

## Eriksson2006Solver Package

Erzeugt von Doxygen 1.8.2

Mit Sep 19 2012 09:29:26



# Inhaltsverzeichnis

<b>1</b>	<b>Dokumentation eines FEM- Solvers</b>	<b>1</b>
1.1	Introduction . . . . .	1
1.2	Installation . . . . .	1
1.2.1	Step 1: Opening the box . . . . .	1
<b>2</b>	<b>Ausstehende Aufgaben</b>	<b>3</b>
<b>3</b>	<b>Modul-Verzeichnis</b>	<b>5</b>
3.1	Module . . . . .	5
<b>4</b>	<b>Modul-Index</b>	<b>7</b>
4.1	Modulliste . . . . .	7
<b>5</b>	<b>Datentyp-Index</b>	<b>9</b>
5.1	Datentyp-Liste . . . . .	9
<b>6</b>	<b>Datei-Verzeichnis</b>	<b>11</b>
6.1	Auflistung der Dateien . . . . .	11
<b>7</b>	<b>Modul-Dokumentation</b>	<b>13</b>
7.1	ElementClass . . . . .	13
7.1.1	Ausführliche Beschreibung . . . . .	14
7.1.2	Funktionen/Unterrouinen-Dokumentation . . . . .	14
7.1.2.1	berechneelement . . . . .	14
7.1.2.2	berechneelementstaggered1 . . . . .	14
7.1.2.3	berechneelementstaggered2 . . . . .	14
7.1.2.4	holematerialtensor . . . . .	15
7.1.2.5	holematerialtensor . . . . .	15
7.1.2.6	holematerialvektor . . . . .	15
7.1.2.7	initialisierebauteilelement . . . . .	15
7.1.2.8	initialisieredgl . . . . .	15
7.1.2.9	initialisieregeometrie . . . . .	15
7.1.2.10	inputtensor . . . . .	15
7.1.2.11	iteriereueberallelemente . . . . .	16

7.1.2.12	iteriereueberallelementestaggered1	16
7.1.2.13	iteriereueberallelementestaggered2	16
7.1.2.14	setzelementzaehler	16
7.1.2.15	setzepointeraufnaechsteselement	16
7.1.3	Variablen-Dokumentation	17
7.1.3.1	element	17
7.1.3.2	elementdof	17
7.1.3.3	elementmatrizen	17
7.1.3.4	elementnummer	17
7.1.3.5	erikssondgl	17
7.1.3.6	feuchtigkeitrandbedingung	17
7.1.3.7	geometrie	17
7.1.3.8	istgebietselement	17
7.1.3.9	material	17
7.1.3.10	materialwerte	17
7.1.3.11	model	17
7.1.3.12	naechsteselement	17
7.1.3.13	solver	17
7.1.3.14	temperaturrandbedingung	17
7.1.3.15	vorherigelsg	17
7.2	Eriksson2006Solve	18
7.2.1	Ausführliche Beschreibung	18
7.2.2	Funktionen/Unterrouinen-Dokumentation	18
7.2.2.1	eriksson2006solve_init	18
7.3	LokaleMatrizenClass	19
7.3.1	Ausführliche Beschreibung	19
7.3.2	Funktionen/Unterrouinen-Dokumentation	20
7.3.2.1	addiere	20
7.3.2.2	gibforce	20
7.3.2.3	gibmass	20
7.3.2.4	gibstiff	20
7.3.2.5	initmatrizen	20
7.3.2.6	initmatrizenkurz	20
7.3.2.7	initmatrizenkurz	20
7.3.2.8	nullifizierematrizen	20
7.3.2.9	schuettleallematrizen	21
7.3.2.10	setzaufwert	21
7.3.2.11	setzematrizen	21
7.4	MaterialClass	22
7.4.1	Ausführliche Beschreibung	23

7.4.2	Funktionen/Unterroutinen-Dokumentation	23
7.4.2.1	gibrandbedingungcauchy	23
7.4.2.2	gibrandbedingundirichlet	23
7.4.2.3	gibrandbedingungneumann	23
7.4.2.4	gibrandbedingungvolumen	23
7.4.2.5	istcauchy	23
7.4.2.6	istdirichlet	23
7.4.2.7	istneumann	23
7.4.2.8	istvolumen	23
7.4.3	Variablen-Dokumentation	24
7.4.3.1	bezugswert	24
7.4.3.2	cauchyrandbedingung	24
7.4.3.3	dirichletrandbedingung	24
7.4.3.4	neumannrandbedingung	24
7.4.3.5	skalierfaktor	24
7.4.3.6	volumenrandbedingung	24
7.4.3.7	wert	24
7.4.3.8	wert	24
7.4.3.9	wert	24
7.5	LokaleGekoppelteMatrizenClass	25
7.5.1	Ausführliche Beschreibung	26
7.5.2	Funktionen/Unterroutinen-Dokumentation	26
7.5.2.1	addiere	26
7.5.2.2	gibforce	26
7.5.2.3	gibmass	26
7.5.2.4	gibstiff	26
7.5.2.5	initialisierematrizen	27
7.5.2.6	initialisierematrizen	27
7.5.2.7	nullifizierematrizen	27
7.5.2.8	schuettleallematrizen	27
7.5.2.9	schuettleallematrizen	27
7.5.2.10	setzeaufwert	27
7.5.2.11	setzematrizen	27
7.5.3	Variablen-Dokumentation	27
7.5.3.1	caa	27
7.5.3.2	cab	27
7.5.3.3	cba	27
7.5.3.4	cbb	27
7.5.3.5	fa	28
7.5.3.6	fb	28

