

FINEL

Generated by Doxygen 1.8.12



# Contents

<b>1</b>	<b>Modules Index</b>	<b>1</b>
1.1	Modules List . . . . .	1
<b>2</b>	<b>Data Type Index</b>	<b>3</b>
2.1	Data Types List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Module Documentation</b>	<b>7</b>
4.1	meshstructure Module Reference . . . . .	7
4.1.1	Detailed Description . . . . .	7
4.1.2	Function/Subroutine Documentation . . . . .	7
4.1.2.1	mallocelem() . . . . .	7
4.1.2.2	mallocnodes() . . . . .	8
4.2	minputreader Module Reference . . . . .	8
4.2.1	Detailed Description . . . . .	9
4.2.2	Function/Subroutine Documentation . . . . .	9
4.2.2.1	analyzefile() . . . . .	9
4.2.2.2	analyzefileinput() . . . . .	10
4.2.2.3	createsimpleinputfile() . . . . .	10
4.2.2.4	findinclude() . . . . .	11
4.2.2.5	findkeyword() . . . . .	11
4.2.2.6	mergeincludecontents() . . . . .	12
4.2.2.7	preparefilelines() . . . . .	13

4.2.2.8	<a href="#">readinputfileds()</a>	13
4.2.2.9	<a href="#">readintarraykeywordvalue()</a>	14
4.2.2.10	<a href="#">readintegerkeywordvalue()</a>	15
4.2.2.11	<a href="#">readrealkeywordvalue()</a>	15
4.2.2.12	<a href="#">readstringkeywordvalue()</a>	16
4.2.3	<a href="#">Variable Documentation</a>	16
4.2.3.1	<a href="#">file_lines</a>	16
4.2.3.2	<a href="#">number_of_lines</a>	16
4.3	<a href="#">mprocessor Module Reference</a>	16
4.3.1	<a href="#">Detailed Description</a>	17
4.3.2	<a href="#">Function/Subroutine Documentation</a>	17
4.3.2.1	<a href="#">applybc()</a>	17
4.3.2.2	<a href="#">assmb()</a>	17
4.3.2.3	<a href="#">drchlt()</a>	18
4.3.2.4	<a href="#">formkf()</a>	19
4.4	<a href="#">mscalar Module Reference</a>	20
4.4.1	<a href="#">Detailed Description</a>	20
4.4.2	<a href="#">Function/Subroutine Documentation</a>	20
4.4.2.1	<a href="#">localelem()</a>	20
4.5	<a href="#">msetup Module Reference</a>	21
4.5.1	<a href="#">Detailed Description</a>	21
4.5.2	<a href="#">Function/Subroutine Documentation</a>	21
4.5.2.1	<a href="#">read_elems()</a>	21
4.5.2.2	<a href="#">read_nodes()</a>	22
4.6	<a href="#">mshapefunctions Module Reference</a>	23
4.6.1	<a href="#">Detailed Description</a>	24
4.6.2	<a href="#">Function/Subroutine Documentation</a>	24
4.6.2.1	<a href="#">setint()</a>	24
4.6.2.2	<a href="#">shpf1d()</a>	24
4.6.3	<a href="#">Variable Documentation</a>	25

4.6.3.1	w	25
4.6.3.2	xi	25
4.7	mutilities Module Reference	25
4.7.1	Detailed Description	25
4.7.2	Function/Subroutine Documentation	26
4.7.2.1	f1()	26
4.7.2.2	linspace()	26
4.7.2.3	print_matrix()	26
4.7.2.4	quad1()	27
4.7.2.5	test_shpf1d()	28
4.8	scalarstructure Module Reference	28
4.8.1	Detailed Description	28
4.8.2	Function/Subroutine Documentation	29
4.8.2.1	mallocelemkf()	29
4.8.2.2	mallocglobalkf()	29
<b>5</b>	<b>Data Type Documentation</b>	<b>31</b>
5.1	meshstructure::mesh Type Reference	31
5.1.1	Detailed Description	32
5.1.2	Member Data Documentation	32
5.1.2.1	ei	32
5.1.2.2	ej	33
5.1.2.3	ek	33
5.1.2.4	flagnode	33
5.1.2.5	gnode	33
5.1.2.6	mat	33
5.1.2.7	nelem	33
5.1.2.8	nelems	33
5.1.2.9	nen	33
5.1.2.10	nintp	34
5.1.2.11	nnodes	34

5.1.2.12	nsd . . . . .	34
5.1.2.13	numat . . . . .	34
5.1.2.14	si . . . . .	34
5.1.2.15	sj . . . . .	34
5.1.2.16	sk . . . . .	34
5.1.2.17	x . . . . .	34
5.1.2.18	xv . . . . .	35
5.1.2.19	y . . . . .	35
5.1.2.20	yv . . . . .	35
5.2	scalarstructure::scalarstructuresystem Type Reference . . . . .	35
5.2.1	Detailed Description . . . . .	36
5.2.2	Member Data Documentation . . . . .	36
5.2.2.1	lhelem . . . . .	36
5.2.2.2	lhsys . . . . .	36
5.2.2.3	rhelem . . . . .	36
5.2.2.4	rhsys . . . . .	36
5.2.2.5	u . . . . .	37
5.2.2.6	valbc . . . . .	37
<b>6</b>	<b>File Documentation</b>	<b>39</b>
6.1	src/driver.F90 File Reference . . . . .	39
6.1.1	Function/Subroutine Documentation . . . . .	39
6.1.1.1	finel() . . . . .	39
6.2	src/meshStructure.F90 File Reference . . . . .	40
6.3	src/mInputReader.F90 File Reference . . . . .	40
6.4	src/processor.F90 File Reference . . . . .	41
6.5	src/scalar.F90 File Reference . . . . .	41
6.6	src/scalarStructure.F90 File Reference . . . . .	41
6.7	src/setup.F90 File Reference . . . . .	42
6.8	src/shapeFunctions.F90 File Reference . . . . .	42
6.9	src/utilities.F90 File Reference . . . . .	43
<b>Index</b>		<b>45</b>

# Chapter 1

## Modules Index

### 1.1 Modules List

Here is a list of all modules with brief descriptions:

<a href="#">meshstructure</a>	Module that contains the data structure of a mesh associate to a problem . . . . .	7
<a href="#">minputreader</a>	Modulo responsavel por reunir subrotinas para leitura do arquivo de entrada . . . . .	8
<a href="#">mprocessor</a>	Processor module to compute, assemble and solve the system . . . . .	16
<a href="#">mscalar</a>	Contains variables and subroutine related to a general scalar problem . . . . .	20
<a href="#">msetup</a>	Module for setup phase by IO procedures . . . . .	21
<a href="#">mshapefunctions</a>	Module for shape functions computations and relate operations . . . . .	23
<a href="#">mutilities</a>	Module for auxiliar routines . . . . .	25
<a href="#">scalarstructure</a>	Module that contains the data structure of a general scalar problem . . . . .	28





