.h.

### virtual ElementType sosicon::sosi::SosiElement::getType ()`[inline]`, `[virtual]`

Get ElementType of current element.

Implements **sosicon::ISosiElement** (*p.96*).

Definition at line 155 of file sosi_element.h.

### bool sosicon::sosi::SosiElement::nextChild (SosiElementSearch & *src*)`[private]`, `[virtual]`

Increment to next child in list.

Definition at line 74 of file sosi_element.cpp.

## Member Data Documentation

### SosiChildrenList sosicon::sosi::SosiElement::mChildren `[private]`

List of children elements.

Definition at line 80 of file sosi_element.h.

### std::string sosicon::sosi::SosiElement::mData `[private]`

Current element's data content.

Definition at line 77 of file sosi_element.h.

### SosiElementMap& sosicon::sosi::SosiElement::mIndex `[private]`

Reference to parser's lookup table.

Definition at line 104 of file sosi_element.h.

### int sosicon::sosi::SosiElement::mLevel `[private]`

Current element's nesting level.

Definition at line 83 of file sosi_element.h.

### std::string sosicon::sosi::SosiElement::mName `[private]`

Current element's name.

Definition at line 86 of file sosi_element.h.

### ObjType sosicon::sosi::SosiElement::mObjType `[private]`

Current element's objtype.

Definition at line 92 of file sosi_element.h.

### std::string sosicon::sosi::SosiElement::mObjTypeStr `[private]`

Current element's objtype.

Definition at line 95 of file sosi_element.h.

### ISosiElement* sosicon::sosi::SosiElement::mRoot `[private]`

Pointer to root element.

Definition at line 101 of file sosi_element.h.

### std::string sosicon::sosi::SosiElement::mSerial `[private]`

Current element's serial number if provided.

Definition at line 98 of file sosi_element.h.

### SosiTranslationTable sosicon::sosi::SosiElement::mTranslation `[private]`

SOSI string translations.

Definition at line 74 of file sosi_element.h.

### ElementType sosicon::sosi::SosiElement::mType `[private]`

Current element's geometric type.

Definition at line 89 of file sosi_element.h.

---

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

- /prosjekter/sosicon/src/sosi/**sosi_element.h**
- /prosjekter/sosicon/src/sosi/**sosi_element.cpp**

# sosicon::sosi::SosiElementSearch Class Reference

`#include <sosi_element_search.h>`

## Public Member Functions

- **SosiElementSearch** ()
- **SosiElementSearch** (**sosi::ElementType** filter)
- SosiChildrenList::size_type **index** ()
- SosiChildrenList::size_type **index** (SosiChildrenList::size_type i)
- **ISosiElement** * **element** ()
- **ISosiElement** * **element** (**ISosiElement** *e)
- **sosi::ElementType** **type** ()
- **sosi::ElementType** **type** (**sosi::ElementType** t)
- void **next** ()

## Private Attributes

- SosiChildrenList::size_type **mIndex**
- **ISosiElement** * **mSosiElement**
- **sosi::ElementType** **mElementType**

## Detailed Description

Definition at line 46 of file sosi_element_search.h.

## Constructor & Destructor Documentation

### sosicon::sosi::SosiElementSearch::SosiElementSearch () `[inline]`

Definition at line 51 of file sosi_element_search.h.

### sosicon::sosi::SosiElementSearch::SosiElementSearch (sosi::ElementType *filter*) `[inline]`

Definition at line 52 of file sosi_element_search.h.

## Member Function Documentation

### ISosiElement* sosicon::sosi::SosiElementSearch::element () `[inline]`

Definition at line 55 of file sosi_element_search.h.

### ISosiElement* sosicon::sosi::SosiElementSearch::element (ISosiElement * *e*) `[inline]`

Definition at line 56 of file sosi_element_search.h.

**SosiChildrenList::size_type sosicon::sosi::SosiElementSearch::index ()**`[inline]`

Definition at line 53 of file sosi_element_search.h.

**SosiChildrenList::size_type sosicon::sosi::SosiElementSearch::index (SosiChildrenList::size_type *i*)**`[inline]`

Definition at line 54 of file sosi_element_search.h.

**void sosicon::sosi::SosiElementSearch::next ()**`[inline]`

Definition at line 59 of file sosi_element_search.h.

**sosi::ElementType sosicon::sosi::SosiElementSearch::type ()**`[inline]`

Definition at line 57 of file sosi_element_search.h.

**sosi::ElementType sosicon::sosi::SosiElementSearch::type (sosi::ElementType *t*)**`[inline]`

Definition at line 58 of file sosi_element_search.h.

## Member Data Documentation

**sosi::ElementType sosicon::sosi::SosiElementSearch::mElementType**`[private]`

Definition at line 49 of file sosi_element_search.h.

**SosiChildrenList::size_type sosicon::sosi::SosiElementSearch::mIndex**`[private]`

Definition at line 47 of file sosi_element_search.h.

**ISosiElement* sosicon::sosi::SosiElementSearch::mSosiElement**`[private]`

Definition at line 48 of file sosi_element_search.h.

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

- /prosjekter/sosicon/src/sosi/**sosi_element_search.h**

# sosicon::sosi::SosiJunctionPoint Class Reference

SOSI Junction point.

`#include <sosi_junction_point.h>`

## Public Member Functions

- **SosiJunctionPoint** (**ISosiElement** *e)
  *Construct new SOSI junction point element.*
- **~SosiJunctionPoint** ()
  *Destructor.*

## Private Attributes

- **ISosiElement * mSosiElement**

## Detailed Description

SOSI Junction point.

Implements SOSI junction point layer specification, as given via the KP element. Norwegian: Knutepunktslag.

Definition at line 43 of file sosi_junction_point.h.

## Constructor & Destructor Documentation

### sosicon::sosi::SosiJunctionPoint::SosiJunctionPoint (ISosiElement * *e*)`[inline]`

Construct new SOSI junction point element.

Definition at line 50 of file sosi_junction_point.h.

### sosicon::sosi::SosiJunctionPoint::~SosiJunctionPoint ()

Destructor.

## Member Data Documentation

### ISosiElement* sosicon::sosi::SosiJunctionPoint::mSosiElement`[private]`

Definition at line 45 of file sosi_junction_point.h.

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

- /prosjekter/sosicon/src/sosi/**sosi_junction_point.h**

# sosicon::sosi::SosiNorthEast Class Reference

SOSI North-east element.

`#include <sosi_north_east.h>`

## Public Member Functions

- void **append** (double n, double e)
- void **append** (double n, double e, double h)
- void **append** (std::string n, std::string e)
- void **append** (std::string n, std::string e, std::string h)
  *Frees allocated memory.*
- void **free** ()
- **SosiNorthEast** (**ISosiElement** *e)
  *Construct new SOSI north-east element.*
- virtual **~SosiNorthEast** ()
  *Destructor.*
- void **dump** ()
  *Debug.*
- void **expandBoundingBox** (double &minX, double &minY, double &maxX, double &maxY)
- **ICoordinate * front** ()
- **ICoordinate * back** ()
- bool **getNext** (**ICoordinate** *&coord)
- int **getNumPoints** ()
- void **reverse** ()
  *Reverse polygon (point order)*
- **SosiNorthEast** & **operator+=** (**SosiOrigoNE** &origo)
- **SosiNorthEast** & **operator/=** (**SosiUnit** &unit)

## Private Member Functions

- void **ragelParseCoordinatesNe** (std::string data)
  *Populate mCoordinates.*
- void **ragelParseCoordinatesNeh** (std::string data)
- void **initHeadMember** (**ISosiHeadMember** &headMember, **ElementType** type)

## Private Attributes

- **ISosiElement * mSosiElement**
- **CoordinateList mCoordinates**
- CoordinateList::iterator **mCoordinatesIterator**
- double **mMinX**
- double **mMinY**
- double **mMaxX**
- double **mMaxY**

## Static Private Attributes

- static **SosiOrigoNE mOrigo**
- static **SosiUnit mUnit**

## Detailed Description

SOSI North-east element.

Implements SOSI north east element, as given via the NØ element.

Definition at line 50 of file sosi_north_east.h.

## Constructor & Destructor Documentation

### sosicon::sosi::SosiNorthEast::SosiNorthEast (ISosiElement * *e*)

Construct new SOSI north-east element.

Definition at line 35 of file sosi_north_east.cpp.

### sosicon::sosi::SosiNorthEast::~SosiNorthEast ()`[virtual]`

Destructor.

Definition at line 54 of file sosi_north_east.cpp.

## Member Function Documentation

### void sosicon::sosi::SosiNorthEast::append (double *n*, double *e*)

Definition at line 89 of file sosi_north_east.cpp.

### void sosicon::sosi::SosiNorthEast::append (double *n*, double *e*, double *h*)

Definition at line 94 of file sosi_north_east.cpp.

### void sosicon::sosi::SosiNorthEast::append (std::string *n*, std::string *e*)

Definition at line 57 of file sosi_north_east.cpp.