7.5.3.7	kaa	28
7.5.3.8	kab	28
7.5.3.9	kba	28
7.5.3.10	kbb	28
7.6	BauteilClass	29
7.6.1	Ausführliche Beschreibung	29
7.6.2	Funktionen/Unterroutinen-Dokumentation	29
7.6.2.1	berechnebauteil	29
7.6.2.2	berechnebauteilstaggered	30
7.6.2.3	berechnerelativefeuchte	30
7.6.2.4	initialisierebauteil	30
7.6.3	Variablen-Dokumentation	30
7.6.3.1	anzahlgebietselemente	30
7.6.3.2	anzahlrandgebietselemente	30
7.6.3.3	elementpointerliste	30
7.6.3.4	nonlineariter	30
7.6.3.5	nonlineartol	30
7.6.3.6	problemdim	30
7.6.3.7	solver	30
7.6.3.8	vorhergehendeloesung	30
7.7	Eriksson2006DGL_Class	31
7.7.1	Ausführliche Beschreibung	31
7.7.2	Variablen-Dokumentation	31
7.7.2.1	feuchtigkeitrandbedingung	31
7.7.2.2	geometrie	31
7.7.2.3	lokalegekoppeltematrizen	31
7.7.2.4	materialwerte	31
7.7.2.5	temperaturrandbedingung	31
<b>8</b>	<b>Modul-Dokumentation</b>	<b>33</b>
8.1	NeumannRandbedingung-Modul-Referenz	33
8.2	VolumenRandbedingung-Modul-Referenz	33
<b>9</b>	<b>Datentyp-Dokumentation</b>	<b>35</b>
9.1	bauteilclass::bauteil_t-Typ-Referenz	35
9.2	bauteilclass-Modul-Referenz	35
9.3	elementclass::bauteilelement_t-Typ-Referenz	36
9.3.1	Ausführliche Beschreibung	36
9.4	bauteilclass::berechne-Interface-Referenz	36
9.5	elementclass::berechne-Interface-Referenz	37
9.6	CauchyRandbedingungClass::CauchyRandbedingung_t-Typ-Referenz	37

9.6.1	Ausführliche Beschreibung	37
9.7	cauchyrandbedingungclass::cauchyrandbedingung_t-Typ-Referenz	37
9.8	cauchyrandbedingungclass-Modul-Referenz	37
9.9	dirichletrandbedingungclass::dirichletrandbedingung_t-Typ-Referenz	38
9.10	DirichletRandbedingung::DirichletRandbedingung_t-Typ-Referenz	38
9.10.1	Ausführliche Beschreibung	38
9.11	dirichletrandbedingungclass-Modul-Referenz	38
9.12	ElementClass-Modul-Referenz	38
9.12.1	Ausführliche Beschreibung	38
9.13	elementclass-Modul-Referenz	39
9.14	eriksson2006dgl_class-Modul-Referenz	40
9.15	eriksson2006dgl_class::eriksson2006dgl_t-Typ-Referenz	40
9.16	lokalematrixenclass::force-Interface-Referenz	40
9.17	lokalegekoppeltematrixenclass::force-Interface-Referenz	40
9.18	geometrischeeigenschaften_class-Modul-Referenz	41
9.18.1	Ausführliche Beschreibung	41
9.19	geometrischeeigenschaften_class::geometrischeeigenschaften_t-Typ-Referenz	41
9.19.1	Ausführliche Beschreibung	41
9.19.2	Dokumentation der Datenelemente	42
9.19.2.1	anzahlknoten	42
9.19.2.2	b	42
9.19.2.3	dbdx	42
9.19.2.4	detj	42
9.19.2.5	dim	42
9.19.2.6	element	42
9.19.2.7	ip	42
9.19.2.8	knotenfreiwerte	42
9.19.2.9	n	42
9.19.2.10	nodes	42
9.20	elementclass::holematerial-Interface-Referenz	42
9.21	mystdmodules::localsystemmatrices_t-Typ-Referenz	42
9.21.1	Dokumentation der Datenelemente	43
9.21.1.1	force	43
9.21.1.2	mass	43
9.21.1.3	stiff	43
9.22	LokaleGekoppelteMatrizen_t-Typ-Referenz	43
9.22.1	Ausführliche Beschreibung	43
9.23	lokalegekoppeltematrixenclass::lokalegekoppeltematrixen_t-Typ-Referenz	43
9.24	lokalegekoppeltematrixenclass-Modul-Referenz	44
9.25	lokalematrixenclass::lokalematrixen_t-Typ-Referenz	44

9.26 LokaleMatrizen_t-Typ-Referenz . . . . .	44
9.26.1 Ausführliche Beschreibung . . . . .	44
9.27 lokalematrizenclass-Modul-Referenz . . . . .	45
9.28 lokalematrizenclass::mass-Interface-Referenz . . . . .	45
9.29 lokalegekoppeltematrizenclass::mass-Interface-Referenz . . . . .	45
9.30 MaterialClass-Modul-Referenz . . . . .	45
9.30.1 Ausführliche Beschreibung . . . . .	46
9.31 materialsammlung_class-Modul-Referenz . . . . .	46
9.31.1 Ausführliche Beschreibung . . . . .	46
9.32 materialsammlung_class::materialsammlung_t-Typ-Referenz . . . . .	46
9.32.1 Dokumentation der Datenelemente . . . . .	47
9.32.1.1 dichte . . . . .	47
9.32.1.2 dichtetensor . . . . .	47
9.32.1.3 diffusionskoeffizient . . . . .	47
9.32.1.4 diffusionskoeffizienttensor . . . . .	47
9.32.1.5 dwdh . . . . .	47
9.32.1.6 e_b . . . . .	47
9.32.1.7 konduktivitaet . . . . .	47
9.32.1.8 konduktivitaetensor . . . . .	47
9.32.1.9 r . . . . .	47
9.32.1.10 relativehumidity . . . . .	47
9.32.1.11 spezifischewaerme . . . . .	47
9.32.1.12 spezifischewaermetensor . . . . .	47
9.32.1.13 trockendichte . . . . .	47
9.32.1.14 trockendichtetensor . . . . .	47
9.33 MaterialSammlungClass-Modul-Referenz . . . . .	48
9.33.1 Ausführliche Beschreibung . . . . .	48
9.34 mystdmodules-Modul-Referenz . . . . .	48
9.34.1 Elementfunktionen/Unterroutinen-Dokumentation . . . . .	48
9.34.1.1 print1darray . . . . .	48
9.34.1.2 print2darray . . . . .	48
9.34.1.3 printstdinfo . . . . .	48
9.34.1.4 returnstdsolverparams . . . . .	48
9.34.1.5 rotate2indextensor . . . . .	48
9.34.1.6 rotate4indextensor . . . . .	48
9.34.1.7 rotateelasticitymatrix . . . . .	49
9.34.1.8 rotateelasticitymatrix2d . . . . .	49
9.34.1.9 rotateelasticitymatrix3d . . . . .	49
9.35 elementclass::neu-Interface-Referenz . . . . .	49
9.36 lokalematrizenclass::neu-Interface-Referenz . . . . .	49



