TiGL

2.1.1

Generated by Doxygen 1.5.6

Mon Jun 16 10:56:43 2014

Contents

1 Module Index

1.1 Modules

Here is a list of all modules:

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

3 Module Documentation

3.1 Enumerations

Typedefs

- typedef unsigned int TiglGeometricComponentType

Enumerations

- enum TiglAlgorithmCode { TIGL_INTERPOLATE_LINEAR_WIRE = 0, TIGL_INTERPOLATE_BSPLINE_WIRE = 1, TIGL_APPROXIMATE_BSPLINE_WIRE = 2 }
- enum TiglBoolean { TIGL_FALSE = 0, TIGL_TRUE = 1 }
- enum TiglImportExportFormat { TIGL_IMPORTEXPORT_IGES = 0, TIGL_IMPORTEXPORT_STEP = 1, TIGL_IMPORTEXPORT_STL = 2, TIGL_IMPORTEXPORT_VTK = 3 }
- enum TiglLogLevel {

  TILOG_SILENT = 0, TILOG_ERROR = 1, TILOG_WARNING = 2, TILOG_INFO = 3,

  TILOG_DEBUG = 4, TILOG_DEBUG1 = 5, TILOG_DEBUG2 = 6, TILOG_DEBUG3 = 7,

  TILOG_DEBUG4 = 8 }
- enum TiglReturnCode {

  TIGL_SUCCESS = 0, TIGL_ERROR = 1, TIGL_NULL_POINTER = 2, TIGL_NOT_FOUND = 3,

  TIGL_XML_ERROR = 4, TIGL_OPEN_FAILED = 5, TIGL_CLOSE_FAILED = 6, TIGL_INDEX_ERROR = 7,

  TIGL_STRING_TRUNCATED = 8, TIGL_WRONG_TIXI_VERSION = 9, TIGL_UID_ERROR = 10, TIGL_WRONG_CPACS_VERSION = 11,

  TIGL_UNINITIALIZED = 12, TIGL_MATH_ERROR = 13 }
- enum TiglSymmetryAxis { TIGL_NO_SYMMETRY = 0, TIGL_X_Y_PLANE = 1, TIGL_X_Z_PLANE = 2, TIGL_Y_Z_PLANE = 3 }

3.1.1 Typedef Documentation

3.1.1.1 typedef unsigned int TiglGeometricComponentType

the type if the component changes the way of behavior.

3.1.2 Enumeration Type Documentation

3.1.2.1 enum TiglAlgorithmCode

these constants to define the algorithm to be used e.g. in interpolations.

Enumerator:

TIGL_INTERPOLATE_LINEAR_WIRE Use a linear interpolation between the points of a wire

TIGL_INTERPOLATE_BSPLINE_WIRE Use a BSpline interpolation between the points of a wire

TIGL_APPROXIMATE_BSPLINE_WIRE Use a BSpline approximation for the points of a wire

3.1.2.2 enum TiglBoolean

Definition of boolean values used in TIGL.

Enumerator:

TIGL_FALSE

TIGL_TRUE

3.1.2.3 enum TiglImportExportFormat

Definition of the different file formats used for import/export used in TIGL.

Enumerator:

TIGL_IMPORTEXPORT_IGES Use IGES format for geometry import/export

TIGL_IMPORTEXPORT_STEP Use STEP format for geometry import/export

TIGL_IMPORTEXPORT_STL Use STL format for geometry import/export

TIGL_IMPORTEXPORT_VTK Use VTK (XML/VTP) format for geometry import/export

3.1.2.4 enum TiglLogLevel

Definition of possible logging levels

Enumerator:

TILOG_SILENT No messages are printed. TiGL is completely silent, even in case of errors.

TILOG_ERROR Only error messages are printed.

TILOG_WARNING Only errors and warnings are printed on console. This is the default log level of TiGL.

TILOG_INFO In addition to TILOG_WANING, also informative messages are printed.

TILOG_DEBUG Also debug messages are printed. Enable this if you want to track down potential errors in TiGL.

TILOG_DEBUG1 This level is only interesting for TiGL developers

TILOG_DEBUG2 This level is only interesting for TiGL developers

TILOG_DEBUG3 This level is only interesting for TiGL developers

TILOG_DEBUG4 This level is only interesting for TiGL developers

3.1.2.5 enum TiglReturnCode

Definition of possible error return codes of some functions.

Enumerator:

TIGL_SUCCESS
TIGL_ERROR
TIGL_NULL_POINTER
TIGL_NOT_FOUND
TIGL_XML_ERROR
TIGL_OPEN_FAILED
TIGL_CLOSE_FAILED
TIGL_INDEX_ERROR
TIGL_STRING_TRUNCATED
TIGL_WRONG_TIXI_VERSION
TIGL_UID_ERROR
TIGL_WRONG_CPACS_VERSION
TIGL_UNINITIALIZED
TIGL_MATH_ERROR

3.1.2.6 enum TiglSymmetryAxis

Definition of Symmetry Axis used in TIGL.

Enumerator:

TIGL_NO_SYMMETRY
TIGL_X_Y_PLANE
TIGL_X_Z_PLANE
TIGL_Y_Z_PLANE

3.2 General TIGL handling functions

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglCloseCPACSConfiguration (TiglCPACSConfigurationHandle cpacsHandle)

  Closes a CPACS configuration and cleans up all memory used by the configuration. After closing a configuration the associated configuration handle is no longer valid. When the CPACS configuration has been closed, the companion tixi document can also be closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglGetCPACSTixiHandle (TiglCPACSConfigurationHandle cpacsHandle, TixiDocumentHandle ∗tixiHandlePtr)

  Returns the underlying TixiDocumentHandle for a given CPACS configuration handle.

- TIGL_COMMON_EXPORT char ∗ tiglGetVersion ()

  Returns the version number of this TIGL version.

- TIGL_COMMON_EXPORT TiglReturnCode tiglIsCPACSConfigurationHandleValid (TiglCPACSConfigurationHandle cpacsHandle, TiglBoolean ∗isValidPtr)

  Checks if a given CPACS configuration handle is a valid.

- TIGL_COMMON_EXPORT TiglReturnCode tiglOpenCPACSConfiguration (TixiDocumentHandle tixiHandle, const char ∗configurationUID, TiglCPACSConfigurationHandle ∗cpacsHandlePtr)

  Opens a CPACS configuration and builds up the data and geometry structure in memory.

3.2.1 Detailed Description

Function to open, create, and close CPACS-files.

3.2.2 Function Documentation

3.2.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglCloseCPACSConfiguration (TiglCPACSConfigurationHandle *cpacsHandle*)

Closes a CPACS configuration and cleans up all memory used by the configuration. After closing a configuration the associated configuration handle is no longer valid. When the CPACS configuration has been closed, the companion tixi document can also be closed.

Fortran syntax:

tigl_close_cpacs_configuration(integer cpacsHandle, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration.

Returns:

- TIGL_SUCCESS if successfully opened the CPACS configuration file
- TIGL_CLOSE_FAILED if closing of the CPACS configuration failed
- TIGL_NOT_FOUND if handle ist not found in handle container
- TIGL_ERROR if some other kind of error occurred

3.2.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglGetCPACSTixiHandle (TiglCPACSConfigurationHandle *cpacsHandle*, TixiDocumentHandle ∗ *tixiHandlePtr*)

Returns the underlying TixiDocumentHandle for a given CPACS configuration handle.

Fortran syntax:

tigl_get_cpacs_tixi_handle(integer cpacsHandle, integer tixiHandlePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

→ ***tixiHandlePtr*** Handle for the TIXI document associated with the CPACS configuration

Returns:

- TIGL_SUCCESS if the TIXI handle was found
- TIGL_NOT_FOUND if the TIXI handle was not found
- TIGL_NULL_POINTER if tixiHandlePtr is an invalid null pointer
- TIGL_ERROR if some other kind of error occurred

3.2.2.3 TIGL_COMMON_EXPORT char∗ tiglGetVersion ()

Returns the version number of this TIGL version.

Fortran syntax: tigl_get_version(character version)

Returns:

- char∗ A string with the version number.

3.2.2.4 TIGL_COMMON_EXPORT TiglReturnCode tiglIsCPACSConfigurationHandleValid (TiglCPACSConfigurationHandle *cpacsHandle*, TiglBoolean ∗ *isValidPtr*)

Checks if a given CPACS configuration handle is a valid.

Fortran syntax:

tigl_is_cpacs_configuration_handle_valid(integer cpacsHandle, integer isValidPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** CPACS configuration handle to check

→ ***isValidPtr*** Contains TIGL_TRUE, if handle is valid, otherwise contains TIGL_FALSE

Returns:

- TIGL_SUCCESS if no error occurred

3.2.2.5 TIGL_COMMON_EXPORT TiglReturnCode tiglOpenCPACSConfiguration (TixiDocumentHandle *tixiHandle*, const char ∗ *configurationUID*, TiglCPACSConfigurationHandle ∗ *cpacsHandlePtr*)

Opens a CPACS configuration and builds up the data and geometry structure in memory.

Fortran syntax:

tigl_open_cpacs_configuration(integer tixiHandle, character configurationUID, integer cpacsHandlePtr, integer returnCode)

Parameters:

← ***tixiHandle*** Handle to a TIXI document. The TIXI document should not be closed until the CPACS configuration is closed. First close the CPACS configuration, then the TIXI document.

← ***configurationUID*** The UID of the configuration that should be loaded by TIGL. Could be NULL or an empty string if the data set contains only one configuration.

→ ***cpacsHandlePtr*** Handle to the CPACS configuration. This handle is used in calls to other TIGL functions.

Returns:

- TIGL_SUCCESS if successfully opened the CPACS configuration
- TIGL_XML_ERROR if file is not well-formed or another XML error occurred
- TIGL_OPEN_FAILED if some other error occurred
- TIGL_NULL_POINTER if cpacsHandlePtr is an invalid null pointer

- TIGL_ERROR if some other kind of error occurred
- TIGL_INVALID_UID is the UID does not exist or an error orrcured with this configuration

3.3 Functions for wing calculations

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglGetWingCount (TiglCPACSConfigurationHandle cpacsHandle, int ∗wingCountPtr)

  Returns the number of wings in a CPACS configuration.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentFindSegment (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, double x, double y, double z, char ∗∗segmentUID, char ∗∗wingUID)

  Returns the segmentUID and wingUID for a given point on a componentSegment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetNumberOfSegments (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, int ∗nsegments)

  Returns the number of segments belonging to a component segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetPoint (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, double eta, double xsi, double ∗x, double ∗y, double ∗z)

  Returns x,y,z koordinates for a given eta and xsi on a componentSegment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetSegmentIntersection (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, const char ∗segmentUID, double csEta1, double csXsi1, double csEta2, double csXsi2, double segmentEta, double ∗segmentXsi)

  Computes the intersection of a line (defined by component segment coordinates) with an iso-eta line on a specified wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetSegmentUID (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, int segmentIndex, char ∗∗segmentUID)

  Returns the segment UID of a segment belonging to a component segment. The segment is specified with its index, which is in the 1...nsegments. The number of segments nsegments can be queried with tiglWingComponentSegmentGetNumberOfSegments.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentPointGetSegmentEtaXsi (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, double eta, double xsi, char ∗∗wingUID, char ∗∗segmentUID, double ∗segmentEta, double ∗segmentXsi)

Returns eta, xsi, segmentUID and wingUID for a given eta and xsi on a componentSegment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetComponentSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int ∗compSegmentCountPtr)

  Returns the number of component segments for a wing in a CPACS configuration.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetComponentSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, const char ∗compSegmentUID, int ∗segmentIndexPtr)

  Returns the Index of a component segment of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetComponentSegmentUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int compSegmentIndex, char ∗∗uidNamePtr)

  Returns the UID of a component segment of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetIndex (TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, int ∗wingIndexPtr)

  Returns the Index of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerConnectedSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int ∗segmentCountPtr)

  Returns the count of wing segments connected to the inner section of a given segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerConnectedSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int n, int ∗connectedIndexPtr)

  Returns the index (number) of the n-th wing segment connected to the inner section of a given segment. n starts at 1.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerSectionAndElementIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int ∗sectionIndexPtr, int ∗elementIndexPtr)

  Returns the section index and section element index of the inner side of a given wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerSectionAndElementUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, char ∗∗sectionUIDPtr, char ∗∗elementUIDPtr)

  Returns the section UID and section element UID of the inner side of a given wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetLowerPoint (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double eta, double xsi, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on the lower wing surface for a a given wing and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetLowerPointAtAngle (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double eta, double xsi, double xAngle, double yAngle, double zAngle, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on the lower wing surface for a a given wing and segment index. This function is different from tiglWingGetLowerPoint, as a point on the cord surface is calculated (given eta and xsi coordinates) and the normal vector at that point is rotated around the x, y, and z axis by the given angles. Finally, the intersection point of the rotated normal and the lower wing shell is computed. The point is returned in absolute world coordinates.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterConnectedSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int ∗segmentCountPtr)

  Returns the count of wing segments connected to the outer section of a given segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterConnectedSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int n, int ∗connectedIndexPtr)

  Returns the index (number) of the n-th wing segment connected to the outer section of a given segment. n starts at 1.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterSectionAndElementIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int ∗sectionIndexPtr, int ∗elementIndexPtr)

  Returns the section index and section element index of the outer side of a given wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterSectionAndElementUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, char ∗∗sectionUIDPtr, char ∗∗elementUIDPtr)

  Returns the section UID and section element UID of the outer side of a given wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetProfileName (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int sectionIndex, int elementIndex, char ∗∗profileNamePtr)

  Returns the name of a wing profile. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSectionUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int sectionIndex, char ∗∗uidNamePtr)

Returns the UID of a section of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int ∗segmentCountPtr)

  Returns the number of segments for a wing in a CPACS configuration.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentEtaXsi (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, double pointX, double pointY, double pointZ, int ∗segmentIndex, double ∗eta, double ∗xsi, int ∗isOnTop)

  Inverse function to tiglWingGetLowerPoint and tiglWingGetLowerPoint. Calculates to a point (x,y,z) in global coordinates the wing segment coordinates and the wing segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, const char ∗segmentUID, int ∗segmentIndexPtr, int ∗wingIndexPtr)

  Returns the Index of a segment of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, char ∗∗uidNamePtr)

  Returns the UID of a segment of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSymmetry (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, TiglSymmetryAxis ∗symmetryAxisPtr)

  Returns the Symmetry Enum if the wing has symmetry-axis.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, char ∗∗uidNamePtr)

  Returns the UID of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetUpperPoint (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double eta, double xsi, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on the upper wing surface for a a given wing and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetUpperPointAtAngle (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double eta, double xsi, double xAngle, double yAngle, double zAngle, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

Returns a point on the upper wing surface for a a given wing and segment index. This function is different from tiglWingGetUpperPoint, as a point on the cord surface is calculated (given eta and xsi coordinates) and the normal vector at that point is rotated around the x, y, and z axis by the given angles. Finally, the intersection point of the rotated normal and the upper wing shell is computed. The point is returned in absolute world coordinates.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingSegmentPointGetComponentSegmentEtaXsi (TiglCPACSConfigurationHandle cpacsHandle, const char ∗segmentUID, const char ∗componentSegmentUID, double segmentEta, double segmentXsi, double ∗eta, double ∗xsi)

  Returns eta, xsi coordinates of a componentSegment given segmentEta and segmentXsi on a wing segment.

3.3.1 Detailed Description

Function to handle wing geometry's with TIGL.

3.3.2 Function Documentation

3.3.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglGetWingCount (TiglCPACSConfigurationHandle *cpacsHandle*, int ∗ *wingCountPtr*)

Returns the number of wings in a CPACS configuration.