## Chapter 2

# Data Type Index

### 2.1 Data Types List

Here are the data types with brief descriptions:

<a href="#">meshstructure::mesh</a>	
Data type for a mesh . . . . .	31
<a href="#">scalarstructure::scalarstructuresystem</a>	
Variables and characteristic data for a scalar problem . . . . .	35



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

src/ <a href="#">driver.F90</a> . . . . .	39
src/ <a href="#">meshStructure.F90</a> . . . . .	40
src/ <a href="#">mInputReader.F90</a> . . . . .	40
src/ <a href="#">processor.F90</a> . . . . .	41
src/ <a href="#">scalar.F90</a> . . . . .	41
src/ <a href="#">scalarStructure.F90</a> . . . . .	41
src/ <a href="#">setup.F90</a> . . . . .	42
src/ <a href="#">shapeFunctions.F90</a> . . . . .	42
src/ <a href="#">utilities.F90</a> . . . . .	43



## Chapter 4

# Module Documentation

### 4.1 meshstructure Module Reference

Module that contains the data structure of a mesh associate to a problem.

#### Data Types

- type `mesh`  
*Data type for a mesh.*

#### Functions/Subroutines

- subroutine `mallocnodes` (meshStrct, n)  
*Routine that allocate memory to node data.*
- subroutine `mallocelem` (meshStrct, n)  
*Routine that allocate memory to element data.*

#### 4.1.1 Detailed Description

Module that contains the data structure of a mesh associate to a problem.

#### Author

Diego T. Volpatto

#### 4.1.2 Function/Subroutine Documentation

##### 4.1.2.1 mallocelem()

```
subroutine meshstructure::mallocelem (  
    type(mesh) meshStrct,  
    integer n )
```

Routine that allocate memory to element data.

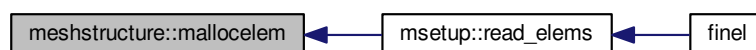
## Parameters

<i>meshStrct</i>	[in/out] mesh structure to allocate
<i>n</i>	[in] number of elements

## Author

Diego T. Volpatto

Here is the caller graph for this function:



## 4.1.2.2 mallocnodes()

```

subroutine meshstructure::mallocnodes (
    type(mesh) meshStrct,
    integer n )
  
```

Routine that allocate memory to node data.

## Parameters

<i>meshStrct</i>	[in/out] mesh structure to allocate
<i>n</i>	[in] number of nodes

## Author

Diego T. Volpatto

Here is the caller graph for this function:



## 4.2 minputreader Module Reference

Modulo responsavel por reunir subrotinas para leitura do arquivo de entrada.

## Functions/Subroutines

- subroutine `readinputfileds` ()  
*Le arquivo de input e armazena seu conteudo em um array.*
- subroutine `createsimpleinputfile` ()  
*Cria a estrutura de input usando um arquivo de entrada sem includes.*
- subroutine `mergeincludecontents` (include\_file, include\_line)  
*Le o conteudo do arquivo de include e armazena no array principal.*
- subroutine `preparefilelines` (include\_indexes, include\_number\_of\_lines, number\_of\_includes, original\_file\_lines)  
*Efetua a alocação da estrutura definitiva, preparando a linha dos arquivos originais para receber os includes.*
- subroutine `analyzefileinput` (number\_of\_lines, number\_of\_includes)  
*Efetua algumas análises no arquivo recebido.*
- subroutine `analyzefile` (file\_name, number\_of\_lines, number\_of\_includes)  
*Efetua algumas análises no arquivo recebido.*
- integer \*4 function `findinclude` (position, file\_lines, number\_of\_lines)  
*Procura a n-esima palavra-chave include.*
- integer \*4 function `findkeyword` (keyword)  
*Procura uma palavra-chave.*
- subroutine `readintegerkeywordvalue` (keyword, target, default\_value)  
*Efetua a leitura de uma palavra-chave to tipo inteiro. Se nao encontrado, associa o valor default fornecido.*
- subroutine `readintarraykeywordvalue` (keyword, target, default\_value)  
*Efetua a leitura de uma palavra-chave do tipo array de inteiro. Se nao encontrado, associa o valor default fornecido.*  
*Obs.: Atentar para o fato de essa sub-rotina ter um do "infinito".*
- subroutine `readstringkeywordvalue` (keyword, target, default\_value)  
*Efetua a leitura de uma palavra-chave to tipo string. Se nao encontrado, associa o valor default fornecido.*
- subroutine `readrealkeywordvalue` (keyword, target, default\_value)  
*Efetua a leitura de uma palavra-chave to tipo real. Se nao encontrado, associa o valor default fornecido.*

## Variables

- character(len=200), dimension(:), allocatable `file_lines`  
*Armazena as linhas do arquivo de input.*
- integer \*4 `number_of_lines`  
*Armazena o numero de linhas no arquivo.*

### 4.2.1 Detailed Description

Modulo responsavel por reunir subrotinas para leitura do arquivo de entrada.

### 4.2.2 Function/Subroutine Documentation

#### 4.2.2.1 analyzefile()

```
subroutine minputreader::analyzefile (
    character(len=200) file_name,
    integer*4 number_of_lines,
    integer*4 number_of_includes )
```

Efetua algumas análises no arquivo recebido.

**Parameters**

<i>file_name</i>	O nome do arquivo.
<i>number_of_lines</i>	Numero de linhas.
<i>number_of_include</i>	Numero de ocorrencias da palavra include.

Here is the caller graph for this function:

**4.2.2.2 analyzefileinput()**

```

subroutine minputreader::analyzefileinput (
    integer*4 number_of_lines,
    integer*4 number_of_includes )
  
```

Efetua algumas analises no arquivo recebido.

**Parameters**

<i>number_of_lines</i>	Numero de linhas.
<i>number_of_include</i>	Numero de ocorrencias da palavra include.

Here is the caller graph for this function:

**4.2.2.3 createsimpleinputfile()**

```

subroutine minputreader::createsimpleinputfile ( )
  
```

Cria a estrutura de input usando um arquivo de entrada sem includes.