9.37	randbedingungclass::neu-Interface-Referenz . . . . .	49
9.38	lokalegekoppeltematrizenclass::neu-Interface-Referenz . . . . .	50
9.39	bauteilclass::neu-Interface-Referenz . . . . .	50
9.40	NeumannRandbedingung::NeumannRandbedingung_t-Typ-Referenz . . . . .	50
9.40.1	Ausführliche Beschreibung . . . . .	50
9.41	neumannrandbedingungclass::neumannrandbedingung_t-Typ-Referenz . . . . .	50
9.42	neumannrandbedingungclass-Modul-Referenz . . . . .	51
9.43	NeumannRandbedingungClass-Modul-Referenz . . . . .	51
9.43.1	Ausführliche Beschreibung . . . . .	51
9.44	lokalematrizenclass::nullifiziere-Interface-Referenz . . . . .	51
9.45	lokalegekoppeltematrizenclass::nullifiziere-Interface-Referenz . . . . .	51
9.46	lokalematrizenclass::operator(+)-Interface-Referenz . . . . .	52
9.47	lokalegekoppeltematrizenclass::operator(+)-Interface-Referenz . . . . .	52
9.47.1	Ausführliche Beschreibung . . . . .	52
9.48	mystdmodules::printarray-Interface-Referenz . . . . .	52
9.48.1	Elementfunktionen/Unterrouinen-Dokumentation . . . . .	52
9.48.1.1	print1darray . . . . .	52
9.48.1.2	print2darray . . . . .	52
9.49	randbedingungclass::randbedingung_t-Typ-Referenz . . . . .	53
9.50	randbedingungclass-Modul-Referenz . . . . .	53
9.51	lokalematrizenclass::schuettlematrizen-Interface-Referenz . . . . .	53
9.52	lokalegekoppeltematrizenclass::schuettlematrizen-Interface-Referenz . . . . .	54
9.53	lokalegekoppeltematrizenclass::setze-Interface-Referenz . . . . .	54
9.54	lokalematrizenclass::setze-Interface-Referenz . . . . .	54
9.55	mystdmodules::stdsolverparams_t-Typ-Referenz . . . . .	54
9.55.1	Dokumentation der Datenelemente . . . . .	54
9.55.1.1	nonlineariter . . . . .	54
9.55.1.2	nonlineartol . . . . .	54
9.55.1.3	solverparams . . . . .	54
9.55.1.4	stdofs . . . . .	54
9.56	lokalematrizenclass::stiff-Interface-Referenz . . . . .	55
9.57	lokalegekoppeltematrizenclass::stiff-Interface-Referenz . . . . .	55
9.58	VolumenRandbedingung::VolumenRandbedingung_t-Typ-Referenz . . . . .	55
9.58.1	Ausführliche Beschreibung . . . . .	55
9.59	volumenrandbedingungclass::volumenrandbedingung_t-Typ-Referenz . . . . .	55
9.60	volumenrandbedingungclass-Modul-Referenz . . . . .	56
<b>10</b>	<b>Datei-Dokumentation</b> . . . . .	<b>57</b>
10.1	BauteilClass.f90-Dateireferenz . . . . .	57
10.2	ElementClass.f90-Dateireferenz . . . . .	57

10.3 Eriksson2006DGL_Class.f90-Dateireferenz . . . . .	57
10.4 Eriksson2006Solve.f90-Dateireferenz . . . . .	57
10.4.1 Funktionen/Unterroutinen-Dokumentation . . . . .	58
10.4.1.1 eriksson2006solve . . . . .	58
10.5 filedocumentation.f90-Dateireferenz . . . . .	58
10.6 GeneralSolverPackage.f90-Dateireferenz . . . . .	58
10.7 LokaleGekoppelteMatrizenClass.f90-Dateireferenz . . . . .	59
10.8 Materialeigenschaften.f90-Dateireferenz . . . . .	59
10.9 Materialfunktionen.f90-Dateireferenz . . . . .	59
10.9.1 Funktionen/Unterroutinen-Dokumentation . . . . .	60
10.9.1.1 cdeliiski . . . . .	60
10.9.1.2 d_t_beech_ihtp . . . . .	60
10.9.1.3 d_t_pine_ihtp . . . . .	60
10.9.1.4 d_w_eriksson . . . . .	60
10.9.1.5 dwdh_zurwitz . . . . .	60
10.9.1.6 eb . . . . .	60
10.9.1.7 emc_zurwitz . . . . .	60
10.9.1.8 rh_zurwitz . . . . .	61
10.10MyStdModules.f90-Dateireferenz . . . . .	61
<b>Index</b>	<b>61</b>

# 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::initialisiertdgl` (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)

Deaktivierung oder spätere Aktivierung funktioniert bisher nicht! Alle Elemente müssen während der gesamten 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

- Prozeduren hinzufügen

# Kapitel 3

## Modul-Verzeichnis

### 3.1 Module

Hier folgt die Aufzählung aller Module:

ElementClass . . . . .	13
Eriksson2006Solve . . . . .	18
LokaleMatrizenClass . . . . .	19
MaterialClass . . . . .	22
LokaleGekoppelteMatrizenClass . . . . .	25
BauteilClass . . . . .	29
Eriksson2006DGL_Class . . . . .	31





# Kapitel 4

## Modul-Index

### 4.1 Modulliste

Hier folgt eine Liste aller Module mit ihren Kurzbeschreibungen:

<a href="#">NeumannRandbedingung</a>	.....	33
<a href="#">VolumenRandbedingung</a>	.....	33