### void sosicon::sosi::SosiNorthEast::append (std::string *n*, std::string *e*, std::string *h*)

Frees allocated memory.

Definition at line 68 of file sosi_north_east.cpp.

### ICoordinate* sosicon::sosi::SosiNorthEast::back ()`[inline]`

Definition at line 96 of file sosi_north_east.h.

### void sosicon::sosi::SosiNorthEast::dump ()

Debug.

Definition at line 132 of file sosi_north_east.cpp.

### void sosicon::sosi::SosiNorthEast::expandBoundingBox (double & *minX*, double & *minY*, double & *maxX*, double & *maxY*)

Definition at line 139 of file sosi_north_east.cpp.

### void sosicon::sosi::SosiNorthEast::free ()

Definition at line 81 of file sosi_north_east.cpp.

### ICoordinate* sosicon::sosi::SosiNorthEast::front ()`[inline]`

Definition at line 94 of file sosi_north_east.h.

### bool sosicon::sosi::SosiNorthEast::getNext (ICoordinate *& *coord*)

Definition at line 147 of file sosi_north_east.cpp.

### int sosicon::sosi::SosiNorthEast::getNumPoints ()`[inline]`

Definition at line 102 of file sosi_north_east.h.

### void sosicon::sosi::SosiNorthEast::initHeadMember (ISosiHeadMember & *headMember*, ElementType *type*)`[private]`

Definition at line 116 of file sosi_north_east.cpp.

### sosicon::sosi::SosiNorthEast & sosicon::sosi::SosiNorthEast::operator+= (SosiOrigoNE & *origo*)

Definition at line 163 of file sosi_north_east.cpp.

### sosicon::sosi::SosiNorthEast & sosicon::sosi::SosiNorthEast::operator/= (SosiUnit & *unit*)

Definition at line 178 of file sosi_north_east.cpp.

### void sosicon::sosi::SosiNorthEast::ragelParseCoordinatesNe (std::string *data*)`[private]`

Populate mCoordinates.

Definition at line 33 of file sosi_north_east.rl.

**void sosicon::sosi::SosiNorthEast::ragelParseCoordinatesNeh (std::string *data*)**`[private]`

Definition at line 33 of file sosi_north_east_height.rl.

**void sosicon::sosi::SosiNorthEast::reverse ()**`[inline]`

Reverse polygon (point order)

Definition at line 105 of file sosi_north_east.h.

## Member Data Documentation

**CoordinateList sosicon::sosi::SosiNorthEast::mCoordinates**`[private]`

Definition at line 54 of file sosi_north_east.h.

**CoordinateList::iterator sosicon::sosi::SosiNorthEast::mCoordinatesIterator**`[private]`

Definition at line 56 of file sosi_north_east.h.

**double sosicon::sosi::SosiNorthEast::mMaxX**`[private]`

Definition at line 64 of file sosi_north_east.h.

**double sosicon::sosi::SosiNorthEast::mMaxY**`[private]`

Definition at line 65 of file sosi_north_east.h.

**double sosicon::sosi::SosiNorthEast::mMinX**`[private]`

Definition at line 62 of file sosi_north_east.h.

**double sosicon::sosi::SosiNorthEast::mMinY**`[private]`

Definition at line 63 of file sosi_north_east.h.

**sosicon::sosi::SosiOrigoNE sosicon::sosi::SosiNorthEast::mOrigo**`[static]`, `[private]`

Definition at line 58 of file sosi_north_east.h.

**ISosiElement* sosicon::sosi::SosiNorthEast::mSosiElement**`[private]`

Definition at line 52 of file sosi_north_east.h.

### sosicon::sosi::SosiUnit sosicon::sosi::SosiNorthEast::mUnit`[static]`, `[private]`

Definition at line 60 of file sosi_north_east.h.

---

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

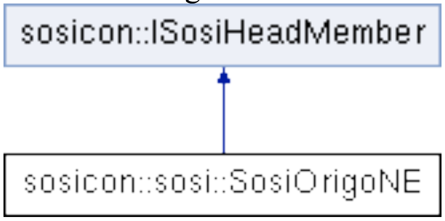
- /prosjekter/sosicon/src/sosi/**sosi_north_east.h**
- /prosjekter/sosicon/src/ragel/**sosi_north_east.rl**
- /prosjekter/sosicon/src/ragel/**sosi_north_east_height.rl**
- /prosjekter/sosicon/src/sosi/**sosi_north_east.cpp**
- /prosjekter/sosicon/src/**sosi_north_east_height_ragel.cpp**
- /prosjekter/sosicon/src/**sosi_north_east_ragel.cpp**

# sosicon::sosi::SosiOrigoNE Class Reference

SOSI Junction point.

`#include <sosi_origo_ne.h>`

Inheritance diagram for sosicon::sosi::SosiOrigoNE:



## Public Member Functions

- **SosiOrigoNE** ()
  *Construct new SOSI junction point element.*
- **SosiOrigoNE** (**ISosiElement** *e)
  *Construct new SOSI junction point element.*
- virtual **~SosiOrigoNE** ()
  *Destructor.*
- int **getN** ()
- int **getE** ()
- virtual void **init** (**ISosiElement** *e)
- virtual bool **initialized** ()
- void **ragelParseSosiOrigoNE** (std::string data)
  *Ragel parse element data.*

## Private Attributes

- **ISosiElement * mSosiElement**
- bool **mInitialized**
- int **mOrigoN**
- int **mOrigoE**

---

## Detailed Description

SOSI Junction point.

Implements SOSI junction point layer specification, as given via the KP element. Norwegian: Knutepunktslag.

Definition at line 44 of file sosi_origo_ne.h.

---

## Constructor & Destructor Documentation

### sosicon::sosi::SosiOrigoNE::SosiOrigoNE ()

Construct new SOSI junction point element.

Definition at line 21 of file sosi_origo_ne.cpp.

### sosicon::sosi::SosiOrigoNE::SosiOrigoNE (ISosiElement * *e*)`[inline]`

Construct new SOSI junction point element.

Definition at line 60 of file sosi_origo_ne.h.

### virtual sosicon::sosi::SosiOrigoNE::~SosiOrigoNE ()`[inline]`, `[virtual]`

Destructor.

Definition at line 63 of file sosi_origo_ne.h.

## Member Function Documentation

### int sosicon::sosi::SosiOrigoNE::getE ()`[inline]`

Definition at line 67 of file sosi_origo_ne.h.

### int sosicon::sosi::SosiOrigoNE::getN ()`[inline]`

Definition at line 65 of file sosi_origo_ne.h.

### void sosicon::sosi::SosiOrigoNE::init (ISosiElement * *e*)`[virtual]`

Implements **sosicon::ISosiHeadMember** (*p.97*).

Definition at line 28 of file sosi_origo_ne.cpp.

### virtual bool sosicon::sosi::SosiOrigoNE::initialized ()`[inline]`, `[virtual]`

Implements **sosicon::ISosiHeadMember** (*p.98*).

Definition at line 71 of file sosi_origo_ne.h.

### void sosicon::sosi::SosiOrigoNE::ragelParseSosiOrigoNE (std::string *data*)

Ragel parse element data.

Definition at line 33 of file sosi_origo_ne.rl.

## Member Data Documentation

### bool sosicon::sosi::SosiOrigoNE::mInitialized`[private]`

Definition at line 48 of file sosi_origo_ne.h.

### int sosicon::sosi::SosiOrigoNE::mOrigoE [private]

Definition at line 52 of file sosi_origo_ne.h.

### int sosicon::sosi::SosiOrigoNE::mOrigoN [private]

Definition at line 50 of file sosi_origo_ne.h.

### ISosiElement* sosicon::sosi::SosiOrigoNE::mSosiElement [private]

Definition at line 46 of file sosi_origo_ne.h.

---

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

- /prosjekter/sosicon/src/sosi/**sosi_origo_ne.h**
- /prosjekter/sosicon/src/ragel/**sosi_origo_ne.rl**
- /prosjekter/sosicon/src/sosi/**sosi_origo_ne.cpp**
- /prosjekter/sosicon/src/**sosi_origo_ne_ragel.cpp**

# sosicon::sosi::SosiRefList Class Reference

SOSI REF list.

`#include <sosi_ref_list.h>`

## Public Member Functions

- **SosiRefList** ()
  *Construct new SOSI REF element.*
- **SosiRefList** (**ISosiElement** *e)
  *Construct new SOSI REF element.*
- **~SosiRefList** ()
  *Destructor.*
- bool **getNextGeometry** (**GeometryRef** *&geometry)
  *Next list of references.*

## Private Member Functions

- void **ragelParseSosiRef** (std::string data)

## Private Attributes

- **ISosiElement * mSosiElement**
- **GeometryCollection mRefListCollection**
- GeometryCollection::size_type **mRefListCollectionIndex**
- GeometryRef::size_type **mRefListIndex**

## Detailed Description

SOSI REF list.

Implements SOSI reference list, as given via the REF element.

Definition at line 42 of file sosi_ref_list.h.