## Parameters

<i>file_name</i>	Nome do arquivo a ser lido.
------------------	-----------------------------

Here is the caller graph for this function:



## 4.2.2.4 findinclude()

```
integer*4 function minputreader::findinclude (
    integer*4 position,
    character(len=200), dimension(:) file_lines,
    integer*4 number_of_lines )
```

Procura a n-esima palavra-chave include.

## Parameters

<i>position</i>	Corresponde a posicao desejada.
<i>file_lines</i>	Linhas do arquivo.
<i>number_of_lines</i>	Numero de linhas atuais.

## Returns

O indice da palavra-chave no array que contem as linhas do arquivo de entrada.

Here is the caller graph for this function:



## 4.2.2.5 findkeyword()

```
integer*4 function minputreader::findkeyword (
    character(50) keyword )
```

Procura uma palavra-chave.

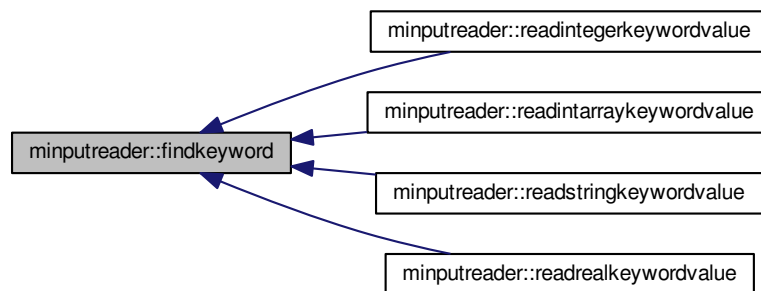
**Parameters**

<i>keyword</i>	A palavra-chave.
----------------	------------------

**Returns**

O indice da palavra-chave no array que contem as linhas do arquivo de entrada.

Here is the caller graph for this function:

**4.2.2.6 mergeincludecontents()**

```

subroutine minputreader::mergeincludecontents (
    character(len=200) include_file,
    integer*4 include_line )
  
```

Le o conteudo do arquivo de include e armazena no array principal.

**Parameters**

<i>include_index</i>	O index do include.
<i>include_files</i>	Array com includes.
<i>include_line</i>	A linha do include.

Here is the caller graph for this function:



## 4.2.2.7 preparefilelines()

```
subroutine minputreader::preparefilelines (
    integer*4, dimension(:) include_indexes,
    integer*4, dimension(:) include_number_of_lines,
    integer*4 number_of_includes,
    character(len=200), dimension(:) original_file_lines )
```

Efetua a alocação da estrutura definitiva, preparando a linha dos arquivos originais para receber os includes.

## Parameters

<i>include_indexes</i>	Array os índices de ocorrências dos includes.
<i>include_number_of_lines</i>	Array com o número de linhas de cada include
<i>number_of_includes</i>	Número de includes.
<i>original_file_lines</i>	Linhas do arquivo de entrada original.

Here is the caller graph for this function:



## 4.2.2.8 readinputfileds()

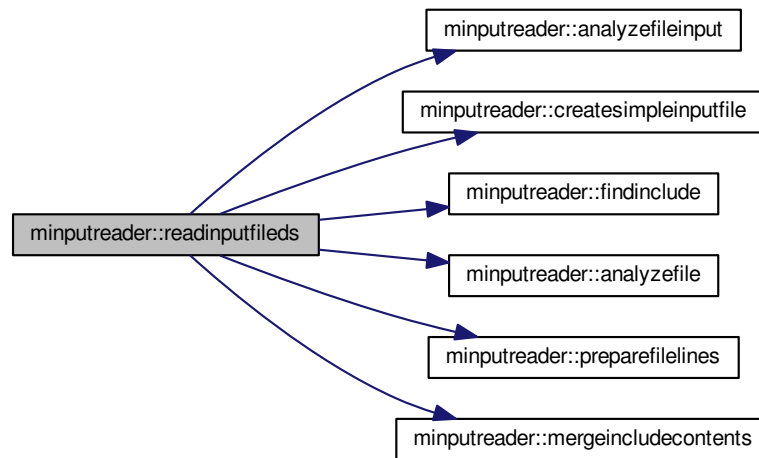
```
subroutine minputreader::readinputfileds ( )
```

Le arquivo de input e armazena seu conteúdo em um array.

## Parameters

<i>file_name</i>	Nome do arquivo a ser lido.
------------------	-----------------------------

Here is the call graph for this function:



#### 4.2.2.9 readintarraykeywordvalue()

```

subroutine minputreader::readintarraykeywordvalue (
    character(50) keyword,
    integer*4, dimension(:), allocatable target,
    integer*4 default_value )
  
```

Efetua a leitura de uma palavra-chave do tipo array de inteiro. Se nao encontrado, associa o valor default fornecido.  
 Obs.: Atentar para o fato de essa sub-rotina ter um do "infinito".

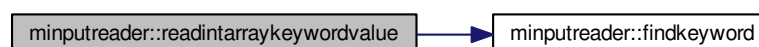
##### Parameters

<i>keyword</i>	A palavra-chave a ser encontrada.
<i>target</i>	Variavel onde o valor inteiro sera atribuido.
<i>default_value</i>	Valor default.

##### Author

Diego Volpatto

Here is the call graph for this function:



## 4.2.2.10 readintegerkeywordvalue()

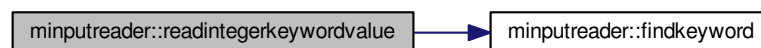
```
subroutine minputreader::readintegerkeywordvalue (
    character(50) keyword,
    target,
    integer*4, target default_value )
```

Efetua a leitura de uma palavra-chave to tipo inteiro. Se nao encontrado, associa o valor default fornecido.

## Parameters

<i>keyword</i>	A palavra-chave a ser encontrada.
<i>target</i>	Variavel onde o valor inteiro sera atribuido.
<i>default_value</i>	Valor default.

Here is the call graph for this function:



## 4.2.2.11 readrealkeywordvalue()

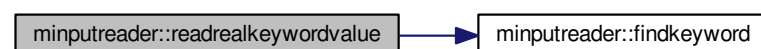
```
subroutine minputreader::readrealkeywordvalue (
    character(50) keyword,
    target,
    real(8), target default_value )
```

Efetua a leitura de uma palavra-chave to tipo real. Se nao encontrado, associa o valor default fornecido.

## Parameters

<i>keyword</i>	A palavra-chave a ser encontrada.
<i>target</i>	Variavel onde o real sera atribuido.
<i>default_value</i>	Valor default.