# Kapitel 5

## Datentyp-Index

### 5.1 Datentyp-Liste

Hier folgen die Datentypen mit Kurzbeschreibungen:

<a href="#">bauteilclass::bauteil_t</a>	35
<a href="#">bauteilclass</a>	35
<a href="#">elementclass::bauteilelement_t</a>	
Sammelt Elementeigenschaften	36
<a href="#">bauteilclass::berechne</a>	36
<a href="#">elementclass::berechne</a>	37
<a href="#">CauchyRandbedingungClass::CauchyRandbedingung_t</a>	
Beinhaltet Werte der Cauchy Randbedingung	37
<a href="#">cauchyrandbedingungclass::cauchyrandbedingung_t</a>	37
<a href="#">cauchyrandbedingungclass</a>	37
<a href="#">dirichletrandbedingungclass::dirichletrandbedingung_t</a>	38
<a href="#">DirichletRandbedingung::DirichletRandbedingung_t</a>	
Beinhaltet den Wert der DirichletRandbedingungen	38
<a href="#">dirichletrandbedingungclass</a>	38
<a href="#">ElementClass</a>	
Sammelt und verarbeitet elementbasierte Informationen	38
<a href="#">elementclass</a>	39
<a href="#">eriksson2006dgl_class</a>	40
<a href="#">eriksson2006dgl_class::eriksson2006dgl_t</a>	40
<a href="#">lokalematrixenclass::force</a>	40
<a href="#">lokalegekoppeltematrixenclass::force</a>	40
<a href="#">geometrischeeigenschaften_class</a>	
Struct dieses Moduls enthält die Werte für die Geometrie eines Elements	41
<a href="#">geometrischeeigenschaften_class::geometrischeeigenschaften_t</a>	
GeometrischeEigenschaften_t Sammlung der für die Simulation nötigen geometrischen Elementeigenschaften	41
<a href="#">elementclass::holematerial</a>	42
<a href="#">mystdmodules::localsystemmatrices_t</a>	42
<a href="#">LokaleGekoppelteMatrizen_t</a>	
Type Struktur zur Sammlung der Systemmatrizen und deren Submatrizen	43
<a href="#">lokalegekoppeltematrixenclass::lokalegekoppeltematrizen_t</a>	43
<a href="#">lokalegekoppeltematrixenclass</a>	44
<a href="#">lokalematrixenclass::lokalematrizen_t</a>	44
<a href="#">LokaleMatrizen_t</a>	
Type Struktur zur Sammlung der Systemmatrizen	44
<a href="#">lokalematrixenclass</a>	45
<a href="#">lokalematrixenclass::mass</a>	45
<a href="#">lokalegekoppeltematrixenclass::mass</a>	45

MaterialClass	
Struktur für einen Materialparameter	45
materialsammlung_class	
Struct enthält die für diese Simulation nötigen materialWerte	46
materialsammlung_class::materialsammlung_t	46
MaterialSammlungClass	
Sammelt die Materialeigenschaften in einer Baumstruktur	48
mystdmodules	48
elementclass::neu	49
lokalematrizenclass::neu	49
randbedingungclass::neu	49
lokalegekoppeltematrizenclass::neu	50
bauteilclass::neu	50
NeumannRandbedingung::NeumannRandbedingung_t	
Beinhaltet den Wert der Neuman-Randbedingungen	50
neumannrandbedingungclass::neumannrandbedingung_t	50
neumannrandbedingungclass	51
NeumannRandbedingungClass	
"RandbedingungClass.f90"	51
lokalematrizenclass::nullifiziere	51
lokalegekoppeltematrizenclass::nullifiziere	51
lokalematrizenclass::operator(+)	52
lokalegekoppeltematrizenclass::operator(+)	
Alles privat, Kommunikation zwischen Modulen erfolgt über Interfaces	52
mystdmodules::printarray	52
randbedingungclass::randbedingung_t	53
randbedingungclass	53
lokalematrizenclass::schuettlematrizen	53
lokalegekoppeltematrizenclass::schuettlematrizen	54
lokalegekoppeltematrizenclass::setze	54
lokalematrizenclass::setze	54
mystdmodules::stdsolverparams_t	54
lokalematrizenclass::stiff	55
lokalegekoppeltematrizenclass::stiff	55
VolumenRandbedingung::VolumenRandbedingung_t	
Beinhaltete Werte der Volumenquellen	55
volumenrandbedingungclass::volumenrandbedingung_t	55
volumenrandbedingungclass	56

# Kapitel 6

## Datei-Verzeichnis

### 6.1 Auflistung der Dateien

Hier folgt die Aufzählung aller Dateien mit einer Kurzbeschreibung:

BauteilClass.f90 . . . . .	57
ElementClass.f90 . . . . .	57
Eriksson2006DGL_Class.f90 . . . . .	57
Eriksson2006Solve.f90 . . . . .	57
filedocumentation.f90 . . . . .	58
GeneralSolverPackage.f90 . . . . .	58
LokaleGekoppelteMatrizenClass.f90 . . . . .	59
Materialeigenschaften.f90 . . . . .	59
Materialfunktionen.f90 . . . . .	59
MyStdModules.f90 . . . . .	61



# 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::initialisiertdgl](#) (this)  
*Übergibt die für dieses Element spezifischen Material- und Geometrieigenschaften 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::initialisiertdgl ( type(bauteilelement\_t), intent(inout) *this* )

Übergibt die für dieses Element spezifischen Material- und Geometrieigenschaften 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 Nachbarlement zu kennen kann eine Linked List erzeugt werden, die es ermöglicht über alle Elemente zu iterieren. Der Pointer wird durch die SUBROUTINE setzePointerAufNaechstesElement 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 Nachbarlement zu kennen kann eine Linked List erzeugt werden, die es ermöglicht über alle Elemente zu iterieren. Der Pointer wird durch die SUBROUTINE setzePointerAufNaechstesElement 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 Nachbarlement zu kennen kann eine Linked List erzeugt werden, die es ermöglicht über alle Elemente zu iterieren. Der Pointer wird durch die SUBROUTINE setzePointerAufNaechstesElement 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 Nachbarlement.

