Eriksson2006Solver Package

Erzeugt von Doxygen 1.8.2

Mit Sep 19 2012 09:29:26

Inhaltsverzeichnis

Kapitel 1

Dokumentation eines FEM- Solvers

Noch zu erledigen Formuliere mainpage aus

1.1 Introduction

This is the introduction.

1.2 Installation

1.2.1 Step 1: Opening the box

etc... /

Kapitel 2

Ausstehende Aufgaben

Unterprogramm bauteilclass::berechnebauteilstaggered (this)

ausbauen

Unterprogramm bauteilclass::berechnerelativefeuchte (this)

Formeln für relative Feuchtigkeit einbetten und Lösungen im Postprozessor als Variable verfügbar machen

page Dokumentation eines FEM- Solvers

Formuliere mainpage aus

Typ ElementClass

Geometrie und Materialwerte nur beim ersten Aufruf pro Zeitschritt setzen

Typ elementclass::bauteilelement_t

Aufbau des structs überdenken

- DGL

Unterprogramm elementclass::berechneelementstaggered1 (this)

allow static Simulationsablauf

allow static Simulationsablauf

Unterprogramm elementclass::berechneelementstaggered2 (this)

allow static Simulationsablauf

allow static Simulationsablauf

Unterprogramm elementclass::initialisieredgl (this)

Werte für randbedingungen einlesen

diese Abfrage optimieren

Unterprogramm elementclass::inputtensor (Tensor, IsScalar, Name, Material, n, NodeIndexes)

allow static Simulationsablauf

allow static Simulationsablauf

Unterprogramm eriksson2006solve (Model, Solver, dt, Transient)

Deactivierung oder spätere Aktivierung funktioniert bisher nicht! Alle Elemente müssen während der gesammten Simulationsdauer aktiv sein. Ursache sind Beschränkungen in ElementClass, in der SUBROUTINE initialisiereBauteilElement, wo geprüft wird ob aktuell der erste Zeitschritt durchlaufen wird.

Typ geometrischeeigenschaften_class

mache geometrischeEigenschaften_Class eigenständig durch eigene Funktionen und als linked List unabhängig von ElementClass, Eriksson2006DGL_Class, etc

Gruppe LokaleGekoppelteMatrizenClass

überlege, welche Anordnung der Elementen nach außen und innen sinnvoll ist, denn laut!! Elmer Forum ist für direkte Solver eine Blockanordnung der Submatrizen ungeeignet, Iterative Solver kommen damit wohl klar

Typ materialsammlung_class

mache MaterialSammlung_Class eigenständig durch eigene Funktionen und als linked List unabhängig von ElementClass, Eriksson2006DGL_Class, etc

Typ NeumannRandbedingungClass

Derived Type ausbauen

- Proceduren hinzufügen

Kapitel 3

Modul-Verzeichnis

3.1 Module

Hier folgt die Aufzählung aller Module:

Kapitel 4

Modul-Index

4.1 Modulliste

Hier folgt eine Liste aller Module mit ihren Kurzbeschreibungen:

Kapitel 5

Datentyp-Index

5.1 Datentyp-Liste

Hier folgen die Datentypen mit Kurzbeschreibungen:

Kapitel 6

Datei-Verzeichnis

6.1 Auflistung der Dateien

Hier folgt die Aufzählung aller Dateien mit einer Kurzbeschreibung:

Kapitel 7

Modul-Dokumentation

7.1 ElementClass

Datentypen

- module elementclass
- type elementclass::bauteilelement_t

  sammelt Elementeigenschaften
- interface elementclass::neu
- interface elementclass::berechne
- interface elementclass::holematerial
- module ElementClass

  sammelt und verarbeitet elementbasierte Informationen

Funktionen/Unterroutinen

- subroutine elementclass::neu::initialisierebauteilelement (this, ElemNr, Solver, Model, istGebietselement)
- subroutine elementclass::berechne::berechneelement (this)

  Löse die definierte DGL für dieses Element.
- subroutine elementclass::holematerial::holematerialvektor (this, materialWert, elmername)

  Gibt mithilfe von Elmers Routine den Wert elmername zurück und macht zusätzliche eine Fehlerprüfung.
- subroutine elementclass::holematerial::holematerialtensor (this, materialWert, elmername)
- subroutine elementclass::setzeelementzaehler (anzahlElemente)
- subroutine elementclass::initialisieregeometrie (this)

  initialisiert die Werte für die Geometrie für die Berechnung der Eriksson2006DGL_Class
- subroutine elementclass::initialisieredgl (this)

  Übergibt die für dieses Element spezifischen Material- und Geometrieeigenschaften an die DGL, sodass diese ihre Gleichungen für dieses Element optimieren kann.
- subroutine elementclass::holematerialtensor (this, materialWert, elmername)
- subroutine elementclass::setzepointeraufnaechsteselement (this, PointerAufNaechstesElement)

  Setze den Pointer auf das Nachbarelement.
- recursive subroutine elementclass::iteriereueberalleelemente (this)

  Löse die DGL für dieses Element und folge dem POINTER auf das naechste Element.
- recursive subroutine elementclass::iteriereueberalleelementestaggered1 (this)

  Löse die DGL für dieses Element und folge dem POINTER auf das naechste Element.
- recursive subroutine elementclass::iteriereueberalleelementestaggered2 (this)

  Löse die DGL für dieses Element und folge dem POINTER auf das naechste Element.
- subroutine elementclass::berechneelementstaggered1 (this)

Löse die definierte DGL für dieses Element.

- subroutine elementclass::berechneelementstaggered2 (this)

  Löse die definierte DGL für dieses Element.
- subroutine elementclass::inputtensor (Tensor, IsScalar, Name, Material, n, NodeIndexes)

  This routine is taken from Elmers Stress.f90 and modified.

Variablen

- integer elementclass::bauteilelement_t::elementnummer
- logical elementclass::bauteilelement_t::istgebietselement
- integer elementclass::bauteilelement_t::elementdof
- type(lokalegekoppeltematrizen_t) elementclass::bauteilelement_t::elementmatrizen
- type(bauteilelement_t), pointer elementclass::bauteilelement_t::naechsteselement
- type(geometrischeeigenschaften_t) elementclass::bauteilelement_t::geometrie
- type(materialsammlung_t) elementclass::bauteilelement_t::materialwerte
- type(randbedingung_t) elementclass::bauteilelement_t::temperaturrandbedingung
- type(randbedingung_t) elementclass::bauteilelement_t::feuchtigkeitrandbedingung
- type(eriksson2006dgl_t) elementclass::bauteilelement_t::erikssondgl
- type(element_t), pointer elementclass::bauteilelement_t::element

  Elmers Elementinfo.
- type(valuelist_t), pointer elementclass::bauteilelement_t::material >NULL()

  Elmers Materialinfo.
- type(solver_t) elementclass::bauteilelement_t::solver
- type(model_t) elementclass::bauteilelement_t::model
- real(kind=dp), dimension(:,:),
  allocatable elementclass::bauteilelement_t::vorherigelsg

7.1.1 Ausführliche Beschreibung

7.1.2 Funktionen/Unterroutinen-Dokumentation

7.1.2.1 subroutine elementclass::berechne::berechneelement (type(bauteilelement_t), intent(inout) *this*)

Löse die definierte DGL für dieses Element.

7.1.2.2 subroutine elementclass::berechneelementstaggered1 (type(bauteilelement_t), intent(inout) *this*)

Löse die definierte DGL für dieses Element.

Noch zu erledigen allow static Simulationsablauf

Noch zu erledigen allow static Simulationsablauf

7.1.2.3 subroutine elementclass::berechneelementstaggered2 (type(bauteilelement_t), intent(inout) *this*)

Löse die definierte DGL für dieses Element.

Noch zu erledigen allow static Simulationsablauf

Noch zu erledigen allow static Simulationsablauf

7.1.2.4 subroutine elementclass::holematerialtensor (type(**bauteilelement_t**), intent(inout) *this,* real(kind=dp), dimension(:,:,:), intent(inout), pointer *materialWert,* character(∗), intent(in) *elmername*)

7.1.2.5 subroutine elementclass::holematerial::holematerialtensor (type(**bauteilelement_t**), intent(inout) *this,* real(kind=dp), dimension(:,:,:), intent(inout), pointer *materialWert,* character(∗), intent(in) *elmername*)

7.1.2.6 subroutine elementclass::holematerial::holematerialvektor (type(**bauteilelement_t**), intent(inout) *this,* real(kind=dp), dimension(:), pointer *materialWert,* character(∗), intent(in) *elmername*)

Gibt mithilfe von Elmers Routine den Wert elmername zurück und macht zusätzliche eine Fehlerprüfung.

Noch zu erledigen Methodenaufruf überdenken

7.1.2.7 subroutine elementclass::neu::initialisierebauteilelement (type(**bauteilelement_t**) *this,* integer *ElemNr,* type(solver_t) *Solver,* type(model_t) *Model,* logical *istGebietselement*)

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.1.2.8 subroutine elementclass::initialisieredgl (type(**bauteilelement_t**), intent(inout) *this*)

Übergibt die für dieses Element spezifischen Material- und Geometrieeigenschaften an die DGL, sodass diese ihre Gleichungen für dieses Element optimieren kann.

Noch zu erledigen Werte für randbedingungen einlesen

Noch zu erledigen diese Abfrage optimieren

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.1.2.9 subroutine elementclass::initialisieregeometrie (type(**bauteilelement_t**), intent(inout) *this*)

initialisiert die Werte für die Geometrie für die Berechnung der Eriksson2006DGL_Class

Dabei sind folgende Werte für die Berechnung der Matrizen notwendig:

- detJ
- ds
- Number of Integrationpoints
- Formfunktionen N
- abgeleitete Formfunktionen B
- anzahlKnoten

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.1.2.10 subroutine elementclass::inputtensor (real(kind=dp), dimension(:,:,:) *Tensor,* logical *IsScalar,* character(len=∗) *Name,* type(valuelist_t), pointer *Material,* integer *n,* integer, dimension(:) *NodeIndexes*)

This routine is taken from Elmers Stress.f90 and modified.

Noch zu erledigen allow static Simulationsablauf

Noch zu erledigen allow static Simulationsablauf

7.1.2.11 recursive subroutine elementclass::iteriereueberalleelemente (type(bauteilelement_t) *this*)

Löse die DGL für dieses Element und folge dem POINTER auf das naechste Element.