## Constructor & Destructor Documentation

### sosicon::sosi::SosiRefList::SosiRefList () `[inline]`

Construct new SOSI REF element.

Definition at line 56 of file sosi_ref_list.h.

### sosicon::sosi::SosiRefList::SosiRefList (ISosiElement * *e*)

Construct new SOSI REF element.

Definition at line 21 of file sosi_ref_list.cpp.

### sosicon::sosi::SosiRefList::~SosiRefList ()

Destructor.

Definition at line 28 of file sosi_ref_list.cpp.

## Member Function Documentation

### bool sosicon::sosi::SosiRefList::getNextGeometry (GeometryRef *& *geometry*)

Next list of references.

A reference list represents a geometry, i.e. a polygon or its holes.

Definition at line 40 of file sosi_ref_list.cpp.

### void sosicon::sosi::SosiRefList::ragelParseSosiRef (std::string *data*) `[private]`

Definition at line 33 of file sosi_ref.rl.

## Member Data Documentation

### GeometryCollection sosicon::sosi::SosiRefList::mRefListCollection `[private]`

Definition at line 46 of file sosi_ref_list.h.

### GeometryCollection::size_type sosicon::sosi::SosiRefList::mRefListCollectionIndex `[private]`

Definition at line 48 of file sosi_ref_list.h.

### GeometryRef::size_type sosicon::sosi::SosiRefList::mRefListIndex `[private]`

Definition at line 49 of file sosi_ref_list.h.

### ISosiElement* sosicon::sosi::SosiRefList::mSosiElement `[private]`

Definition at line 44 of file sosi_ref_list.h.

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

- /prosjekter/sosicon/src/sosi/**sosi_ref_list.h**
- /prosjekter/sosicon/src/ragel/**sosi_ref.rl**
- /prosjekter/sosicon/src/sosi/**sosi_ref_list.cpp**
- /prosjekter/sosicon/src/**sosi_ref_ragel.cpp**

# sosicon::sosi::SosiTranslationTable Class Reference

```
#include <sosi_translation_table.h>
```

## Public Member Functions

- **SosiTranslationTable** ()
- **CoordSys** & **sysCodeToCoordSys** (int sysCode)
- **ElementType** **sosiNameToType** (std::string typeName)
- std::string **sosiTypeToName** (**ElementType** elementType)
- **ObjType** **sosiObjNameToType** (std::string objTypeName)
- std::string **sosiTypeToObjName** (**ObjType** objType)

## Private Member Functions

- template<typename Key , typename Val > Key **reverseLookup** (std::map< Key, Val > &c, Val v)
  *Scan container looking for value, returning key.*

## Static Private Attributes

- static const int **MAX_COORDSYS_TABLE**
  *Number of entries in KOORDSYS lookup table.*
- static std::map< std::string,
- **ElementType** > **mTypeNameMap**
  *SOSI element name map.*
- static std::map< std::string,
- **ObjType** > **mObjTypeNameMap**
  *OBJTYPE name lookup table.*
- static **CoordSys** **mCoordSysTable** [**MAX_COORDSYS_TABLE**+1]
  *KOORDSYS code lookup table.*

## Detailed Description

Definition at line 36 of file sosi_translation_table.h.

## Constructor & Destructor Documentation

### sosicon::sosi::SosiTranslationTable::SosiTranslationTable ()

Definition at line 26 of file sosi_translation_table.cpp.

## Member Function Documentation

### template<typename Key , typename Val > Key sosicon::sosi::SosiTranslationTable::reverseLookup (std::map< Key, Val > & *c*, Val *v*)`[inline]`, `[private]`

Scan container looking for value, returning key.

Definition at line 64 of file sosi_translation_table.h.

### ElementType sosicon::sosi::SosiTranslationTable::sosiNameToType (std::string *typeName*)`[inline]`

Definition at line 80 of file sosi_translation_table.h.

### ObjType sosicon::sosi::SosiTranslationTable::sosiObjNameToType (std::string *objTypeName*)`[inline]`

Definition at line 89 of file sosi_translation_table.h.

### std::string sosicon::sosi::SosiTranslationTable::sosiTypeToName (ElementType *elementType*)`[inline]`

Definition at line 85 of file sosi_translation_table.h.

### std::string sosicon::sosi::SosiTranslationTable::sosiTypeToObjName (ObjType *objType*)`[inline]`

Definition at line 94 of file sosi_translation_table.h.

### CoordSys& sosicon::sosi::SosiTranslationTable::sysCodeToCoordSys (int *sysCode*)`[inline]`

Definition at line 75 of file sosi_translation_table.h.

## Member Data Documentation

### const int sosicon::sosi::SosiTranslationTable::MAX_COORDSYS_TABLE`[static]`, `[private]`

Number of entries in KOORDSYS lookup table.

Definition at line 39 of file sosi_translation_table.h.

### sosicon::sosi::CoordSys sosicon::sosi::SosiTranslationTable::mCoordSysTable`[static]`, `[private]`

KOORDSYS code lookup table.

Primitive array, int key.

Definition at line 60 of file sosi_translation_table.h.

### std::map< std::string, sosicon::sosi::ObjType > sosicon::sosi::SosiTranslationTable::mObjTypeNameMap `[static]`, `[private]`

OBJTYPE name lookup table.

STL map with string key.

Definition at line 54 of file sosi_translation_table.h.

### std::map< std::string, sosicon::sosi::ElementType > sosicon::sosi::SosiTranslationTable::mTypeNameMap `[static]`, `[private]`

SOSI element name map.

STL map with string key.

Definition at line 48 of file sosi_translation_table.h.

---

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

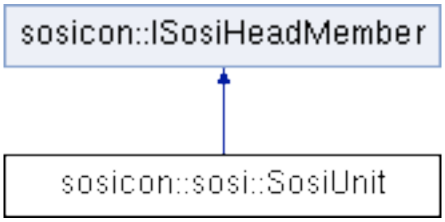
- /prosjekter/sosicon/src/sosi/**sosi_translation_table.h**
- /prosjekter/sosicon/src/sosi/**sosi_translation_table.cpp**

## sosicon::sosi::SosiUnit Class Reference

SOSI Unit.

`#include <sosi_unit.h>`

Inheritance diagram for sosicon::sosi::SosiUnit:



### Public Member Functions

- **SosiUnit** ()
  *Construct new SOSI Unit element.*
- virtual ~**SosiUnit** ()
  *Destructor.*
- **SosiUnit** (**ISosiElement** *e)
  *Construct new SOSI Unit element.*
- int **getDivisor** ()
- virtual void **init** (**ISosiElement** *e)
  *Initnialize SOSI Unit element.*
- virtual bool **initialized** ()

### Private Attributes

- **ISosiElement** * **mSosiElement**
- bool **mInitialized**
- int **mDivisor**

### Detailed Description

SOSI Unit.

Implements SOSI unit (coordinate resolution), as given via the ENHET element.

Definition at line 43 of file sosi_unit.h.

### Constructor & Destructor Documentation

#### sosicon::sosi::SosiUnit::SosiUnit ()

Construct new SOSI Unit element.

Definition at line 21 of file sosi_unit.cpp.

#### virtual sosicon::sosi::SosiUnit::~SosiUnit () `[inline]`, `[virtual]`

Destructor.

Definition at line 57 of file sosi_unit.h.

### sosicon::sosi::SosiUnit::SosiUnit (ISosiElement * *e*)`[inline]`

Construct new SOSI Unit element.

Definition at line 60 of file sosi_unit.h.

## Member Function Documentation

### int sosicon::sosi::SosiUnit::getDivisor ()`[inline]`

Definition at line 62 of file sosi_unit.h.

### void sosicon::sosi::SosiUnit::init (ISosiElement * *e*)`[virtual]`

Initnialize SOSI Unit element.

Implements **sosicon::ISosiHeadMember** (*p.97*).

Definition at line 28 of file sosi_unit.cpp.

### virtual bool sosicon::sosi::SosiUnit::initialized ()`[inline]`, `[virtual]`

Implements **sosicon::ISosiHeadMember** (*p.98*).

Definition at line 67 of file sosi_unit.h.

## Member Data Documentation

### int sosicon::sosi::SosiUnit::mDivisor`[private]`

Definition at line 49 of file sosi_unit.h.

### bool sosicon::sosi::SosiUnit::mInitialized`[private]`

Definition at line 47 of file sosi_unit.h.

### ISosiElement* sosicon::sosi::SosiUnit::mSosiElement`[private]`

Definition at line 45 of file sosi_unit.h.