Here is the call graph for this function:



#### 4.2.2.12 readstringkeywordvalue()

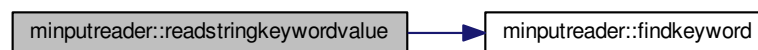
```
subroutine minputreader::readstringkeywordvalue (
    character(50) keyword,
    character(len=*) target,
    character(len=*) default_value )
```

Efetua a leitura de uma palavra-chave to tipo string. Se nao encontrado, associa o valor default fornecido.

##### Parameters

<i>keyword</i>	A palavra-chave a ser encontrada.
<i>target</i>	Variavel onde a string sera atribuido.
<i>default_value</i>	Valor default.

Here is the call graph for this function:



### 4.2.3 Variable Documentation

#### 4.2.3.1 file\_lines

```
character(len=200), dimension(:), allocatable minputreader::file_lines
```

Armazena as linhas do arquivo de input.

#### 4.2.3.2 number\_of\_lines

```
integer*4 minputreader::number_of_lines
```

Armazena o numero de linhas no arquivo.

## 4.3 mprocessor Module Reference

Processor module to compute, assemble and solve the system.

## Functions/Subroutines

- subroutine `formkf` (mesh\_, scalar\_)  
*Form and assemble  $Ku = F$  system.*
- subroutine `assmb` (mesh\_, scalar\_, nel)  
*Assemble element stiffness matrix and load vector to global stiffness matrix and load vector, respectively.*
- subroutine `drchlt` (mesh\_, scalar\_, n, val)  
*Apply Dirichlet Boundary Condition.*
- subroutine `applybc` (mesh\_, scalar\_)

### 4.3.1 Detailed Description

Processor module to compute, assemble and solve the system.

#### Author

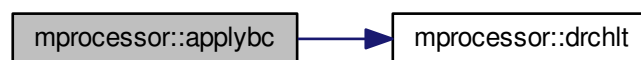
Diego T. Volpatto

### 4.3.2 Function/Subroutine Documentation

#### 4.3.2.1 `applybc()`

```
subroutine mprocessor::applybc (
    type(mesh) mesh_,
    type(scalarstructuresystem) scalar_ )
```

Here is the call graph for this function:



#### 4.3.2.2 `assmb()`

```
subroutine mprocessor::assmb (
    type(mesh) mesh_,
    type(scalarstructuresystem) scalar_,
    integer nel )
```

Assemble element stiffness matrix and load vector to global stiffness matrix and load vector, respectively.

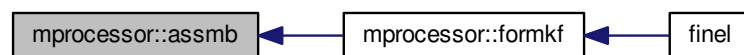
## Parameters

<i>mesh</i> ↔ —	A mesh structure
<i>scalar</i> ↔ —	A scalar structure
<i>nel</i>	Index of current element

## Author

Diego Volpatto

Here is the caller graph for this function:



## 4.3.2.3 drchlt()

```

subroutine mprocessor::drchlt (
    type(mesh) mesh_,
    type(scalarstructuresystem) scalar_,
    integer n,
    real*8 val )
  
```

Apply Dirichlet Boundary Condition.

## Parameters

<i>mesh</i> ↔ —	A mesh structure
<i>scalar</i> ↔ —	A scalar structure
<i>n</i>	Node index of BC
<i>val</i>	Value of BC



**Author**

Diego Volpatto

Here is the caller graph for this function:

**4.3.2.4 formkf()**

```

subroutine mprocessor::formkf (
    type(mesh) mesh_,
    type(scalarstructuresystem) scalar_ )
  
```

Form and assemble  $Ku = F$  system.

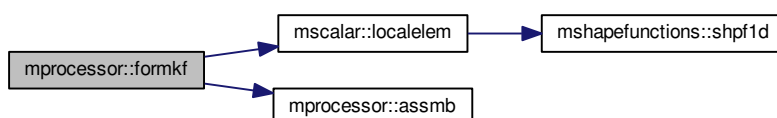
**Parameters**

<i>mesh</i> ↔ —	[in/out] A mesh structure
<i>scalar</i> ↔ —	[in/out] A scalar structure

**Author**

Diego Volpatto

Here is the call graph for this function:



Here is the caller graph for this function:



## 4.4 mscalar Module Reference

Contains variables and subroutine related to a general scalar problem.

### Functions/Subroutines

- subroutine [localelem](#) (mesh\_, scalar\_, nel)  
*Computes a master element contribution – 1D.*

#### 4.4.1 Detailed Description

Contains variables and subroutine related to a general scalar problem.

##### Author

Diego T. Volpatto

#### 4.4.2 Function/Subroutine Documentation

##### 4.4.2.1 localelem()

```

subroutine mscalar::localelem (
    type(mesh) mesh_,
    type(scalarstructuresystem) scalar_,
    integer nel )
  
```

Computes a master element contribution – 1D.

##### Parameters

<i>mesh</i> ↔	[in/out] A mesh structure
<i>scalar</i> ↔	[in/out] A scalar structure
<i>nel</i>	[in] Index of current element

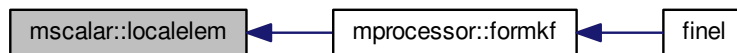
**Author**

Diego Volpatto

Here is the call graph for this function:



Here is the caller graph for this function:



## 4.5 msetup Module Reference

Module for setup phase by IO procedures.

**Functions/Subroutines**

- subroutine `read_nodes` (`mesh_`)  
*Subroutine to read node data file generated by EasyMesh.*
- subroutine `read_elems` (`mesh_`)  
*Subroutine to read element data file generated by EasyMesh.*

### 4.5.1 Detailed Description

Module for setup phase by IO procedures.

**Author**

Diego T. Volpatto

### 4.5.2 Function/Subroutine Documentation

#### 4.5.2.1 `read_elems()`

```

subroutine msetup::read_elems (
    type(mesh) mesh_ )
  
```

Subroutine to read element data file generated by EasyMesh.

**Parameters**

<i>mesh</i> ↔	[in/out] a mesh structure
—	

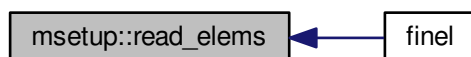
**Author**

Diego Volpatto

Here is the call graph for this function:



Here is the caller graph for this function:

**4.5.2.2 read\_nodes()**

```

subroutine msetup::read_nodes (
    type(mesh) mesh_ )
  
```

Subroutine to read node data file generated by EasyMesh.

**Parameters**

<i>mesh</i> ↔	[in/out] a mesh structure
—	

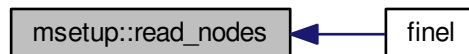
**Author**

Diego Volpatto

Here is the call graph for this function:



Here is the caller graph for this function:



## 4.6 mshapefunctions Module Reference

Module for shape functions computations and relate operations.