Fortran syntax:

tigl_get_wing_count(integer cpacsHandle, integer wingCountPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

→ ***wingCountPtr*** Pointer to the number of wings

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if wingCountPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentFindSegment (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentSegmentUID*, double *x*, double *y*, double *z*, char ∗∗ *segmentUID*, char ∗∗ *wingUID*)

Returns the segmentUID and wingUID for a given point on a componentSegment.

Fortran syntax:

tigl_wing_componentsegment_find_segment(integer cpacsHandle, character∗n componentSegmentUID, real x, real y, real z, character∗n segmentUID, character∗n wingUID, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***componentSegmentUID*** UID of the componentSegment to search for

← ***x,y,z*** Coordinates of the point of the componentSegment

→ ***segmentUID*** UID of the segment that fits to the given point and componentSegment. In contrast to old releases, the returned string **must not be freed** by the user!

→ ***wingUID*** UID of the wing that fits to the given point and componentSegment In contrast to old releases, the returned string **must not be freed** by the user!

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NOT_FOUND if the point does not lie on the wing component segment within 1cm tolerance.
- TIGL_INDEX_ERROR if wingIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.3.2.3 TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetNumberOfSegments (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentSegmentUID*, int ∗ *nsegments*)

Returns the number of segments belonging to a component segment.

Fortran syntax:

tigl_wing_component_segment_get_number_of_segments(integer cpacsHandle, character∗n componentSegmentUID, integer nsegments, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***componentSegmentUID*** UID of the componentSegment

→ ***nsegments*** Number of segments belonging to the component segment

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_UID_ERROR if the component segment does not exist
- TIGL_ERROR if some other error occurred

3.3.2.4 TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetPoint (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentSegmentUID*, double *eta*, double *xsi*, double ∗ *x*, double ∗ *y*, double ∗ *z*)

Returns x,y,z koordinates for a given eta and xsi on a componentSegment.

Fortran syntax:

tigl_wing_component_segment_get_point(integer cpacsHandle, character∗n componentSegmentUID, real eta, real xsi real x, real y, real z, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***componentSegmentUID*** UID of the componentSegment to search for

← ***eta,xsi*** Eta and Xsi of the point of the componentSegment

→ ***x*** X coordinate of the point on the corresponding segment.

→ ***y*** Y coordinate of the point on the corresponding segment.

→ ***z*** Z coordinate of the point on the corresponding segment.

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_UID_ERROR if the componentSegment does not exist
- TIGL_ERROR if some other error occurred

3.3.2.5 TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetSegmentIntersection (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentSegmentUID*, const char ∗ *segmentUID*, double *csEta1*, double *csXsi1*, double *csEta2*, double *csXsi2*, double *segmentEta*, double ∗ *segmentXsi*)

Computes the intersection of a line (defined by component segment coordinates) with an iso-eta line on a specified wing segment.

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***componentSegmentUID*** UID of the componentSegment

← ***segmentUID*** UID of the segment, the intersection should be calculated with

← ***csEta1,csEta2*** Start and end eta coordinates of the intersection line (given as component segment coordinates)

← ***csXsi1,csXsi2*** Start and end xsi coordinates of the intersection line (given as component segment coordinates)

← ***segmentEta*** Eta coordinate of the iso-eta segment intersection line

→ ***segmentXsi*** Xsi coordinate of the intersection point on the wing segment

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_UID_ERROR if the segment or the component segment does not exist
- TIGL_MATH_ERROR if the intersection could not be computed (e.g. if no intersection exists)
- TIGL_ERROR if some other error occurred

3.3.2.6 TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetSegmentUID (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentSegmentUID*, int *segmentIndex*, char ∗∗ *segmentUID*)

Returns the segment UID of a segment belonging to a component segment. The segment is specified with its index, which is in the 1...nsegments. The number of segments nsegments can be queried with tiglWingComponentSegmentGetNumberOfSegments.

Fortran syntax:

tigl_wing_component_segment_get_segment_uid(integer cpacsHandle, character∗n componentSegmentUID, integer segmentIndex, character∗n segmentUID, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***componentSegmentUID*** UID of the componentSegment

← ***segmentIndex*** Index of the segment (1 <= index <= nsegments)

→ ***segmentUID*** UID of the segment

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_UID_ERROR if the component segment does not exist
- TIGL_INDEX_ERROR if the segment index is invalid
- TIGL_ERROR if some other error occurred

3.3.2.7 TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentPointGetSegmentEtaXsi (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentSegmentUID*, double *eta*, double *xsi*, char ∗∗ *wingUID*, char ∗∗ *segmentUID*, double ∗ *segmentEta*, double ∗ *segmentXsi*)

Returns eta, xsi, segmentUID and wingUID for a given eta and xsi on a componentSegment.

If the given component segment point deviates more than 1 cm from any segment, the function returns TIGL_MATH_ERROR. If the point is outside any segment but deviates less than 1 cm from a segment, the point is first projected onto the segment. Then, this point is transformed into segment coordinates. In this case, a warning is generated to inform the user, that he can fix his setup.

Fortran syntax:

tigl_wing_component_segment_point_get_segment_eta_xsi(integer cpacsHandle, character∗n componentSegmentUID, real eta, real xsi character∗n wingUID, character∗n segmentUID, real segmentEta, real segmentXsi integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***componentSegmentUID*** UID of the componentSegment to search for

← ***eta,xsi*** Eta and Xsi of the point of the componentSegment

→ ***wingUID*** UID of the wing that fits to the given point and componentSegment. In contrast to old releases, the returned string **must not be freed** by the user!

→ ***segmentUID*** UID of the segment that fits to the given point and componentSegment. In contrast to old releases, the returned string **must not be freed** by the user!

→ ***segmentEta*** Eta of the point on the corresponding segment.

→ ***segmentXsi*** Xsi of the point on the corresponding segment.

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_UID_ERROR if the componentSegment does not exist
- TIGL_MATH_ERROR if the given wing component segment point can not be transformed into a segment point. This might happen, if the component segment contains twisted section.
- TIGL_ERROR if some other error occurred

3.3.2.8 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetComponentSegmentCount (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int ∗ *compSegmentCountPtr*)

Returns the number of component segments for a wing in a CPACS configuration.

Fortran syntax:

tigl_wing_get_component_segment_count(integer cpacsHandle, integer wingIndex, integer compSegmentCountPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of the wing, starting at 1

→ ***compSegmentCountPtr*** Pointer to the number of component segments

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex is not valid
- TIGL_NULL_POINTER if compSegmentCountPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.9 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetComponentSegmentIndex (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, const char ∗ *compSegmentUID*, int ∗ *segmentIndexPtr*)

Returns the Index of a component segment of a wing.

Fortran syntax:

tigl_wing_get_segment_index(integer cpacsHandle, integer wingIndex, character∗n uIDNamePtr, integer segmentIndex, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***compSegmentUID*** The uid of the wing

→ ***segmentIndexPtr*** The index of a segment, starting at 1

Usage example:

```
TiglReturnCode returnCode;
int segmentIndex;
returnCode = tiglWingGetComponentSegmentIndex(cpacsHandle, wing, uidName, &segmentIndex);
printf("The Index of the component segment of wing %d is %d\n", wing, segmentIndex);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex is not valid
- TIGL_UID_ERROR if the compSegmentUID does not exist
- TIGL_NULL_POINTER if compSegmentUID is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.10 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetComponentSegmentUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *compSegmentIndex*, char ∗∗ *uidNamePtr*)

Returns the UID of a component segment of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

Fortran syntax:

tigl_wing_get_component_segment_uid(integer cpacsHandle, integer wingIndex, integer segmentIndex, character∗n uIDNamePtr, integer returnCode)

Parameters:

$\leftarrow$ ***cpacsHandle*** Handle for the CPACS configuration

$\leftarrow$ ***wingIndex*** The index of a wing, starting at 1

$\leftarrow$ ***compSegmentIndex*** The index of a segment, starting at 1

$\rightarrow$ ***uidNamePtr*** The uid of the wing

Usage example:

```
TiglReturnCode returnCode;
char* uidPtr = 0;
returnCode = tiglWingGetComponentSegmentUID(cpacsHandle, wing, segmentID, &uidPtr);
printf("The UID of the component segment of wing %d is %s\n", wing, uidPtr);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex or the compSegmentIndex are not valid
- TIGL_NULL_POINTER if uidNamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.11 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetIndex (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *wingUID*, int ∗ *wingIndexPtr*)

Returns the Index of a wing.

Fortran syntax:

tigl_wing_get_index(integer cpacsHandle, character∗n uIDNamePtr, integer wingIndex, integer returnCode)

Parameters:

$\leftarrow$ ***cpacsHandle*** Handle for the CPACS configuration

$\leftarrow$ ***wingUID*** The uid of the wing

→ ***wingIndexPtr*** The index of a wing, starting at 1

Usage example:

```
TiglReturnCode returnCode;
int wingIndex;
returnCode = tiglWingGetUID(cpacsHandle, wingUID, &wingIndex);
printf("The Index of the wing is %d\n", wingIndex);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_UID_ERROR if wingUID does not exist
- TIGL_NULL_POINTER if wingUID is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.12 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerConnectedSegmentCount (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, int ∗ *segmentCountPtr*)

Returns the count of wing segments connected to the inner section of a given segment.

Fortran syntax:

tigl_wing_get_inner_connected_segment_count(integer cpacsHandle, integer wingIndex, integer segmentIndex, integer segmentCountPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***segmentCountPtr*** Pointer to the count of connected segments

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if segmentCountPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.13 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerConnectedSegmentIndex (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, int *n*, int ∗ *connectedIndexPtr*)

Returns the index (number) of the n-th wing segment connected to the inner section of a given segment. n starts at 1.

Fortran syntax:

tigl_wing_get_inner_connected_segment_index(integer cpacsHandle, integer wingIndex, integer segmentIndex, integer n, integer connectedIndexPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

← ***n*** n-th segment searched, 1 <= n <= tiglWingGetInnerConnectedSegmentCount(...)

→ ***connectedIndexPtr*** Pointer to the segment index of the n-th connected segment

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle or no n-th connected segment was found
- TIGL_INDEX_ERROR if wingIndex, segmentIndex or n are not valid
- TIGL_NULL_POINTER if segmentIndexPtr is a null pointer
- TIGL_ERROR if some other error occured

3.3.2.14 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerSectionAndElementIndex (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, int ∗ *sectionIndexPtr*, int ∗ *elementIndexPtr*)

Returns the section index and section element index of the inner side of a given wing segment.

Fortran syntax:

tigl_wing_get_inner_section_and_element_index(integer cpacsHandle, integer wingIndex, integer segmentIndex, integer sectionUIDPtr, integer elementUIDPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***sectionIndexPtr*** The section index of the inner side

→ ***elementIndexPtr*** The section element index of the inner side

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if sectionIndexPtr or elementIndexPtr are a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.15 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerSectionAndElementUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, char ∗∗ *sectionUIDPtr*, char ∗∗ *elementUIDPtr*)

Returns the section UID and section element UID of the inner side of a given wing segment.

Important change: The memory necessary for the two UIDs must not be freed by the user anymore.

Fortran syntax:

tigl_wing_get_inner_section_and_element_UID(integer cpacsHandle, integer wingIndex, integer segmentIndex, character sectionUIDPtr, character elementUIDPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***sectionUIDPtr*** The section UID of the inner side

→ ***elementUIDPtr*** The section element UID of the inner side

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if sectionIndexPtr or elementIndexPtr are a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.16 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetLowerPoint (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, double *eta*, double *xsi*, double ∗ *pointXPtr*, double ∗ *pointYPtr*, double ∗ *pointZPtr*)

Returns a point on the lower wing surface for a a given wing and segment index.

Returns a point on the lower wing surface in dependence of parameters eta and xsi, which range from 0.0 to 1.0. For eta = 0.0, xsi = 0.0 the point is equal to the leading edge on the inner section of the given segment. For eta = 1.0, xsi = 1.0 the point is equal to the trailing edge on the outer section of the given segment. The point is returned in absolute world coordinates.

Fortran syntax:

tigl_wing_get_lower_point(integer cpacsHandle, integer wingIndex, integer segmentIndex, real eta, real xsi, real pointXPtr, real pointYPtr, real pointZPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of the wing, starting at 1

← ***segmentIndex*** The index of the segment of the wing, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***xsi*** xsi in the range 0.0 <= xsi <= 1.0

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occured

3.3.2.17 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetLowerPointAtAngle (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, double *eta*, double *xsi*, double *xAngle*, double *yAngle*, double *zAngle*, double ∗ *pointXPtr*, double ∗ *pointYPtr*, double ∗ *pointZPtr*)

Returns a point on the lower wing surface for a a given wing and segment index. This function is different from tiglWingGetLowerPoint, as a point on the cord surface is calculated (given eta and xsi coordinates) and the normal vector at that point is rotated around the x, y, and z axis by the given angles. Finally, the intersection point of the rotated normal and the lower wing shell is computed. The point is returned in absolute world coordinates.

Fortran syntax:

tigl_wing_get_lower_point_at_angle(integer cpacsHandle, integer wingIndex, integer segmentIndex, real eta, real xsi, real xangle, real yangle, real zangle, real pointXPtr, real pointYPtr, real pointZPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of the wing, starting at 1

← ***segmentIndex*** The index of the segment of the wing, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0; eta = 0 for inner section , eta = 1 for outer section

← ***xsi*** xsi in the range 0.0 <= xsi <= 1.0; xsi = 0 for Leading Edge, xsi = 1 for Trailing Edge

← ***xAngle*** Angle of normal vector rotation around the x-Axis (in degrees)

← ***yAngle*** Angle of normal vector rotation around the y-Axis (in degrees)

← ***zAngle*** Angle of normal vector rotation around the z-Axis (in degrees)

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.3.2.18 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterConnectedSegmentCount (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, int ∗ *segmentCountPtr*)

Returns the count of wing segments connected to the outer section of a given segment.

Fortran syntax:

tigl_wing_get_outer_connected_segment_count(integer cpacsHandle, integer wingIndex, integer segmentIndex, integer segmentCountPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***segmentCountPtr*** Pointer to the count of connected segments

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if segmentCountPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.19 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterConnectedSegmentIndex (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, int *n*, int ∗ *connectedIndexPtr*)

Returns the index (number) of the n-th wing segment connected to the outer section of a given segment. n starts at 1.

Fortran syntax:

tigl_wing_get_outer_connected_segment_index(integer cpacsHandle, integer wingIndex, integer segmentIndex, integer n, integer connectedIndexPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

← ***n*** n-th segment searched, 1 <= n <= tiglWingGetOuterConnectedSegmentCount(...)

→ ***connectedIndexPtr*** Pointer to the segment index of the n-th connected segment

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle or no n-th connected segment was found
- TIGL_INDEX_ERROR if wingIndex, segmentIndex or n are not valid
- TIGL_NULL_POINTER if segmentIndexPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.20 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterSectionAndElementIndex (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, int $*$ *sectionIndexPtr*, int $*$ *elementIndexPtr*)

Returns the section index and section element index of the outer side of a given wing segment.

Fortran syntax:

tigl_wing_get_outer_section_and_element_uid(integer cpacsHandle, integer wingIndex, integer segmentIndex, integer sectionIndexPtr, integer elementIndexPtr, integer returnCode)

Parameters:

$\leftarrow$ ***cpacsHandle*** Handle for the CPACS configuration

$\leftarrow$ ***wingIndex*** The index of a wing, starting at 1

$\leftarrow$ ***segmentIndex*** The index of a segment, starting at 1

$\rightarrow$ ***sectionIndexPtr*** The section index of the outer side

$\rightarrow$ ***elementIndexPtr*** The section element index of the outer side

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if sectionIndexPtr or elementIndexPtr are a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.21 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterSectionAndElementUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, char $**$ *sectionUIDPtr*, char $**$ *elementUIDPtr*)

Returns the section UID and section element UID of the outer side of a given wing segment.

Important change: The memory necessary for the two UIDs must not be freed by the user anymore.

Fortran syntax:

tigl_wing_get_outer_section_and_element_uid(integer cpacsHandle, integer wingIndex, integer segmentIndex, character sectionUIDPtr, character elementUIDPtr, integer returnCode)

Parameters:

$\leftarrow$ ***cpacsHandle*** Handle for the CPACS configuration

$\leftarrow$ ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***sectionUIDPtr*** The section UID of the outer side

→ ***elementUIDPtr*** The section element UID of the outer side

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if sectionIndexPtr or elementIndexPtr are a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.22 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetProfileName (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *sectionIndex*, int *elementIndex*, char ∗∗ *profileNamePtr*)

Returns the name of a wing profile. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

Fortran syntax:

tigl_wing_get_profile_name(integer cpacsHandle, integer wingIndex, integer sectionIndex, integer elementIndex, character∗n profileNamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***sectionIndex*** The index of a section, starting at 1

← ***elementIndex*** The index of an element on the section

→ ***profileNamePtr*** The name of the wing profile

Usage example:

```
TiglReturnCode returnCode;
char* namePtr = 0;
returnCode = tiglWingGetProfileName(cpacsHandle, wing, section, element, &namePtr);
printf("Profile name is %s\n", namePtr);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if profileNamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.23 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSectionUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *sectionIndex*, char ∗∗ *uidNamePtr*)

Returns the UID of a section of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

Fortran syntax:

tigl_wing_get_section_uid(integer cpacsHandle, integer wingIndex, integer sectionIndex, character∗n uIDNamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***sectionIndex*** The index of a section, starting at 1

→ ***uidNamePtr*** The uid of the wing

Usage example:

```
TiglReturnCode returnCode;
char* uidPtr = 0;
returnCode = tiglWingGetSectionUID(cpacsHandle, wing, sectionUID, &uidPtr);
printf("The UID of the section of wing %d is %s\n", wing, uidPtr);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if profileNamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.24 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentCount (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int ∗ *segmentCountPtr*)

Returns the number of segments for a wing in a CPACS configuration.

Fortran syntax:

tigl_wing_get_segment_count(integer cpacsHandle, integer wingIndex, integer segmentCountPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of the wing, starting at 1

→ ***segmentCountPtr*** Pointer to the number of segments

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex is not valid
- TIGL_NULL_POINTER if segmentCountPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.25 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentEtaXsi (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, double *pointX*, double *pointY*, double *pointZ*, int ∗ *segmentIndex*, double ∗ *eta*, double ∗ *xsi*, int ∗ *isOnTop*)

Inverse function to tiglWingGetLowerPoint and tiglWingGetLowerPoint. Calculates to a point (x,y,z) in global coordinates the wing segment coordinates and the wing segment index.

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of the wing, starting at 1

← ***pointX*** X-Coordinate of the global point

← ***pointY*** Y-Coordinate of the global point

← ***pointZ*** Z-Coordinate of the global point

→ ***segmentIndex*** The index of the segment of the wing, starting at 1

→ ***eta*** Eta value in segment coordinates

→ ***xsi*** Xsi value in segment coordinates

→ ***isOnTop*** isOnTop is 1, if the point lies on the upper wing face, else 0.

Returns:

- TIGL_SUCCESS if a solution was found
- TIGL_NOT_FOUND if the point does not lie on the wing
- TIGL_INDEX_ERROR if wingIndex is not valid
- TIGL_NULL_POINTER if eta, xsi or isOnTop are null pointers
- TIGL_ERROR if some other error occurred

3.3.2.26 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentIndex (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *segmentUID*, int ∗ *segmentIndexPtr*, int ∗ *wingIndexPtr*)

Returns the Index of a segment of a wing.

Fortran syntax:

tigl_wing_get_segment_index(integer cpacsHandle, integer wingIndex, character∗n uIDNamePtr, integer segmentIndex, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***segmentUID*** The uid of the wing

→ ***segmentIndexPtr*** The index of a segment, starting at 1

→ ***wingIndexPtr*** The index of a wing, starting at 1

Usage example:

```
TiglReturnCode returnCode;
int segmentIndex, wingIndex;
returnCode = tiglWingGetSegmentIndex(cpacsHandle, segmentUID, &segmentIndex, &wingIndex);
printf("The Index of the segment of wing %d is %d\n", wingIndex, segmentIndex);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex is not valid
- TIGL_UID_ERROR if the segmentUID does not exist
- TIGL_NULL_POINTER if segmentUID is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.27 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, char ∗∗ *uidNamePtr*)

Returns the UID of a segment of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

Fortran syntax:

tigl_wing_get_segment_uid(integer cpacsHandle, integer wingIndex, integer segmentIndex, character∗n uIDNamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***uidNamePtr*** The uid of the wing

Usage example:

```
TiglReturnCode returnCode;
char* uidPtr = 0;
returnCode = tiglWingGetSegmentUID(cpacsHandle, wing, segmentID, &uidPtr);
printf("The UID of the segment of wing %d is %s\n", wing, uidPtr);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if profileNamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.28 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSymmetry (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, TiglSymmetryAxis ∗ *symmetryAxisPtr*)

Returns the Symmetry Enum if the wing has symmetry-axis.

Fortran syntax:

tigl_wing_get_symmetry(integer cpacsHandle, int wingIndex, integer symmetryAxisPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** Index of the Wing to export

→ ***symmetryAxisPtr*** Returning TiglSymmetryAxis enum pointer

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.3.2.29 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, char ** *uidNamePtr*)

Returns the UID of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

Fortran syntax:

tigl_wing_get_uid(integer cpacsHandle, integer wingIndex, character*n uIDNamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration
← ***wingIndex*** The index of a wing, starting at 1
→ ***uidNamePtr*** The uid of the wing

Usage example:

```
TiglReturnCode returnCode;
char* uidPtr = 0;
returnCode = tiglWingGetUID(cpacsHandle, wing, &uidPtr);
printf("The UID of the wing is %s\n", uidPtr);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if profileNamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.3.2.30 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetUpperPoint (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, double *eta*, double *xsi*, double * *pointXPtr*, double * *pointYPtr*, double * *pointZPtr*)

Returns a point on the upper wing surface for a a given wing and segment index.

Returns a point on the upper wing surface in dependence of parameters eta and xsi, which range from 0.0 to 1.0. For eta = 0.0, xsi = 0.0 the point is equal to the leading edge on the inner section of the given segment. For eta = 1.0, xsi = 1.0 the point is equal to the trailing edge on the outer section of the given segment. The point is returned in absolute world coordinates.

Fortran syntax:

tigl_wing_get_upper_point(integer cpacsHandle, integer wingIndex, integer segmentIndex, real eta, real xsi, real pointXPtr, real pointYPtr, real pointZPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of the wing, starting at 1

← ***segmentIndex*** The index of the segment of the wing, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0; eta = 0 for inner section , eta = 1 for outer section

← ***xsi*** xsi in the range 0.0 <= xsi <= 1.0; xsi = 0 for Leading Edge, xsi = 1 for Trailing Edge

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.3.2.31 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetUpperPointAtAngle (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, double *eta*, double *xsi*, double *xAngle*, double *yAngle*, double *zAngle*, double ∗ *pointXPtr*, double ∗ *pointYPtr*, double ∗ *pointZPtr*)

Returns a point on the upper wing surface for a a given wing and segment index. This function is different from tiglWingGetUpperPoint, as a point on the cord surface is calculated (given eta and xsi coordinates) and the normal vector at that point is rotated around the x, y, and z axis by the given angles. Finally, the intersection point of the rotated normal and the upper wing shell is computed. The point is returned in absolute world coordinates.

Fortran syntax:

tigl_wing_get_upper_point_at_angle(integer cpacsHandle, integer wingIndex, integer segmentIndex, real eta, real xsi, real xangle, real yangle, real zangle, real pointXPtr, real pointYPtr, real pointZPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of the wing, starting at 1

← ***segmentIndex*** The index of the segment of the wing, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0; eta = 0 for inner section , eta = 1 for outer section

← ***xsi*** xsi in the range 0.0 <= xsi <= 1.0; xsi = 0 for Leading Edge, xsi = 1 for Trailing Edge

← ***xAngle*** Angle of normal vector rotation around the x-Axis (in degrees)

← ***yAngle*** Angle of normal vector rotation around the y-Axis (in degrees)

← ***zAngle*** Angle of normal vector rotation around the z-Axis (in degrees)

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if wingIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.3.2.32 TIGL_COMMON_EXPORT TiglReturnCode tiglWingSegmentPointGetComponentSegmentEtaXsi (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *segmentUID*, const char ∗ *componentSegmentUID*, double *segmentEta*, double *segmentXsi*, double ∗ *eta*, double ∗ *xsi*)

Returns eta, xsi coordinates of a componentSegment given segmentEta and segmentXsi on a wing segment.

Fortran syntax:

tigl_wing_segment_point_get_component_segment_eta_xsi(integer cpacsHandle, character∗n segmentUID, character∗n componentSegmentUID, real segmentEta, real segmentXsi real eta, real xsi, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***segmentUID*** UID of the wing segment to search for

← ***componentSegmentUID*** UID of the associated componentSegment

← ***segmentEta,segmentXsi*** Eta and Xsi coordinates of the point on the wing segment

→ ***eta*** Eta of the point on the corresponding component segment.

→ ***xsi*** Xsi of the point on the corresponding component segment.

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_UID_ERROR if the segment or the component segment does not exist
- TIGL_ERROR if some other error occurred

3.4 Functions for fuselage calculations

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetCircumference (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double ∗circumferencePtr)

  Returns the circumference of a fuselage surface for a given fuselage and segment index and an eta.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndConnectedSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int ∗segmentCountPtr)

  Returns the count of segments connected to the end section of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndConnectedSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int n, int ∗connectedIndexPtr)

  Returns the index (number) of the n-th segment connected to the end section of a given fuselage segment. n starts at 1.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndSectionAndElementIndex (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int ∗sectionIndexPtr, int ∗elementIndexPtr)

  Returns the section index and section element index of the end side of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndSectionAndElementUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, char ∗∗sectionUIDPtr, char ∗∗elementUIDPtr)

  Returns the section UID and section element UID of the end side of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetMinumumDistanceToGround (TiglCPACSConfigurationHandle cpacsHandle, char ∗fuselageUID, double axisPntX, double axisPntY, double axisPntZ, double axisDirX, double axisDirY, double axisDirZ, double angle, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns the point where the distance between the selected fuselage and the ground is at minimum. The Fuselage could be turned with a given angle at at given axis, specified by a point and a direction.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetNumPointsOnXPlane (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double xpos, int ∗numPointsPtr)

  Returns the number of points on a fuselage surface for a given fuselage and a give x-position.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetNumPointsOnYPlane (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double ypos, int ∗numPointsPtr)

  Returns the number of points on a fuselage surface for a given fuselage and a give y-position.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPoint (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double zeta, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointAngle (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double alpha, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index and an angle alpha (degree).

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointAngleTranslated (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double alpha, double y_cs, double z_cs, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index and an angle alpha (degree). 0 degree of the angle alpha is meant to be "up" in the direction of the positive z-axis like specifies in cpacs. The origin of the line that will be rotated with the angle alpha could be translated via the parameters y_cs and z_cs.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointOnXPlane (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double xpos, int pointIndex, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointOnYPlane (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double ypos, int pointIndex, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetProfileName (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int sectionIndex, int elementIndex, char ∗∗profileNamePtr)

Returns the name of a fuselage profile. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSectionUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int sectionIndex, char ∗∗uidNamePtr)

  Returns the UID of a section of a fuselage. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int ∗segmentCountPtr)

  Returns the number of segments for a fuselage in a CPACS configuration.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, char ∗∗uidNamePtr)

  Returns the UID of a segment of a fuselage. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartConnectedSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int ∗segmentCountPtr)

  Returns the count of segments connected to the start section of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartConnectedSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int n, int ∗connectedIndexPtr)

  Returns the index (number) of the n-th segment connected to the start section of a given fuselage segment. n starts at 1.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartSectionAndElementIndex (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int ∗sectionIndexPtr, int ∗elementIndexPtr)

  Returns the section index and section element index of the start side of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartSectionAndElementUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, char ∗∗sectionUIDPtr, char ∗∗elementUIDPtr)

  Returns the section UID and section element UID of the start side of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSymmetry (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, TiglSymmetryAxis ∗symmetryAxisPtr)

  Returns the Symmetry Enum if the fuselage has symmetry-axis.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, char ∗∗uidNamePtr)

  Returns the UID of a fuselage. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglGetFuselageCount (TiglCPACSConfigurationHandle cpacsHandle, int ∗fuselageCountPtr)

  Returns the number of fuselages in a CPACS configuration.

3.4.1 Detailed Description

Function to handle fuselage geometry's with TIGL.

3.4.2 Function Documentation

3.4.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetCircumference (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double *eta*, double ∗ *circumferencePtr*)

Returns the circumference of a fuselage surface for a given fuselage and segment index and an eta.

Returns the circumference of a fuselage segment of a given fuselage in dependence of parameters eta with 0.0 <= eta <= 1.0. For eta = 0.0 the point lies on the start profile of the segment, for eta = 1.0 it lies on the end profile of the segment.

Fortran syntax:

tigl_fuselage_get_circumference(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real eta, real circumference, integer returnCode)

Parameters:

← ***cpacsHandle***	Handle for the CPACS configuration
← ***fuselageIndex***	The index of the fuselage, starting at 1
← ***segmentIndex***	The index of the segment of the fuselage, starting at 1
← ***eta***	eta in the range 0.0 <= eta <= 1.0
→ ***circumferencePtr***	The Circumference of the fuselage at the given position

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.4.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndConnectedSegmentCount (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, int $*$ *segmentCountPtr*)

Returns the count of segments connected to the end section of a given fuselage segment.

Fortran syntax:

tigl_fuselage_get_end_connected_segment_count(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, integer segmentCountPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***segmentCountPtr*** Pointer to the count of connected segments

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if segmentCountPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.3 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndConnectedSegmentIndex (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, int *n*, int $*$ *connectedIndexPtr*)

Returns the index (number) of the n-th segment connected to the end section of a given fuselage segment. n starts at 1.

Fortran syntax:

tigl_fuselage_get_end_connected_segment_index(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, integer n, integer connectedIndexPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

← ***n*** n-th segment searched, 1 <= n <= tiglFuselageGetEndConnectedSegmentCount(...)

→ ***connectedIndexPtr*** Pointer to the segment index of the n-th connected segment

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle or no n-th connected segment was found
- TIGL_INDEX_ERROR if fuselageIndex, segmentIndex or n are not valid
- TIGL_NULL_POINTER if segmentIndexPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.4 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndSectionAndElementIndex (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, int * *sectionIndexPtr*, int * *elementIndexPtr*)

Returns the section index and section element index of the end side of a given fuselage segment.

Fortran syntax:

tigl_fuselage_get_end_section_and_element_index(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, integer sectionIndexPtr, integer elementIndexPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration
← ***fuselageIndex*** The index of a fuselage, starting at 1
← ***segmentIndex*** The index of a segment, starting at 1
→ ***sectionIndexPtr*** The section index UID the end side
→ ***elementIndexPtr*** The section element UID of the end side

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if sectionIndexPtr or elementIndexPtr are a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.5 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndSectionAndElementUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, char ** *sectionUIDPtr*, char ** *elementUIDPtr*)

Returns the section UID and section element UID of the end side of a given fuselage segment.

Important change: The memory necessary for the two UIDs must not to be freed by the user anymore.