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

<i>Model</i>	bereitgestellt von Elmer
<i>Solver</i>	bereitgestellt von Elmer
<i>dt</i>	Schrittweite, bereitgestellt von Elmer
<i>Transient</i>	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 einzelnen 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 gesamten 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 lokalematrixclass::operator(+)::addiere ( type(lokalematrixen_t), intent(in) this, type(lokalematrixen_t), intent(in) LM2 )`

addiert zwei LokaleMatrizen indem die einzelnen 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 lokalematrixclass::force::gibforce ( type(lokalematrixen_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 lokalematrixclass::mass::gibmass ( type(lokalematrixen_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 lokalematrixclass::stiff::gibstiff ( type(lokalematrixen_t), intent(in) this )`

gibSTIFF gibt die Steifigkeitsmatrix zurück

#### Parameter

<code>in</code>	<code>this</code>	<code>LokaleMatrizen_t</code> : this Returns REAL(KIND=dp), DIMENSION(SHAPE(STIFF))
-----------------	-------------------	---

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

7.3.2.5 `subroutine lokalematrixclass::neu::initmatrizen ( type( lokalematrixen_t ), intent(inout) this, integer, dimension(2), intent(in) maxElementMatrixSize )`

initMatrizen allokiert Speicherplatz für Systemmatrizen

Um Types mit dynamische großen Elementen zu erzeugen ist es nötig POINTER zu verwenden, allokierte Arrays innerhalb von Type Definitionen nicht erlaubt sind. Die POINTER bekommen im folgenden ähnlich allokiertbarer Arrays einen Speicherbereich ihrer Größe entsprechend zugewiesen.

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

7.3.2.6 `subroutine lokalematrixclass::initmatrizenkurz ( type( lokalematrixen_t ), intent(inout) this, integer, intent(in) dof )`

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

7.3.2.7 `subroutine lokalematrixclass::neu::initmatrizenkurz ( type( lokalematrixen_t ), intent(inout) this, integer, intent(in) dof )`

7.3.2.8 `subroutine lokalematrixclass::nullifiziere::nullifizierematrizen ( type(lokalematrixen_t), intent(inout) this )`

7.3.2.9 subroutine lokalematrizenclass::schuettlematrizen::schuettleallematrizen ( type(lokaletmatrizen\_t), intent(inout) *this* )

7.3.2.10 subroutine lokalematrizenclass::setze::setzeaufwert ( type(lokaletmatrizen\_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 gesamten 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

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

7.3.2.11 subroutine lokalematrizenclass::neu::setzematrizen ( type(lokaletmatrizen\_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 einzelnen 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 gesamten 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 einzelnen 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

<i>this</i>	
-------------	--

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 gesamten 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

<i>this</i>	Sammlung der Matrizen
<i>wertFORCE</i>	0.0d0, falls nicht vorhanden
<i>wertstiff</i>	alter Wert falls nicht vorhanden
<i>wertmass</i>	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/Unterrountinen

- 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/Unterrountinen-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` )

initialisiert 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. Berechnung der Elementmatrizen nach der Galerkin Methode für das gekoppelte Temperatur-Feuchtigkeitsfeldproblem nach Eriksson2006. Das Feldproblem wird nach einem Staggered 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)  
*initialisiert und steuert die Eriksson2006Solve berechnung für dieses Mesh*
- subroutine [berechnrelativefeuchte](#) (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

<i>elementNumber</i>	vielleicht überflüssig
<i>ist-Gebietselement</i>	seperiert in Gebiets Elemente 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)  
*initialisiert 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

<i>Wert</i>	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 Gesamtmatrizen 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 [setzelementzaehler](#) (anzahlElemente)
- subroutine [initialisieregeometrie](#) (this)  
*initialisiert die Werte für die Geometrie für die Berechnung der Eriksson2006DGL\_Class*
- subroutine [initialisiertdgl](#) (this)  
*Übergibt die für dieses Element spezifischen Material- und Geometrieigenschaften 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 Funktionen 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 Funktionen gleichzeitig auf alle Arrays anzuwenden. Es werden allozierbare POINTER als Behelflösung verwendet, da allozierbare 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 lokalematrixenclass-Modul-Referenz

### Datentypen

- interface [force](#)
- type [lokalematrixen\\_t](#)
- interface [mass](#)
- interface [neu](#)
- interface [nullifiziere](#)
- interface [operator\(+\)](#)
- interface [schuettlematrixen](#)
- interface [setze](#)
- interface [stiff](#)

### Öffentliche Methoden

- subroutine [initmatrixenkurz](#) (this, dof)

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

- [GeneralSolverPackage.f90](#)

## 9.28 lokalematrixenclass::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 lokalegekoppeltematrixenclass::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 enthält die für diese Simulation nötigen materialWerte

### Datentypen

- type [materialsammlung\\_t](#)

### 9.31.1 Ausführliche Beschreibung

struct enthält 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 [konduktivitaetensor](#) >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::konduktivitaetensor >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 [initialisierbauteilelement](#) (this, ElemNr, Solver, Model, istGebietselement)

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

- [ElementClass.f90](#)

## 9.36 lokalematrixenclass::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

<i>Wert</i>	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
  - Prozeduren hinzufügen

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

- [GeneralSolverPackage.f90](#)

## 9.44 lokalematrixnclass::nullifiziere-Interface-Referenz

### Öffentliche Methoden

- subroutine [nullifizierematrizen](#) (this)

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

- [GeneralSolverPackage.f90](#)

## 9.45 lokalegekoppeltematrixnclass::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 einzelnen 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 einzelnen 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 lokalematrixnclass::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 gesamten 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 gesamten 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 lokalematrixclass::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 lokalegekoppeltematrixclass::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 Gebiets-elemente, 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

<i>Model</i>	bereitgestellt von Elmer
<i>Solver</i>	bereitgestellt von Elmer
<i>dt</i>	Schrittweite, bereitgestellt von Elmer
<i>Transient</i>	bereitgestellt von Elmer

**Noch zu erledigen** Deaktivierung oder spätere Aktivierung funktioniert bisher nicht! Alle Elemente müssen während der gesamten 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 [lokalematrixclass](#)
- type [lokalematrixclass::lokalematrixen\\_t](#)
- interface [lokalematrixclass::operator\(+\)](#)
- interface [lokalematrixclass::neu](#)
- interface [lokalematrixclass::setze](#)
- interface [lokalematrixclass::nullifiziere](#)
- interface [lokalematrixclass::mass](#)
- interface [lokalematrixclass::force](#)
- interface [lokalematrixclass::stiff](#)
- interface [lokalematrixclass::schuettlematrixen](#)
- 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 enthält 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) model, integer, intent(in) n, real(kind=dp), intent(in) mc )`

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

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

10.9.1.2 `subroutine d_t_beech_ihnp ( 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  $W/(mK)$ .