Da jedes Element die Möglichkeit hat sein Nachbarelement zu kennen kann eine Linked List erzeugt werden, die es ermöglicht über alle Elemente zu iterieren. Der Pointer wird durch die SUBROUTINE setzePointerAufNaechstes-Element gesetzt. Wird diese SUBROUTINE aufgerufen werden zuerst die Systemmatrizen für dieses Element berechnet und dann wird zum nächsten Element gegangen. Dort werden die Matrizen berechnet, usw., bis das Letzte Element keinen Pointer mehr auf ein weiteres Element mehr hat.

Siehe auch

setzePointerAufNaechstesElement

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.1.2.12 recursive subroutine elementclass::iteriereueberalleelementestaggered1 (type(bauteilelement_t) *this*)

Löse die DGL für dieses Element und folge dem POINTER auf das naechste Element.

Da jedes Element die Möglichkeit hat sein Nachbarelement zu kennen kann eine Linked List erzeugt werden, die es ermöglicht über alle Elemente zu iterieren. Der Pointer wird durch die SUBROUTINE setzePointerAufNaechstes-Element gesetzt. Wird diese SUBROUTINE aufgerufen werden zuerst die Systemmatrizen für dieses Element berechnet und dann wird zum nächsten Element gegangen. Dort werden die Matrizen berechnet, usw., bis das Letzte Element keinen Pointer mehr auf ein weiteres Element mehr hat.

Siehe auch

setzePointerAufNaechstesElement

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.1.2.13 recursive subroutine elementclass::iteriereueberalleelementestaggered2 (type(bauteilelement_t) *this*)

Löse die DGL für dieses Element und folge dem POINTER auf das naechste Element.

Da jedes Element die Möglichkeit hat sein Nachbarelement zu kennen kann eine Linked List erzeugt werden, die es ermöglicht über alle Elemente zu iterieren. Der Pointer wird durch die SUBROUTINE setzePointerAufNaechstes-Element gesetzt. Wird diese SUBROUTINE aufgerufen werden zuerst die Systemmatrizen für dieses Element berechnet und dann wird zum nächsten Element gegangen. Dort werden die Matrizen berechnet, usw., bis das Letzte Element keinen Pointer mehr auf ein weiteres Element mehr hat.

Siehe auch

setzePointerAufNaechstesElement

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.1.2.14 subroutine elementclass::setzeelementzaehler (integer *anzahlElemente*)

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.1.2.15 subroutine elementclass::setzepointeraufnaechsteselement (type(bauteilelement_t) *this*, type(bauteilelement_t), pointer *PointerAufNaechstesElement*)

Setze den Pointer auf das Nachbarelement.

Normalerweise ist der Pointer auf das nächste Element auf NULL gesetzt und muss erst gesetzt werden, um die Möglichkeit zu geben durch alle Elementen als eine Linked List zu iterieren.

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.1.3 Variablen-Dokumentation

7.1.3.1 type(element_t), pointer elementclass::bauteilelement_t::element

Elmers Elementinfo.

7.1.3.2 integer elementclass::bauteilelement_t::elementdof

7.1.3.3 type(lokalegekoppeltematrizen_t) elementclass::bauteilelement_t::elementmatrizen

7.1.3.4 integer elementclass::bauteilelement_t::elementnummer

7.1.3.5 type(eriksson2006dgl_t) elementclass::bauteilelement_t::erikssondgl

7.1.3.6 type(randbedingung_t) elementclass::bauteilelement_t::feuchtigkeitrandbedingung

7.1.3.7 type(geometrischeeigenschaften_t) elementclass::bauteilelement_t::geometrie

7.1.3.8 logical elementclass::bauteilelement_t::istgebietselement

7.1.3.9 type(valuelist_t), pointer elementclass::bauteilelement_t::material >NULL()

Elmers Materialinfo.

7.1.3.10 type(materialsammlung_t) elementclass::bauteilelement_t::materialwerte

7.1.3.11 type(model_t) elementclass::bauteilelement_t::model

7.1.3.12 type(bauteilelement_t), pointer elementclass::bauteilelement_t::naechsteselement

7.1.3.13 type(solver_t) elementclass::bauteilelement_t::solver

7.1.3.14 type(randbedingung_t) elementclass::bauteilelement_t::temperaturrandbedingung

7.1.3.15 real(kind=dp), dimension(:,:), allocatable elementclass::bauteilelement_t::vorherigelsg

7.2 Eriksson2006Solve

Eriksson2006Solve.f90 beinhaltet die SUBROUTINE Eriksson2006Solve welche durch den Elmer Kern in jeder Steady state Iteration aufgerufen wird.

Funktionen/Unterroutinen

- subroutine eriksson2006solve_init (Model, Solver, dt, Transient)

  Initialisiere einige Standardparameter für die Simulation.

7.2.1 Ausführliche Beschreibung

Eriksson2006Solve.f90 beinhaltet die SUBROUTINE Eriksson2006Solve welche durch den Elmer Kern in jeder Steady state Iteration aufgerufen wird.

7.2.2 Funktionen/Unterroutinen-Dokumentation

7.2.2.1 subroutine eriksson2006solve_init (type(model_t) *Model,* type(solver_t) *Solver,* real(kind=dp) *dt,* logical *Transient*)

Initialisiere einige Standardparameter für die Simulation.

Wird automatisch beim Aufruf der SUBROUTINE Eriksson2006Solve aufgerufen.

Parameter

Model	bereitgestellt von Elmer
Solver	bereitgestellt von Elmer
dt	Schrittweite, bereitgestellt von Elmer
Transient	bereitgestellt von Elmer

7.3 LokaleMatrizenClass

Verwaltung Elementmatrizen der FEM.

Datentypen

- module lokalematrizenclass
- type lokalematrizenclass::lokalematrizen_t
- interface lokalematrizenclass::operator(+)
- interface lokalematrizenclass::neu
- interface lokalematrizenclass::setze
- interface lokalematrizenclass::nullifiziere
- interface lokalematrizenclass::mass
- interface lokalematrizenclass::force
- interface lokalematrizenclass::stiff
- interface lokalematrizenclass::schuettlematrizen

Funktionen/Unterroutinen

- TYPE(LokaleMatrizen_t) function lokalematrizenclass::operator(+)::addiere (this, LM2)

  addiert zwei LokaleMatrizen indem die einzelenen Arrays (FORCE, STIFF, MASS) einzeln und elementweise addiert werden
- subroutine lokalematrizenclass::neu::setzematrizen (this, aFORCE, aSTIFF, aMASS)

  setzeMatrizen setzt die Systemmatrizen
- subroutine lokalematrizenclass::neu::initmatrizen (this, maxElementMatrixSize)

  initMatrizen allokiert Speicherplatz für Systemmatrizen
- subroutine lokalematrizenclass::neu::initmatrizenkurz (this, dof)
- subroutine lokalematrizenclass::setze::setzeaufwert (this, wertFORCE, wertSTIFF, wertMASS)

  setzeAufWert setzt die gesammten Matrizen auf Werte
- subroutine lokalematrizenclass::nullifiziere::nullifizierematrizen (this)
- real(kind=dp) function,
  dimension(:,:), pointer lokalematrizenclass::mass::gibmass (this)

  gibMASS gibt die Massematrix zurück
- real(kind=dp) function,
  dimension(:), pointer lokalematrizenclass::force::gibforce (this)

  gibFORCE gibt den Kraftvektor zurück
- real(kind=dp) function,
  dimension(:,:), pointer lokalematrizenclass::stiff::gibstiff (this)

  gibSTIFF gibt die Steifigkeitsmatrix zurück
- subroutine lokalematrizenclass::schuettlematrizen::schuettleallematrizen (this)
- subroutine lokalematrizenclass::initmatrizenkurz (this, dof)

7.3.1 Ausführliche Beschreibung

Verwaltung Elementmatrizen der FEM. Diese Elementmatrizen werden in einem Type gesammelt und geschlossen behandelt. Dazu stellt diese KLasse diverse Funktionen bereit um Matrizen zu manipulieren.

7.3.2 Funktionen/Unterroutinen-Dokumentation

7.3.2.1 TYPE(LokaleMatrizen_t) function lokalematrizenclass::operator(+)::addiere (type(lokalematrizen_t), intent(in) *this,* type(lokalematrizen_t), intent(in) *LM2*)

addiert zwei LokaleMatrizen indem die einzelenen Arrays (FORCE, STIFF, MASS) einzeln und elementweise addiert werden

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.3.2.2 real(kind=dp) function, dimension(:), pointer lokalematrizenclass::force::gibforce (type(lokalematrizen_t), intent(in) *this*)

gibFORCE gibt den Kraftvektor zurück

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.3.2.3 real(kind=dp) function, dimension(:,:), pointer lokalematrizenclass::mass::gibmass (type(lokalematrizen_t), intent(in) *this*)

gibMASS gibt die Massematrix zurück

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.3.2.4 real(kind=dp) function, dimension(:,:), pointer lokalematrizenclass::stiff::gibstiff (type(lokalematrizen_t), intent(in) *this*)

gibSTIFF gibt die Steifigkeitsmatrix zurück

Parameter

in	*this*	LokaleMatrizen_t:: this Returns REAL(KIND=dp), DIMENSION(SHAPE(STIFF)

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.3.2.5 subroutine lokalematrizenclass::neu::initmatrizen (type(lokalematrizen_t), intent(inout) *this,* integer, dimension(2), intent(in) *maxElementMatrixSize*)

initMatrizen allokiert Speicherplatz für Systemmatrizen

Um Types mit dynamische gro0en Elementen+ zu erzeugen ist es nötig POINTER zu verwenden, allokierbare Arrays innerhalb von Type Definitionen nicht erlaubt sind. Die POINTER bekommenim folgenden ähnlich allokierbarer Arrays einen Speicherbereich ihrer Größe entsprechend zugewiesen.