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

- /prosjekter/sosicon/src/sosi/**sosi_unit.h**

- /prosjekter/sosicon/src/sosi/**sosi_unit.cpp**

# File Documentation

## /prosjekter/sosicon/src/byte_order.cpp File Reference

```
#include "byte_order.h"
```

# /prosjekter/sosicon/src/byte_order.h File Reference

```
#include <inttypes.h>
#include <algorithm>
#include <cmath>
#include <iostream>
```

## Namespaces

- **sosicon**
- *Application root.* **sosicon::byteOrder**

## *Big/low-endian conversions.* Enumerations

- enum **sosicon::byteOrder::Endianness** { **sosicon::byteOrder::not_set**, **sosicon::byteOrder::big**, **sosicon::byteOrder::little** }

## *Big/little flag.* Functions

- Endianness **sosicon::byteOrder::determine** ()
  *Determines system endianness.*
- void **sosicon::byteOrder::doubleToLittleEndian** (double from, char *to)
  *Writes little endian representation of double.*
- void **sosicon::byteOrder::toBigEndian** (const char *from, char *to, size_t bufSize)
  *Reverses buffer to big endian if required.*
- void **sosicon::byteOrder::toLittleEndian** (const char *from, char *to, size_t bufSize)
  *Reverses buffer to little endian if required.*

## Variables

- enum **sosicon::byteOrder::Endianness sosicon::byteOrder::endianness**
  *Stores system endianness.*

# /prosjekter/sosicon/src/command_line.cpp File Reference

```
#include "command_line.h"
```

## /prosjekter/sosicon/src/command_line.h File Reference

```
#include <stdio.h>
#include <iostream>
#include <vector>
#include <string>
#include <unistd.h>
#include "utils.h"
```

### Classes

- class **sosicon::CommandLine**

### *Command-line parser.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/common_types.h File Reference

```
#include <vector>
#include "interface/i_coordinate.h"
```

## Namespaces

- **sosicon**

## *Application root.* Typedefs

- typedef std::vector
- < ICoordinate * > **sosicon::CoordinateList**

# /prosjekter/sosicon/src/converter_sosi2shp.cpp File Reference

```
#include "converter_sosi2shp.h"
```

## /prosjekter/sosicon/src/converter_sosi2shp.h File Reference

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <vector>
#include <sstream>
#include <string>
#include "interface/i_converter.h"
#include "interface/i_sosi_element.h"
#include "command_line.h"
#include "parser.h"
#include "utils.h"
#include "shape/shapefile.h"
```

### Classes

- class **sosicon::ConverterSosi2shp**

### *SOSI to ESRI Shape converter.* Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/converter_sosi2tsv.cpp File Reference

```
#include "converter_sosi2tsv.h"
```

## /prosjekter/sosicon/src/converter_sosi2tsv.h File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
#include "interface/i_converter.h"
#include "command_line.h"
#include "parser.h"
```

### Classes

- class **sosicon::ConverterSosi2tsv**

### *SOSI to TSV converter.* Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/converter_sosi2xml.cpp File Reference

```
#include "converter_sosi2xml.h"
```

# /prosjekter/sosicon/src/converter_sosi2xml.h File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
#include "interface/i_converter.h"
#include "interface/i_sosi_element.h"
#include "command_line.h"
#include "utils.h"
#include "parser.h"
```

## Classes

- class **sosicon::ConverterSosi2xml**

## *SOSI to ESRI Shape converter.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/converter_sosi_stat.cpp File Reference

```
#include "converter_sosi_stat.h"
```

# /prosjekter/sosicon/src/converter_sosi_stat.h File Reference

```
#include <iostream>
#include <fstream>
#include <map>
#include "interface/i_converter.h"
#include "interface/i_sosi_element.h"
#include "sosi/sosi_types.h"
#include "sosi/sosi_element_search.h"
#include "command_line.h"
#include "utils.h"
#include "parser.h"
```

## Classes

- class **sosicon::ConverterSosiStat**

## *SOSI to ESRI Shape converter.* Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/coordinate.h File Reference

```
#include <string>
#include <iostream>
#include <sstream>
#include <ios>
#include "interface/i_coordinate.h"
```

### Classes

- class **sosicon::Coordinate**

### *Coordinate* ***container.*** Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/coordinate_collection.cpp File Reference

```
#include "coordinate_collection.h"
```

## /prosjekter/sosicon/src/coordinate_collection.h File Reference

```
#include <algorithm>
#include <limits>
#include <vector>
#include <iostream>
#include "common_types.h"
#include "sosi/sosi_types.h"
#include "sosi/sosi_element_search.h"
#include "sosi/sosi_ref_list.h"
#include "sosi/sosi_north_east.h"
#include "interface/i_coordinate.h"
#include "interface/i_sosi_element.h"
```

### Classes

- class **sosicon::CoordinateCollection**

### *Coordinate* ***container.*** Namespaces

- **sosicon**

### ***Application root.*** Functions

- bool **sosicon::getNext** (ICoordinate *&coord, sosi::NorthEastList &list, sosi::NorthEastList::iterator &i)
  *Get next coordinate in list.*
- bool **sosicon::getNextOffset** (int &offset, std::vector< int > &offsets, std::vector< int >::iterator &iterator)
  *Get next offset in part offsets list.*

## /prosjekter/sosicon/src/factory.cpp File Reference

```
#include "factory.h"
```

## /prosjekter/sosicon/src/factory.h File Reference

```
#include "interface/i_converter.h"
#include "converter_sosi2shp.h"
#include "converter_sosi2xml.h"
#include "converter_sosi2tsv.h"
#include "converter_sosi_stat.h"
```

### Classes

- class **sosicon::Factory**

### *Factory* ***class.*** Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/interface/i_binary_streamable.h File Reference

```
#include <iostream>
```

### Classes

- class **sosicon::IBinaryStreamable**

### *Interface: Binary streamable object.* Namespaces

- **sosicon**

### *Application root.* Functions

- std::ostream & **sosicon::operator<<** (std::ostream &os, IBinaryStreamable &binaryStreamable)
  *Stream output operator.*

# /prosjekter/sosicon/src/interface/i_converter.h File Reference

```
#include "../command_line.h"
```

## Classes

- class **sosicon::IConverter**

## *Interface: Converter.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_coordinate.h File Reference

```
#include <string>
```

## Classes

- class **sosicon::ICoordinate**

## ***Interface:*** *Coordinate***.** Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_lookup_table.h File Reference

`#include <string>`

## Classes

- class **sosicon::ILookupTable**

## *Interface: Lookup table.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_rectangle.h File Reference

## Classes

- class **sosicon::IRectangle**

## *Interface: Rectangle.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_shape_element.h File Reference

```
#include "i_binary_streamable.h"
#include "i_sosi_element.h"
#include "i_rectangle.h"
```

## Classes

- class **sosicon::IShapeElement**

## *Interface: Shape element.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_shape_element_header.h File Reference

```
#include "i_binary_streamable.h"
```

## Classes

- class **sosicon::IShapeElementHeader**

## *Interface: Shape element header.* Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/interface/i_shape_header.h File Reference

```
#include "i_binary_streamable.h"
#include "../shape/shapefile_types.h"
#include "../shape/bounding_box.h"
```

### Classes

- class **sosicon::IShapeHeader**

### *Interface: Shape element.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_shapefile.h File Reference

```
#include "i_shapefile_shp_part.h"
#include "i_shapefile_shx_part.h"
#include "i_shapefile_dbf_part.h"
#include "i_shapefile_prj_part.h"
#include "i_sosi_element.h"
#include "../sosi/sosi_types.h"
```

## Classes

- class **sosicon::IShapefile**

## *Interface: Shapefile.* Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/interface/i_shapefile_dbf_part.h File Reference

```
#include <iostream>
#include "i_binary_streamable.h"
```

### Classes

- class **sosicon::IShapefileDbfPart**

### *Interface: ShapefileDbfPart.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_shapefile_prj_part.h File Reference

```
#include <iostream>
#include "i_binary_streamable.h"
```

## Classes

- class **sosicon::IShapefilePrjPart**

## *Interface: ShapefilePrjPart.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_shapefile_shp_part.h File Reference

```
#include <iostream>
#include "i_binary_streamable.h"
```

## Classes

- class **sosicon::IShapefileShpPart**

## *Interface: ShapefileShpPart.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_shapefile_shx_part.h File Reference

```
#include <iostream>
#include "i_binary_streamable.h"
```

## Classes

- class **sosicon::IShapefileShxPart**

## *Interface: ShapefileShxPart.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_sosi_element.h File Reference

```
#include <string>
#include <vector>
#include <map>
#include "../sosi/sosi_types.h"
#include "../sosi/sosi_element_search.h"
```

## Classes

- class **sosicon::ISosiElement**

## *Interface: SOSI element.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/interface/i_sosi_head_member.h File Reference

```
#include "i_sosi_element.h"
```

## Classes

- class **sosicon::ISosiHeadMember**

## *Interface: SOSI header element.* Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/inttypes.h File Reference