### Functions/Subroutines

- subroutine [setint](#)  
*Gauss quadrature data set routine.*
- subroutine [shpf1d](#) (xl, n, psi, dpsi)  
*Calculates the values of the shape functions and their derivatives.*

### Variables

- real \*8, dimension(4, 4) [xi](#)  
*Gauss point integration.*
- real \*8, dimension(4, 4) [w](#)  
*Gauss weights.*

### 4.6.1 Detailed Description

Module for shape functions computations and relate operations.

Author

Diego T. Volpatto

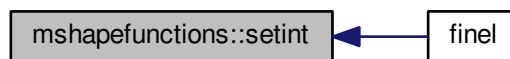
### 4.6.2 Function/Subroutine Documentation

#### 4.6.2.1 `setint()`

```
subroutine mshapefunctions::setint ( )
```

Gauss quadrature data set routine.

Here is the caller graph for this function:



#### 4.6.2.2 `shpf1d()`

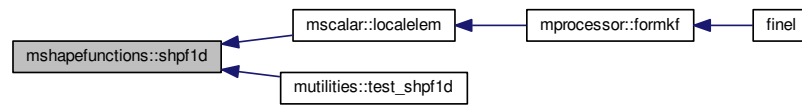
```
subroutine mshapefunctions::shpf1d (
    real*8 xl,
    integer n,
    real*8, dimension(n) psi,
    real*8, dimension(n) dpsi )
```

Calculates the values of the shape functions and their derivatives.

Parameters

<i>xl</i>	[in] specified value of master element coord
<i>n</i>	[in] number of element nodes
<i>psi</i>	[out] shape function values
<i>dpsi</i>	[out] derivatives shape functions values

Here is the caller graph for this function:



### 4.6.3 Variable Documentation

#### 4.6.3.1 w

`real*8, dimension(4,4) mshapefunctions::w`

Gauss weights.

#### 4.6.3.2 xi

`real*8, dimension(4,4) mshapefunctions::xi`

Gauss point integration.

## 4.7 mutilities Module Reference

Module for auxiliar routines.

### Functions/Subroutines

- subroutine `linspace` (`x1`, `x2`, `nintv`, `x`)  
*Generate points between `x1` and `x2` equally spaced in `x(i)`. Same idea of numpy subroutine.*
- real \*8 function `f1` (`x`)  
*A function to test purpose.*
- subroutine `quad1` (`n`, `x1`, `x2`)  
*Subroutine that computes gaussian quadrature of `f1`.*
- subroutine `test_shpf1d` (`n`, `nelem`, `x`)  
*Check if `shpf1d` works properly.*
- subroutine `print_matrix` (`A`, `n`, `m`)  
*Prints in the screen a matrix `A(n,m)`*

### 4.7.1 Detailed Description

Module for auxiliar routines.

Author

Diego T. Volpatto

## 4.7.2 Function/Subroutine Documentation

### 4.7.2.1 f1()

```
real*8 function mutilities::f1 (
    real*8 x )
```

A function to test purpose.

#### Parameters

<i>x</i>	input coordinate
----------	------------------

Here is the caller graph for this function:



### 4.7.2.2 linspace()

```
subroutine mutilities::linspace (
    real*8 x1,
    real*8 x2,
    integer nintv,
    real*8, dimension(:), allocatable x )
```

Generate points between `x1` and `x2` equally spaced in `x(i)`. Same idea of numpy subroutine.

#### Parameters

<i>x1</i>	interval lower bound
<i>x2</i>	interval upper bound
<i>nintv</i>	num of intervals
<i>x</i>	vector to assemble the values

### 4.7.2.3 print\_matrix()

```
subroutine mutilities::print_matrix (
    real*8, dimension(n,m) A,
    integer n,
    integer m )
```

Prints in the screen a matrix `A(n,m)`



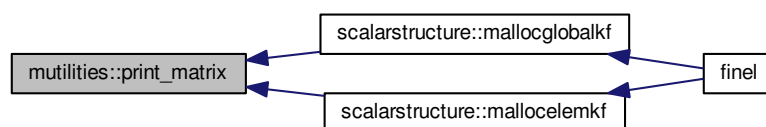
## Parameters

$A$	A matrix
$n$	Number of lines of $A$
$m$	Number of columns of $A$

## Author

Diego Volpatto

Here is the caller graph for this function:



## 4.7.2.4 quad1()

```

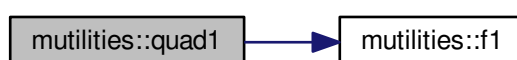
subroutine mutilities::quad1 (
    integer n,
    real*8 x1,
    real*8 x2 )
  
```

Subroutine that computes gaussian quadrature of f1.

## Parameters

$n$	quadrature order
$x1$	integral lower bound
$x2$	integral upper bound

Here is the call graph for this function:



#### 4.7.2.5 test\_shpf1d()

```
subroutine mutilities::test_shpf1d (
    integer n,
    integer nelem,
    real*8, dimension(nelem+1) x )
```

Check if shpf1d works properly.

##### Parameters

<i>n</i>	element node numbers
<i>nelem</i>	num of discrete intervals
<i>x</i>	master element's coordinates

Here is the call graph for this function:



## 4.8 scalarstructure Module Reference

Module that contains the data structure of a general scalar problem.

### Data Types

- type [scalarstructuresystem](#)  
*Variables and characteristic data for a scalar problem.*

### Functions/Subroutines

- subroutine [mallocglobalkf](#) (scalar\_, n)  
*Routine to allocate and clear the  $Ku = F$  system.*
- subroutine [mallocelemkf](#) (scalar\_, n)  
*Routine to allocate and clear the element  $KF$ .*

#### 4.8.1 Detailed Description

Module that contains the data structure of a general scalar problem.

##### Author

Diego T. Volpatto

## 4.8.2 Function/Subroutine Documentation

### 4.8.2.1 mallocelemkf()

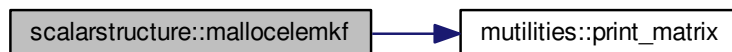
```
subroutine scalarstructure::mallocelemkf (
    type(scalarstructuresystem) scalar_,
    integer n )
```

Routine to allocate and clear the element KF.