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.3.2.6 subroutine lokalematrizenclass::initmatrizenkurz (type(lokalematrizen_t), intent(inout) *this,* integer, intent(in) *dof*)

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.3.2.7 subroutine lokalematrizenclass::neu::initmatrizenkurz (type(lokalematrizen_t), intent(inout) *this,* integer, intent(in) *dof*)

7.3.2.8 subroutine lokalematrizenclass::nullifiziere::nullifizierematrizen (type(lokalematrizen_t), intent(inout) *this*)

7.3.2.9 subroutine lokalematrizenclass::schuettlematrizen::schuettleallematrizen (type(lokalematrizen_t), intent(inout) *this*)

7.3.2.10 subroutine lokalematrizenclass::setze::setzeaufwert (type(lokalematrizen_t), intent(inout) *this,* real(kind=dp), dimension(:), optional *wertFORCE,* real(kind=dp), dimension(:,:), optional *wertSTIFF,* real(kind=dp), dimension(:,:), optional *wertMASS*)

setzeAufWert setzt die gesammten Matrizen auf Werte

FORCE, STIFF und MASS werden geschlossen auf Werte gesetzt. Das ist z.B. nützlich, um die Matrizen schnell auf 0 zu setzen

Parameter

this	Sammlung der Matrizen
wertFORCE	0.0d0, falls nicht vorhanden
wertforce	alter Wert falls nicht vorhanden
wertstiff	alter Wert falls nicht vorhanden
wertmass	alter Wert falls nicht vorhanden

7.3.2.11 subroutine lokalematrizenclass::neu::setzematrizen (type(lokalematrizen_t), intent(inout) *this,* real(kind=dp), dimension(:), intent(in) *aFORCE,* real(kind=dp), dimension(:,:), intent(in) *aSTIFF,* real(kind=dp), dimension(:,:), intent(in) *aMASS*)

setzeMatrizen setzt die Systemmatrizen

und prüft die Matrizengrößen der Eingangsmatrizen , ruft initMatrizen auf und ordnet die Matrizen in den Lokale-Matrizen_t ein

Noch zu erledigen mache diese Funktion unabhängig von Eingangsmatrizen übergebe stattdessen die Anzahl der Freiheitsgrade pro Element

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.4 MaterialClass

Datentypen

- module neumannrandbedingungclass
- type neumannrandbedingungclass::neumannrandbedingung_t
- type CauchyRandbedingungClass::CauchyRandbedingung_t

  Beinhaltet Werte der Cauchy Randbedingung.
- module cauchyrandbedingungclass
- type cauchyrandbedingungclass::cauchyrandbedingung_t
- module dirichletrandbedingungclass
- type dirichletrandbedingungclass::dirichletrandbedingung_t
- module volumenrandbedingungclass
- type volumenrandbedingungclass::volumenrandbedingung_t
- module randbedingungclass
- interface randbedingungclass::neu
- type randbedingungclass::randbedingung_t
- module MaterialClass

  Struktur für einen Materialparameter.
- module MaterialSammlungClass

  Sammelt die Materialeigenschaften in einer Baumstruktur.
- module NeumannRandbedingungClass

  "RandbedingungClass.f90"
- type DirichletRandbedingung::DirichletRandbedingung_t

  Beinhaltet den Wert der DirichletRandbedingungen.

Funktionen/Unterroutinen

- LOGICAL function, public randbedingungclass::istvolumen (this)
- LOGICAL function, public randbedingungclass::istneumann (this)
- LOGICAL function, public randbedingungclass::istcauchy (this)
- LOGICAL function, public randbedingungclass::istdirichlet (this)
- real(kind=dp) function,
  dimension(:), pointer, public randbedingungclass::gibrandbedingungdirichlet (this)
- real(kind=dp) function,
  dimension(:), pointer, public randbedingungclass::gibrandbedingungneumann (this)
- real(kind=dp) function,
  dimension(:), pointer, public randbedingungclass::gibrandbedingungcauchy (this, i)
- real(kind=dp) function,
  dimension(:), pointer, public randbedingungclass::gibrandbedingungvolumen (this)

Variablen

- real(kind=dp), dimension(:),
  pointer neumannrandbedingungclass::neumannrandbedingung_t::wert
- real(kind=dp), dimension(:),
  pointer cauchyrandbedingungclass::cauchyrandbedingung_t::bezugswert >NULL()
- real(kind=dp), dimension(:),
  pointer cauchyrandbedingungclass::cauchyrandbedingung_t::skalierfaktor >NULL()
- real(kind=dp), dimension(:),
  pointer dirichletrandbedingungclass::dirichletrandbedingung_t::wert
- real(kind=dp), dimension(:),
  pointer volumenrandbedingungclass::volumenrandbedingung_t::wert

- type(volumenrandbedingung_t),
  pointer randbedingungclass::randbedingung_t::volumenrandbedingung >NULL()
- type(neumannrandbedingung_t),
  pointer randbedingungclass::randbedingung_t::neumannrandbedingung >NULL()
- type(cauchyrandbedingung_t),
  pointer randbedingungclass::randbedingung_t::cauchyrandbedingung >NULL()
- type(dirichletrandbedingung_t),
  pointer randbedingungclass::randbedingung_t::dirichletrandbedingung >NULL()

7.4.1 Ausführliche Beschreibung

7.4.2 Funktionen/Unterroutinen-Dokumentation

7.4.2.1 real(kind=dp) function, dimension(:), pointer, public randbedingungclass::gibrandbedingungcauchy (type(**randbedingung_t**), intent(in) *this,* integer, intent(in) *i*)

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.4.2.2 real(kind=dp) function, dimension(:), pointer, public randbedingungclass::gibrandbedingungdirichlet (type(**randbedingung_t**), intent(in) *this*)

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.4.2.3 real(kind=dp) function, dimension(:), pointer, public randbedingungclass::gibrandbedingungneumann (type(**randbedingung_t**), intent(in) *this*)

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.4.2.4 real(kind=dp) function, dimension(:), pointer, public randbedingungclass::gibrandbedingungvolumen (type(**randbedingung_t**), intent(in) *this*)

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.4.2.5 LOGICAL function, public randbedingungclass::istcauchy (type(**randbedingung_t**), intent(in) *this*)

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.4.2.6 LOGICAL function, public randbedingungclass::istdirichlet (type(**randbedingung_t**), intent(in) *this*)

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.4.2.7 LOGICAL function, public randbedingungclass::istneumann (type(**randbedingung_t**), intent(in) *this*)

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.4.2.8 LOGICAL function, public randbedingungclass::istvolumen (type(**randbedingung_t**), intent(in) *this*)

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

7.4.3 Variablen-Dokumentation

7.4.3.1 real(kind=dp), dimension(:), pointer cauchyrandbedingungclass::cauchyrandbedingung_t::bezugswert >NULL()

7.4.3.2 type(cauchyrandbedingung_t), pointer randbedingungclass::randbedingung_t::cauchyrandbedingung >NULL()

7.4.3.3 type(dirichletrandbedingung_t), pointer randbedingungclass::randbedingung_t::dirichletrandbedingung >NULL()

7.4.3.4 type(neumannrandbedingung_t), pointer randbedingungclass::randbedingung_t::neumannrandbedingung >NULL()

7.4.3.5 real(kind=dp), dimension(:), pointer cauchyrandbedingungclass::cauchyrandbedingung_t::skalierfaktor >NULL()

7.4.3.6 type(volumenrandbedingung_t), pointer randbedingungclass::randbedingung_t::volumenrandbedingung >NULL()

7.4.3.7 real(kind=dp), dimension(:), pointer neumannrandbedingungclass::neumannrandbedingung_t::wert

7.4.3.8 real(kind=dp), dimension(:), pointer volumenrandbedingungclass::volumenrandbedingung_t::wert

7.4.3.9 real(kind=dp), dimension(:), pointer dirichletrandbedingungclass::dirichletrandbedingung_t::wert

7.5 LokaleGekoppelteMatrizenClass

Verwaltung gekoppelte Elementmatrizen der FEM.

Datentypen

- module lokalegekoppeltematrizenclass
- type lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t
- interface lokalegekoppeltematrizenclass::operator(+)

  alles privat, Kommunikation zwischen Modulen erfolgt über Interfaces
- interface lokalegekoppeltematrizenclass::neu
- interface lokalegekoppeltematrizenclass::setze
- interface lokalegekoppeltematrizenclass::nullifiziere
- interface lokalegekoppeltematrizenclass::mass
- interface lokalegekoppeltematrizenclass::force
- interface lokalegekoppeltematrizenclass::stiff
- interface lokalegekoppeltematrizenclass::schuettlematrizen

Funktionen/Unterroutinen

- subroutine lokalegekoppeltematrizenclass::initialisierematrizen (this, dof)
- subroutine lokalegekoppeltematrizenclass::schuettleallematrizen (this)

Variablen

- real(kind=dp), dimension(:,:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::kaa
- real(kind=dp), dimension(:,:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::kab
- real(kind=dp), dimension(:,:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::kba
- real(kind=dp), dimension(:,:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::kbb
- real(kind=dp), dimension(:,:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::caa
- real(kind=dp), dimension(:,:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::cab
- real(kind=dp), dimension(:,:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::cba
- real(kind=dp), dimension(:,:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::cbb
- real(kind=dp), dimension(:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::fa
- real(kind=dp), dimension(:),
  pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::fb

- TYPE(LokaleGekoppelteMatrizen_t)
  function lokalegekoppeltematrizenclass::operator(+)::addiere (this, LM2)

  addiert zwei LokaleMatrizen indem die einzelenen Arrays (FORCE, STIFF, MASS) einzeln und elementweise addiert werden
- subroutine lokalegekoppeltematrizenclass::neu::setzematrizen (this, aFORCE, aSTIFF, aMASS)

  setzeMatrizen setzt die Systemmatrizen
- subroutine lokalegekoppeltematrizenclass::neu::initialisierematrizen (this, dof)

- subroutine lokalegekoppeltematrizenclass::setze::setzeaufwert (this, wertFORCE, wertSTIFF, wertMASS)

  setzeAufWert setzt die gesammten Matrizen auf Werte
- subroutine lokalegekoppeltematrizenclass::nullifiziere::nullifizierematrizen (this)
- real(kind=dp) function, dimension(:,:), pointer lokalegekoppeltematrizenclass::mass::gibmass (this)

  gibMASS gibt die Massematrix zurück
- real(kind=dp) function, dimension(:), pointer lokalegekoppeltematrizenclass::force::gibforce (this)

  gibFORCE gibt den Kraftvektor zurück
- real(kind=dp) function, dimension(:,:), pointer lokalegekoppeltematrizenclass::stiff::gibstiff (this)

  gibSTIFF gibt die Steifigkeitsmatrix zurück
- subroutine lokalegekoppeltematrizenclass::schuettlematrizen::schuettleallematrizen (this)

7.5.1 Ausführliche Beschreibung

Verwaltung gekoppelte Elementmatrizen der FEM. Diese Elementmatrizen werden in einem Type gesammelt und geschlossen behandelt. Diese Klasse erlaubt die Modifikation der Submatrizen.