```
#include "stdint.h"
```

## Classes

- struct **imaxdiv_t**

## Macros

- #define **PRId8**
- #define **PRIi8**
- #define **PRIdLEAST8**
- #define **PRIiLEAST8**
- #define **PRIdFAST8**
- #define **PRIiFAST8**
- #define **PRId16**
- #define **PRIi16**
- #define **PRIdLEAST16**
- #define **PRIiLEAST16**
- #define **PRIdFAST16**
- #define **PRIiFAST16**
- #define **PRId32**
- #define **PRIi32**
- #define **PRIdLEAST32**
- #define **PRIiLEAST32**
- #define **PRIdFAST32**
- #define **PRIiFAST32**
- #define **PRId64**
- #define **PRIi64**
- #define **PRIdLEAST64**
- #define **PRIiLEAST64**
- #define **PRIdFAST64**
- #define **PRIiFAST64**
- #define **PRIdMAX**
- #define **PRIiMAX**
- #define **PRIdPTR**
- #define **PRIiPTR**
- #define **PRIo8**
- #define **PRIu8**
- #define **PRIx8**
- #define **PRIX8**
- #define **PRIoLEAST8**
- #define **PRIuLEAST8**
- #define **PRIxLEAST8**
- #define **PRIXLEAST8**
- #define **PRIoFAST8**
- #define **PRIuFAST8**
- #define **PRIxFAST8**
- #define **PRIXFAST8**
- #define **PRIo16**
- #define **PRIu16**
- #define **PRIx16**

- #define **PRIX16**
- #define **PRIoLEAST16**
- #define **PRIuLEAST16**
- #define **PRIxLEAST16**
- #define **PRIXLEAST16**
- #define **PRIoFAST16**
- #define **PRIuFAST16**
- #define **PRIxFAST16**
- #define **PRIXFAST16**
- #define **PRIo32**
- #define **PRIu32**
- #define **PRIx32**
- #define **PRIX32**
- #define **PRIoLEAST32**
- #define **PRIuLEAST32**
- #define **PRIxLEAST32**
- #define **PRIXLEAST32**
- #define **PRIoFAST32**
- #define **PRIuFAST32**
- #define **PRIxFAST32**
- #define **PRIXFAST32**
- #define **PRIo64**
- #define **PRIu64**
- #define **PRIx64**
- #define **PRIX64**
- #define **PRIoLEAST64**
- #define **PRIuLEAST64**
- #define **PRIxLEAST64**
- #define **PRIXLEAST64**
- #define **PRIoFAST64**
- #define **PRIuFAST64**
- #define **PRIxFAST64**
- #define **PRIXFAST64**
- #define **PRIoMAX**
- #define **PRIuMAX**
- #define **PRIxMAX**
- #define **PRIXMAX**
- #define **PRIoPTR**
- #define **PRIuPTR**
- #define **PRIxPTR**
- #define **PRIXPTR**
- #define **SCNd8**
- #define **SCNi8**
- #define **SCNdLEAST8**
- #define **SCNiLEAST8**
- #define **SCNdFAST8**
- #define **SCNiFAST8**
- #define **SCNd16**
- #define **SCNi16**
- #define **SCNdLEAST16**
- #define **SCNiLEAST16**
- #define **SCNdFAST16**
- #define **SCNiFAST16**

- #define **SCNd32**
- #define **SCNi32**
- #define **SCNdLEAST32**
- #define **SCNiLEAST32**
- #define **SCNdFAST32**
- #define **SCNiFAST32**
- #define **SCNd64**
- #define **SCNi64**
- #define **SCNdLEAST64**
- #define **SCNiLEAST64**
- #define **SCNdFAST64**
- #define **SCNiFAST64**
- #define **SCNdMAX**
- #define **SCNiMAX**
- #define **SCNdPTR**
- #define **SCNiPTR**
- #define **SCNo8**
- #define **SCNu8**
- #define **SCNx8**
- #define **SCNX8**
- #define **SCNoLEAST8**
- #define **SCNuLEAST8**
- #define **SCNxLEAST8**
- #define **SCNXLEAST8**
- #define **SCNoFAST8**
- #define **SCNuFAST8**
- #define **SCNxFAST8**
- #define **SCNXFAST8**
- #define **SCNo16**
- #define **SCNu16**
- #define **SCNx16**
- #define **SCNX16**
- #define **SCNoLEAST16**
- #define **SCNuLEAST16**
- #define **SCNxLEAST16**
- #define **SCNXLEAST16**
- #define **SCNoFAST16**
- #define **SCNuFAST16**
- #define **SCNxFAST16**
- #define **SCNXFAST16**
- #define **SCNo32**
- #define **SCNu32**
- #define **SCNx32**
- #define **SCNX32**
- #define **SCNoLEAST32**
- #define **SCNuLEAST32**
- #define **SCNxLEAST32**
- #define **SCNXLEAST32**
- #define **SCNoFAST32**
- #define **SCNuFAST32**
- #define **SCNxFAST32**
- #define **SCNXFAST32**
- #define **SCNo64**

- #define **SCNu64**
- #define **SCNx64**
- #define **SCNX64**
- #define **SCNoLEAST64**
- #define **SCNuLEAST64**
- #define **SCNxLEAST64**
- #define **SCNXLEAST64**
- #define **SCNoFAST64**
- #define **SCNuFAST64**
- #define **SCNxFAST64**
- #define **SCNXFAST64**
- #define **SCNoMAX**
- #define **SCNuMAX**
- #define **SCNxMAX**
- #define **SCNXMAX**
- #define **SCNoPTR**
- #define **SCNuPTR**
- #define **SCNxPTR**
- #define **SCNXPTR**
- #define **imaxabs**
- #define **strtoimax**
- #define **strtoumax**
- #define **wcstoimax**
- #define **wcstoumax**

## Functions

- **imaxdiv_t** __cdecl **imaxdiv** (intmax_t numer, intmax_t denom)

---

## Macro Definition Documentation

### #define imaxabs

Definition at line 269 of file inttypes.h.

### #define PRId16

Definition at line 64 of file inttypes.h.

### #define PRId32

Definition at line 71 of file inttypes.h.

### #define PRId64

Definition at line 78 of file inttypes.h.

#### #define PRId8

Definition at line 57 of file inttypes.h.

#### #define PRIdFAST16

Definition at line 68 of file inttypes.h.

#### #define PRIdFAST32

Definition at line 75 of file inttypes.h.

#### #define PRIdFAST64

Definition at line 82 of file inttypes.h.

#### #define PRIdFAST8

Definition at line 61 of file inttypes.h.

#### #define PRIdLEAST16

Definition at line 66 of file inttypes.h.

#### #define PRIdLEAST32

Definition at line 73 of file inttypes.h.

#### #define PRIdLEAST64

Definition at line 80 of file inttypes.h.

#### #define PRIdLEAST8

Definition at line 59 of file inttypes.h.

#### #define PRIdMAX

Definition at line 85 of file inttypes.h.

#### #define PRIdPTR

Definition at line 88 of file inttypes.h.

#### #define PRIi16

Definition at line 65 of file inttypes.h.

#### #define PRIi32

Definition at line 72 of file inttypes.h.

#### #define PRIi64

Definition at line 79 of file inttypes.h.

#### #define PRIi8

Definition at line 58 of file inttypes.h.

#### #define PRIiFAST16

Definition at line 69 of file inttypes.h.

#### #define PRIiFAST32

Definition at line 76 of file inttypes.h.

#### #define PRIiFAST64

Definition at line 83 of file inttypes.h.

#### #define PRIiFAST8

Definition at line 62 of file inttypes.h.

#### #define PRIiLEAST16

Definition at line 67 of file inttypes.h.

#### #define PRIiLEAST32

Definition at line 74 of file inttypes.h.

#### #define PRIiLEAST64

Definition at line 81 of file inttypes.h.

#### #define PRIiLEAST8

Definition at line 60 of file inttypes.h.

#### #define PRIiMAX

Definition at line 86 of file inttypes.h.

#### #define PRIiPTR

Definition at line 89 of file inttypes.h.

#### #define PRIo16

Definition at line 105 of file inttypes.h.

#### #define PRIo32

Definition at line 118 of file inttypes.h.

#### #define PRIo64

Definition at line 131 of file inttypes.h.

#### #define PRIo8

Definition at line 92 of file inttypes.h.

#### #define PRIoFAST16

Definition at line 113 of file inttypes.h.

#### #define PRIoFAST32

Definition at line 126 of file inttypes.h.

#### #define PRIoFAST64

Definition at line 139 of file inttypes.h.

#### #define PRIoFAST8

Definition at line 100 of file inttypes.h.

#### #define PRIoLEAST16

Definition at line 109 of file inttypes.h.

#### #define PRIoLEAST32

Definition at line 122 of file inttypes.h.

#### #define PRIoLEAST64

Definition at line 135 of file inttypes.h.

#### #define PRIoLEAST8

Definition at line 96 of file inttypes.h.

#### #define PRIoMAX

Definition at line 144 of file inttypes.h.

#### #define PRIoPTR

Definition at line 149 of file inttypes.h.

#### #define PRIu16

Definition at line 106 of file inttypes.h.

#### #define PRIu32

Definition at line 119 of file inttypes.h.

#### #define PRIu64

Definition at line 132 of file inttypes.h.

#### #define PRIu8

Definition at line 93 of file inttypes.h.

#### #define PRIuFAST16

Definition at line 114 of file inttypes.h.

#### #define PRIuFAST32

Definition at line 127 of file inttypes.h.

#### #define PRIuFAST64

Definition at line 140 of file inttypes.h.

#### #define PRIuFAST8

Definition at line 101 of file inttypes.h.

#### #define PRIuLEAST16

Definition at line 110 of file inttypes.h.

#### #define PRIuLEAST32

Definition at line 123 of file inttypes.h.

#### #define PRIuLEAST64

Definition at line 136 of file inttypes.h.

#### #define PRIuLEAST8

Definition at line 97 of file inttypes.h.

#### #define PRIuMAX

Definition at line 145 of file inttypes.h.