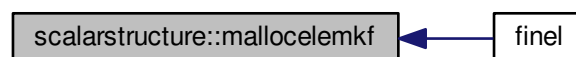
#### Parameters

<i>scalar</i> ↔	[in/out] A general scalar structure
—	
<i>n</i>	[in] Number of element nodes

Here is the call graph for this function:



Here is the caller graph for this function:



### 4.8.2.2 mallocglobalkf()

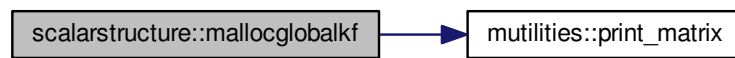
```
subroutine scalarstructure::mallocglobalkf (
    type(scalarstructuresystem) scalar_,
    integer n )
```

Routine to allocate and clear the  $Ku = F$  system.

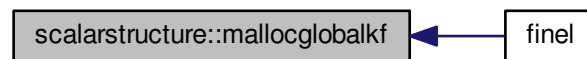
#### Parameters

<i>scalar</i> ↔	[in/out] A general scalar structure
—	
<i>n</i>	[in] Number of global nodes

Here is the call graph for this function:



Here is the caller graph for this function:



## Chapter 5

# Data Type Documentation

### 5.1 meshstructure::mesh Type Reference

Data type for a mesh.

Collaboration diagram for meshstructure::mesh:

meshstructure::mesh
<div><div><div><div><div><div></div><div>+ numat</div></div></div><div><div><div></div><div>+ nsd</div></div></div><div><div><div></div><div>+ nintp</div></div></div><div><div><div></div><div>+ nnodes</div></div></div><div><div><div></div><div>+ nelems</div></div></div><div><div><div></div><div>+ nen</div></div></div><div><div><div></div><div>+ x</div></div></div><div><div><div></div><div>+ y</div></div></div><div><div><div></div><div>+ flagnode</div></div></div><div><div><div></div><div>+ nelem</div></div></div><div><div><div></div><div>and 10 more...</div></div></div></div></div></div>

#### Public Attributes

- integer [numat](#)  
*Number of materials.*
- integer [nsd](#)  
*Number of spatial.*
- integer [nintp](#)  
*Number of integration points.*

- integer [nnodes](#)  
*Number of nodes.*
- integer [nelems](#)  
*Number of elements.*
- integer [nen](#)  
*Number of element's nodes.*
- real \*8, dimension(:), allocatable [x](#)  
*x coordinates nodes*
- real \*8, dimension(:), allocatable [y](#)  
*y coordinates nodes*
- integer \*4, dimension(:), allocatable [flagnode](#)  
*boundary flag*
- integer [nelem](#)  
*Number of elements.*
- real \*8, dimension(:), allocatable [xv](#)  
*Circumcenter Elem xcoord.*
- real \*8, dimension(:), allocatable [yv](#)  
*Circumcenter Elem ycoord.*
- integer \*4, dimension(:, :), allocatable [gnode](#)  
*Global node.*
- integer \*4, dimension(:), allocatable [mat](#)  
*Element material kind.*
- integer \*4, dimension(:), allocatable [ei](#)  
*i-opposite element*
- integer \*4, dimension(:), allocatable [ej](#)  
*j-opposite element*
- integer \*4, dimension(:), allocatable [ek](#)  
*k-opposite element*
- integer \*4, dimension(:), allocatable [si](#)  
*Opposite i-side.*
- integer \*4, dimension(:), allocatable [sj](#)  
*Opposite j-side.*
- integer \*4, dimension(:), allocatable [sk](#)  
*Opposite k-side.*

### 5.1.1 Detailed Description

Data type for a mesh.

### 5.1.2 Member Data Documentation

#### 5.1.2.1 [ei](#)

```
integer*4, dimension(:), allocatable meshstructure::mesh::ei
```

i-opposite element

#### 5.1.2.2 ej

`integer*4, dimension(:), allocatable meshstructure::mesh::ej`

j-opposite element

#### 5.1.2.3 ek

`integer*4, dimension(:), allocatable meshstructure::mesh::ek`

k-opposite element

#### 5.1.2.4 flagnode

`integer*4, dimension(:), allocatable meshstructure::mesh::flagnode`

boundary flag

#### 5.1.2.5 gnode

`integer*4, dimension(:, :), allocatable meshstructure::mesh::gnode`

Global node.

#### 5.1.2.6 mat

`integer*4, dimension(:), allocatable meshstructure::mesh::mat`

Element material kind.

#### 5.1.2.7 nelem

`integer meshstructure::mesh::nelem`

Number of elements.

#### 5.1.2.8 nelems

`integer meshstructure::mesh::nelems`

Number of elements.

#### 5.1.2.9 nen

`integer meshstructure::mesh::nen`

Number of element's nodes.

#### 5.1.2.10 nintp

```
integer meshstructure::mesh::nintp
```

Number of integration points.

#### 5.1.2.11 nnodes

```
integer meshstructure::mesh::nnodes
```

Number of nodes.

#### 5.1.2.12 nsd

```
integer meshstructure::mesh::nsd
```

Number of spatial.

#### 5.1.2.13 numat

```
integer meshstructure::mesh::numat
```

Number of materials.

#### 5.1.2.14 si

```
integer*4, dimension(:), allocatable meshstructure::mesh::si
```

Opposite i-side.

#### 5.1.2.15 sj

```
integer*4, dimension(:), allocatable meshstructure::mesh::sj
```

Opposite j-side.

#### 5.1.2.16 sk

```
integer*4, dimension(:), allocatable meshstructure::mesh::sk
```

Opposite k-side.

#### 5.1.2.17 x

```
real*8, dimension(:), allocatable meshstructure::mesh::x
```

x coordinates nodes



## 5.1.2.18 xv

```
real*8, dimension(:), allocatable meshstructure::mesh::xv
```

Circumcenter Elem xcoor.

## 5.1.2.19 y

```
real*8, dimension(:), allocatable meshstructure::mesh::y
```

y coordinates nodes

## 5.1.2.20 yv

```
real*8, dimension(:), allocatable meshstructure::mesh::yv
```

Circumcenter Elem ycoor.

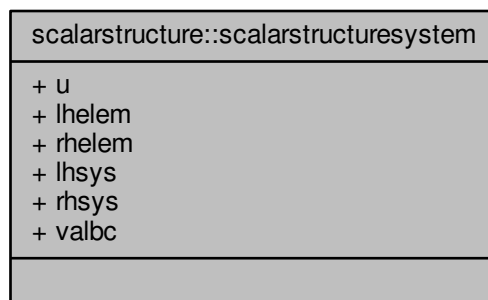
The documentation for this type was generated from the following file:

- [src/meshStructure.F90](#)

## 5.2 scalarstructure::scalarstructuresystem Type Reference

Variables and characteristic data for a scalar problem.