Noch zu erledigen

- überlege, welche Anordnung der Elementen nach außen und innen sinnvoll ist, denn laut!! Elmer Forum ist für direkte Solver eine Blockanordnung der Submatrizen ungeeignet, Iterative Solver kommen damit wohl klar

7.5.2 Funktionen/Unterroutinen-Dokumentation

7.5.2.1 TYPE(LokaleGekoppelteMatrizen_t) function lokalegekoppeltematrizenclass::operator(+)::addiere (type(lokalegekoppeltematrizen_t), intent(in) *this,* type(lokalegekoppeltematrizen_t), intent(in) *LM2*)

addiert zwei LokaleMatrizen indem die einzelenen Arrays (FORCE, STIFF, MASS) einzeln und elementweise addiert werden

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.5.2.2 real(kind=dp) function, dimension(:), pointer lokalegekoppeltematrizenclass::force::gibforce (type(lokalegekoppeltematrizen_t), intent(in) *this*)

gibFORCE gibt den Kraftvektor zurück

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.5.2.3 real(kind=dp) function, dimension(:,:), pointer lokalegekoppeltematrizenclass::mass::gibmass (type(lokalegekoppeltematrizen_t), intent(in) *this*)

gibMASS gibt die Massematrix zurück

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.5.2.4 real(kind=dp) function, dimension(:,:), pointer lokalegekoppeltematrizenclass::stiff::gibstiff (type(lokalegekoppeltematrizen_t), intent(in) *this*)

gibSTIFF gibt die Steifigkeitsmatrix zurück

Parameter

this	

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.5.2.5 subroutine lokalegekoppeltematrizenclass::initialisierematrizen (type(**lokalegekoppeltematrizen_t**), intent(inout) *this,* integer, intent(in) *dof*)

7.5.2.6 subroutine lokalegekoppeltematrizenclass::neu::initialisierematrizen (type(**lokalegekoppeltematrizen_t**), intent(inout) *this,* integer, intent(in) *dof*)

7.5.2.7 subroutine lokalegekoppeltematrizenclass::nullifiziere::nullifizierematrizen (type(**lokalegekoppeltematrizen_t**), intent(inout) *this*)

7.5.2.8 subroutine lokalegekoppeltematrizenclass::schuettleallematrizen (type(**lokalegekoppeltematrizen_t**), intent(inout) *this*)

7.5.2.9 subroutine lokalegekoppeltematrizenclass::schuettlematrizen::schuettleallematrizen (type(**lokalegekoppeltematrizen_t**), intent(inout) *this*)

7.5.2.10 subroutine lokalegekoppeltematrizenclass::setze::setzeaufwert (type(**lokalegekoppeltematrizen_t**), intent(inout) *this,* real(kind=dp), dimension(:), optional *wertFORCE,* real(kind=dp), dimension(:,:), optional *wertSTIFF,* real(kind=dp), dimension(:,:), optional *wertMASS*)

setzeAufWert setzt die gesammten Matrizen auf Werte

FORCE, STIFF und MASS werden geschlossen auf Werte gesetzt. Das ist z.B. nützlich, um die Matrizen schnell auf 0 zu setzen

Parameter

this	Sammlung der Matrizen
wertFORCE	0.0d0, falls nicht vorhanden
wertstiff	alter Wert falls nicht vorhanden
wertmass	alter Wert falls nicht vorhanden

7.5.2.11 subroutine lokalegekoppeltematrizenclass::neu::setzematrizen (type(**lokalegekoppeltematrizen_t**), intent(inout) *this,* real(kind=dp), dimension(:), intent(in) *aFORCE,* real(kind=dp), dimension(:,:), intent(in) *aSTIFF,* real(kind=dp), dimension(:,:), intent(in) *aMASS*)

setzeMatrizen setzt die Systemmatrizen

und prüft die Matrizengrößen der Eingangsmatrizen , ruft initMatrizen auf und ordnet die Matrizen in den Lokale-Matrizen_t ein

Noch zu erledigen füge weitere Routine für skalaren Input ein

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.5.3 Variablen-Dokumentation

7.5.3.1 real(kind=dp), dimension(:,:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::caa

7.5.3.2 real(kind=dp), dimension(:,:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::cab

7.5.3.3 real(kind=dp), dimension(:,:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::cba

7.5.3.4 real(kind=dp), dimension(:,:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::cbb

7.5.3.5 real(kind=dp), dimension(:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::fa

7.5.3.6 real(kind=dp), dimension(:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::fb

7.5.3.7 real(kind=dp), dimension(:,:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::kaa

7.5.3.8 real(kind=dp), dimension(:,:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::kab

7.5.3.9 real(kind=dp), dimension(:,:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::kba

7.5.3.10 real(kind=dp), dimension(:,:), pointer lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t::kbb

7.6 BauteilClass

BauteilClass speichert die vorhergehende Lösung und verwaltet den Simulationsablauf.

Datentypen

- module bauteilclass
- type bauteilclass::bauteil_t
- interface bauteilclass::neu
- interface bauteilclass::berechne

Funktionen/Unterroutinen

- subroutine bauteilclass::neu::initialisierebauteil (this, Solver, Model)

  Füllen der Werte des Bauteil_t.
- subroutine bauteilclass::berechne::berechnebauteil (this)

  intialisiert und steuert die Eriksson2006Solve berechnung für dieses Mesh
- subroutine bauteilclass::berechnebauteilstaggered (this)

  intialisiert und steuert die Eriksson2006Solve berechnung für dieses Mesh
- subroutine bauteilclass::berechnerelativefeuchte (this)

  berechne relative Feuchtigkeit aus dem Feuchtegehalt und der Temperatur

Variablen

- real(kind=dp), pointer bauteilclass::bauteil_t::vorhergehendeloesung
- real(kind=dp) bauteilclass::bauteil_t::nonlineartol 0.0d0
- integer bauteilclass::bauteil_t::anzahlgebietselemente 0
- integer bauteilclass::bauteil_t::anzahlrandgebietselemente 0
- integer bauteilclass::bauteil_t::problemdim 1
- integer bauteilclass::bauteil_t::nonlineariter 0
- type(elementepointer),
  dimension(:), allocatable bauteilclass::bauteil_t::elementepointerliste
- type(solver_t) bauteilclass::bauteil_t::solver

7.6.1 Ausführliche Beschreibung

BauteilClass speichert die vorhergehende Lösung und verwaltet den Simulationsablauf. Da die Simulation nach einem sogenannten Staggerd Iteration Scheme ablaufen soll muss jeweils der vorhergehende Lösungsschritt gespeichert werden. Weiterhin verwaltet BauteilClass seine Elemente vom Typ ElementeClass

7.6.2 Funktionen/Unterroutinen-Dokumentation

7.6.2.1 subroutine bauteilclass::berechne::berechnebauteil (type(bauteil_t), intent(inout) *this*)

intialisiert und steuert die Eriksson2006Solve berechnung für dieses Mesh

Noch zu erledigen ausbauen

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.6.2.2 subroutine bauteilclass::berechnebauteilstaggered (type(bauteil_t), intent(inout) *this*)

intialisiert und steuert die Eriksson2006Solve berechnung für dieses Mesh

Noch zu erledigen ausbauen

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.6.2.3 subroutine bauteilclass::berechnerelativefeuchte (type(bauteil_t), intent(inout) *this*)

berechne relative Feuchtigkeit aus dem Feuchtegehalt und der Temperatur

Die DGL gib den Feuchtegehalt als Lösung dieser muss für Quellung noch in die relative Feuchtigkeit umgerechnet werden. Das geschieht mithilfe von sogenannten Sorptionsisothermen.

Noch zu erledigen
- Formeln für relative Feuchtigkeit einbetten und Lösungen im Postprozessor als Variable verfügbar machen

7.6.2.4 subroutine bauteilclass::neu::initialisierebauteil (type(bauteil_t), intent(inout) *this,* type(solver_t), intent(in) *Solver,* type(model_t), intent(in) *Model*)

Füllen der Werte des Bauteil_t.

Initialisiert ein Bauteil für die Simulation. Dazu werden die für die global wichtigen Parameter aus dem Solver_t und Model_t gezogen, welche Elmer bereitstellt. Weiterhin werden die Elemente (Gebiets- und Randgebietselemente) initialisiert

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

7.6.3 Variablen-Dokumentation

7.6.3.1 integer bauteilclass::bauteil_t::anzahlgebietselemente 0

7.6.3.2 integer bauteilclass::bauteil_t::anzahlrandgebietselemente 0

7.6.3.3 type(elementepointer), dimension(:), allocatable bauteilclass::bauteil_t::elementepointerliste

7.6.3.4 integer bauteilclass::bauteil_t::nonlineariter 0

7.6.3.5 real(kind=dp) bauteilclass::bauteil_t::nonlineartol 0.0d0

7.6.3.6 integer bauteilclass::bauteil_t::problemdim 1

7.6.3.7 type(solver_t) bauteilclass::bauteil_t::solver

7.6.3.8 real(kind=dp), pointer bauteilclass::bauteil_t::vorhergehendeloesung

7.7 Eriksson2006DGL_Class

Numerische Implementierung des gekoppelten Feldproblems nach Eriksson2006DGL.

Datentypen

- module eriksson2006dgl_class
- type eriksson2006dgl_class::eriksson2006dgl_t

Variablen

- type(lokalegekoppeltematrizen_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::lokalegekoppeltematrizen
- type(materialsammlung_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::materialwerte > NULL()
- type(geometrischeeigenschaften_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::geometrie > NULL()
- type(randbedingung_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::temperaturrandbedingung >NULL()
- type(randbedingung_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::feuchtigkeitrandbedingung >NULL()

7.7.1 Ausführliche Beschreibung

Numerische Implementierung des gekoppelten Feldproblems nach Eriksson2006DGL. Berechnug der Elementmatrizen nach der Galerkin Methode für das gekoppelte Temperatur-Feuchtigkeitsfeldproblem nach Eriksson2006. Das Feldproblem wird nach einem Staggerd Iteration Schema gelöst, wobei die beiden Felder getrennt gelöst werden

7.7.2 Variablen-Dokumentation

7.7.2.1 type(randbedingung_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::feuchtigkeitrandbedingung >NULL()

7.7.2.2 type(geometrischeeigenschaften_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::geometrie > NULL()

7.7.2.3 type(lokalegekoppeltematrizen_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::lokalegekoppeltematrizen

7.7.2.4 type(materialsammlung_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::materialwerte > NULL()

7.7.2.5 type(randbedingung_t), pointer eriksson2006dgl_class::eriksson2006dgl_t::temperaturrandbedingung >NULL()

Kapitel 8

Modul-Dokumentation

8.1 NeumannRandbedingung-Modul-Referenz

Datentypen

- type NeumannRandbedingung_t

  Beinhaltet den Wert der Neuman-Randbedingungen.