#### #define PRIuPTR

Definition at line 150 of file inttypes.h.

#### #define PRIx16

Definition at line 107 of file inttypes.h.

#### #define PRIX16

Definition at line 108 of file inttypes.h.

#### #define PRIx32

Definition at line 120 of file inttypes.h.

#### #define PRIX32

Definition at line 121 of file inttypes.h.

#### #define PRIx64

Definition at line 133 of file inttypes.h.

#### #define PRIX64

Definition at line 134 of file inttypes.h.

#### #define PRIx8

Definition at line 94 of file inttypes.h.

#### #define PRIX8

Definition at line 95 of file inttypes.h.

#### #define PRIxFAST16

Definition at line 115 of file inttypes.h.

#### #define PRIXFAST16

Definition at line 116 of file inttypes.h.

#### #define PRIxFAST32

Definition at line 128 of file inttypes.h.

#### #define PRIXFAST32

Definition at line 129 of file inttypes.h.

#### #define PRIxFAST64

Definition at line 141 of file inttypes.h.

### #define PRIXFAST64

Definition at line 142 of file inttypes.h.

### #define PRIxFAST8

Definition at line 102 of file inttypes.h.

### #define PRIXFAST8

Definition at line 103 of file inttypes.h.

### #define PRIxLEAST16

Definition at line 111 of file inttypes.h.

### #define PRIXLEAST16

Definition at line 112 of file inttypes.h.

### #define PRIxLEAST32

Definition at line 124 of file inttypes.h.

### #define PRIXLEAST32

Definition at line 125 of file inttypes.h.

### #define PRIxLEAST64

Definition at line 137 of file inttypes.h.

### #define PRIXLEAST64

Definition at line 138 of file inttypes.h.

### #define PRIxLEAST8

Definition at line 98 of file inttypes.h.

### #define PRIXLEAST8

Definition at line 99 of file inttypes.h.

#### #define PRIxMAX

Definition at line 146 of file inttypes.h.

#### #define PRIXMAX

Definition at line 147 of file inttypes.h.

#### #define PRIxPTR

Definition at line 151 of file inttypes.h.

#### #define PRIXPTR

Definition at line 152 of file inttypes.h.

#### #define SCNd16

Definition at line 162 of file inttypes.h.

#### #define SCNd32

Definition at line 169 of file inttypes.h.

#### #define SCNd64

Definition at line 176 of file inttypes.h.

#### #define SCNd8

Definition at line 155 of file inttypes.h.

#### #define SCNdFAST16

Definition at line 166 of file inttypes.h.

#### #define SCNdFAST32

Definition at line 173 of file inttypes.h.

#### #define SCNdFAST64

Definition at line 180 of file inttypes.h.

#### #define SCNdFAST8

Definition at line 159 of file inttypes.h.

#### #define SCNdLEAST16

Definition at line 164 of file inttypes.h.

#### #define SCNdLEAST32

Definition at line 171 of file inttypes.h.

#### #define SCNdLEAST64

Definition at line 178 of file inttypes.h.

#### #define SCNdLEAST8

Definition at line 157 of file inttypes.h.

#### #define SCNdMAX

Definition at line 183 of file inttypes.h.

#### #define SCNdPTR

Definition at line 190 of file inttypes.h.

#### #define SCNi16

Definition at line 163 of file inttypes.h.

#### #define SCNi32

Definition at line 170 of file inttypes.h.

#### #define SCNi64

Definition at line 177 of file inttypes.h.

#### #define SCNi8

Definition at line 156 of file inttypes.h.

#### #define SCNiFAST16

Definition at line 167 of file inttypes.h.

#### #define SCNiFAST32

Definition at line 174 of file inttypes.h.

#### #define SCNiFAST64

Definition at line 181 of file inttypes.h.

#### #define SCNiFAST8

Definition at line 160 of file inttypes.h.

#### #define SCNiLEAST16

Definition at line 165 of file inttypes.h.

#### #define SCNiLEAST32

Definition at line 172 of file inttypes.h.

#### #define SCNiLEAST64

Definition at line 179 of file inttypes.h.

#### #define SCNiLEAST8

Definition at line 158 of file inttypes.h.

#### #define SCNiMAX

Definition at line 184 of file inttypes.h.

#### #define SCNiPTR

Definition at line 191 of file inttypes.h.

#### #define SCNo16

Definition at line 208 of file inttypes.h.

#### #define SCNo32

Definition at line 221 of file inttypes.h.

#### #define SCNo64

Definition at line 234 of file inttypes.h.

#### #define SCNo8

Definition at line 195 of file inttypes.h.

#### #define SCNoFAST16

Definition at line 216 of file inttypes.h.

#### #define SCNoFAST32

Definition at line 229 of file inttypes.h.

#### #define SCNoFAST64

Definition at line 242 of file inttypes.h.

#### #define SCNoFAST8

Definition at line 203 of file inttypes.h.

#### #define SCNoLEAST16

Definition at line 212 of file inttypes.h.

#### #define SCNoLEAST32

Definition at line 225 of file inttypes.h.

#### #define SCNoLEAST64

Definition at line 238 of file inttypes.h.

#### #define SCNoLEAST8

Definition at line 199 of file inttypes.h.

#### #define SCNoMAX

Definition at line 247 of file inttypes.h.

#### #define SCNoPTR

Definition at line 258 of file inttypes.h.

#### #define SCNu16

Definition at line 209 of file inttypes.h.

#### #define SCNu32

Definition at line 222 of file inttypes.h.

#### #define SCNu64

Definition at line 235 of file inttypes.h.

#### #define SCNu8

Definition at line 196 of file inttypes.h.

#### #define SCNuFAST16

Definition at line 217 of file inttypes.h.

#### #define SCNuFAST32

Definition at line 230 of file inttypes.h.

#### #define SCNuFAST64

Definition at line 243 of file inttypes.h.

#### #define SCNuFAST8

Definition at line 204 of file inttypes.h.

#### #define SCNuLEAST16

Definition at line 213 of file inttypes.h.

#### #define SCNuLEAST32

Definition at line 226 of file inttypes.h.

#### #define SCNuLEAST64

Definition at line 239 of file inttypes.h.

#### #define SCNuLEAST8

Definition at line 200 of file inttypes.h.

#### #define SCNuMAX

Definition at line 248 of file inttypes.h.

#### #define SCNuPTR

Definition at line 259 of file inttypes.h.

#### #define SCNx16

Definition at line 210 of file inttypes.h.

#### #define SCNX16

Definition at line 211 of file inttypes.h.

#### #define SCNx32

Definition at line 223 of file inttypes.h.

#### #define SCNX32

Definition at line 224 of file inttypes.h.

#### #define SCNx64

Definition at line 236 of file inttypes.h.

#### #define SCNX64

Definition at line 237 of file inttypes.h.

#### #define SCNx8

Definition at line 197 of file inttypes.h.

#### #define SCNX8

Definition at line 198 of file inttypes.h.

#### #define SCNxFAST16

Definition at line 218 of file inttypes.h.

#### #define SCNXFAST16

Definition at line 219 of file inttypes.h.

#### #define SCNxFAST32

Definition at line 231 of file inttypes.h.

#### #define SCNXFAST32

Definition at line 232 of file inttypes.h.

#### #define SCNxFAST64

Definition at line 244 of file inttypes.h.

#### #define SCNXFAST64

Definition at line 245 of file inttypes.h.

#### #define SCNxFAST8

Definition at line 205 of file inttypes.h.

#### #define SCNXFAST8

Definition at line 206 of file inttypes.h.

#### #define SCNxLEAST16

Definition at line 214 of file inttypes.h.

### #define SCNXLEAST16

Definition at line 215 of file inttypes.h.

### #define SCNxLEAST32

Definition at line 227 of file inttypes.h.

### #define SCNXLEAST32

Definition at line 228 of file inttypes.h.

### #define SCNxLEAST64

Definition at line 240 of file inttypes.h.

### #define SCNXLEAST64

Definition at line 241 of file inttypes.h.

### #define SCNxLEAST8

Definition at line 201 of file inttypes.h.

### #define SCNXLEAST8

Definition at line 202 of file inttypes.h.

### #define SCNxMAX

Definition at line 249 of file inttypes.h.

### #define SCNXMAX

Definition at line 250 of file inttypes.h.

### #define SCNxPTR

Definition at line 260 of file inttypes.h.

### #define SCNXPTR

Definition at line 261 of file inttypes.h.

### #define strtoimax

Definition at line 297 of file inttypes.h.

### #define strtoumax

Definition at line 298 of file inttypes.h.

### #define wcstoimax

Definition at line 301 of file inttypes.h.

### #define wcstoumax

Definition at line 302 of file inttypes.h.

## Function Documentation

### imaxdiv_t __cdecl imaxdiv (intmax_t *numer*, intmax_t *denom*)`[inline]`

Definition at line 280 of file inttypes.h.

# /prosjekter/sosicon/src/main.cpp File Reference