Fortran syntax:

tigl_fuselage_get_end_section_and_element_uid(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, character sectionUIDPtr, character elementUIDPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***sectionUIDPtr*** The section UID the end side

→ ***elementUIDPtr*** The section element UID of the end side

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_ERROR if some other error occurred

3.4.2.6 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetMinumumDistanceToGround (TiglCPACSConfigurationHandle *cpacsHandle*, char ∗ *fuselageUID*, double *axisPntX*, double *axisPntY*, double *axisPntZ*, double *axisDirX*, double *axisDirY*, double *axisDirZ*, double *angle*, double ∗ *pointXPtr*, double ∗ *pointYPtr*, double ∗ *pointZPtr*)

Returns the point where the distance between the selected fuselage and the ground is at minimum. The Fuselage could be turned with a given angle at at given axis, specified by a point and a direction.

Fortran syntax:

tigl_fuselage_get_minumum_distance_to_ground(integer cpacsHandle, character∗n fuselageUID, real axisPntX, real axisPntY, real axisPntZ, real axisDirX, real axisDirY, real axisDirZ, real angle, real pointXPtr, real pointYPtr, real pointZPtr)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageUID*** The uid of the fuselage

← ***axisPntX*** X-coordinate of the point that specifies the axis of rotation

← ***axisPntY*** Y-coordinate of the point that specifies the axis of rotation

← ***axisPntZ*** Z-coordinate of the point that specifies the axis of rotation

← ***axisDirX*** X-coordinate of the direction that specifies the axis of rotation

← ***axisDirY*** Y-coordinate of the direction that specifies the axis of rotation

← ***axisDirZ*** Z-coordinate of the direction that specifies the axis of rotation

← ***angle*** The angle (in Degree) by which the fuselage should be turned on the axis of rotation

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageUID is not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.4.2.7 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetNumPointsOnXPlane (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double *eta*, double *xpos*, int ∗ *numPointsPtr*)

Returns the number of points on a fuselage surface for a given fuselage and a give x-position.

Returns the number of points on a fuselage segment of a given fuselage in dependence of parameters eta and at all x-positions with 0.0 <= eta <= 1.0. For eta = 0.0 the point lies on the start profile of the segment, for eta = 1.0 it lies on the end profile of the segment.

Fortran syntax:

tigl_fuselage_get_num_points_on_xplane(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real eta, real xpos, integer numPoints, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of the fuselage, starting at 1

← ***segmentIndex*** The index of the segment of the fuselage, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***xpos*** X position

→ ***numPointsPtr*** Pointer to a integer for the number of intersection points

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.4.2.8 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetNumPointsOnYPlane (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double *eta*, double *ypos*, int ∗ *numPointsPtr*)

Returns the number of points on a fuselage surface for a given fuselage and a give y-position.

Returns the number of points on a fuselage segment of a given fuselage in dependence of parameters eta and at all y-positions with 0.0 <= eta <= 1.0. For eta = 0.0 the point lies on the start profile of the segment, for eta = 1.0 it lies on the end profile of the segment.

Fortran syntax:

tigl_fuselage_get_num_points_on_yplane(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real eta, real ypos, integer numPoints, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of the fuselage, starting at 1

← ***segmentIndex*** The index of the segment of the fuselage, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***ypos*** Y position

→ ***numPointsPtr*** Pointer to a interger for the number of intersection points

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.4.2.9 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPoint (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double *eta*, double *zeta*, double $*$ *pointXPtr*, double $*$ *pointYPtr*, double $*$ *pointZPtr*)

Returns a point on a fuselage surface for a given fuselage and segment index.

Returns a point on a fuselage segment of a given fuselage in dependence of parameters eta and zeta with 0.0 <= eta <= 1.0 and 0.0 <= zeta <= 1.0. For eta = 0.0 the point lies on the start profile of the segment, for eta = 1.0 it lies on the end profile of the segment. For zeta = 0.0 the point is the identical to the start point of the profile wire, for zeta = 1.0 it is identical to the last profile point. The point is returned in absolute world coordinates.

Fortran syntax:

tigl_fuselage_get_point(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real eta, real zeta, real pointXPtr, real pointYPtr, real pointZPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of the fuselage, starting at 1

← ***segmentIndex*** The index of the segment of the fuselage, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***zeta*** zeta in the range 0.0 <= zeta <= 1.0

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.4.2.10 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointAngle (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double *eta*, double *alpha*, double $*$ *pointXPtr*, double $*$ *pointYPtr*, double $*$ *pointZPtr*)

Returns a point on a fuselage surface for a given fuselage and segment index and an angle alpha (degree).

Returns a point on a fuselage segment of a given fuselage in dependence of parameters eta and at all y-positions with 0.0 <= eta <= 1.0. For eta = 0.0 the point lies on the start profile of the segment, for eta = 1.0 it lies on the end profile of the segment. The angle alpha is calculated in degrees. Alpha=0 degree is meant to be "up" in the direction of the positive z-axis like specifies in cpacs. It's orientation is the mathematical negative rotation direction around the X-axis, i.e. looking in flight direction, an angle of 45 degrees resembles a point on the top-left fuselage. The point is returned in absolute world coordinates.

Fortran syntax:

tigl_fuselage_get_point_angle(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real eta, real alpha, real pointXPtr, real pointYPtr, real pointZPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of the fuselage, starting at 1

← ***segmentIndex*** The index of the segment of the fuselage, starting at 1

← ***eta*** Eta in the range 0.0 <= eta <= 1.0

← ***alpha*** Angle alpha in degrees. No range restrictions.

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred, for example if there is no point at the given eta.

3.4.2.11 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointAngleTranslated (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double *eta*, double *alpha*, double *y_cs*, double *z_cs*, double ∗ *pointXPtr*, double ∗ *pointYPtr*, double ∗ *pointZPtr*)

Returns a point on a fuselage surface for a given fuselage and segment index and an angle alpha (degree). 0 degree of the angle alpha is meant to be "up" in the direction of the positive z-axis like specifies in cpacs. The origin of the line that will be rotated with the angle alpha could be translated via the parameters y_cs and z_cs.

Returns a point on a fuselage segment of a given fuselage in dependence of parameters eta and at all y-positions with 0.0 <= eta <= 1.0. For eta = 0.0 the point lies on the start profile of the segment, for eta = 1.0 it lies on the end profile of the segment. The angle alpha is calculated in degrees. It's orientation is the mathematical negative rotation direction around the X-axis, i.e. looking in flight direction, an angle of 45 degrees resembles a point on the top-left fuselage. The parameters x_cs and z_cs must be in absolute world coordinates. The point is returned in absolute world coordinates.

Fortran syntax:

tigl_fuselage_get_point_angle_translated(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real eta, real alpha, real y_cs, real z_cs real pointXPtr, real pointYPtr, real pointZPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of the fuselage, starting at 1

← ***segmentIndex*** The index of the segment of the fuselage, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***alpha*** Angle alpha in degrees. No range restrictions.

← ***y_cs*** Shifts the origin of the angle alpha in y-direction.

← ***z_cs*** Shifts the origin of the angle alpha in z-direction.

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred, for example if there is no point at the given eta and the given shifting parameters y_cs and z_cs.

3.4.2.12 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointOnXPlane (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double *eta*, double *xpos*, int *pointIndex*, double ∗ *pointXPtr*, double ∗ *pointYPtr*, double ∗ *pointZPtr*)

Returns a point on a fuselage surface for a given fuselage and segment index.

Returns a point on a fuselage segment of a given fuselage in dependence of parameters eta and at all y-positions with 0.0 <= eta <= 1.0. For eta = 0.0 the point lies on the start profile of the segment, for eta = 1.0 it lies on the end profile of the segment. The point is returned in absolute world coordinates.

Fortran syntax:

tigl_fuselage_get_point_on_xplane(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real eta, real xpos, integer pointIndex, real pointXPtr, real pointYPtr, real pointZPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of the fuselage, starting at 1

← ***segmentIndex*** The index of the segment of the fuselage, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***xpos*** x position of a cutting plane

← ***pointIndex*** Defines witch point if more than one.

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.4.2.13 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointOnYPlane (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double *eta*, double *ypos*, int *pointIndex*, double * *pointXPtr*, double * *pointYPtr*, double * *pointZPtr*)

Returns a point on a fuselage surface for a given fuselage and segment index.

Returns a point on a fuselage segment of a given fuselage in dependence of parameters eta and at all y-positions with 0.0 <= eta <= 1.0. For eta = 0.0 the point lies on the start profile of the segment, for eta = 1.0 it lies on the end profile of the segment. The point is returned in absolute world coordinates.

Fortran syntax:

tigl_fuselage_get_point_on_yplane(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real eta, real ypos, integer pointIndex, real pointXPtr, real pointYPtr, real pointZPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of the fuselage, starting at 1

← ***segmentIndex*** The index of the segment of the fuselage, starting at 1

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***ypos*** Y position

← ***pointIndex*** Defines witch point if more than one.

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.4.2.14 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetProfileName (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *sectionIndex*, int *elementIndex*, char ∗∗ *profileNamePtr*)

Returns the name of a fuselage profile. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

Fortran syntax:

tigl_fuselage_get_profile_name(integer cpacsHandle, integer fuselageIndex, integer sectionIndex, integer elementIndex, character∗n profileNamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***sectionIndex*** The index of a section, starting at 1

← ***elementIndex*** The index of an element on the section

→ ***profileNamePtr*** The name of the wing profile

Usage example:

```
TiglReturnCode returnCode;
char* namePtr = 0;
returnCode = tiglFuselageGetProfileName(cpacsHandle, fuselage, section, element, &namePtr);
printf("Profile name is %s\n", namePtr);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if profileNamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.15 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSectionUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *sectionIndex*, char ∗∗ *uidNamePtr*)

Returns the UID of a section of a fuselage. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

Fortran syntax:

tigl_fuselage_get_section_uid(integer cpacsHandle, integer fuselageIndex, integer sectionIndex, character∗n uIDNamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***sectionIndex*** The index of a section, starting at 1

→ ***uidNamePtr*** The uid of the fuselage

Usage example:

```
TiglReturnCode returnCode;
char* uidPtr = 0;
returnCode = tiglFuselageGetSectionUID(cpacsHandle, fuselage, sectionUID, &uidPtr);
printf("The UID of the section of fuselage %d is %s\n", fuselage, uidPtr);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if profileNamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.16 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentCount (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int ∗ *segmentCountPtr*)

Returns the number of segments for a fuselage in a CPACS configuration.

Fortran syntax:

tigl_fuselage_get_segment_count(integer cpacsHandle, integer fuselageIndex, integer segmentCountPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration
← ***fuselageIndex*** The index of the fuselage, starting at 1
→ ***segmentCountPtr*** Pointer to the number of segments

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex is not valid
- TIGL_NULL_POINTER if segmentCountPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.17 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, char ∗∗ *uidNamePtr*)

Returns the UID of a segment of a fuselage. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

Fortran syntax:

tigl_fuselage_get_segment_uid(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, character∗n uIDNamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration
← ***fuselageIndex*** The index of a fuselage, starting at 1
← ***segmentIndex*** The index of a segment, starting at 1
→ ***uidNamePtr*** The uid of the fuselage

Usage example:

```
TiglReturnCode returnCode;
char* uidPtr = 0;
returnCode = tiglFuselageGetSegmentUID(cpacsHandle, fuselage, segmentID, &uidPtr);
printf("The UID of the segment of fuselage %d is %s\n", fuselage, uidPtr);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if profileNamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.18 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartConnectedSegmentCount (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, int ∗ *segmentCountPtr*)

Returns the count of segments connected to the start section of a given fuselage segment.

Fortran syntax:

tigl_fuselage_get_start_connected_segment_count(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, integer segmentCountPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***segmentCountPtr*** Pointer to the count of connected segments

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if segmentCountPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.19 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartConnectedSegmentIndex (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, int *n*, int ∗ *connectedIndexPtr*)

Returns the index (number) of the n-th segment connected to the start section of a given fuselage segment. n starts at 1.

Fortran syntax:

tigl_fuselage_get_start_connected_segment_index(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, integer n, integer connectedIndexPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

← ***n*** n-th segment searched, 1 <= n <= tiglFuselageGetStartConnectedSegment-Count(...)

→ ***connectedIndexPtr*** Pointer to the segment index of the n-th connected segment

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle or no n-th connected segment was found
- TIGL_INDEX_ERROR if fuselageIndex, segmentIndex or n are not valid
- TIGL_NULL_POINTER if segmentIndexPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.20 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartSectionAndElementIndex (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, int ∗ *sectionIndexPtr*, int ∗ *elementIndexPtr*)

Returns the section index and section element index of the start side of a given fuselage segment.

Fortran syntax:

tigl_fuselage_get_start_section_and_element_index(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, integer sectionIndexPtr, integer elementIndex-Ptr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***sectionIndexPtr*** The section UID of the start side

→ ***elementIndexPtr*** The section element UID of the start side

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_NULL_POINTER if sectionIndexPtr or elementIndexPtr are a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.21 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartSectionAndElementUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, char ∗∗ *sectionUIDPtr*, char ∗∗ *elementUIDPtr*)

Returns the section UID and section element UID of the start side of a given fuselage segment.

Important change: The memory necessary for the two UIDs must not to be freed by the user anymore.

Fortran syntax:

tigl_fuselage_get_start_section_and_element_uid(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, character sectionUIDPtr, character elementUIDPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***sectionUIDPtr*** The section UID of the start side

→ ***elementUIDPtr*** The section element UID of the start side

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex or segmentIndex are not valid
- TIGL_ERROR if some other error occurred

3.4.2.22 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSymmetry (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, TiglSymmetryAxis ∗ *symmetryAxisPtr*)

Returns the Symmetry Enum if the fuselage has symmetry-axis.

Fortran syntax:

tigl_fuselage_get_symmetry(integer cpacsHandle, int fuselageIndex, integer symmetryAxisPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** Index of the Wing to export

→ ***symmetryAxisPtr*** Returning TiglSymmetryAxis enum pointer

Returns:

- TIGL_SUCCESS if no error occurred

- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.4.2.23 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetUID (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, char ∗∗ *uidNamePtr*)

Returns the UID of a fuselage. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

Fortran syntax:

tigl_fuselage_get_uid(integer cpacsHandle, integer fuselageIndex, character∗n uID-NamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

→ ***uidNamePtr*** The uid of the fuselage

Usage example:

```
TiglReturnCode returnCode;
char* uidPtr = 0;
returnCode = tiglFuselageGetUID(cpacsHandle, fuselage, &uidPtr);
printf("The UID of the fuselage is %s\n", uidPtr);
```

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if profileNamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.4.2.24 TIGL_COMMON_EXPORT TiglReturnCode tiglGetFuselageCount (TiglCPACSConfigurationHandle *cpacsHandle*, int ∗ *fuselageCountPtr*)

Returns the number of fuselages in a CPACS configuration.

Fortran syntax:

tigl_get_fuselage_count(integer cpacsHandle, integer fuselageCountPtr, integer return-Code)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

→ ***fuselageCountPtr*** Pointer to the number of fuselages

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if fuselageCountPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.5 Functions for boolean calculations

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglComponentIntersectionLineCount (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentUidOne, const char ∗componentUidTwo, int ∗numWires)

  Returns the number if intersection lines of two geometric components.

- TIGL_COMMON_EXPORT TiglReturnCode tiglComponentIntersectionPoint (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentUidOne, const char ∗componentUidTwo, int lineID, double eta, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on the intersection line of two geometric components. Often there are more that one intersection line, therefore you need to specify the line.

- TIGL_COMMON_EXPORT TiglReturnCode tiglComponentIntersectionPoints (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentUidOne, const char ∗componentUidTwo, int lineID, const double ∗etaArray, int numberOfPoints, double ∗pointXArray, double ∗pointYArray, double ∗pointZArray)

  Convienience function to returns a list of points on the intersection line of two geometric components. Often there are more that one intersection line, therefore you need to specify the line.

3.5.1 Detailed Description

Function for boolean calculations on wings/fuselages.

3.5.2 Function Documentation

3.5.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglComponentIntersectionLineCount (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentUidOne*, const char ∗ *componentUidTwo*, int ∗ *numWires*)

Returns the number if intersection lines of two geometric components.

Fortran syntax:

TiglReturnCode tigl_component_intersection_line_count(integer cpacsHandle, integer fuselageIndex, integer∗ wingIndex, integer returnCode);

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***componentUidOne*** The UID of the first component

← ***componentUidTwo*** The UID of the second component

→ ***numWires*** The number of intersection lines

Returns:

- TIGL_SUCCESS if there are not intersection lines
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_ERROR if some other error occurred

3.5.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglComponentIntersectionPoint (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentUidOne*, const char ∗ *componentUidTwo*, int *lineID*, double *eta*, double ∗ *pointXPtr*, double ∗ *pointYPtr*, double ∗ *pointZPtr*)

Returns a point on the intersection line of two geometric components. Often there are more that one intersection line, therefore you need to specify the line.

Returns a point on the intersection line between a surface and a wing in dependence of parameter eta which range from 0.0 to 1.0. The point is returned in absolute world coordinates.

Fortran syntax:

TiglReturnCode tigl_component_intersection_point(integer cpacsHandle, integer fuselageIndex, integer wingIndex, real eta, real∗ pointXPtr, real∗ pointYPtr, real∗ pointZPtr, integer returnCode);

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***componentUidOne*** The UID of the first component

← ***componentUidTwo*** The UID of the second component

← ***lineID*** The index of the intersection wire, get wire number with "tiglComponentIntersectionLineCount"

← ***eta*** eta in the range 0.0 <= eta <= 1.0

→ ***pointXPtr*** Pointer to the x-coordinate of the point in absolute world coordinates

→ ***pointYPtr*** Pointer to the y-coordinate of the point in absolute world coordinates

→ ***pointZPtr*** Pointer to the z-coordinate of the point in absolute world coordinates

Returns:

- TIGL_SUCCESS if a point was found
- TIGL_NOT_FOUND if no point was found or the cpacs handle is not valid
- TIGL_NULL_POINTER if pointXPtr, pointYPtr or pointZPtr are null pointers
- TIGL_ERROR if some other error occurred

3.5.2.3 TIGL_COMMON_EXPORT TiglReturnCode tiglComponentIntersectionPoints (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentUidOne*, const char ∗ *componentUidTwo*, int *lineID*, const double ∗ *etaArray*, int *numberOfPoints*, double ∗ *pointXArray*, double ∗ *pointYArray*, double ∗ *pointZArray*)

Convienience function to returns a list of points on the intersection line of two geometric components. Often there are more that one intersection line, therefore you need to specify the line.

Returns a point on the intersection line between a surface and a wing in dependence of parameter eta which range from 0.0 to 1.0. The point is returned in absolute world coordinates.

Fortran syntax:

TiglReturnCode tigl_component_intersection_point(integer cpacsHandle, integer fuselageIndex, integer wingIndex, real∗ etaArray, integer numberOfPoints, real∗ pointXPtr, real∗ pointYPtr, real∗ pointZPtr, integer returnCode);

3.6 Export Functions

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportFusedSTEP (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the fused/trimmed geometry of a CPACS configuration to STEP format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportFusedWingFuselageIGES (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the boolean fused geometry of a CPACS configuration to IGES format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportFuselageColladaByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗fuselageUID, const char ∗filename, double deflection)

  Exports the boolean fused geometry of a fuselage (selected by uid) meshed to Collada (∗.dae) format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportIGES (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the geometry of a CPACS configuration to IGES format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageSTL (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a fuselage meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageSTLByUID (TiglCPACSConfigurationHandle cpacsHandle, const char ∗fuselageUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a fuselage meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageVTKByIndex (const TiglCPACSConfigurationHandle cpacsHandle, const int fuselageIndex, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a fuselage (selected by index) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageVTKByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗fuselageUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a fuselage (selected by uid) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageVTKSimpleByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗fuselageUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a fuselage (selected by uid) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedGeometrySTL (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of the whole configuration meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedGeometryVTK (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of the whole configuration meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedGeometryVTKSimple (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of the whole configuration meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingSTL (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a wing meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingSTL-ByUID (TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a wing meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingVTK-ByIndex (const TiglCPACSConfigurationHandle cpacsHandle, const int wingIndex, const char ∗filenamePtr, const double deflection)

  Exports the boolean fused geometry of a wing (selected by id) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingVTK-ByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a wing (selected by UID) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingVTKSim-pleByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a wing (selected by UID) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportSTEP (TiglCPAC-SConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the geometry of a CPACS configuration to STEP format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportStructuredIGES (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the geometry of a CPACS configuration to IGES format. In this version structure, names and metadata will also be exported as much as it is possible.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportStructuredSTEP (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the geometry of a CPACS configuration to STEP format. In this version structure, names and metadata will also be exported as much as it is possible.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportWingColladaByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, const char ∗filename, double deflection)

  Exports the boolean fused geometry of a wing (selected by uid) meshed to Collada (∗.dae) format.

3.6.1 Detailed Description

Functions for export of wings/fuselages.

# VTK-Export # There a various different VTK exports functions in TIGL. All functions starting with 'tiglExportVTK[Fuselage|Wing]...' are exporting a special triangulation with no duplicated points into a VTK file formated in XML (file extension .vtp) with some custom informations added to the file.

In addition to the triangulated geometry, additional data are written to the VTK file. These data currently include:

- uID: The UID of the fuselage or wing component segment on which the triangle exists.
- segmentIndex: The segmentIndex of the fuselage or wing component segment on which the triangle exists. Kind of redundant to the UID.
- eta/xsi: The parameters in the surface parametrical space of the triangle.
- isOnTop: Flag that indicates whether the triangle is on the top of the wing or not. Please see the cpacs documentation how "up" is defined for wings.

Please note that at this time these information are only valid for wings!

There are two ways, how these additional data are attached to the VTK file. The first is the official VTK way to declare additional data for each polygon/triangle. For each data entry, a <DataArray> tag is added under the xpath

```
/VTKFile/PolyData/Piece/CellData
```

Each CellData contains a vector(list) of values, each of them corresponding the data of one triangle. For example the data entry for the wing segment eta coordinates for 4 triangles looks like.

```
<DataArray type="Float64" Name="eta" NumberOfComponents="1"
    format="ascii" RangeMin="0.000000" RangeMax="1.000000">
        0.25 0.5 0.75 1.0
</DataArray>
```

The second way these data are stored is by using the "MetaData" mechanism of VTK. Here, a <MetaData> tag is added under the xpath

```
/VTKFile/PolyData/Piece/Polys
```

A typical exported MetaData tag looks like the following:

```
<MetaData elements="uID segmentIndex eta xsi isOnTop">
  "rootToInnerkink" 1 3.18702 0.551342 0
  "rootToInnerkink" 1 2.93939 0.581634 0
  "rootToInnerkink" 1 4.15239 0.520915 0
  ...
</MetaData>
```

The 'elements' attribute indicates the number and the names of the additional information tags as a whitespace separated list. In this example you could see 5 information fields with the name.

3.6.2 Function Documentation

3.6.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglExportFusedSTEP (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *filenamePtr*)

Exports the fused/trimmed geometry of a CPACS configuration to STEP format.

In order to fuse the geometry, boolean operations are performed. Depending on the complexity of the configuration, the fusing can take several minutes.

Fortran syntax:

tigl_export_fused_step(integer cpacsHandle, character∗n filenamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***filenamePtr*** Pointer to an STEP export file name

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.6.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglExportFusedWingFuselageIGES (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *filenamePtr*)

Exports the boolean fused geometry of a CPACS configuration to IGES format.

To maintain compatibility with CATIA, the file suffix should be ".igs".

Fortran syntax:

tigl_export_fused_wing_fuselage_iges(integer cpacsHandle, character∗n filenamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***filenamePtr*** Pointer to an IGES export file name

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.6.2.3 TIGL_COMMON_EXPORT TiglReturnCode tiglExportFuselageColladaByUID (const TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *fuselageUID*, const char ∗ *filename*, double *deflection*)

Exports the boolean fused geometry of a fuselage (selected by uid) meshed to Collada (∗.dae) format.

Fortran syntax:

tigl_export_fuselage_collada_by_uid(integer cpacsHandle, character∗n fuselageUID, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageUID*** UID of the Fuselage to export

← ***filename*** Filename of the resulting collada file. It should contain the .dae file name ending.

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filename is a null pointer
- TIGL_INDEX_ERROR if fuselageUID does not exists
- TIGL_ERROR if some other error occurred

3.6.2.4 TIGL_COMMON_EXPORT TiglReturnCode tiglExportIGES (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *filenamePtr*)

Exports the geometry of a CPACS configuration to IGES format.

To maintain compatibility with CATIA, the file suffix should be ".igs".

Fortran syntax:

tigl_export_iges(integer cpacsHandle, character∗n filenamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***filenamePtr*** Pointer to an IGES export file name

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.6.2.5 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageSTL (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of a fuselage meshed to STL format.

Fortran syntax:

tigl_export_meshed_fuselage_stl(integer cpacsHandle, int fuselageIndex, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** Index of the Fuselage to export

← ***filenamePtr*** Pointer to an STL export file name

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if fuselageIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.6.2.6 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageSTLByUID (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *fuselageUID*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of a fuselage meshed to STL format.

Fortran syntax:

tigl_export_meshed_fuselage_stl_by_uid(integer cpacsHandle, character∗n fuselageUID, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageUID*** UID of the Fuselage to export

← ***filenamePtr*** Pointer to an STL export file name

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if fuselageIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.6.2.7 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageVTKByIndex (const TiglCPACSConfigurationHandle *cpacsHandle*, const int *fuselageIndex*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of a fuselage (selected by index) meshed to VTK format.

Fortran syntax:

tigl_export_meshed_fuselage_vtk_by_index(integer cpacsHandle, int fuselageIndex, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** Index of the Fuselage to export

← ***filenamePtr*** Pointer to an VTK export file name (∗.vtp = polygonal XML_-VTK)

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if fuselageIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.6.2.8 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageVTKByUID (const TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *fuselageUID*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of a fuselage (selected by uid) meshed to VTK format.

Fortran syntax:

tigl_export_meshed_fuselage_vtk_by_uid(integer cpacsHandle, character∗n fuselageUID, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageUID*** UID of the Fuselage to export

← ***filenamePtr*** Pointer to an VTK export file name (∗.vtp = polygonal XML_-VTK)

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if fuselageUID does not exists
- TIGL_ERROR if some other error occurred

3.6.2.9 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageVTKSimpleByUID (const TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *fuselageUID*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of a fuselage (selected by uid) meshed to VTK format.

This function does only a very simple, but also very fast meshing on the fuselage and exports them to a VTK file. No additional CPACS relevant information are computed.

Fortran syntax:

tigl_export_meshed_fuselage_vtk_simple_by_uid(integer cpacsHandle, character∗n fuselageUID, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageUID*** UID of the Fuselage to export

← ***filenamePtr*** Pointer to an VTK export file name (∗.vtp = polygonal XML_-VTK)

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if fuselageUID does not exists
- TIGL_ERROR if some other error occurred

3.6.2.10 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedGeometrySTL (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of the whole configuration meshed to STL format.

Fortran syntax:

tigl_export_meshed_geometry_stl(integer cpacsHandle, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***filenamePtr*** Pointer to an STL export file name

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.6.2.11 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedGeometryVTK (const TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of the whole configuration meshed to VTK format.

Fortran syntax:

tigl_export_meshed_geometry_vtk(integer cpacsHandle, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***filenamePtr*** Pointer to an VTK export file name (∗.vtp = polygonal XML_-VTK)

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.6.2.12 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedGeometryVTKSimple (const TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of the whole configuration meshed to VTK format.

This function does only a very simple, but also very fast meshing on the geometry and exports them to a VTK file. No additional CPACS relevant information are computed.

Fortran syntax:

tigl_export_meshed_geometry_vtk_simple(integer cpacsHandle, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***filenamePtr*** Pointer to an VTK export file name (∗.vtp = polygonal XML_-VTK)

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.6.2.13 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingSTL (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of a wing meshed to STL format.

Fortran syntax:

tigl_export_meshed_wing_stl(integer cpacsHandle, int wingIndex, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** Index of the Wing to export

← ***filenamePtr*** Pointer to an STL export file name

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if wingIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.6.2.14 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingSTLByUID (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *wingUID*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of a wing meshed to STL format.

Fortran syntax:

tigl_export_meshed_wing_by_uid(integer cpacsHandle, character∗n wingUID, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingUID*** UID of the Wing to export

← ***filenamePtr*** Pointer to an STL export file name

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if wingIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.6.2.15 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingVTKByIndex (const TiglCPACSConfigurationHandle *cpacsHandle*, const int *wingIndex*, const char ∗ *filenamePtr*, const double *deflection*)

Exports the boolean fused geometry of a wing (selected by id) meshed to VTK format.

Fortran syntax:

tigl_export_meshed_wing_vtk_by_index(integer cpacsHandle, int wingIndex, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** Index of the Wing to export

← ***filenamePtr*** Pointer to an VTK export file name (∗.vtp = polygonal XML_-VTK)

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if wingIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.6.2.16 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingVTKByUID (const TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *wingUID*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of a wing (selected by UID) meshed to VTK format.

Fortran syntax:

tigl_export_meshed_wing_vtk_by_uid(integer cpacsHandle, character∗n wingUID, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingUID*** UID of the Wing to export

← ***filenamePtr*** Pointer to an VTK export file name (∗.vtp = polygonal XML_-VTK)

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if the wing UID does not exists
- TIGL_ERROR if some other error occurred

3.6.2.17 TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingVTKSimpleByUID (const TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *wingUID*, const char ∗ *filenamePtr*, double *deflection*)

Exports the boolean fused geometry of a wing (selected by UID) meshed to VTK format.

This function does only a very simple, but also very fast meshing on the wing segments and exports them to a VTK file. No additional CPACS relevant information are computed.

Fortran syntax:

tigl_export_meshed_wing_vtk_simple_by_uid(integer cpacsHandle, character∗n wingUID, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration
← ***wingUID*** UID of the Wing to export
← ***filenamePtr*** Pointer to an VTK export file name (∗.vtp = polygonal XML_-VTK)
← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_INDEX_ERROR if the wing UID does not exists
- TIGL_ERROR if some other error occurred

3.6.2.18 TIGL_COMMON_EXPORT TiglReturnCode tiglExportSTEP (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *filenamePtr*)

Exports the geometry of a CPACS configuration to STEP format.

Fortran syntax:

tigl_export_step(integer cpacsHandle, character∗n filenamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration
← ***filenamePtr*** Pointer to an STEP export file name

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.6.2.19 TIGL_COMMON_EXPORT TiglReturnCode tiglExportStructuredIGES (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *filenamePtr*)

Exports the geometry of a CPACS configuration to IGES format. In this version structure, names and metadata will also be exported as much as it is possible.

To maintain compatibility with CATIA, the file suffix should be ".igs".

Fortran syntax:

tigl_export_structured_iges(integer cpacsHandle, character∗n filenamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***filenamePtr*** Pointer to an IGES export file name

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.6.2.20 TIGL_COMMON_EXPORT TiglReturnCode tiglExportStructuredSTEP (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *filenamePtr*)

Exports the geometry of a CPACS configuration to STEP format. In this version structure, names and metadata will also be exported as much as it is possible.

Fortran syntax:

tigl_export_structured_step(integer cpacsHandle, character∗n filenamePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***filenamePtr*** Pointer to an STEP export file name

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filenamePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.6.2.21 TIGL_COMMON_EXPORT TiglReturnCode tiglExportWingColladaByUID (const TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *wingUID*, const char ∗ *filename*, double *deflection*)

Exports the boolean fused geometry of a wing (selected by uid) meshed to Collada (∗.dae) format.

Fortran syntax:

tigl_export_wing_collada_by_uid(integer cpacsHandle, character∗n fuselageUID, character∗n filenamePtr, integer returnCode, real deflection)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingUID*** UID of the Wing to export

← ***filename*** Filename of the resulting collada file. It should contain the .dae file name ending.

← ***deflection*** Maximum deflection of the triangulation from the real surface

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if filename is a null pointer
- TIGL_INDEX_ERROR if fuselageUID does not exists
- TIGL_ERROR if some other error occurred

3.7 Material functions

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetMaterialCount (TiglCPACSConfigurationHandle cpacsHandle, const char ∗compSegmentUID, TiglStructureType structureType, double eta, double xsi, int ∗materialCount)

  Returns the number of materials defined at a point on the wing component segment surface.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetMaterialThickness (TiglCPACSConfigurationHandle cpacsHandle, const char ∗compSegmentUID, TiglStructureType structureType, double eta, double xsi, int materialIndex, double ∗thickness)

  Returns one of the material thicknesses of a given point on the wing component segment surface. The number of materials on that point has to be first queried using tiglWingComponentSegmentGetMaterialCount.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetMaterialUID (TiglCPACSConfigurationHandle cpacsHandle, const char ∗compSegmentUID, TiglStructureType structureType, double eta, double xsi, int materialIndex, char ∗∗uid)

  Returns one of the material UIDs of a given point on the wing component segment surface. The number of materials on that point has to be first queried using tiglWingComponentSegmentGetMaterialCount.

3.7.1 Detailed Description

Functions to query material information of wings/fuselages. Materials are currently ony implemented for the wing component segment. Here, materials for the lower and upper wing surface can be queried, which can be a material for the whole skin/surface or a material defined inside a wing cell. A wing cell material overwrites the global skin material, i.e. if the whole wing skin material is aluminum and the trailing edge is made of radar absorbing material, only the absorbing material is returned by the querying functions.

3.7.2 Function Documentation

3.7.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetMaterialCount (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *compSegmentUID*, TiglStructureType *structureType*, double *eta*, double *xsi*, int ∗ *materialCount*)

Returns the number of materials defined at a point on the wing component segment surface.

Fortran syntax:

tigl_wing_component_segment_get_material_count(integer cpacsHandle, character∗n compSegmentUID, integer structType, real eta, real xsi, integer matIndex, character∗n materialUID, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***compSegmentUID*** UID of the component segment

← ***structureType*** Type of structure, where the materials are queried

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***xsi*** xsi in the range 0.0 <= xsi <= 1.0

→ ***materialCount*** Number of materials defined at the given coordinate

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle

- TIGL_NULL_POINTER if compSegmentUID or materialCount is a null pointer
- TIGL_INDEX_ERROR if compSegmentUID or materialIndex is invalid
- TIGL_ERROR if some other error occurred

3.7.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetMaterialThickness (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *compSegmentUID*, TiglStructureType *structureType*, double *eta*, double *xsi*, int *materialIndex*, double ∗ *thickness*)

Returns one of the material thicknesses of a given point on the wing component segment surface. The number of materials on that point has to be first queried using tiglWingComponentSegmentGetMaterialCount.

Fortran syntax:

tigl_wing_component_segment_get_material_thickness(integer cpacsHandle, character∗n compSegmentUID, integer structType, real eta, real xsi, integer matIndex, real thickness, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***compSegmentUID*** UID of the component segment

← ***structureType*** Type of structure, where the materials are queried

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***xsi*** xsi in the range 0.0 <= xsi <= 1.0

← ***materialIndex*** Index of the material to query (1 <= index <= materialCount)

→ ***thickness*** Material thickness at the given coordinate. If no thickness is defined, thickness gets a negative value and TIGL_UNINITIALIZED is returned.

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if compSegmentUID or thickness is a null pointer
- TIGL_INDEX_ERROR if compSegmentUID or materialIndex is invalid
- TIGL_UNINITIALIZED if no thickness is defined for the material
- TIGL_ERROR if some other error occurred

3.7.2.3 TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetMaterialUID (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *compSegmentUID*, TiglStructureType *structureType*, double *eta*, double *xsi*, int *materialIndex*, char ∗∗ *uid*)

Returns one of the material UIDs of a given point on the wing component segment surface. The number of materials on that point has to be first queried using tiglWing-ComponentSegmentGetMaterialCount.

Fortran syntax:

tigl_wing_component_segment_get_material_uid(integer cpacsHandle, character∗n compSegmentUID, integer structType, real eta, real xsi, integer matIndex, character∗n materialUID, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***compSegmentUID*** UID of the component segment

← ***structureType*** Type of structure, where the materials are queried

← ***eta*** eta in the range 0.0 <= eta <= 1.0

← ***xsi*** xsi in the range 0.0 <= xsi <= 1.0

← ***materialIndex*** Index of the material to query (1 <= index <= materialCount)

→ ***uid*** Material uid at the given coordinate

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if compSegmentUID is a null pointer
- TIGL_INDEX_ERROR if compSegmentUID or materialIndex is invalid
- TIGL_ERROR if some other error occurred

3.8 Functions for volume calculations

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentVolume (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double ∗volumePtr)

  Returns the volume of a segment of a fuselage.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetVolume (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, double ∗volumePtr)

  Returns the volume of the fuselage.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentVolume (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double ∗volumePtr)