8.2 VolumenRandbedingung-Modul-Referenz

Datentypen

- type VolumenRandbedingung_t

  Beinhaltete Werte der Volumenquellen.

Kapitel 9

Datentyp-Dokumentation

9.1 bauteilclass::bauteil_t-Typ-Referenz

Zusammengehörigkeiten von bauteilclass::bauteil_t:

Öffentliche Attribute

- real(kind=dp), pointer vorhergehendeloesung
- real(kind=dp) nonlineartol 0.0d0
- integer anzahlgebietselemente 0
- integer anzahlrandgebietselemente 0
- integer problemdim 1
- integer nonlineariter 0
- type(elementepointer),
  dimension(:), allocatable elementepointerliste
- type(solver_t) solver

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- BauteilClass.f90

9.2 bauteilclass-Modul-Referenz

Datentypen

- type bauteil_t
- interface berechne
- interface neu

Öffentliche Methoden

- subroutine berechnebauteilstaggered (this)

  intialisiert und steuert die Eriksson2006Solve berechnung für dieses Mesh
- subroutine berechnerelativefeuchte (this)

  berechne relative Feuchtigkeit aus dem Feuchtegehalt und der Temperatur

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- BauteilClass.f90

9.3 elementclass::bauteilelement_t-Typ-Referenz

sammelt Elementeigenschaften

Zusammengehörigkeiten von elementclass::bauteilelement_t:

Öffentliche Attribute

- integer elementnummer
- logical istgebietselement
- integer elementdof
- type(lokalegekoppeltematrizen_t) elementmatrizen
- type(bauteilelement_t), pointer naechsteselement
- type(geometrischeeigenschaften_t) geometrie
- type(materialsammlung_t) materialwerte
- type(randbedingung_t) temperaturrandbedingung
- type(randbedingung_t) feuchtigkeitrandbedingung
- type(eriksson2006dgl_t) erikssondgl
- type(element_t), pointer element

  Elmers Elementinfo.
- type(valuelist_t), pointer material >NULL()

  Elmers Materialinfo.
- type(solver_t) solver
- type(model_t) model
- real(kind=dp), dimension(:,:), allocatable vorherigelsg

9.3.1 Ausführliche Beschreibung

sammelt Elementeigenschaften

Parameter

elementNumber	vielleicht überflüssig
ist-Gebietselement	seperiert in Gebietselemente und Randgebietselemente

Noch zu erledigen
- Aufbau des structs überdenken
- DGL

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- ElementClass.f90

9.4 bauteilclass::berechne-Interface-Referenz

Öffentliche Methoden

- subroutine berechnebauteil (this)

  intialisiert und steuert die Eriksson2006Solve berechnung für dieses Mesh

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- BauteilClass.f90

9.5 elementclass::berechne-Interface-Referenz

Öffentliche Methoden

- subroutine berechneelement (this)

  Löse die definierte DGL für dieses Element.

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- ElementClass.f90

9.6 CauchyRandbedingungClass::CauchyRandbedingung_t-Typ-Referenz

Beinhaltet Werte der Cauchy Randbedingung.

9.6.1 Ausführliche Beschreibung

Beinhaltet Werte der Cauchy Randbedingung.

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.7 cauchyrandbedingungclass::cauchyrandbedingung_t-Typ-Referenz

Öffentliche Attribute

- real(kind=dp), dimension(:),
  pointer bezugswert >NULL()
- real(kind=dp), dimension(:),
  pointer skalierfaktor >NULL()

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.8 cauchyrandbedingungclass-Modul-Referenz

Datentypen

- type cauchyrandbedingung_t

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.9 dirichletrandbedingungclass::dirichletrandbedingung_t-Typ-Referenz

Öffentliche Attribute

- real(kind=dp), dimension(:),
  pointer wert

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.10 DirichletRandbedingung::DirichletRandbedingung_t-Typ-Referenz

Beinhaltet den Wert der DirichletRandbedingungen.

9.10.1 Ausführliche Beschreibung

Beinhaltet den Wert der DirichletRandbedingungen.

Parameter

Wert	speichert den Wert der DirichletRandbedingung für ein Element

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.11 dirichletrandbedingungclass-Modul-Referenz

Datentypen

- type dirichletrandbedingung_t

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.12 ElementClass-Modul-Referenz

sammelt und verarbeitet elementbasierte Informationen

```
#include "ElementClass.f90"
```

9.12.1 Ausführliche Beschreibung

sammelt und verarbeitet elementbasierte Informationen

Elmers Simulationsablauf ist elementbasiert. Das heißt für jedes Element werden die Systemmatrizen berechnet und durch Elmers Routinen in die Gesammtmatrizen eingepflegt. Diese Klasse sammelt und verarbeitet die für die Komposition der Lokalen Matrizen notwendigen Informationen, wie

- Randbedingung

- Elementmatrizen
- Algorithmus zum berechnen der Elementmatrizen

Noch zu erledigen

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- ElementClass.f90

9.13 elementclass-Modul-Referenz

Datentypen

- type bauteilelement_t

  sammelt Elementeigenschaften
- interface berechne
- interface holematerial
- interface neu

Öffentliche Methoden

- subroutine setzeelementzaehler (anzahlElemente)
- subroutine initialisieregeometrie (this)

  initialisiert die Werte für die Geometrie für die Berechnung der Eriksson2006DGL_Class
- subroutine initialisieredgl (this)

  Übergibt die für dieses Element spezifischen Material- und Geometrieeigenschaften an die DGL, sodass diese ihre Gleichungen für dieses Element optimieren kann.
- subroutine holematerialtensor (this, materialWert, elmername)
- subroutine setzepointeraufnaechsteselement (this, PointerAufNaechstesElement)

  Setze den Pointer auf das Nachbarelement.
- recursive subroutine iteriereueberalleelemente (this)

  Löse die DGL für dieses Element und folge dem POINTER auf das naechste Element.
- recursive subroutine iteriereueberalleelementestaggered1 (this)

  Löse die DGL für dieses Element und folge dem POINTER auf das naechste Element.
- recursive subroutine iteriereueberalleelementestaggered2 (this)

  Löse die DGL für dieses Element und folge dem POINTER auf das naechste Element.
- subroutine berechneelementstaggered1 (this)

  Löse die definierte DGL für dieses Element.
- subroutine berechneelementstaggered2 (this)

  Löse die definierte DGL für dieses Element.
- subroutine inputtensor (Tensor, IsScalar, Name, Material, n, NodeIndexes)

  This routine is taken from Elmers Stress.f90 and modified.

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- ElementClass.f90

9.14 eriksson2006dgl_class-Modul-Referenz

Datentypen

- type eriksson2006dgl_t

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- Eriksson2006DGL_Class.f90

9.15 eriksson2006dgl_class::eriksson2006dgl_t-Typ-Referenz

Öffentliche Attribute

- type(lokalegekoppeltematrizen_t), pointer lokalegekoppeltematrizen
- type(materialsammlung_t), pointer materialwerte > NULL()
- type(geometrischeeigenschaften_t), pointer geometrie > NULL()
- type(randbedingung_t), pointer temperaturrandbedingung >NULL()
- type(randbedingung_t), pointer feuchtigkeitrandbedingung >NULL()

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- Eriksson2006DGL_Class.f90

9.16 lokalematrizenclass::force-Interface-Referenz

Öffentliche Methoden

- real(kind=dp) function, dimension(:), pointer gibforce (this)

  gibFORCE gibt den Kraftvektor zurück

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.17 lokalegekoppeltematrizenclass::force-Interface-Referenz

Öffentliche Methoden

- real(kind=dp) function, dimension(:), pointer gibforce (this)

  gibFORCE gibt den Kraftvektor zurück

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.18 geometrischeeigenschaften_class-Modul-Referenz

struct dieses Moduls enthält die Werte für die geometrie eines Elements

Datentypen

- type geometrischeeigenschaften_t

  geometrischeEigenschaften_t Sammlung der für die Simulation nötigen geometrischen Elementeigenschaften

9.18.1 Ausführliche Beschreibung

struct dieses Moduls enthält die Werte für die geometrie eines Elements

Alle hier gelisteten Werte werden momentan in ElementClass gesetzt und durch Eriksson2006DGL_Class in der Berechnung verwendet

Noch zu erledigen mache geometrischeEigenschaften_Class eigenständig durch eigene Funktionen und als linked List unabhängig von ElementClass, Eriksson2006DGL_Class, etc

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- Materialeigenschaften.f90

9.19 geometrischeeigenschaften_class::geometrischeeigenschaften_t-Typ-Referenz

geometrischeEigenschaften_t Sammlung der für die Simulation nötigen geometrischen Elementeigenschaften

Öffentliche Attribute

- real(kind=dp) detj
- real(kind=dp), dimension(:,:), allocatable n

  Achtung beim setzen der DIMENSION für N und B.
- real(kind=dp), dimension(:,:), allocatable b
- real(kind=dp), dimension(:,:,:), allocatable dbdx
- type(gaussintegrationpoints_t) ip
- integer anzahlknoten
- integer knotenfreiwerte
- type(nodes_t) nodes
- integer dim
- type(element_t), pointer element

9.19.1 Ausführliche Beschreibung

geometrischeEigenschaften_t Sammlung der für die Simulation nötigen geometrischen Elementeigenschaften

9.19.2 Dokumentation der Datenelemente

9.19.2.1 integer geometrischeeigenschaften_class::geometrischeeigenschaften_t::anzahlknoten

9.19.2.2 real(kind=dp), dimension(:,:), allocatable geometrischeeigenschaften_class::geometrischeeigenschaften_t::b

9.19.2.3 real(kind=dp), dimension(:,:,:), allocatable geometrischeeigenschaften_class::geometrischeeigenschaften_t::dbdx

9.19.2.4 real(kind=dp) geometrischeeigenschaften_class::geometrischeeigenschaften_t::detj

9.19.2.5 integer geometrischeeigenschaften_class::geometrischeeigenschaften_t::dim

9.19.2.6 type(element_t), pointer geometrischeeigenschaften_class::geometrischeeigenschaften_t::element

9.19.2.7 type(gaussintegrationpoints_t) geometrischeeigenschaften_class::geometrischeeigenschaften_t::ip

9.19.2.8 integer geometrischeeigenschaften_class::geometrischeeigenschaften_t::knotenfreiwerte

9.19.2.9 real(kind=dp), dimension(:,:), allocatable geometrischeeigenschaften_class::geometrischeeigenschaften_t::n

Achtung beim setzen der DIMENSION für N und B.