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

10.9.1.3 `subroutine d_t_pine_ihnp ( 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  $W/(mK)$ .

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

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.

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

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  $MC = \left[ -T \frac{\ln(1-h)}{c_2(1-\frac{T}{T_c})^{c_1}} \right] \frac{1}{c_3 T^{c_4}}$   $MC$  und die Temperatur sind für jeden

Knoten bekannt und es folgt nach umstellen:  $\frac{\delta \omega}{\delta H} = \{c_3 T^{c_4}\} \{T\} \{(1-h)c_2(1-\frac{T}{T_c})^{c_1}\}^{\{1-c_3 T^{c_4}\}} \{c_3 T^{c_4}\}$

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) model, integer, intent(in) n, real(kind=dp), intent(in) mc )`

Aktivierungsenergie gebundenen Wassers als Funktion des Feuchtegehalt.

In Eriksson2006 wird 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

- addiere
  - LokaleGekoppelteMatrizenClass, 26
  - LokaleMatrizenClass, 20
- anzahlgebietselemente
  - BauteilClass, 30
- anzahlknoten
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- anzahlrandgebietselemente
  - BauteilClass, 30
- b
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- BauteilClass, 29
  - anzahlgebietselemente, 30
  - anzahlrandgebietselemente, 30
  - berechnebauteil, 29
  - berechnebauteilstaggered, 29
  - berechnerelativefeuchte, 30
  - elementpointerliste, 30
  - initialisierebauteil, 30
  - nonlineariter, 30
  - nonlineartol, 30
  - problemdim, 30
  - solver, 30
  - vorhergehendeloesung, 30
- BauteilClass.f90, 57
- bauteilclass, 35
- bauteilclass::bauteil\_t, 35
- bauteilclass::berechne, 36
- bauteilclass::neu, 50
- berechnebauteil
  - BauteilClass, 29
- berechnebauteilstaggered
  - BauteilClass, 29
- berechneelement
  - ElementClass, 14
- berechneelementstaggered1
  - ElementClass, 14
- berechneelementstaggered2
  - ElementClass, 14
- berechnerelativefeuchte
  - BauteilClass, 30
- bezugswert
  - MaterialClass, 24
- caa
  - LokaleGekoppelteMatrizenClass, 27
- cab
  - LokaleGekoppelteMatrizenClass, 27
- CauchyRandbedingungClass::CauchyRandbedingung\_t, 37
- cauchyrandbedingung
  - MaterialClass, 24
- cauchyrandbedingungclass, 37
- cauchyrandbedingungclass::cauchyrandbedingung\_t, 37
- cba
  - LokaleGekoppelteMatrizenClass, 27
- cbb
  - LokaleGekoppelteMatrizenClass, 27
- cauchyrandbedingungclass::cauchyrandbedingung\_t, 37
- Materialfunktionen.f90, 60
- d\_t\_beech\_ihtp
  - Materialfunktionen.f90, 60
- d\_t\_pine\_ihtp
  - Materialfunktionen.f90, 60
- d\_w\_eriksson
  - Materialfunktionen.f90, 60
- dbdx
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- detj
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- dichte
  - materialsammlung\_class::materialsammlung\_t, 47
- dichtetensor
  - materialsammlung\_class::materialsammlung\_t, 47
- diffusionskoeffizient
  - materialsammlung\_class::materialsammlung\_t, 47
- diffusionskoeffizienttensor
  - materialsammlung\_class::materialsammlung\_t, 47
- dim
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- DirichletRandbedingung::DirichletRandbedingung\_t, 38
- dirichletrandbedingung
  - MaterialClass, 24
- dirichletrandbedingungclass, 38
- dirichletrandbedingungclass::dirichletrandbedingung\_t, 38
- dwdh
  - materialsammlung\_class::materialsammlung\_t, 47
- dwdh\_zurwitz
  - Materialfunktionen.f90, 60
- e\_b