Collaboration diagram for scalarstructure::scalarstructuresystem:



## Public Attributes

- `real *8, dimension(:), allocatable u`  
*Solution vector.*
- `real *8, dimension(:, :), allocatable lhelem`  
*Element left-hand system.*
- `real *8, dimension(:), allocatable rhelem`  
*Element right-hand system.*
- `real *8, dimension(:, :), allocatable lhsys`  
*Global left-hand system.*
- `real *8, dimension(:), allocatable rhsys`  
*Global right-hand system.*
- `real *8, dimension(:), allocatable valbc`  
*Dirichlet BC values.*

### 5.2.1 Detailed Description

Variables and characteristic data for a scalar problem.

### 5.2.2 Member Data Documentation

#### 5.2.2.1 lhelem

```
real*8, dimension(:, :), allocatable scalarstructure::scalarstructuresystem::lhelem
```

Element left-hand system.

#### 5.2.2.2 lhsys

```
real*8, dimension(:, :), allocatable scalarstructure::scalarstructuresystem::lhsys
```

Global left-hand system.

#### 5.2.2.3 rhelem

```
real*8, dimension(:), allocatable scalarstructure::scalarstructuresystem::rhelem
```

Element right-hand system.

#### 5.2.2.4 rhsys

```
real*8, dimension(:), allocatable scalarstructure::scalarstructuresystem::rhsys
```

Global right-hand system.

### 5.2.2.5 u

```
real*8, dimension(:), allocatable scalarstructure::scalarstructuresystem::u
```

Solution vector.

### 5.2.2.6 valbc

```
real*8, dimension(:), allocatable scalarstructure::scalarstructuresystem::valbc
```

Dirichlet BC values.

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

- [src/scalarStructure.F90](#)



## Chapter 6

# File Documentation

### 6.1 src/driver.F90 File Reference

#### Functions/Subroutines

- program [finel](#)

*A FINite ELeMent program for general purpose problems. The present is based in the book "Finite Elements: An Introduction" wrote by Eric Becker, Graham Carey and Tinsley Oden.*

#### 6.1.1 Function/Subroutine Documentation

##### 6.1.1.1 finel()

```
program finel ( )
```

A FINite ELeMent program for general purpose problems. The present is based in the book "Finite Elements: An Introduction" wrote by Eric Becker, Graham Carey and Tinsley Oden.

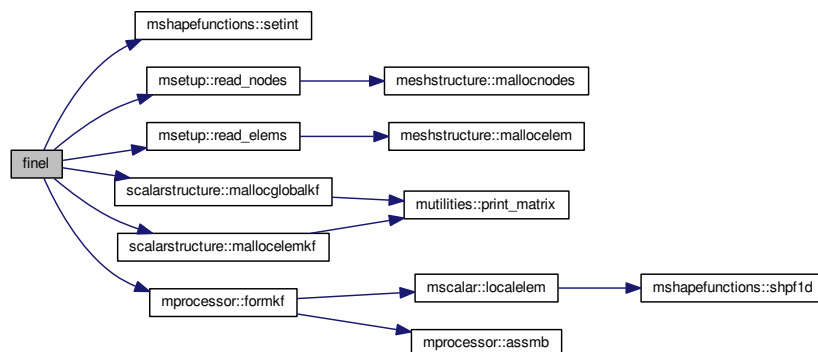
Due to the evolution of Fortran programming language, the code developed here incorporate several changes comparing to the original given in the book cited before. Modular paradigm was employed, as well a little of derived data structure.

Implementations by Diego T. Volpatto. email: [volpatto@lncc.br](mailto:volpatto@lncc.br) or [dtvolpatto@gmail.com](mailto:dtvolpatto@gmail.com)

#### Author

Diego Tavares Volpatto

Here is the call graph for this function:



## 6.2 src/meshStructure.F90 File Reference

### Data Types

- type `meshstructure::mesh`  
*Data type for a mesh.*

### Modules

- module `meshstructure`  
*Module that contains the data structure of a mesh associate to a problem.*

### Functions/Subroutines

- subroutine `meshstructure::mallocnodes` (meshStrct, n)  
*Routine that allocate memory to node data.*
- subroutine `meshstructure::mallocelem` (meshStrct, n)  
*Routine that allocate memory to element data.*

## 6.3 src/minputReader.F90 File Reference

### Modules

- module `minputreader`  
*Modulo responsavel por reunir subrotinas para leitura do arquivo de entrada.*

### Functions/Subroutines

- subroutine `minputreader::readinputfileds` ()  
*Le arquivo de input e armazena seu conteudo em um array.*
- subroutine `minputreader::createsimpleinputfile` ()  
*Cria a estrutura de input usando um arquivo de entrada sem includes.*
- subroutine `minputreader::mergeincludecontents` (include\_file, include\_line)  
*Le o conteudo do arquivo de include e armazena no array principal.*
- subroutine `minputreader::preparefilelines` (include\_indexes, include\_number\_of\_lines, number\_of\_includes, original\_file\_lines)  
*Efetua a alocação da estrutura definitiva, preparando a linha dos arquivos originais para receber os includes.*
- subroutine `minputreader::analyzefileinput` (number\_of\_lines, number\_of\_includes)  
*Efetua algumas análises no arquivo recebido.*
- subroutine `minputreader::analyzefile` (file\_name, number\_of\_lines, number\_of\_includes)  
*Efetua algumas análises no arquivo recebido.*
- integer \*4 function `minputreader::findinclude` (position, file\_lines, number\_of\_lines)  
*Procura a n-esima palavra-chave include.*
- integer \*4 function `minputreader::findkeyword` (keyword)  
*Procura uma palavra-chave.*
- subroutine `minputreader::readintegerkeywordvalue` (keyword, target, default\_value)  
*Efetua a leitura de uma palavra-chave to tipo inteiro. Se nao encontrado, associa o valor default fornecido.*
- subroutine `minputreader::readintarraykeywordvalue` (keyword, target, default\_value)  
*Efetua a leitura de uma palavra-chave do tipo array de inteiro. Se nao encontrado, associa o valor default fornecido. Obs.: Atentar para o fato de essa sub-rotina ter um do "infinito".*
- subroutine `minputreader::readstringkeywordvalue` (keyword, target, default\_value)  
*Efetua a leitura de uma palavra-chave to tipo string. Se nao encontrado, associa o valor default fornecido.*
- subroutine `minputreader::readrealkeywordvalue` (keyword, target, default\_value)  
*Efetua a leitura de uma palavra-chave to tipo real. Se nao encontrado, associa o valor default fornecido.*