9.19.2.10 type(nodes_t) geometrischeeigenschaften_class::geometrischeeigenschaften_t::nodes

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- Materialeigenschaften.f90

9.20 elementclass::holematerial-Interface-Referenz

Öffentliche Methoden

- subroutine holematerialvektor (this, materialWert, elmername)

  Gibt mithilfe von Elmers Routine den Wert elmername zurück und macht zusätzliche eine Fehlerprüfung.
- subroutine holematerialtensor (this, materialWert, elmername)

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- ElementClass.f90

9.21 mystdmodules::localsystemmatrices_t-Typ-Referenz

Öffentliche Attribute

- real(kind=dp), dimension(:,:), allocatable mass
- real(kind=dp), dimension(:,:), allocatable stiff
- real(kind=dp), dimension(:), allocatable force

9.21.1 Dokumentation der Datenelemente

9.21.1.1 real(kind=dp), dimension(:), allocatable mystdmodules::localsystemmatrices_t::force

9.21.1.2 real(kind=dp), dimension(:,:), allocatable mystdmodules::localsystemmatrices_t::mass

9.21.1.3 real(kind=dp), dimension(:,:), allocatable mystdmodules::localsystemmatrices_t::stiff

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- MyStdModules.f90

9.22 LokaleGekoppelteMatrizen_t-Typ-Referenz

Type Struktur zur Sammlung der Systemmatrizen und deren Submatrizen.

9.22.1 Ausführliche Beschreibung

Type Struktur zur Sammlung der Systemmatrizen und deren Submatrizen.

vereint die MASS, STIFF and FORCE Arrays, sowie Pointer auf die Submatrizen Kaa, Kab, Kba, Kbb, Caa, Cab, Cba, Cbb, fa und fb in einem TYPE um Parameterlisten übersichtlicher zu gestalten und Functionen gleichzeitig auf alle Arrays

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.23 lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t-Typ-Referenz

Öffentliche Attribute

- real(kind=dp), dimension(:,:), pointer kaa
- real(kind=dp), dimension(:,:), pointer kab
- real(kind=dp), dimension(:,:), pointer kba
- real(kind=dp), dimension(:,:), pointer kbb
- real(kind=dp), dimension(:,:), pointer caa
- real(kind=dp), dimension(:,:), pointer cab
- real(kind=dp), dimension(:,:), pointer cba
- real(kind=dp), dimension(:,:), pointer cbb
- real(kind=dp), dimension(:), pointer fa
- real(kind=dp), dimension(:), pointer fb

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.24 lokalegekoppeltematrizenclass-Modul-Referenz

Datentypen

- interface force
- type lokalegekoppeltematrizen_t
- interface mass
- interface neu
- interface nullifiziere
- interface operator(+)

  alles privat, Kommunikation zwischen Modulen erfolgt über Interfaces
- interface schuettlematrizen
- interface setze
- interface stiff

Öffentliche Methoden

- subroutine initialisierematrizen (this, dof)
- subroutine schuettleallematrizen (this)

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.25 lokalematrizenclass::lokalematrizen_t-Typ-Referenz

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.26 LokaleMatrizen_t-Typ-Referenz

Type Struktur zur Sammlung der Systemmatrizen.

9.26.1 Ausführliche Beschreibung

Type Struktur zur Sammlung der Systemmatrizen.

vereint die MASS, STIFF and FORCE Arrays in einem TYPE um Parameterlisten übersichtlicher zu gestalten und Functionen gleichzeitig auf alle Arrays anzuwenden. Es werden allokierbare POINTER als Behelflösung verwendet, da allokierbare Arrays innerhalb von TYPE Definitionen nicht erlaubt sind. Die Matrizen mit Shake∗ geben den für elmer umsortierten Matrizenelementanordnungen wieder.

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.27 lokalematrizenclass-Modul-Referenz

Datentypen

- interface force
- type lokalematrizen_t
- interface mass
- interface neu
- interface nullifiziere
- interface operator(+)
- interface schuettlematrizen
- interface setze
- interface stiff

Öffentliche Methoden

- subroutine initmatrizenkurz (this, dof)

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.28 lokalematrizenclass::mass-Interface-Referenz

Öffentliche Methoden

- real(kind=dp) function,
  dimension(:,:), pointer gibmass (this)

  gibMASS gibt die Massematrix zurück

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.29 lokalegekoppeltematrizenclass::mass-Interface-Referenz

Öffentliche Methoden

- real(kind=dp) function,
  dimension(:,:), pointer gibmass (this)

  gibMASS gibt die Massematrix zurück

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.30 MaterialClass-Modul-Referenz

Struktur für einen Materialparameter.

9.30.1 Ausführliche Beschreibung

Struktur für einen Materialparameter.

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.31 materialsammlung_class-Modul-Referenz

struct enhält die für diese Simulation nötigen materialWerte

Datentypen

- type materialsammlung_t

9.31.1 Ausführliche Beschreibung

struct enhält die für diese Simulation nötigen materialWerte

Alle hier gelisteten Materialwerte werden momentan in ElementClass gesetzt und durch Eriksson2006DGL_Class in der Berechnung verwendet

Noch zu erledigen mache MaterialSammlung_Class eigenständig durch eigene Funktionen und als linked List unabhängig von ElementClass, Eriksson2006DGL_Class, etc

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- Materialeigenschaften.f90

9.32 materialsammlung_class::materialsammlung_t-Typ-Referenz

Öffentliche Attribute

- real(kind=dp), dimension(:), pointer dichte >NULL()
- real(kind=dp), dimension(:), pointer trockendichte >NULL()
- real(kind=dp), dimension(:), pointer diffusionskoeffizient >NULL()
- real(kind=dp), dimension(:), pointer konduktivitaet >NULL()
- real(kind=dp), dimension(:), pointer spezifischewaerme >NULL()
- real(kind=dp), dimension(:), pointer e_b >NULL()

  Activation energy of bound water.
- real(kind=dp), dimension(:), pointer r >NULL()

  gas konstant
- real(kind=dp), dimension(:), pointer relativehumidity >NULL()

- real(kind=dp), dimension(:),
  pointer dwdh >NULL()
- real(kind=dp), dimension(:,:,:),
  pointer dichtetensor >NULL()
- real(kind=dp), dimension(:,:,:),
  pointer trockendichtetensor >NULL()
- real(kind=dp), dimension(:,:,:),
  pointer diffusionskoeffizienttensor >NULL()
- real(kind=dp), dimension(:,:,:),
  pointer konduktivitaettensor >NULL()
- real(kind=dp), dimension(:,:,:),
  pointer spezifischewaermetensor >NULL()

9.32.1 Dokumentation der Datenelemente

9.32.1.1 real(kind=dp), dimension(:), pointer materialsammlung_class::materialsammlung_t::dichte >NULL()

9.32.1.2 real(kind=dp), dimension(:,:,:), pointer materialsammlung_class::materialsammlung_t::dichtetensor >NULL()

9.32.1.3 real(kind=dp), dimension(:), pointer materialsammlung_class::materialsammlung_t::diffusionskoeffizient >NULL()

9.32.1.4 real(kind=dp), dimension(:,:,:), pointer materialsammlung_class::materialsammlung_t::diffusionskoeffizienttensor >NULL()

9.32.1.5 real(kind=dp), dimension(:), pointer materialsammlung_class::materialsammlung_t::dwdh >NULL()

9.32.1.6 real(kind=dp), dimension(:), pointer materialsammlung_class::materialsammlung_t::e_b >NULL()

Activation energy of bound water.

9.32.1.7 real(kind=dp), dimension(:), pointer materialsammlung_class::materialsammlung_t::konduktivitaet >NULL()

9.32.1.8 real(kind=dp), dimension(:,:,:), pointer materialsammlung_class::materialsammlung_t::konduktivitaettensor >NULL()

9.32.1.9 real(kind=dp), dimension(:), pointer materialsammlung_class::materialsammlung_t::r >NULL()

gas konstant

9.32.1.10 real(kind=dp), dimension(:), pointer materialsammlung_class::materialsammlung_t::relativehumidity >NULL()

9.32.1.11 real(kind=dp), dimension(:), pointer materialsammlung_class::materialsammlung_t::spezifischewaerme >NULL()

9.32.1.12 real(kind=dp), dimension(:,:,:), pointer materialsammlung_class::materialsammlung_t::spezifischewaermetensor >NULL()

9.32.1.13 real(kind=dp), dimension(:), pointer materialsammlung_class::materialsammlung_t::trockendichte >NULL()

9.32.1.14 real(kind=dp), dimension(:,:,:), pointer materialsammlung_class::materialsammlung_t::trockendichtetensor >NULL()

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- Materialeigenschaften.f90

9.33 MaterialSammlungClass-Modul-Referenz

Sammelt die Materialeigenschaften in einer Baumstruktur.