- materialsammlung\_class::materialsammlung\_t, 47
- eb
  - Materialfunktionen.f90, 60
- element
  - ElementClass, 17
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- ElementClass, 13, 38
  - berechneelement, 14
  - berechneelementstaggered1, 14
  - berechneelementstaggered2, 14
  - element, 17
  - elementdof, 17
  - elementmatrizen, 17
  - elementnummer, 17
  - erikssondgl, 17
  - feuchtigkeitrandbedingung, 17
  - geometrie, 17
  - holematerialtensor, 14, 15
  - holematerialvektor, 15
  - initialisierebauteilelement, 15
  - initialisieredgl, 15
  - initialisiergeometrie, 15
  - inputtensor, 15
  - istgebietsselement, 17
  - iteriereueberalleelemente, 15
  - iteriereueberalleelementestaggered1, 16
  - iteriereueberalleelementestaggered2, 16
  - material, 17
  - materialwerte, 17
  - model, 17
  - naechsteselement, 17
  - setzelementzaehler, 16
  - setzepointeraufnaechsteselement, 16
  - solver, 17
  - temperaturrandbedingung, 17
  - vorherigelsg, 17
- ElementClass.f90, 57
- elementclass, 39
- elementclass::bauteilelement\_t, 36
- elementclass::berechne, 37
- elementclass::holematerial, 42
- elementclass::neu, 49
- elementdof
  - ElementClass, 17
- elementpointerliste
  - BauteilClass, 30
- elementmatrizen
  - ElementClass, 17
- elementnummer
  - ElementClass, 17
- emc\_zurwitz
  - Materialfunktionen.f90, 60
- Eriksson2006DGL\_Class, 31
  - feuchtigkeitrandbedingung, 31
  - geometrie, 31
  - lokalegekoppeltematrizen, 31
  - materialwerte, 31
  - temperaturrandbedingung, 31
  - Eriksson2006DGL\_Class.f90, 57
  - Eriksson2006Solve, 18
    - eriksson2006solve\_init, 18
  - Eriksson2006Solve.f90, 57
  - eriksson2006solve, 58
  - eriksson2006dgl\_class, 40
  - eriksson2006dgl\_class::eriksson2006dgl\_t, 40
  - eriksson2006solve
    - Eriksson2006Solve.f90, 58
  - eriksson2006solve\_init
    - Eriksson2006Solve, 18
  - erikssondgl
    - ElementClass, 17
- fa
  - LokaleGekoppelteMatrizenClass, 27
- fb
  - LokaleGekoppelteMatrizenClass, 28
- feuchtigkeitrandbedingung
  - ElementClass, 17
  - Eriksson2006DGL\_Class, 31
- filedocumentation.f90, 58
- force
  - mystdmodules::localsystemmatrices\_t, 43
- GeneralSolverPackage.f90, 58
- geometrie
  - ElementClass, 17
  - Eriksson2006DGL\_Class, 31
- geometrischeeigenschaften\_class, 41
- geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 41
  - anzahlknoten, 42
  - b, 42
  - dbdx, 42
  - detj, 42
  - dim, 42
  - element, 42
  - ip, 42
  - knotenfreiwerte, 42
  - n, 42
  - nodes, 42
- gibforce
  - LokaleGekoppelteMatrizenClass, 26
  - LokaleMatrizenClass, 20
- gibmass
  - LokaleGekoppelteMatrizenClass, 26
  - LokaleMatrizenClass, 20
- gibrandbedingungcauchy
  - MaterialClass, 23
- gibrandbedingungdirichlet
  - MaterialClass, 23
- gibrandbedingungneumann
  - MaterialClass, 23
- gibrandbedingungvolumen
  - MaterialClass, 23
- gibstiff
  - LokaleGekoppelteMatrizenClass, 26

- LokaleMatrizenClass, 20
- holematerialtensor
  - ElementClass, 14, 15
- holematerialvektor
  - ElementClass, 15
- initialisierebauteil
  - BauteilClass, 30
- initialisierebauteilelement
  - ElementClass, 15
- initialisieredgl
  - ElementClass, 15
- initialisieregeometrie
  - ElementClass, 15
- initialisierematrizen
  - LokaleGekoppelteMatrizenClass, 27
- initmatrizen
  - LokaleMatrizenClass, 20
- initmatrizenkurz
  - LokaleMatrizenClass, 20
- inputtensor
  - ElementClass, 15
- ip
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- istcauchy
  - MaterialClass, 23
- istdirichlet
  - MaterialClass, 23
- istgebietselement
  - ElementClass, 17
- istneumann
  - MaterialClass, 23
- istvolumen
  - MaterialClass, 23
- iteriereueberalleelemente
  - ElementClass, 15
- iteriereueberalleelementestaggered1
  - ElementClass, 16
- iteriereueberalleelementestaggered2
  - ElementClass, 16
- kaa
  - LokaleGekoppelteMatrizenClass, 28
- kab
  - LokaleGekoppelteMatrizenClass, 28
- kba
  - LokaleGekoppelteMatrizenClass, 28
- kbb
  - LokaleGekoppelteMatrizenClass, 28
- knotenfreiwerte
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- konduktivitaet
  - materialsammlung\_class::materialsammlung\_t, 47
- konduktivitaetensor
  - materialsammlung\_class::materialsammlung\_t, 47
- LokaleGekoppelteMatrizen\_t, 43
- LokaleGekoppelteMatrizenClass, 25
  - addiere, 26
  - caa, 27
  - cab, 27
  - cba, 27
  - cbb, 27
  - fa, 27
  - fb, 28
  - gibforce, 26
  - gibmass, 26
  - gibstiff, 26
  - initialisierematrizen, 27
  - kaa, 28
  - kab, 28
  - kba, 28
  - kbb, 28
  - nullifizierematrizen, 27
  - schuettleallematrizen, 27
  - setzeaufwert, 27
  - setzematrizen, 27
- LokaleGekoppelteMatrizenClass.f90, 59
- LokaleMatrizen\_t, 44
- LokaleMatrizenClass, 19
  - addiere, 20
  - gibforce, 20
  - gibmass, 20
  - gibstiff, 20
  - initmatrizen, 20
  - initmatrizenkurz, 20
  - nullifizierematrizen, 20
  - schuettleallematrizen, 20
  - setzeaufwert, 21
  - setzematrizen, 21
- lokalegekoppeltematrizen
  - Eriksson2006DGL\_Class, 31
- lokalegekoppeltematrizenclass, 44
- lokalegekoppeltematrizenclass::force, 40
- lokalegekoppeltematrizenclass::lokalegekoppeltematrizen\_t, 43
- lokalegekoppeltematrizenclass::mass, 45
- lokalegekoppeltematrizenclass::neu, 50
- lokalegekoppeltematrizenclass::nullifiziere, 51
- lokalegekoppeltematrizenclass::operator(+), 52
- lokalegekoppeltematrizenclass::schuettlematrizen, 54
- lokalegekoppeltematrizenclass::setze, 54
- lokalegekoppeltematrizenclass::stiff, 55
- lokalematrizenclass, 45
- lokalematrizenclass::force, 40
- lokalematrizenclass::lokalematrizen\_t, 44
- lokalematrizenclass::mass, 45
- lokalematrizenclass::neu, 49
- lokalematrizenclass::nullifiziere, 51
- lokalematrizenclass::operator(+), 52
- lokalematrizenclass::schuettlematrizen, 53
- lokalematrizenclass::setze, 54
- lokalematrizenclass::stiff, 55
- mass