```
#include "main.h"
```

## Functions

- int **main** (int argc, char *argv[])

---

## Function Documentation

### int main (int *argc*, char * *argv*[])

Definition at line 20 of file main.cpp.

# /prosjekter/sosicon/src/main.h File Reference

```
#include <exception>
#include <ios>
#include <iostream>
#include <locale>
#include "command_line.h"
#include "factory.h"
#include "interface/i_converter.h"
```

## Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/parser.cpp File Reference

```
#include "parser.h"
```

# /prosjekter/sosicon/src/parser.h File Reference

```
#include <iostream>
#include <algorithm>
#include <string>
#include <sstream>
#include <vector>
#include <map>
#include "utils.h"
#include "command_line.h"
#include "sosi/sosi_element.h"
#include "interface/i_sosi_element.h"
```

## Classes

- class **sosicon::Parser**

## *SOSI file parser.* Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/parser_ragel.cpp File Reference

```
#include "parser.h"
```

### Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/ragel/parser.rl File Reference

```
#include "parser.h"
```

## Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/ragel/sosi_north_east.rl File Reference

```
#include "sosi/sosi_north_east.h"
```

## Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/ragel/sosi_north_east_height.rl File Reference

```
#include "sosi/sosi_north_east.h"
```

### Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/ragel/sosi_origo_ne.rl File Reference

```
#include "sosi/sosi_origo_ne.h"
```

## Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/ragel/sosi_ref.rl File Reference

```
#include "sosi/sosi_ref_list.h"
```

## Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/shape/shapefile.cpp File Reference

```
#include "shapefile.h"
```

# /prosjekter/sosicon/src/shape/shapefile.h File Reference

```
#include <algorithm>
#include <ctime>
#include <string>
#include <vector>
#include <iostream>
#include "shapefile_types.h"
#include "../byte_order.h"
#include "../utils.h"
#include "../coordinate_collection.h"
#include "../sosi/sosi_types.h"
#include "../sosi/sosi_element.h"
#include "../sosi/sosi_element_search.h"
#include "../interface/i_shapefile.h"
#include "../interface/i_coordinate.h"
```

## Classes

- class **sosicon::shape::Shapefile**

## *Shapefile* ***implementation.*** Namespaces

- **sosicon**
- *Application root.* **sosicon::shape**

## ***ESRI Shape.*** Functions

- ShapeType **sosicon::shape::getShapeEquivalent** (sosi::ElementType sosiType)
  *Resolve geometry type.*

# /prosjekter/sosicon/src/shape/shapefile_types.h File Reference

```
#include <stdint.h>
#include <map>
#include <algorithm>
#include <limits>
```

## Classes

- union **sosicon::shape::Int8Field**
- *8 bit integer / byte field* union **sosicon::shape::Int16Field**
- *16 bit integer / byte field* union **sosicon::shape::Int32Field**
- *32 bit integer / byte field* union **sosicon::shape::Int32TField**
- *32 bit integer / byte / geom::ShapeType field* union **sosicon::shape::DoubleField**
- *32 bit double / byte field* struct **sosicon::shape::ShxIndex**

## Namespaces

- **sosicon**
- *Application root.* **sosicon::shape**

## *ESRI Shape.* Typedefs

- typedef std::map< std::string,
- std::string > **sosicon::shape::DbfRecord**
- typedef std::vector< DbfRecord > **sosicon::shape::DbfRecordSet**
- typedef std::map< std::string,
- int > **sosicon::shape::DbfFieldLengths**
- typedef std::vector< ShxIndex > **sosicon::shape::ShxOffsets**

## Enumerations

- enum **sosicon::shape::ShapeType** { **sosicon::shape::shape_type_none**, **sosicon::shape::shape_type_nullShape**, **sosicon::shape::shape_type_point**, **sosicon::shape::shape_type_polyLine**, **sosicon::shape::shape_type_polygon**, **sosicon::shape::shape_type_multipoint**, **sosicon::shape::shape_type_pointZ**, **sosicon::shape::shape_type_polyLineZ**, **sosicon::shape::shape_type_polygonZ**, **sosicon::shape::shape_type_multipointZ**, **sosicon::shape::shape_type_pointM**, **sosicon::shape::shape_type_polyLineM**, **sosicon::shape::shape_type_polygonM**, **sosicon::shape::shape_type_multiPointM**, **sosicon::shape::shape_type_multiPatch** }

*Geometry types.*

# /prosjekter/sosicon/src/sosi/sosi_element.cpp File Reference

```
#include "sosi_element.h"
```

# /prosjekter/sosicon/src/sosi/sosi_element.h File Reference

```
#include <iostream>
#include <vector>
#include <string>
#include "sosi_element_search.h"
#include "sosi_translation_table.h"
#include "sosi_types.h"
#include "../interface/i_sosi_element.h"
```

## Classes

- class **sosicon::sosi::SosiElement**

## *Basic SOSI element.* Namespaces

- **sosicon**
- *Application root.* **sosicon::sosi**

## *SOSI.* Functions

- CoordSys **sosicon::sosi::sysCodeToCoordSys** (int sysCode)
  *Convert SOSI SYSKODE value to coordinate system data.*
- ElementType **sosicon::sosi::sosiNameToType** (std::string sosiElementName)
  *Convert SOSI element names to ElementType enum value.*
- ObjType **sosicon::sosi::sosiObjNameToType** (std::string sosiObjTypeName)
  *Convert SOSI objtype names to ObjType enum value.*

## /prosjekter/sosicon/src/sosi/sosi_element_search.h File Reference

```
#include <map>
#include <vector>
#include "sosi_types.h"
```

### Classes

- class **sosicon::sosi::SosiElementSearch**

### Namespaces

- **sosicon**
- *Application root.* **sosicon::sosi**

### *SOSI.* Typedefs

- typedef std::map< std::string,
- ISosiElement * > **sosicon::sosi::SosiElementMap**
  *Element index type.*
- typedef std::vector
- < ISosiElement * > **sosicon::sosi::SosiChildrenList**
- typedef SosiChildrenList::iterator **sosicon::sosi::SosiChildrenIterator**

# /prosjekter/sosicon/src/sosi/sosi_junction_point.h File Reference

```
#include "../interface/i_sosi_element.h"
#include "sosi_types.h"
#include <iostream>
#include <string>
#include <vector>
```

## Classes

- class **sosicon::sosi::SosiJunctionPoint**

## *SOSI Junction point.* Namespaces

- **sosicon**
- *Application root.* **sosicon::sosi**

*SOSI.*

## /prosjekter/sosicon/src/sosi/sosi_north_east.cpp File Reference

```
#include "sosi_north_east.h"
```

# /prosjekter/sosicon/src/sosi/sosi_north_east.h File Reference

```
#include "../interface/i_sosi_element.h"
#include "../interface/i_coordinate.h"
#include "../common_types.h"
#include "../coordinate.h"
#include "sosi_types.h"
#include "sosi_origo_ne.h"
#include "sosi_unit.h"
#include <algorithm>
#include <limits>
#include <iostream>
#include <string>
#include <sstream>
#include <vector>
```

## Classes

- class **sosicon::sosi::SosiNorthEast**

## *SOSI North-east element.* Namespaces

- **sosicon**
- *Application root.* **sosicon::sosi**

## *SOSI.* Typedefs

- typedef std::vector
- < SosiNorthEast * > **sosicon::sosi::NorthEastList**
  *List of SosiSNorthEast elements.*

## Functions

- void **sosicon::sosi::deleteNorthEasts** (NorthEastList &lst)
  *Deletes* ***SosiNorthEast*** *elements of NorthEastList.*

# /prosjekter/sosicon/src/sosi/sosi_origo_ne.cpp File Reference

```
#include "sosi_origo_ne.h"
```

# /prosjekter/sosicon/src/sosi/sosi_origo_ne.h File Reference

```
#include "../interface/i_sosi_element.h"
#include "../interface/i_sosi_head_member.h"
#include "sosi_types.h"
#include <iostream>
#include <sstream>
#include <string>
```

## Classes

- class **sosicon::sosi::SosiOrigoNE**

## *SOSI Junction point.* Namespaces

- **sosicon**
- *Application root.* **sosicon::sosi**

*SOSI.*

# /prosjekter/sosicon/src/sosi/sosi_ref_list.cpp File Reference

```
#include "sosi_ref_list.h"
```

## /prosjekter/sosicon/src/sosi/sosi_ref_list.h File Reference