  Returns the volume of a segment of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetVolume (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, double ∗volumePtr)

  Returns the volume of the wing.

3.8.1 Detailed Description

Function for volume calculations on wings/fuselages.

3.8.2 Function Documentation

3.8.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentVolume (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double ∗ *volumePtr*)

Returns the volume of a segment of a fuselage.

Fortran syntax:

tigl_fuselage_get_segment_volume(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real volumePtr, integer returnCode)

Parameters:

← ***cpacsHandle***	Handle for the CPACS configuration
← ***fuselageIndex***	The index of a fuselage, starting at 1
← ***segmentIndex***	The index of a segment, starting at 1
→ ***volumePtr***	The pointer to a variable for the volume of the fuselage

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if volumePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.8.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetVolume (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, double ∗ *volumePtr*)

Returns the volume of the fuselage.

Fortran syntax:

tigl_fuselage_get_volume(integer cpacsHandle, int fuselageIndex, real volumePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** Index of the fuselage to calculate the volume, starting at 1

→ ***volumePtr*** The volume of the fuselage

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.8.2.3 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentVolume (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, double ∗ *volumePtr*)

Returns the volume of a segment of a wing.

Fortran syntax:

tigl_wing_get_segment_volume(integer cpacsHandle, integer wingIndex, integer segmentIndex, real volumePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***volumePtr*** The pointer to a variable for the volume of the wing

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if volumePtr is a null pointer
- TIGL_ERROR if some other error occurred

3.8.2.4 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetVolume (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, double ∗ *volumePtr*)

Returns the volume of the wing.

Fortran syntax:

tigl_wing_get_volume(integer cpacsHandle, int wingIndex, real volumePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** Index of the Wing to calculate the volume, starting at 1

→ ***volumePtr*** The volume of the wing

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.9 Functions for surface area calculations

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentSurfaceArea (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double ∗surfaceAreaPtr)

  Returns the surface area of a segment of a fuselage.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSurfaceArea (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, double ∗surfaceAreaPtr)

  Returns the surface area of the fuselage. Currently, the area includes also the faces on the fuselage symmetry plane (in case of a symmetric wing). This is in particular a problem for fuselages, where only one half side is defined in CPACS. In future releases, these faces will not belong anymore to the surface area calculation.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetReferenceArea (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, TiglSymmetryAxis symPlane, double ∗referenceAreaPtr)

  Returns the reference area of the wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentSurfaceArea (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double ∗surfaceAreaPtr)

  Returns the surface area of a segment of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSurfaceArea (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, double ∗surfaceAreaPtr)

  Returns the surface area of the wing. Currently, the area includes also the faces on the wing symmetry plane (in case of a symmetric wing). In coming releases, these faces will not belong anymore to the surface area calculation.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetWettedArea (TiglCPACSConfigurationHandle cpacsHandle, char ∗wingUID, double ∗wettedAreaPtr)

  Returns the wetted area of the wing.

3.9.1 Detailed Description

Function for surface area calculations off wings/fuselages.

3.9.2 Function Documentation

3.9.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentSurfaceArea (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, int *segmentIndex*, double ∗ *surfaceAreaPtr*)

Returns the surface area of a segment of a fuselage.

Fortran syntax:

tigl_fuselage_get_segment_surface_area(integer cpacsHandle, integer fuselageIndex, integer segmentIndex, real surfaceAreaPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** The index of a fuselage, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***surfaceAreaPtr*** The pointer to a variable for the surface area of the fuselage-segment

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if surfaceAreaPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.9.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSurfaceArea (TiglCPACSConfigurationHandle *cpacsHandle*, int *fuselageIndex*, double ∗ *surfaceAreaPtr*)

Returns the surface area of the fuselage. Currently, the area includes also the faces on the fuselage symmetry plane (in case of a symmetric wing). This is in particular a problem for fuselages, where only one half side is defined in CPACS. In future releases, these faces will not belong anymore to the surface area calculation.

Fortran syntax:

tigl_fuselage_get_surface_area(integer cpacsHandle, int fuselageIndex, real surfaceAreaPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***fuselageIndex*** Index of the Fuselage to calculate the area, starting at 1

→ ***surfaceAreaPtr*** The surface area of the wing

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if fuselageIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.9.2.3 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetReferenceArea (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, TiglSymmetryAxis *symPlane*, double ∗ *referenceAreaPtr*)

Returns the reference area of the wing.

The reference area of the wing is calculated by taking account the quadrilateral portions of each wing segment by projecting the wing segments into the plane defined by the user. If projection should be avoided, use TIGL_NO_SYMMETRY as symPlane argument.

Fortran syntax:

tigl_wing_get_reference_area(integer cpacsHandle, int wingIndex, real referenceAreaPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** Index of the Wing to calculate the area, starting at 1

← ***symPlane*** Plane on which the wing is projected for calculating the refarea. Values can be:

- TIGL_NO_SYMMETRY, the wing is not projected but its true 3D area is calculated
- TIGL_X_Y_PLANE, the wing is projected onto the x-y plane (use for e.g. main wings and HTPs)
- TIGL_X_Z_PLANE, the wing is projected onto the x-z plane (use for e.g. VTPs)
- TIGL_Y_Z_PLANE, the wing is projected onto the y-z plane

→ ***referenceAreaPtr*** The refence area of the wing

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.9.2.4 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentSurfaceArea (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, int *segmentIndex*, double ∗ *surfaceAreaPtr*)

Returns the surface area of a segment of a wing.

Fortran syntax:

tigl_wing_get_segment_surface_area(integer cpacsHandle, integer wingIndex, integer segmentIndex, real surfaceAreaPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** The index of a wing, starting at 1

← ***segmentIndex*** The index of a segment, starting at 1

→ ***surfaceAreaPtr*** The pointer to a variable for the surface area of the wing

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex, sectionIndex or elementIndex are not valid
- TIGL_NULL_POINTER if surfaceAreaPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.9.2.5 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSurfaceArea (TiglCPACSConfigurationHandle *cpacsHandle*, int *wingIndex*, double ∗ *surfaceAreaPtr*)

Returns the surface area of the wing. Currently, the area includes also the faces on the wing symmetry plane (in case of a symmetric wing). In coming releases, these faces will not belong anymore to the surface area calculation.

Fortran syntax:

tigl_wing_get_surface_area(integer cpacsHandle, int wingIndex, real surfaceAreaPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingIndex*** Index of the Wing to calculate the area, starting at 1

→ ***surfaceAreaPtr*** The surface area of the wing

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_INDEX_ERROR if wingIndex is less or equal zero
- TIGL_ERROR if some other error occurred

3.9.2.6 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetWettedArea (TiglCPACSConfigurationHandle *cpacsHandle*, char ∗ *wingUID*, double ∗ *wettedAreaPtr*)

Returns the wetted area of the wing.

Fortran syntax:

tigl_wing_get_wetted_area(integer cpacsHandle, character∗n wingUID, real referenceAreaPtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***wingUID*** UID of the Wing to calculate the wetted area

→ ***wettedAreaPtr*** The wetted area of the wing

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_UID_ERROR if wingUID is wrong
- TIGL_NULL_POINTER if wingIUD is NULL
- TIGL_ERROR if some other error occurred

3.10 Logging functions.

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogSetFileEnding (const char ∗ending)

  Sets the file ending for logging files. Default is "log".

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogSetTimeInFilenameEnabled (TiglBoolean enabled)

  Enables or disables appending a unique date/time identifier inside the log file name (behind the file prefix). By default, the time indentifier is enabled.

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogSetVerbosity (TiglLogLevel level)

  Set the console verbosity level.

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogToFileDisabled ()

  Disabled file logging. If a log file is currently opened by TiGL it will be closed. The log messages are printed to console. This is the default logging mechanism of TIGL.

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogToFileEnabled (const char ∗filePrefix)

  Sets up the tigl logging mechanism to send all log messages into a file.

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogToFileStreamEnabled (FILE ∗fp)

  Sets up the tigl logging mechanism to send all log messages into an already opened file.

3.10.1 Detailed Description

The following functions are used to customize the behaviour how messages are handled by TiGL. By default, only error messages and warnings are printed to the console.

In addition, TiGL can be told to write message into a log file using the function tiglLogToFileEnabled and tiglLogToFileStreamEnabled. By enabling file logging, other log messages than errors and warnings can be inspected. File logging is disabled by default.

In order to change the verbosity of the TiGL messages printed on console, use tiglLogSetVerbosity.

3.10.2 Function Documentation

3.10.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglLogSetFileEnding (const char ∗ *ending*)

Sets the file ending for logging files. Default is "log".

This function has to be called before tiglLogToFileEnabled to have the desired effect.

Fortran syntax:

tigl_log_set_file_ending(character∗n ending)

Parameters:

← ***ending*** File ending of the logging file. Default is "log".

Returns:

- TIGL_SUCCESS if no error occurred

- TIGL_NULL_POINTER if ending is NULL
- TIGL_ERROR if some error occurred

3.10.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglLogSetTimeInFilenameEnabled (TiglBoolean *enabled*)

Enables or disables appending a unique date/time identifier inside the log file name (behind the file prefix). By default, the time indentifier is enabled.

This function has to be called before tiglLogToFileEnabled to have the desired effect.

Fortran syntax:

tigl_log_set_data_in_filename_enabled(logical enabled)

Parameters:

← ***enabled*** Set to true, if time identifier should be enabled.

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_ERROR if some error occurred

3.10.2.3 TIGL_COMMON_EXPORT TiglReturnCode tiglLogSetVerbosity (TiglLogLevel *level*)

Set the console verbosity level.

This function shall be used change, what kind of logging information is displayed on the console. By default, only errors and warnings are printed on console.

Fortran syntax:

tigl_log_set_verbosity(TiglLogLevel level)

Parameters:

← ***level*** Verbosity level for console messages.

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_ERROR if some error occurred

3.10.2.4 TIGL_COMMON_EXPORT TiglReturnCode tiglLogToFileDisabled ()

Disabled file logging. If a log file is currently opened by TiGL it will be closed. The log messages are printed to console. This is the default logging mechanism of TIGL.

Fortran syntax:

tigl_log_to_file_disabled()

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_ERROR if some other error occurred

3.10.2.5 TIGL_COMMON_EXPORT TiglReturnCode tiglLogToFileEnabled (const char ∗ *filePrefix*)

Sets up the tigl logging mechanism to send all log messages into a file.

Typically this function has to be called before opening any cpacs configuration. The console logging mechanism remains untouched.

Fortran syntax:

tigl_log_to_file_enabled(character∗n filePrefix)

Parameters:

← ***filePrefix*** Prefix of the filename to be created. The filename consists of the prefix, a date and time string and the ending ".log".

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NULL_POINTER if filePrefix is NULL
- TIGL_OPEN_FAILED if file can not be opened for writing
- TIGL_ERROR if some other error occurred

3.10.2.6 TIGL_COMMON_EXPORT TiglReturnCode tiglLogToFileStreamEnabled (FILE ∗ *fp*)

Sets up the tigl logging mechanism to send all log messages into an already opened file.

In contrast to tiglLogToFileEnabled, the messages are appended to an already opened file. This file might be an already opened logging file set up by another library or program. The console logging mechanism remains untouched.

Typically this function has to be called before opening any cpacs configuration.

Fortran syntax:

This function is and will be not available for Fortran due to incompatible file I/O!

Parameters:

← ***fp*** File pointer to an already opened file.

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NULL_POINTER if fp is NULL , i.e. not opened for writing.
- TIGL_ERROR if some other error occurred

3.11 Generic utility functions.

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglComponentGetHashCode (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentUID, int ∗hashCodePtr)

  Returns a unique HashCode for a geometric components. The component, for example a wing or a fuselage, could be specified via its UID. The HashCode is the same as long as the geometry of this component has not changed. The HashCode is valid through the current session only! Computes a hash value to represent this shape. This value is computed from the value of the underlying shape reference and the location. Orientation is not taken into account.

- TIGL_COMMON_EXPORT TiglReturnCode tiglConfigurationGetLength (TiglCPACSConfigurationHandle cpacsHandle, double ∗pLength)

  Returns the length of the plane.

- TIGL_COMMON_EXPORT const char ∗ tiglGetErrorString (TiglReturnCode errorCode)

  Translates an error code into a string.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetMAC (TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, double ∗mac_chord, double ∗mac_x, double ∗mac_y, double ∗mac_z)
- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSpan (TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, double ∗pSpan)

  Returns the span of a wing.

3.11.1 Detailed Description

Generic utility functions for geometric components that fits not only to wings _or_ fuselages.

3.11.2 Function Documentation

3.11.2.1 TIGL_COMMON_EXPORT TiglReturnCode tiglComponentGetHashCode (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *componentUID*, int ∗ *hashCodePtr*)

Returns a unique HashCode for a geometric components. The component, for example a wing or a fuselage, could be specified via its UID. The HashCode is the same as long as the geometry of this component has not changed. The HashCode is valid through the current session only! Computes a hash value to represent this shape. This value is computed from the value of the underlying shape reference and the location. Orientation is not taken into account.

Fortran syntax:

tigl_component_get_hash_code(integer cpacsHandle, character componentUID, integer hashCodePtr, integer returnCode)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

← ***componentUID*** The uid of the component for which the hash should be computed

→ ***hashCodePtr*** The pointer to a hash value to represent this shape

Returns:

- TIGL_SUCCESS if no error occurred
- TIGL_NOT_FOUND if no configuration was found for the given handle
- TIGL_NULL_POINTER if surfaceAreaPtr is a null pointer
- TIGL_ERROR if some other error occurred

3.11.2.2 TIGL_COMMON_EXPORT TiglReturnCode tiglConfigurationGetLength (TiglCPACSConfigurationHandle *cpacsHandle*, double ∗ *pLength*)

Returns the length of the plane.

The calculation of the airplane lenght is realized as follows:

All part of the configuration (currently all wing and fuselage segments) are put into a bounding box. The length of the plane is returned as the length of the box in x-direction.

Fortran syntax:

tigl_configuration_get_length(real length, integer return_code)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration

→ ***pLength*** Length of plane

Returns:

Error code

3.11.2.3 TIGL_COMMON_EXPORT const char∗ tiglGetErrorString (TiglReturnCode *errorCode*)

Translates an error code into a string.

Fortran syntax:

tigl_get_error_string(integer return_code, character∗n error_msg)

Parameters:

← ***errorCode*** Return value of a tigl function

Returns:

Error code as a string.

3.11.2.4 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetMAC (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *wingUID*, double ∗ *mac_chord*, double ∗ *mac_x*, double ∗ *mac_y*, double ∗ *mac_z*)

3.11.2.5 TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSpan (TiglCPACSConfigurationHandle *cpacsHandle*, const char ∗ *wingUID*, double ∗ *pSpan*)

Returns the span of a wing.

The calculation of the wing span is realized as follows:

∗ If the wing is mirrored at a symmetry plane (like the main wing), the wing body and its mirrored counterpart are computed and are put into a bounding box. The length of the box in a specific space dimension is returned as the wing span depending on the symmetry plane (y direction for x-z planes, z direction for x-y planes, x direction for y-z symmetry planes).

∗ If no symmetry plane is defined (e.g. for the fins), the largest dimension of the bounding box around the wing is returned.

Fortran syntax:

tigl_wing_get_span(real length, character∗n winguid, integer return_code)

Parameters:

← ***cpacsHandle*** Handle for the CPACS configuration
← ***wingUID*** UID of the Wing
→ ***pSpan*** Wing span

Returns:

Error code

4 File Documentation

4.1 /localdata2/lett_pa/Software/Build/TIGL-2.1.1-Source/src/tigl.h File Reference

Declaration of the TIGL C interface.