9.33.1 Ausführliche Beschreibung

Sammelt die Materialeigenschaften in einer Baumstruktur.

materialSammlung_t ist die Wurzel einer Baumstruktur, welche die Materialwerte für skalare, isotrope und anisotrope Parameter in linked lists ablegt. Ein beliebiger Wert kann aus den Linked lists geholt oder weitere hinzugefügt werden. Durch die Struktur muss die Anzahl der Parameter nicht vor der Programmausführung bekannt sein

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.34 mystdmodules-Modul-Referenz

Datentypen

- type localsystemmatrices_t
- interface printarray
- type stdsolverparams_t

Öffentliche Methoden

- type(stdsolverparams_t) function returnstdsolverparams (Solver)
- subroutine printstdinfo (str, iter, NonlinearIter)
- subroutine print2darray (array, str)
- subroutine print1darray (array, str)
- subroutine rotateelasticitymatrix (C, T, dim)
- subroutine rotateelasticitymatrix2d (C, T)
- subroutine rotateelasticitymatrix3d (C, T)
- subroutine rotate2indextensor (C, T, dim)
- subroutine rotate4indextensor (C, T, dim)

9.34.1 Elementfunktionen/Unterroutinen-Dokumentation

9.34.1.1 subroutine mystdmodules::print1darray (real(kind=dp), dimension(:) *array,* character(len=∗) *str*)

9.34.1.2 subroutine mystdmodules::print2darray (real(kind=dp), dimension(:,:) *array,* character(len=∗) *str*)

9.34.1.3 subroutine mystdmodules::printstdinfo (character(len=∗) *str,* integer *iter,* integer *NonlinearIter*)

9.34.1.4 type(stdsolverparams_t) function mystdmodules::returnstdsolverparams (type(solver_t) *Solver*)

9.34.1.5 subroutine mystdmodules::rotate2indextensor (real(kind=dp), dimension(:,:) *C,* real(kind=dp), dimension(:,:) *T,* integer *dim*)

9.34.1.6 subroutine mystdmodules::rotate4indextensor (real(kind=dp), dimension(:,:,:,:) *C,* real(kind=dp), dimension(:,:) *T,* integer *dim*)

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

9.34.1.7 subroutine mystdmodules::rotateelasticitymatrix (real(kind=dp), dimension(:,:) *C,* real(kind=dp), dimension(:,:) *T,* integer *dim*)

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

9.34.1.8 subroutine mystdmodules::rotateelasticitymatrix2d (real(kind=dp), dimension(:,:) *C,* real(kind=dp), dimension(:,:) *T*)

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

9.34.1.9 subroutine mystdmodules::rotateelasticitymatrix3d (real(kind=dp), dimension(:,:) *C,* real(kind=dp), dimension(:,:) *T*)

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- MyStdModules.f90

9.35 elementclass::neu-Interface-Referenz

Öffentliche Methoden

- subroutine initialisierebauteilelement (this, ElemNr, Solver, Model, istGebietselement)

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- ElementClass.f90

9.36 lokalematrizenclass::neu-Interface-Referenz

Öffentliche Methoden

- subroutine setzematrizen (this, aFORCE, aSTIFF, aMASS)

  setzeMatrizen setzt die Systemmatrizen
- subroutine initmatrizen (this, maxElementMatrixSize)

  initMatrizen allokiert Speicherplatz für Systemmatrizen
- subroutine initmatrizenkurz (this, dof)

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.37 randbedingungclass::neu-Interface-Referenz

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.38 lokalegekoppeltematrizenclass::neu-Interface-Referenz

Öffentliche Methoden

- subroutine setzematrizen (this, aFORCE, aSTIFF, aMASS)

  setzeMatrizen setzt die Systemmatrizen
- subroutine initialisierematrizen (this, dof)

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.39 bauteilclass::neu-Interface-Referenz

Öffentliche Methoden

- subroutine initialisierebauteil (this, Solver, Model)

  Füllen der Werte des Bauteil_t.

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- BauteilClass.f90

9.40 NeumannRandbedingung::NeumannRandbedingung_t-Typ-Referenz

Beinhaltet den Wert der Neuman-Randbedingungen.

9.40.1 Ausführliche Beschreibung

Beinhaltet den Wert der Neuman-Randbedingungen.

Parameter

Wert	speichert den Wert der Neuman-Randbedingung für ein Element

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.41 neumannrandbedingungclass::neumannrandbedingung_t-Typ-Referenz

Öffentliche Attribute

- real(kind=dp), dimension(:),
  pointer wert

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.42 neumannrandbedingungclass-Modul-Referenz

Datentypen

- type neumannrandbedingung_t

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.43 NeumannRandbedingungClass-Modul-Referenz

"RandbedingungClass.f90"

9.43.1 Ausführliche Beschreibung

"RandbedingungClass.f90"

Einbinden von Neuman Randbedingungen

Die Klasse wird dynamisch eingebunden, falls für das Element Neuman-Randbedingungen definiert sind

Noch zu erledigen
- Derived Type ausbauen
- Proceduren hinzufügen

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.44 lokalematrizenclass::nullifiziere-Interface-Referenz

Öffentliche Methoden

- subroutine nullifizierematrizen (this)

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.45 lokalegekoppeltematrizenclass::nullifiziere-Interface-Referenz

Öffentliche Methoden

- subroutine nullifizierematrizen (this)

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.46 lokalematrizenclass::operator(+)-Interface-Referenz

Öffentliche Methoden

- TYPE(LokaleMatrizen_t) function addiere (this, LM2)

  addiert zwei LokaleMatrizen indem die einzelenen Arrays (FORCE, STIFF, MASS) einzeln und elementweise addiert werden

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.47 lokalegekoppeltematrizenclass::operator(+)-Interface-Referenz

alles privat, Kommunikation zwischen Modulen erfolgt über Interfaces

Öffentliche Methoden

- TYPE(LokaleGekoppelteMatrizen_t)
  function addiere (this, LM2)

  addiert zwei LokaleMatrizen indem die einzelenen Arrays (FORCE, STIFF, MASS) einzeln und elementweise addiert werden

9.47.1 Ausführliche Beschreibung

alles privat, Kommunikation zwischen Modulen erfolgt über Interfaces

Interfaces als Schnittstelle nach Ausen ermöglichen das sicher Überladen von Funktionen und Subroutinen

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.48 mystdmodules::printarray-Interface-Referenz

Öffentliche Methoden

- subroutine print1darray (array, str)
- subroutine print2darray (array, str)

9.48.1 Elementfunktionen/Unterroutinen-Dokumentation

9.48.1.1 subroutine mystdmodules::printarray::print1darray (real(kind=dp), dimension(:) *array,* character(len=∗) *str*)

9.48.1.2 subroutine mystdmodules::printarray::print2darray (real(kind=dp), dimension(:,:) *array,* character(len=∗) *str*)

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- MyStdModules.f90

9.49 randbedingungclass::randbedingung_t-Typ-Referenz

Öffentliche Attribute

- type(volumenrandbedingung_t),
  pointer volumenrandbedingung >NULL()
- type(neumannrandbedingung_t),
  pointer neumannrandbedingung >NULL()
- type(cauchyrandbedingung_t),
  pointer cauchyrandbedingung >NULL()
- type(dirichletrandbedingung_t),
  pointer dirichletrandbedingung >NULL()

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.50 randbedingungclass-Modul-Referenz

Datentypen

- interface neu
- type randbedingung_t

Öffentliche Methoden

- LOGICAL function, public istvolumen (this)
- LOGICAL function, public istneumann (this)
- LOGICAL function, public istcauchy (this)
- LOGICAL function, public istdirichlet (this)
- real(kind=dp) function,
  dimension(:), pointer, public gibrandbedingungdirichlet (this)
- real(kind=dp) function,
  dimension(:), pointer, public gibrandbedingungneumann (this)
- real(kind=dp) function,
  dimension(:), pointer, public gibrandbedingungcauchy (this, i)
- real(kind=dp) function,
  dimension(:), pointer, public gibrandbedingungvolumen (this)

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.51 lokalematrizenclass::schuettlematrizen-Interface-Referenz

Öffentliche Methoden

- subroutine schuettleallematrizen (this)

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.52 lokalegekoppeltematrizenclass::schuettlematrizen-Interface-Referenz

Öffentliche Methoden

- subroutine schuettleallematrizen (this)

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.53 lokalegekoppeltematrizenclass::setze-Interface-Referenz

Öffentliche Methoden

- subroutine setzeaufwert (this, wertFORCE, wertSTIFF, wertMASS)
  setzeAufWert setzt die gesammten Matrizen auf Werte

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.54 lokalematrizenclass::setze-Interface-Referenz

Öffentliche Methoden

- subroutine setzeaufwert (this, wertFORCE, wertSTIFF, wertMASS)
  setzeAufWert setzt die gesammten Matrizen auf Werte

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.55 mystdmodules::stdsolverparams_t-Typ-Referenz

Öffentliche Attribute

- real(kind=dp) nonlineartol
- integer nonlineariter
- integer stdofs
- type(valuelist_t), pointer solverparams

9.55.1 Dokumentation der Datenelemente

9.55.1.1 integer mystdmodules::stdsolverparams_t::nonlineariter

9.55.1.2 real(kind=dp) mystdmodules::stdsolverparams_t::nonlineartol

9.55.1.3 type(valuelist_t), pointer mystdmodules::stdsolverparams_t::solverparams

9.55.1.4 integer mystdmodules::stdsolverparams_t::stdofs

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- MyStdModules.f90

9.56 lokalematrizenclass::stiff-Interface-Referenz

Öffentliche Methoden

- real(kind=dp) function,
  dimension(:,:), pointer gibstiff (this)

  gibSTIFF gibt die Steifigkeitsmatrix zurück

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.57 lokalegekoppeltematrizenclass::stiff-Interface-Referenz

Öffentliche Methoden

- real(kind=dp) function,
  dimension(:,:), pointer gibstiff (this)

  gibSTIFF gibt die Steifigkeitsmatrix zurück

Die Dokumentation für dieses Interface wurde aus der folgenden Datei erzeugt:

- LokaleGekoppelteMatrizenClass.f90

9.58 VolumenRandbedingung::VolumenRandbedingung_t-Typ-Referenz

Beinhaltete Werte der Volumenquellen.

9.58.1 Ausführliche Beschreibung

Beinhaltete Werte der Volumenquellen.

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.59 volumenrandbedingungclass::volumenrandbedingung_t-Typ-Referenz

Öffentliche Attribute

- real(kind=dp), dimension(:),
  pointer wert

Die Dokumentation für diesen Typ wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

9.60 volumenrandbedingungclass-Modul-Referenz

Datentypen

- type volumenrandbedingung_t

Die Dokumentation für dieses Modul wurde aus der folgenden Datei erzeugt:

- GeneralSolverPackage.f90

Kapitel 10

Datei-Dokumentation

10.1 BauteilClass.f90-Dateireferenz

Datentypen

- module bauteilclass
- type bauteilclass::bauteil_t
- interface bauteilclass::neu
- interface bauteilclass::berechne

10.2 ElementClass.f90-Dateireferenz

Datentypen

- module elementclass
- type elementclass::bauteilelement_t

  sammelt Elementeigenschaften
- interface elementclass::neu
- interface elementclass::berechne
- interface elementclass::holematerial

10.3 Eriksson2006DGL_Class.f90-Dateireferenz

Datentypen

- module eriksson2006dgl_class
- type eriksson2006dgl_class::eriksson2006dgl_t

10.4 Eriksson2006Solve.f90-Dateireferenz

Funktionen/Unterroutinen

- subroutine eriksson2006solve_init (Model, Solver, dt, Transient)

  Initialisiere einige Standardparameter für die Simulation.
- subroutine eriksson2006solve (Model, Solver, dt, Transient)

  Eriksson2006Solve organisiert den Wärme-Feuchtigkeitstransportsolver.