- mystdmodules::localsystemmatrices\_t, 43
- material
  - ElementClass, 17
- MaterialClass, 22, 45
  - bezugswert, 24
  - cauchyrandbedingung, 24
  - dirichletrandbedingung, 24
  - gibrandbedingungcauchy, 23
  - gibrandbedingungdirichlet, 23
  - gibrandbedingungneumann, 23
  - gibrandbedingungvolumen, 23
  - istcauchy, 23
  - istdirichlet, 23
  - istneumann, 23
  - istvolumen, 23
  - neumannrandbedingung, 24
  - skalierfaktor, 24
  - volumenrandbedingung, 24
  - wert, 24
- MaterialSammlungClass, 48
- Materialeigenschaften.f90, 59
- Materialfunktionen.f90, 59
  - cdeliiski, 60
  - d\_t\_beech\_ihtp, 60
  - d\_t\_pine\_ihtp, 60
  - d\_w\_eriksson, 60
  - dwdh\_zurwitz, 60
  - eb, 60
  - emc\_zurwitz, 60
  - rh\_zurwitz, 60
- materialsammlung\_class, 46
- materialsammlung\_class::materialsammlung\_t, 46
  - dichte, 47
  - dichtetensor, 47
  - diffusionskoeffizient, 47
  - diffusionskoeffizienttensor, 47
  - dwdh, 47
  - e\_b, 47
  - konduktivitaet, 47
  - konduktivitaetensor, 47
  - r, 47
  - relativehumidity, 47
  - spezifischewaerme, 47
  - spezifischewaermetensor, 47
  - trockendichte, 47
  - trockendichtetensor, 47
- materialwerte
  - ElementClass, 17
  - Eriksson2006DGL\_Class, 31
- model
  - ElementClass, 17
- MyStdModules.f90, 61
- mystdmodules, 48
  - print1darray, 48
  - print2darray, 48
  - printstdinfo, 48
  - returnstdsolverparams, 48
  - rotate2indextensor, 48
  - rotate4indextensor, 48
  - rotateelasticitymatrix, 48
  - rotateelasticitymatrix2d, 49
  - rotateelasticitymatrix3d, 49
- mystdmodules::localsystemmatrices\_t, 42
  - force, 43
  - mass, 43
  - stiff, 43
- mystdmodules::printarray, 52
  - print1darray, 52
  - print2darray, 52
- mystdmodules::stdsolverparams\_t, 54
  - nonlineariter, 54
  - nonlineartol, 54
  - solverparams, 54
  - stdofs, 54
- n
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- naechsteselement
  - ElementClass, 17
- NeumannRandbedingung, 33
- NeumannRandbedingung::NeumannRandbedingung\_t, 50
- NeumannRandbedingungClass, 51
- neumannrandbedingung
  - MaterialClass, 24
- neumannrandbedingungclass, 51
- neumannrandbedingungclass::neumannrandbedingung\_t, 50
- nodes
  - geometrischeeigenschaften\_class::geometrischeeigenschaften\_t, 42
- nonlineariter
  - BauteilClass, 30
  - mystdmodules::stdsolverparams\_t, 54
- nonlineartol
  - BauteilClass, 30
  - mystdmodules::stdsolverparams\_t, 54
- nullifizierematrizen
  - LokaleGekoppelteMatrizenClass, 27
  - LokaleMatrizenClass, 20
- print1darray
  - mystdmodules, 48
  - mystdmodules::printarray, 52
- print2darray
  - mystdmodules, 48
  - mystdmodules::printarray, 52
- printstdinfo
  - mystdmodules, 48
- problemdim
  - BauteilClass, 30
- r
  - materialsammlung\_class::materialsammlung\_t, 47
- randbedingungclass, 53
- randbedingungclass::neu, 49

- randbedingungclass::randbedingung\_t, 53
- relativehumidity
  - materialsammlung\_class::materialsammlung\_t, 47
- returnstdsolverparams
  - mystdmodules, 48
- rh\_zurwitz
  - Materialfunktionen.f90, 60
- rotate2indextensor
  - mystdmodules, 48
- rotate4indextensor
  - mystdmodules, 48
- rotateelasticitymatrix
  - mystdmodules, 48
- rotateelasticitymatrix2d
  - mystdmodules, 49
- rotateelasticitymatrix3d
  - mystdmodules, 49
  
- schuettleallematrizen
  - LokaleGekoppelteMatrizenClass, 27
  - LokaleMatrizenClass, 20
- setzeaufwert
  - LokaleGekoppelteMatrizenClass, 27
  - LokaleMatrizenClass, 21
- setzeelementzaehler
  - ElementClass, 16
- setzematrizen
  - LokaleGekoppelteMatrizenClass, 27
  - LokaleMatrizenClass, 21
- setzepointeraufnaechsteselement
  - ElementClass, 16
- skalierfaktor
  - MaterialClass, 24
- solver
  - BauteilClass, 30
  - ElementClass, 17
- solverparams
  - mystdmodules::stdsolverparams\_t, 54
- spezifischewaerme
  - materialsammlung\_class::materialsammlung\_t, 47
- spezifischewaermetensor
  - materialsammlung\_class::materialsammlung\_t, 47
- stdofs
  - mystdmodules::stdsolverparams\_t, 54
- stiff
  - mystdmodules::localsystemmatrices\_t, 43
  
- temperaturrandbedingung
  - ElementClass, 17
  - Eriksson2006DGL\_Class, 31
- trockendichte
  - materialsammlung\_class::materialsammlung\_t, 47
- trockendichtetensor
  - materialsammlung\_class::materialsammlung\_t, 47
  
- VolumenRandbedingung, 33
- VolumenRandbedingung::VolumenRandbedingung\_t, 55
- volumenrandbedingung
  - MaterialClass, 24
  - volumenrandbedingungclass, 56
  - volumenrandbedingungclass::volumenrandbedingung\_t, 55
  - vorhergehendeloesung
    - BauteilClass, 30
  - vorherigelsg
    - ElementClass, 17
- wert
  - MaterialClass, 24