```
#include "tixi.h"
#include <stdio.h>
```

Defines

- #define TIGL_COMMON_EXPORT
- #define TIGL_COMPONENT_FUSELAGE 8
- #define TIGL_COMPONENT_FUSELSEGMENT 128
- #define TIGL_COMPONENT_LOGICAL 2
- #define TIGL_COMPONENT_PHYSICAL 1
- #define TIGL_COMPONENT_PLANE 4
- #define TIGL_COMPONENT_SEGMENT 32
- #define TIGL_COMPONENT_WING 16
- #define TIGL_COMPONENT_WINGCOMPSEGMENT 256
- #define TIGL_COMPONENT_WINGSEGMENT 64
- #define TIGL_COMPONENT_WINGSHELL 512
- #define TIGL_VERSION "2.0"

  Definition of the TIGL version number.

- #define TIGL_VERSION_MAJOR 2

Typedefs

- typedef enum TiglAlgorithmCode TiglAlgorithmCode

  Typedef for possible algorithm types used in calculations.

- typedef enum TiglBoolean TiglBoolean

  Definition of boolean type used in TIGL.

- typedef enum TiglContinuity TiglContinuity
- typedef int TiglCPACSConfigurationHandle

  Datatype for a CPACS configuration handle.

- typedef unsigned int TiglGeometricComponentType
- typedef enum TiglImportExportFormat TiglImportExportFormat

  Definition of the different file formats used for import/export used in TIGL.

- typedef enum TiglLogLevel TiglLogLevel

  Definition of logging levels.

- typedef enum TiglReturnCode TiglReturnCode

  Definition of error return type.

- typedef const char ** TiglStringList
- typedef enum TiglStructureType TiglStructureType
- typedef enum TiglSymmetryAxis TiglSymmetryAxis

  Definition of Symmetry Axis used in TIGL.

Enumerations

- enum TiglAlgorithmCode { TIGL_INTERPOLATE_LINEAR_WIRE = 0, TIGL_INTERPOLATE_BSPLINE_WIRE = 1, TIGL_APPROXIMATE_-BSPLINE_WIRE = 2 }
- enum TiglBoolean { TIGL_FALSE = 0, TIGL_TRUE = 1 }
- enum TiglContinuity { C0 = 0, C1 = 1, C2 = 2 }
- enum TiglImportExportFormat { TIGL_IMPORTEXPORT_IGES = 0, TIGL_-IMPORTEXPORT_STEP = 1, TIGL_IMPORTEXPORT_STL = 2, TIGL_-IMPORTEXPORT_VTK = 3 }
- enum TiglLogLevel {

  TILOG_SILENT = 0, TILOG_ERROR = 1, TILOG_WARNING = 2, TILOG_-INFO = 3,

  TILOG_DEBUG = 4, TILOG_DEBUG1 = 5, TILOG_DEBUG2 = 6, TILOG_-DEBUG3 = 7,

  TILOG_DEBUG4 = 8 }
- enum TiglReturnCode {

  TIGL_SUCCESS = 0, TIGL_ERROR = 1, TIGL_NULL_POINTER = 2, TIGL_NOT_FOUND = 3,

  TIGL_XML_ERROR = 4, TIGL_OPEN_FAILED = 5, TIGL_CLOSE_FAILED = 6, TIGL_INDEX_ERROR = 7,

  TIGL_STRING_TRUNCATED = 8, TIGL_WRONG_TIXI_VERSION = 9, TIGL_UID_ERROR = 10, TIGL_WRONG_CPACS_VERSION = 11,

  TIGL_UNINITIALIZED = 12, TIGL_MATH_ERROR = 13 }
- enum TiglStructureType { UPPER_SHELL = 0, LOWER_SHELL = 1, INNER_STRUCTURE = 2 }
- enum TiglSymmetryAxis { TIGL_NO_SYMMETRY = 0, TIGL_X_Y_PLANE = 1, TIGL_X_Z_PLANE = 2, TIGL_Y_Z_PLANE = 3 }

Functions

- TIGL_COMMON_EXPORT TiglReturnCode tiglCloseCPACSConfiguration (TiglCPACSConfigurationHandle cpacsHandle)

  Closes a CPACS configuration and cleans up all memory used by the configuration. After closing a configuration the associated configuration handle is no longer valid. When the CPACS configuration has been closed, the companion tixi document can also be closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglComponentGetHashCode (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentUID, int ∗hashCodePtr)

  Returns a unique HashCode for a geometric components. The component, for example a wing or a fuselage, could be specified via its UID. The HashCode is the same as long as the geometry of this component has not changed. The HashCode is valid through the current session only! Computes a hash value to represent this shape. This value is computed from the value of the underlying shape reference and the location. Orientation is not taken into account.

- TIGL_COMMON_EXPORT TiglReturnCode tiglComponentIntersectionLineCount (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentUidOne, const char ∗componentUidTwo, int ∗numWires)

  Returns the number if intersection lines of two geometric components.

- TIGL_COMMON_EXPORT TiglReturnCode tiglComponentIntersectionPoint (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentUidOne, const char ∗componentUidTwo, int lineID, double eta, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on the intersection line of two geometric components. Often there are more that one intersection line, therefore you need to specify the line.

- TIGL_COMMON_EXPORT TiglReturnCode tiglComponentIntersectionPoints (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentUidOne, const char ∗componentUidTwo, int lineID, const double ∗etaArray, int numberOfPoints, double ∗pointXArray, double ∗pointYArray, double ∗pointZArray)

  Convienience function to returns a list of points on the intersection line of two geometric components. Often there are more that one intersection line, therefore you need to specify the line.

- TIGL_COMMON_EXPORT TiglReturnCode tiglConfigurationGetLength (TiglCPACSConfigurationHandle cpacsHandle, double ∗pLength)

  Returns the length of the plane.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportFusedSTEP (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the fused/trimmed geometry of a CPACS configuration to STEP format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportFusedWingFuselageIGES (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the boolean fused geometry of a CPACS configuration to IGES format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportFuselageColladaByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗fuselageUID, const char ∗filename, double deflection)

  Exports the boolean fused geometry of a fuselage (selected by uid) meshed to Collada (∗.dae) format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportIGES (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the geometry of a CPACS configuration to IGES format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageSTL (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, const char ∗filenamePtr, double deflection)

Exports the boolean fused geometry of a fuselage meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageSTLByUID (TiglCPACSConfigurationHandle cpacsHandle, const char ∗fuselageUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a fuselage meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageVTKByIndex (const TiglCPACSConfigurationHandle cpacsHandle, const int fuselageIndex, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a fuselage (selected by index) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageVTKByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗fuselageUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a fuselage (selected by uid) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedFuselageVTKSimpleByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗fuselageUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a fuselage (selected by uid) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedGeometrySTL (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of the whole configuration meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedGeometryVTK (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of the whole configuration meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedGeometryVTKSimple (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of the whole configuration meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingSTL (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a wing meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingSTLByUID (TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, const char ∗filenamePtr, double deflection)

Exports the boolean fused geometry of a wing meshed to STL format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingVTKByIndex (const TiglCPACSConfigurationHandle cpacsHandle, const int wingIndex, const char ∗filenamePtr, const double deflection)

  Exports the boolean fused geometry of a wing (selected by id) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingVTKByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a wing (selected by UID) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportMeshedWingVTKSimpleByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, const char ∗filenamePtr, double deflection)

  Exports the boolean fused geometry of a wing (selected by UID) meshed to VTK format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportSTEP (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the geometry of a CPACS configuration to STEP format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportStructuredIGES (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the geometry of a CPACS configuration to IGES format. In this version structure, names and metadata will also be exported as much as it is possible.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportStructuredSTEP (TiglCPACSConfigurationHandle cpacsHandle, const char ∗filenamePtr)

  Exports the geometry of a CPACS configuration to STEP format. In this version structure, names and metadata will also be exported as much as it is possible.

- TIGL_COMMON_EXPORT TiglReturnCode tiglExportWingColladaByUID (const TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, const char ∗filename, double deflection)

  Exports the boolean fused geometry of a wing (selected by uid) meshed to Collada (∗.dae) format.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetCircumference (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double ∗circumferencePtr)

  Returns the circumference of a fuselage surface for a given fuselage and segment index and an eta.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndConnectedSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int ∗segmentCountPtr)

Returns the count of segments connected to the end section of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndConnectedSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int n, int ∗connectedIndexPtr)

  Returns the index (number) of the n-th segment connected to the end section of a given fuselage segment. n starts at 1.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndSectionAndElementIndex (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int ∗sectionIndexPtr, int ∗elementIndexPtr)

  Returns the section index and section element index of the end side of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetEndSectionAndElementUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, char ∗∗sectionUIDPtr, char ∗∗elementUIDPtr)

  Returns the section UID and section element UID of the end side of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetMinumumDistanceToGround (TiglCPACSConfigurationHandle cpacsHandle, char ∗fuselageUID, double axisPntX, double axisPntY, double axisPntZ, double axisDirX, double axisDirY, double axisDirZ, double angle, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns the point where the distance between the selected fuselage and the ground is at minimum. The Fuselage could be turned with a given angle at at given axis, specified by a point and a direction.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetNumPointsOnXPlane (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double xpos, int ∗numPointsPtr)

  Returns the number of points on a fuselage surface for a given fuselage and a give x-position.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetNumPointsOnYPlane (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double ypos, int ∗numPointsPtr)

  Returns the number of points on a fuselage surface for a given fuselage and a give y-position.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPoint (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double zeta, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointAngle (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double alpha, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index and an angle alpha (degree).

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointAngleTranslated (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double alpha, double y_cs, double z_cs, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index and an angle alpha (degree). 0 degree of the angle alpha is meant to be "up" in the direction of the positive z-axis like specifies in cpacs. The origin of the line that will be rotated with the angle alpha could be translated via the parameters y_cs and z_cs.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointOnXPlane (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double xpos, int pointIndex, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetPointOnYPlane (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double eta, double ypos, int pointIndex, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on a fuselage surface for a given fuselage and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetProfileName (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int sectionIndex, int elementIndex, char ∗∗profileNamePtr)

  Returns the name of a fuselage profile. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSectionUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int sectionIndex, char ∗∗uidNamePtr)

  Returns the UID of a section of a fuselage. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int ∗segmentCountPtr)

  Returns the number of segments for a fuselage in a CPACS configuration.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentSurfaceArea (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double ∗surfaceAreaPtr)

Returns the surface area of a segment of a fuselage.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, char ∗∗uidNamePtr)

  Returns the UID of a segment of a fuselage. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSegmentVolume (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, double ∗volumePtr)

  Returns the volume of a segment of a fuselage.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartConnectedSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int ∗segmentCountPtr)

  Returns the count of segments connected to the start section of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartConnectedSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int n, int ∗connectedIndexPtr)

  Returns the index (number) of the n-th segment connected to the start section of a given fuselage segment. n starts at 1.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartSectionAndElementIndex (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, int ∗sectionIndexPtr, int ∗elementIndexPtr)

  Returns the section index and section element index of the start side of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetStartSectionAndElementUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, int segmentIndex, char ∗∗sectionUIDPtr, char ∗∗elementUIDPtr)

  Returns the section UID and section element UID of the start side of a given fuselage segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSurfaceArea (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, double ∗surfaceAreaPtr)

  Returns the surface area of the fuselage. Currently, the area includes also the faces on the fuselage symmetry plane (in case of a symmetric wing). This is in particular a problem for fuselages, where only one half side is defined in CPACS. In future releases, these faces will not belong anymore to the surface area calculation.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetSymmetry (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, TiglSymmetryAxis ∗symmetryAxisPtr)

  Returns the Symmetry Enum if the fuselage has symmetry-axis.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetUID (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, char ∗∗uidNamePtr)

  Returns the UID of a fuselage. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglFuselageGetVolume (TiglCPACSConfigurationHandle cpacsHandle, int fuselageIndex, double ∗volumePtr)

  Returns the volume of the fuselage.

- TIGL_COMMON_EXPORT TiglReturnCode tiglGetCPACSTixiHandle (TiglCPACSConfigurationHandle cpacsHandle, TixiDocumentHandle ∗tixiHandlePtr)

  Returns the underlying TixiDocumentHandle for a given CPACS configuration handle.

- TIGL_COMMON_EXPORT const char ∗ tiglGetErrorString (TiglReturnCode errorCode)

  Translates an error code into a string.

- TIGL_COMMON_EXPORT TiglReturnCode tiglGetFuselageCount (TiglCPACSConfigurationHandle cpacsHandle, int ∗fuselageCountPtr)

  Returns the number of fuselages in a CPACS configuration.

- TIGL_COMMON_EXPORT char ∗ tiglGetVersion ()

  Returns the version number of this TIGL version.

- TIGL_COMMON_EXPORT TiglReturnCode tiglGetWingCount (TiglCPACSConfigurationHandle cpacsHandle, int ∗wingCountPtr)

  Returns the number of wings in a CPACS configuration.

- TIGL_COMMON_EXPORT TiglReturnCode tiglIsCPACSConfigurationHandleValid (TiglCPACSConfigurationHandle cpacsHandle, TiglBoolean ∗isValidPtr)

  Checks if a given CPACS configuration handle is a valid.

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogSetFileEnding (const char ∗ending)

  Sets the file ending for logging files. Default is "log".

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogSetTimeInFilenameEnabled (TiglBoolean enabled)

  Enables or disables appending a unique date/time identifier inside the log file name (behind the file prefix). By default, the time indentifier is enabled.

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogSetVerbosity (TiglLogLevel level)

Set the console verbosity level.

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogToFileDisabled ()

  Disabled file logging. If a log file is currently opened by TiGL it will be closed. The log messages are printed to console. This is the default logging mechanism of TIGL.

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogToFileEnabled (const char ∗filePrefix)

  Sets up the tigl logging mechanism to send all log messages into a file.

- TIGL_COMMON_EXPORT TiglReturnCode tiglLogToFileStreamEnabled (FILE ∗fp)

  Sets up the tigl logging mechanism to send all log messages into an already opened file.

- TIGL_COMMON_EXPORT TiglReturnCode tiglOpenCPACSConfiguration (TixiDocumentHandle tixiHandle, const char ∗configurationUID, TiglCPACSConfigurationHandle ∗cpacsHandlePtr)

  Opens a CPACS configuration and builds up the data and geometry structure in memory.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentFindSegment (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, double x, double y, double z, char ∗∗segmentUID, char ∗∗wingUID)

  Returns the segmentUID and wingUID for a given point on a componentSegment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetMaterialCount (TiglCPACSConfigurationHandle cpacsHandle, const char ∗compSegmentUID, TiglStructureType structureType, double eta, double xsi, int ∗materialCount)

  Returns the number of materials defined at a point on the wing component segment surface.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetMaterialThickness (TiglCPACSConfigurationHandle cpacsHandle, const char ∗compSegmentUID, TiglStructureType structureType, double eta, double xsi, int materialIndex, double ∗thickness)

  Returns one of the material thicknesses of a given point on the wing component segment surface. The number of materials on that point has to be first queried using tiglWingComponentSegmentGetMaterialCount.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetMaterialUID (TiglCPACSConfigurationHandle cpacsHandle, const char ∗compSegmentUID, TiglStructureType structureType, double eta, double xsi, int materialIndex, char ∗∗uid)

  Returns one of the material UIDs of a given point on the wing component segment surface. The number of materials on that point has to be first queried using tiglWingComponentSegmentGetMaterialCount.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetNumberOfSegments (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, int ∗nsegments)

  Returns the number of segments belonging to a component segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetPoint (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, double eta, double xsi, double ∗x, double ∗y, double ∗z)

  Returns x,y,z koordinates for a given eta and xsi on a componentSegment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetSegmentIntersection (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, const char ∗segmentUID, double csEta1, double csXsi1, double csEta2, double csXsi2, double segmentEta, double ∗segmentXsi)

  Computes the intersection of a line (defined by component segment coordinates) with an iso-eta line on a specified wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentGetSegmentUID (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, int segmentIndex, char ∗∗segmentUID)

  Returns the segment UID of a segment belonging to a component segment. The segment is specified with its index, which is in the 1...nsegments. The number of segments nsegments can be queried with tiglWingComponentSegmentGetNumberOfSegments.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingComponentSegmentPointGetSegmentEtaXsi (TiglCPACSConfigurationHandle cpacsHandle, const char ∗componentSegmentUID, double eta, double xsi, char ∗∗wingUID, char ∗∗segmentUID, double ∗segmentEta, double ∗segmentXsi)

  Returns eta, xsi, segmentUID and wingUID for a given eta and xsi on a componentSegment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetComponentSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int ∗compSegmentCountPtr)

  Returns the number of component segments for a wing in a CPACS configuration.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetComponentSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, const char ∗compSegmentUID, int ∗segmentIndexPtr)

  Returns the Index of a component segment of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetComponentSegmentUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int compSegmentIndex, char ∗∗uidNamePtr)

Returns the UID of a component segment of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetIndex (TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, int ∗wingIndexPtr)

  Returns the Index of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerConnectedSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int ∗segmentCountPtr)

  Returns the count of wing segments connected to the inner section of a given segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerConnectedSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int n, int ∗connectedIndexPtr)

  Returns the index (number) of the n-th wing segment connected to the inner section of a given segment. n starts at 1.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerSectionAndElementIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int ∗sectionIndexPtr, int ∗elementIndexPtr)

  Returns the section index and section element index of the inner side of a given wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetInnerSectionAndElementUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, char ∗∗sectionUIDPtr, char ∗∗elementUIDPtr)

  Returns the section UID and section element UID of the inner side of a given wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetLowerPoint (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double eta, double xsi, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on the lower wing surface for a a given wing and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetLowerPointAtAngle (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double eta, double xsi, double xAngle, double yAngle, double zAngle, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on the lower wing surface for a a given wing and segment index. This function is different from tiglWingGetLowerPoint, as a point on the cord surface is calculated (given eta and xsi coordinates) and the normal vector at that point is rotated around the x, y, and z axis by the given angles. Finally, the intersection point of the rotated normal and the lower wing shell is computed. The point is returned in absolute world coordinates.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetMAC (TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, double ∗mac_chord, double ∗mac_x, double ∗mac_y, double ∗mac_z)
- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterConnectedSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int ∗segmentCountPtr)

  Returns the count of wing segments connected to the outer section of a given segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterConnectedSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int n, int ∗connectedIndexPtr)

  Returns the index (number) of the n-th wing segment connected to the outer section of a given segment. n starts at 1.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterSectionAndElementIndex (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, int ∗sectionIndexPtr, int ∗elementIndexPtr)

  Returns the section index and section element index of the outer side of a given wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetOuterSectionAndElementUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, char ∗∗sectionUIDPtr, char ∗∗elementUIDPtr)

  Returns the section UID and section element UID of the outer side of a given wing segment.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetProfileName (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int sectionIndex, int elementIndex, char ∗∗profileNamePtr)

  Returns the name of a wing profile. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetReferenceArea (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, TiglSymmetryAxis symPlane, double ∗referenceAreaPtr)

  Returns the reference area of the wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSectionUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int sectionIndex, char ∗∗uidNamePtr)

  Returns the UID of a section of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentCount (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int ∗segmentCountPtr)

  Returns the number of segments for a wing in a CPACS configuration.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentEtaXsi (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, double pointX, double pointY, double pointZ, int ∗segmentIndex, double ∗eta, double ∗xsi, int ∗isOnTop)

  Inverse function to tiglWingGetLowerPoint and tiglWingGetLowerPoint. Calculates to a point (x,y,z) in global coordinates the wing segment coordinates and the wing segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentIndex (TiglCPACSConfigurationHandle cpacsHandle, const char ∗segmentUID, int ∗segmentIndexPtr, int ∗wingIndexPtr)

  Returns the Index of a segment of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentSurfaceArea (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double ∗surfaceAreaPtr)

  Returns the surface area of a segment of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, char ∗∗uidNamePtr)

  Returns the UID of a segment of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSegmentVolume (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double ∗volumePtr)

  Returns the volume of a segment of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSpan (TiglCPACSConfigurationHandle cpacsHandle, const char ∗wingUID, double ∗pSpan)

  Returns the span of a wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSurfaceArea (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, double ∗surfaceAreaPtr)

  Returns the surface area of the wing. Currently, the area includes also the faces on the wing symmetry plane (in case of a symmetric wing). In coming releases, these faces will not belong anymore to the surface area calculation.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetSymmetry (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, TiglSymmetryAxis ∗symmetryAxisPtr)

  Returns the Symmetry Enum if the wing has symmetry-axis.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetUID (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, char ∗∗uidNamePtr)

  Returns the UID of a wing. The string returned must not be deleted by the caller via free(). It will be deleted when the CPACS configuration is closed.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetUpperPoint (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double eta, double xsi, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on the upper wing surface for a a given wing and segment index.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetUpperPointAtAngle (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, int segmentIndex, double eta, double xsi, double xAngle, double yAngle, double zAngle, double ∗pointXPtr, double ∗pointYPtr, double ∗pointZPtr)

  Returns a point on the upper wing surface for a a given wing and segment index. This function is different from tiglWingGetUpperPoint, as a point on the cord surface is calculated (given eta and xsi coordinates) and the normal vector at that point is rotated around the x, y, and z axis by the given angles. Finally, the intersection point of the rotated normal and the upper wing shell is computed. The point is returned in absolute world coordinates.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetVolume (TiglCPACSConfigurationHandle cpacsHandle, int wingIndex, double ∗volumePtr)

  Returns the volume of the wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingGetWettedArea (TiglCPACSConfigurationHandle cpacsHandle, char ∗wingUID, double ∗wettedAreaPtr)

  Returns the wetted area of the wing.

- TIGL_COMMON_EXPORT TiglReturnCode tiglWingSegmentPointGetComponentSegmentEtaXsi (TiglCPACSConfigurationHandle cpacsHandle, const char ∗segmentUID, const char ∗componentSegmentUID, double segmentEta, double segmentXsi, double ∗eta, double ∗xsi)

  Returns eta, xsi coordinates of a componentSegment given segmentEta and segmentXsi on a wing segment.

4.1.1 Detailed Description

Declaration of the TIGL C interface.

4.1.2 Define Documentation

4.1.2.1 #define TIGL_COMMON_EXPORT

4.1.2.2 #define TIGL_COMPONENT_FUSELAGE 8

The Component is a fuselage

4.1.2.3 #define TIGL_COMPONENT_FUSELSEGMENT 128

4.1.2.4 #define TIGL_COMPONENT_LOGICAL 2

4.1.2.5 #define TIGL_COMPONENT_PHYSICAL 1

4.1.2.6 #define TIGL_COMPONENT_PLANE 4

4.1.2.7 #define TIGL_COMPONENT_SEGMENT 32

4.1.2.8 #define TIGL_COMPONENT_WING 16

The Component is a wing

4.1.2.9 #define TIGL_COMPONENT_WINGCOMPSEGMENT 256

4.1.2.10 #define TIGL_COMPONENT_WINGSEGMENT 64

4.1.2.11 #define TIGL_COMPONENT_WINGSHELL 512

4.1.2.12 #define TIGL_VERSION "2.0"

Definition of the TIGL version number.

4.1.2.13 #define TIGL_VERSION_MAJOR 2

4.1.3 Typedef Documentation

4.1.3.1 typedef enum TiglAlgorithmCode TiglAlgorithmCode

Typedef for possible algorithm types used in calculations.

Possible values for variables of type TiglAlgorithmCode are:

- TIGL_INTERPOLATE_LINEAR_WIRE: Use a linear interpolation between the points of a wire.
- TIGL_INTERPOLATE_BSPLINE_WIRE: Use a BSpline interpolation between the points of a wire.
- TIGL_APPROXIMATE_BSPLINE_WIRE: Use a BSpline approximation for the points of a wire.

4.1.3.2 typedef enum TiglBoolean TiglBoolean

Definition of boolean type used in TIGL.

Possible values are:

- TIGL_FALSE
- TIGL_TRUE

4.1.3.3 typedef enum TiglContinuity TiglContinuity

4.1.3.4 typedef int TiglCPACSConfigurationHandle

Datatype for a CPACS configuration handle.

4.1.3.5 typedef enum TiglImportExportFormat TiglImportExportFormat

Definition of the different file formats used for import/export used in TIGL.

Possible values are:

- TIGL_IMPORTEXPORT_IGES: Use IGES format for geometry import/export.
- TIGL_IMPORTEXPORT_STEP: Use STEP format for geometry import/export.
- TIGL_IMPORTEXPORT_STL: Use STL format for geometry import/export.
- TIGL_IMPORTEXPORT_VTK: Use VTK format for geometry import/export.

4.1.3.6 typedef enum TiglLogLevel TiglLogLevel

Definition of logging levels.

Possible values are:

- TILOG_SILENT
- TILOG_ERROR
- TILOG_WARNING
- TILOG_INFO
- TILOG_DEBUG
- TILOG_DEBUG1
- TILOG_DEBUG2
- TILOG_DEBUG3
- TILOG_DEBUG4

4.1.3.7 typedef enum TiglReturnCode TiglReturnCode

Definition of error return type.

Possible values are:

- TIGL_SUCCESS
- TIGL_ERROR
- TIGL_NULL_POINTER
- TIGL_NOT_FOUND
- TIGL_XML_ERROR
- TIGL_OPEN_FAILED
- TIGL_CLOSE_FAILED
- TIGL_INDEX_ERROR
- TIGL_STRING_TRUNCATED
- TIGL_WRONG_TIXI_VERSION
- TIGL_UID_ERROR

4.1.3.8 typedef const char∗∗ TiglStringList

4.1.3.9 typedef enum TiglStructureType TiglStructureType

4.1.3.10 typedef enum TiglSymmetryAxis TiglSymmetryAxis

Definition of Symmetry Axis used in TIGL.

Possible values are:

- TIGL_NO_SYMMETRY
- TIGL_X_Y_PLANE
- TIGL_X_Z_PLANE
- TIGL_Y_Z_PLANE

4.1.4 Enumeration Type Documentation

4.1.4.1 enum TiglContinuity

Enumerator:

C0
C1
C2

4.1.4.2 enum TiglStructureType

Enumerator:

- *UPPER_SHELL*
- *LOWER_SHELL*
- *INNER_STRUCTURE*