10.4.1 Funktionen/Unterroutinen-Dokumentation

10.4.1.1 subroutine eriksson2006solve (type(model_t) *Model,* type(solver_t) *Solver,* real(kind=dp) *dt,* logical *Transient*)

Eriksson2006Solve organisiert den Wärme-Feuchtigkeitstransportsolver.

Aufruf erfolgt durch Elmer Der Solver übernimmt die Assemblierung der Elementmatrizen für Rand- und Gebietselemente, berücksichtigt dabei die Randbedingungen für Dirichlet-, Neumann- und Cauchyrandbedingungen und prüft die Fehlerschranken für die nichtlineare Iterationsschleife, nachdem das Gleichungssystem durch, von Elmer bereitgestellte, Routinen gelöst wurde

Siehe auch

http://www.elmerfem.org/doxygen

Parameter

Model	bereitgestellt von Elmer
Solver	bereitgestellt von Elmer
dt	Schrittweite, bereitgestellt von Elmer
Transient	bereitgestellt von Elmer

Noch zu erledigen Deactivierung oder spätere Aktivierung funktioniert bisher nicht! Alle Elemente müssen während der gesammten Simulationsdauer aktiv sein. Ursache sind Beschränkungen in Element-Class, in der SUBROUTINE initialisiereBauteilElement, wo geprüft wird ob aktuell der erste Zeitschritt durchlaufen wird.

10.5 filedocumentation.f90-Dateireferenz

10.6 GeneralSolverPackage.f90-Dateireferenz

Datentypen

- module lokalematrizenclass
- type lokalematrizenclass::lokalematrizen_t
- interface lokalematrizenclass::operator(+)
- interface lokalematrizenclass::neu
- interface lokalematrizenclass::setze
- interface lokalematrizenclass::nullifiziere
- interface lokalematrizenclass::mass
- interface lokalematrizenclass::force
- interface lokalematrizenclass::stiff
- interface lokalematrizenclass::schuettlematrizen
- module neumannrandbedingungclass
- type neumannrandbedingungclass::neumannrandbedingung_t
- module cauchyrandbedingungclass
- type cauchyrandbedingungclass::cauchyrandbedingung_t
- module dirichletrandbedingungclass
- type dirichletrandbedingungclass::dirichletrandbedingung_t
- module volumenrandbedingungclass
- type volumenrandbedingungclass::volumenrandbedingung_t
- module randbedingungclass
- type randbedingungclass::randbedingung_t
- interface randbedingungclass::neu

10.7 LokaleGekoppelteMatrizenClass.f90-Dateireferenz

Datentypen

- module lokalegekoppeltematrizenclass
- type lokalegekoppeltematrizenclass::lokalegekoppeltematrizen_t
- interface lokalegekoppeltematrizenclass::operator(+)

  alles privat, Kommunikation zwischen Modulen erfolgt über Interfaces
- interface lokalegekoppeltematrizenclass::neu
- interface lokalegekoppeltematrizenclass::setze
- interface lokalegekoppeltematrizenclass::nullifiziere
- interface lokalegekoppeltematrizenclass::mass
- interface lokalegekoppeltematrizenclass::force
- interface lokalegekoppeltematrizenclass::stiff
- interface lokalegekoppeltematrizenclass::schuettlematrizen

10.8 Materialeigenschaften.f90-Dateireferenz

Datentypen

- module materialsammlung_class

  struct enhält die für diese Simulation nötigen materialWerte
- type materialsammlung_class::materialsammlung_t
- module geometrischeeigenschaften_class

  struct dieses Moduls enthält die Werte für die geometrie eines Elements
- type geometrischeeigenschaften_class::geometrischeeigenschaften_t

  geometrischeEigenschaften_t Sammlung der für die Simulation nötigen geometrischen Elementeigenschaften

10.9 Materialfunktionen.f90-Dateireferenz

Funktionen/Unterroutinen

- real(kind=dp) function rh_zurwitz (model, n, mc)

  gibt die relative Feuchtigkeit passend zur Temperatur und dem Feuchtegehalt nach Zurwitz et. al (Avramidis1989) zurück
- real(kind=dp) function dwdh_zurwitz (model, n, mc)

  gibt $\frac{\delta\omega}{\delta H}$ passend zur Temperatur und dem Feuchtegehalt nach Zurwitz et. al (Avramidis1989) zurück
- real(kind=dp) function emc_zurwitz (model, n, time)
- real(kind=dp) function cdeliiski (model, n, mc)

  Specific Heat capacity according to Olek2003 in $J/(KgK)$.
- real(kind=dp) function eb (model, n, mc)

  Activierungsenergie gebundenen Wassers als Funktion des Feuchtegehalt.
- subroutine d_t_beech_ihtp (model, n, T, D_T)

  Heat conductivity tensor of european beech wood according to Olek2003 in $W/(mK)$.
- subroutine d_t_pine_ihtp (model, n, T, D_T)

  Heat conductivity tensor of scots pine wood according to Olek2003 in $W/(mK)$.
- subroutine d_w_eriksson (model, n, mc, Diffusivity)

  Diffusion Coefficient matrix according to Eriksson2006.

10.9.1 Funktionen/Unterroutinen-Dokumentation

10.9.1.1 real(kind=dp) function cdeliiski (type(model_t), intent(in) *model,* integer, intent(in) *n,* real(kind=dp), intent(in) *mc*)

Specific Heat capacity according to Olek2003 in $J/(KgK)$.

$c = \frac{0.0022}{1+0.01*M} * T^2 + \frac{3.32*0.01*M+2.95}{1+0.01M} T + \frac{4057*0.01M+526}{1+0.01M} M$ in % T in Kelvin

10.9.1.2 subroutine d_t_beech_ihtp (type(model_t) *model,* integer *n,* real(kind=dp) *T,* real(kind=dp), dimension(:,:), pointer *D_T*)

Heat conductivity tensor of european beech wood according to Olek2003 in $W/(mK)$.

$k_T = 0.19933 + 0.18888 * 10^{-3} * (T - 293.15)$ k_R = 0.19958+0.33211*10^{-3}(T-293.15) $k_L = 0.29937 + 0.70147 * 10^{-3}(T - 293.15)$ $diag(D_T) = (k_T, k_R, k_L)$

10.9.1.3 subroutine d_t_pine_ihtp (type(model_t) *model,* integer *n,* real(kind=dp) *T,* real(kind=dp), dimension(:,:), pointer *D_T*)

Heat conductivity tensor of scots pine wood according to Olek2003 in $W/(mK)$.

$k_T = 0.1989 + 0.8313 * 10^{-4} * (T - 293.15)$ k_R = 0.1990+0.8393*10^{-4}(T-293.15) $k_L = 0.2991 + 0.6184 * 10^{-4}(T - 293.15)$ $diag(D_T) = (k_T, k_R, k_L)$

10.9.1.4 subroutine d_w_eriksson (type(model_t) *model,* integer *n,* real(kind=dp) *mc,* real(kind=dp), dimension(:,:), pointer *Diffusivity*)

Diffusion Coefficient matrix according to Eriksson2006.

$D_{} = 2*10^{-9} m^2/s * e^{0.0641+0.04867*}

10.9.1.5 real(kind=dp) function dwdh_zurwitz (type(model_t) *model,* integer *n,* real(kind=dp), intent(in) *mc*)

gibt $\frac{\delta\omega}{\delta H}$ passend zur Temperatur und dem Feuchtegehalt nach Zurwitz et. al (Avramidis1989) zurück

Zurwitz definiert folgenden Zusammenhang $MC = \left[-T \frac{ln(1-h)}{c_2\left(1-\frac{T}{T_c}\right)^{c_1}} \right]^{\frac{1}{c_3 T^{c_4}}}$ MC und die Temperatur sind für jeden Knoten bekannt und es folgt nach umstellen: $\frac{\delta\omega}{\delta H}$= {1}{c_3 T^{c_4}}({T}{(1-h)c_2(1-{T}{T_c}^{c_1})^{{1-c_3 T^{c_-4}}{c_3 T^{c_4}}

Parameter

n	Knotennummer

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:

10.9.1.6 real(kind=dp) function eb (type(model_t), intent(in) *model,* integer, intent(in) *n,* real(kind=dp), intent(in) *mc*)

Activierungsenergie gebundenen Wassers als Funktion des Feuchtegehalt.

In Eriksson2006 wir für die Aktivierungsenergie gebundenen Wassers

10.9.1.7 real(kind=dp) function emc_zurwitz (type(model_t) *model,* integer *n,* real(kind=dp), intent(in) *time*)

Parameter

n	Knotennummer

10.9.1.8 real(kind=dp) function rh_zurwitz (type(model_t) *model,* integer *n,* real(kind=dp), intent(in) *mc*)

gibt die relative Feuchtigkeit passend zur Temperatur und dem Feuchtegehalt nach Zurwitz et. al (Avramidis1989) zurück

Zurwitz definiert folgenden Zusammenhang $MC = \left[-T\frac{ln(1-h)}{c_2\left(1-\frac{T}{T_c}\right)^{c_1}}\right]^{\frac{1}{c_3 T^{c_4}}}$ MC und die Temperatur sind für jeden Knoten bekannt und es folgt nach umstellen: $RH = 1 - \exp\left(\frac{1}{-T}M^{c_3 T^{c_4}} c_2 \left(1-\frac{T}{T_c}\right)^{c_1}\right)$

Parameter

n	Knotennummer

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

10.10 MyStdModules.f90-Dateireferenz

Datentypen

- module mystdmodules
- type mystdmodules::stdsolverparams_t
- type mystdmodules::localsystemmatrices_t
- interface mystdmodules::printarray

Index