```
#include "../interface/i_sosi_element.h"
#include "sosi_types.h"
#include <iostream>
#include <string>
#include <vector>
```

### Classes

- class **sosicon::sosi::SosiRefList**

### *SOSI REF list.* Namespaces

- **sosicon**
- *Application root.* **sosicon::sosi**

*SOSI.*

## /prosjekter/sosicon/src/sosi/sosi_translation_table.cpp File Reference

```
#include "sosi_translation_table.h"
```

## /prosjekter/sosicon/src/sosi/sosi_translation_table.h File Reference

```
#include <map>
#include <vector>
#include "sosi_types.h"
```

### Classes

- class **sosicon::sosi::SosiTranslationTable**

### Namespaces

- **sosicon**
- *Application root.* **sosicon::sosi**

*SOSI.*

# /prosjekter/sosicon/src/sosi/sosi_types.h File Reference

```
#include "../interface/i_coordinate.h"
#include <string>
#include <vector>
#include <map>
```

## Classes

- struct **sosicon::sosi::ReferenceData**
- *SOSI reference number.* class **sosicon::sosi::CoordSys**

## *SOSI coordinate system.* Namespaces

- **sosicon**
- *Application root.* **sosicon::sosi**

## *SOSI.* Typedefs

- typedef std::vector
- < ReferenceData * > **sosicon::sosi::GeometryRef**
  *List of SOSI references.*
- typedef std::vector
- < GeometryRef * > **sosicon::sosi::GeometryCollection**
  *Collection of SOSI reference lists.*

## Enumerations

- enum **sosicon::sosi::ElementType** { **sosicon::sosi::sosi_element_unknown**, **sosicon::sosi::sosi_element_airport_roads**, **sosicon::sosi::sosi_element_airport_type**, **sosicon::sosi::sosi_element_area**, **sosicon::sosi::sosi_element_charset**, **sosicon::sosi::sosi_element_coordsys**, **sosicon::sosi::sosi_element_curve**, **sosicon::sosi::sosi_element_eof**, **sosicon::sosi::sosi_element_head**, **sosicon::sosi::sosi_element_height**, **sosicon::sosi::sosi_element_iata_code**, **sosicon::sosi::sosi_element_icao_code**, **sosicon::sosi::sosi_element_kp**, **sosicon::sosi::sosi_element_level**, **sosicon::sosi::sosi_element_max_ne**, **sosicon::sosi::sosi_element_min_ne**, **sosicon::sosi::sosi_element_municipality**, **sosicon::sosi::sosi_element_name**, **sosicon::sosi::sosi_element_ne**, **sosicon::sosi::sosi_element_neh**, **sosicon::sosi::sosi_element_objtype**, **sosicon::sosi::sosi_element_origo_ne**, **sosicon::sosi::sosi_element_owner**, **sosicon::sosi::sosi_element_point**, **sosicon::sosi::sosi_element_quality**, **sosicon::sosi::sosi_element_ref**, **sosicon::sosi::sosi_element_surface**, **sosicon::sosi::sosi_element_text**, **sosicon::sosi::sosi_element_traffic_type**, **sosicon::sosi::sosi_element_transpar**, **sosicon::sosi::sosi_element_unit**, **sosicon::sosi::sosi_element_updatedate**, **sosicon::sosi::sosi_element_water_width**, **sosicon::sosi::sosi_element_vendor**, **sosicon::sosi::sosi_element_version** }
- *List of SOSI element types.* enum **sosicon::sosi::ObjType** { **sosicon::sosi::sosi_objtype_unknown**, **sosicon::sosi::sosi_objtype_airport**, **sosicon::sosi::sosi_objtype_airport_type**, **sosicon::sosi::sosi_objtype_baseline**, **sosicon::sosi::sosi_objtype_carriageway**, **sosicon::sosi::sosi_objtype_cadastral_address**, **sosicon::sosi::sosi_objtype_coastline**, **sosicon::sosi::sosi_objtype_county_boundary**, **sosicon::sosi::sosi_objtype_data_delineation**, **sosicon::sosi::sosi_objtype_edge_view**, **sosicon::sosi::sosi_objtype_fictious_dividing_line**, **sosicon::sosi::sosi_objtype_forest**, **sosicon::sosi::sosi_objtype_developed_area**, **sosicon::sosi::sosi_objtype_golf_course**, **sosicon::sosi::sosi_objtype_industrial_area**, **sosicon::sosi::sosi_objtype_lake**, **sosicon::sosi::sosi_objtype_lane**, **sosicon::sosi::sosi_objtype_lake_edge**, **sosicon::sosi::sosi_objtype_lake_river_barrier**, **sosicon::sosi::sosi_objtype_land_use_boundary**, **sosicon::sosi::sosi_objtype_level_crossing**, **sosicon::sosi::sosi_objtype_municipal_divide**, **sosicon::sosi::sosi_objtype_municipality**, **sosicon::sosi::sosi_objtype_municipality_boundary**,

**sosicon::sosi::sosi_objtype_marsh**, **sosicon::sosi::sosi_objtype_national_border**, **sosicon::sosi::sosi_objtype_pedestrian_bicycle_road_centre_line**, **sosicon::sosi::sosi_objtype_sea_river_delineation**, **sosicon::sosi::sosi_objtype_snow_field**, **sosicon::sosi::sosi_objtype_open_land**, **sosicon::sosi::sosi_objtype_river_brook**, **sosicon::sosi::sosi_objtype_river_brook_edge**, **sosicon::sosi::sosi_objtype_road_block**, **sosicon::sosi::sosi_objtype_road_centre_line**, **sosicon::sosi::sosi_objtype_road_under_railway**, **sosicon::sosi::sosi_objtype_sea_surface**, **sosicon::sosi::sosi_objtype_sidewalk**, **sosicon::sosi::sosi_objtype_spelling**, **sosicon::sosi::sosi_objtype_stone_quarry**, **sosicon::sosi::sosi_objtype_street_address**, **sosicon::sosi::sosi_objtype_territorial_boundary**, **sosicon::sosi::sosi_objtype_turn_connecting_segment** }

- *List of SOSI OBJTYPEs.* enum **sosicon::sosi::JunctionPoint** { **sosicon::sosi::sosi_junction_node**, **sosicon::sosi::sosi_junction_connection**, **sosicon::sosi::sosi_junction_open_end** }

*Default SOSI junction point layer types.*

# /prosjekter/sosicon/src/sosi/sosi_unit.cpp File Reference

```
#include "sosi_unit.h"
```

## /prosjekter/sosicon/src/sosi/sosi_unit.h File Reference

```
#include "../interface/i_sosi_element.h"
#include "../interface/i_sosi_head_member.h"
#include "sosi_types.h"
#include <iostream>
#include <string>
#include <vector>
```

### Classes

- class **sosicon::sosi::SosiUnit**

### *SOSI Unit.* Namespaces

- **sosicon**
- *Application root.* **sosicon::sosi**

*SOSI.*

# /prosjekter/sosicon/src/sosi_north_east_height_ragel.cpp File Reference

`#include "sosi/sosi_north_east.h"`

## Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/sosi_north_east_ragel.cpp File Reference

```
#include "sosi/sosi_north_east.h"
```

## Namespaces

- **sosicon**

*Application root.*

## /prosjekter/sosicon/src/sosi_origo_ne_ragel.cpp File Reference

```
#include "sosi/sosi_origo_ne.h"
```

### Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/sosi_ref_ragel.cpp File Reference

```
#include "sosi/sosi_ref_list.h"
```

## Namespaces

- **sosicon**

*Application root.*

# /prosjekter/sosicon/src/utils.cpp File Reference

```
#include "utils.h"
```

# /prosjekter/sosicon/src/utils.h File Reference

```
#include "memory.h"
#include <sys/stat.h>
#include <locale>
#include <string>
#include <ctype.h>
```

## Namespaces

- **sosicon**
- *Application root.* **sosicon::utils**

## *String manipulation routines.* Functions

- std::string **sosicon::utils::className2FileName** (const std::string &className)
  *Converts Class name to file name string.*
- bool **sosicon::utils::fileExists** (const std::string &name)
  *Test if file exists.*
- std::string **sosicon::utils::normalizeAppClassName** (const std::string &className)
  *Asserts correct name of application classes.*
- std::string **sosicon::utils::repeat** (const std::string &seq, unsigned int count)
  *Repeat string N times.*
- std::string **sosicon::utils::replaceAll** (const std::string &from, const std::string &to, const std::string &subject)
  *Replace all occurences of one string witn another.*
- std::string **sosicon::utils::trim** (const std::string &str)
  *Removes leading and trailing space characters.*
- std::string **sosicon::utils::trimLeft** (const std::string &str)
- std::string **sosicon::utils::trimRight** (const std::string &str)
- std::string **sosicon::utils::toLower** (const std::string &from)
- std::string **sosicon::utils::ucFirst** (const std::string &str)
- void **sosicon::utils::getPathInfo** (std::string path, std::string &dir, std::string &tit, std::string &ext)

# Index