## Variables

- character(len=200), dimension(:), allocatable [minputreader::file\\_lines](#)  
*Armazena as linhas do arquivo de input.*
- integer \*4 [minputreader::number\\_of\\_lines](#)  
*Armazena o numero de linhas no arquivo.*

## 6.4 src/processor.F90 File Reference

### Modules

- module [mprocessor](#)  
*Processor module to compute, assemble and solve the system.*

### Functions/Subroutines

- subroutine [mprocessor::formkf](#) (mesh\_, scalar\_)  
*Form and assemble  $Ku = F$  system.*
- subroutine [mprocessor::assmb](#) (mesh\_, scalar\_, nel)  
*Assemble element stiffness matrix and load vector to global stiffness matrix and load vector, respectively.*
- subroutine [mprocessor::drchlt](#) (mesh\_, scalar\_, n, val)  
*Apply Dirichlet Boundary Condition.*
- subroutine [mprocessor::applybc](#) (mesh\_, scalar\_)

## 6.5 src/scalar.F90 File Reference

### Modules

- module [mscalar](#)  
*Contains variables and subroutine related to a general scalar problem.*

### Functions/Subroutines

- subroutine [mscalar::localelem](#) (mesh\_, scalar\_, nel)  
*Computes a master element contribution – 1D.*

## 6.6 src/scalarStructure.F90 File Reference

### Data Types

- type [scalarstructure::scalarstructuresystem](#)  
*Variables and characteristic data for a scalar problem.*

## Modules

- module [scalarstructure](#)  
*Module that contains the data structure of a general scalar problem.*

## Functions/Subroutines

- subroutine [scalarstructure::malloccglobalkf](#) (scalar\_, n)  
*Routine to allocate and clear the  $Ku = F$  system.*
- subroutine [scalarstructure::mallocelemkf](#) (scalar\_, n)  
*Routine to allocate and clear the element  $KF$ .*

## 6.7 src/setup.F90 File Reference

### Modules

- module [msetup](#)  
*Module for setup phase by IO procedures.*

### Functions/Subroutines

- subroutine [msetup::read\\_nodes](#) (mesh\_)  
*Subroutine to read node data file generated by EasyMesh.*
- subroutine [msetup::read\\_elems](#) (mesh\_)  
*Subroutine to read element data file generated by EasyMesh.*

## 6.8 src/shapeFunctions.F90 File Reference

### Modules

- module [mshapefunctions](#)  
*Module for shape functions computations and relate operations.*

### Functions/Subroutines

- subroutine [mshapefunctions::setint](#)  
*Gauss quadrature data set routine.*
- subroutine [mshapefunctions::shpf1d](#) (xl, n, psi, dpsi)  
*Calculates the values of the shape functions and their derivatives.*

### Variables

- real \*8, dimension(4, 4) [mshapefunctions::xi](#)  
*Gauss point integration.*
- real \*8, dimension(4, 4) [mshapefunctions::w](#)  
*Gauss weights.*



## 6.9 src/utilities.F90 File Reference

### Modules

- module `mutilities`  
*Module for auxiliar routines.*

### Functions/Subroutines

- subroutine `mutilities::linspace` ( $x1$ ,  $x2$ ,  $nintv$ ,  $x$ )  
*Generate points between  $x1$  and  $x2$  equally spaced in  $x(i)$ . Same idea of numpy subroutine.*
- real \*8 function `mutilities::f1` ( $x$ )  
*A function to test purpose.*
- subroutine `mutilities::quad1` ( $n$ ,  $x1$ ,  $x2$ )  
*Subroutine that computes gaussian quadrature of  $f1$ .*
- subroutine `mutilities::test_shpf1d` ( $n$ ,  $nelem$ ,  $x$ )  
*Check if `shpf1d` works properly.*
- subroutine `mutilities::print_matrix` ( $A$ ,  $n$ ,  $m$ )  
*Prints in the screen a matrix  $A(n,m)$*



# Index

analyzefile  
    minputreader, [9](#)  
analyzefileinput  
    minputreader, [10](#)  
applybc  
    mprocessor, [17](#)  
assmb  
    mprocessor, [17](#)  
  
createsimpleinputfile  
    minputreader, [10](#)  
  
drchlt  
    mprocessor, [18](#)  
driver.F90  
    finel, [39](#)  
  
ei  
    meshstructure::mesh, [32](#)  
ej  
    meshstructure::mesh, [32](#)  
ek  
    meshstructure::mesh, [33](#)  
  
f1  
    mutilities, [26](#)  
file\_lines  
    minputreader, [16](#)  
findinclude  
    minputreader, [11](#)  
findkeyword  
    minputreader, [11](#)  
finel  
    driver.F90, [39](#)  
flagnode  
    meshstructure::mesh, [33](#)  
formkf  
    mprocessor, [19](#)  
  
gnode  
    meshstructure::mesh, [33](#)  
  
lhelem  
    scalarstructure::scalarstructuresystem, [36](#)  
lhsys  
    scalarstructure::scalarstructuresystem, [36](#)  
linspace  
    mutilities, [26](#)  
localelem  
    mscalar, [20](#)  
  
mallocelem  
    meshstructure, [7](#)  
mallocelemkf  
    scalarstructure, [29](#)  
mallocglobalkf  
    scalarstructure, [29](#)  
mallocnodes  
    meshstructure, [8](#)  
mat  
    meshstructure::mesh, [33](#)  
mergeincludecontents  
    minputreader, [12](#)  
meshstructure, [7](#)  
    mallocelem, [7](#)  
    mallocnodes, [8](#)  
meshstructure::mesh, [31](#)  
    ei, [32](#)  
    ej, [32](#)  
    ek, [33](#)  
    flagnode, [33](#)  
    gnode, [33](#)  
    mat, [33](#)  
    nelem, [33](#)  
    nelems, [33](#)  
    nen, [33](#)  
    nintp, [33](#)  
    nnodes, [34](#)  
    nsd, [34](#)  
    numat, [34](#)  
    si, [34](#)  
    sj, [34](#)  
    sk, [34](#)  
    x, [34](#)  
    xv, [34](#)  
    y, [35](#)  
    yv, [35](#)  
minputreader, [8](#)  
    analyzefile, [9](#)  
    analyzefileinput, [10](#)  
    createsimpleinputfile, [10](#)  
    file\_lines, [16](#)  
    findinclude, [11](#)  
    findkeyword, [11](#)  
    mergeincludecontents, [12](#)  
    number\_of\_lines, [16](#)  
    preparefilelines, [12](#)  
    readinputfiles, [13](#)  
    readintarraykeywordvalue, [14](#)  
    readintegerkeywordvalue, [14](#)

- readrealkeywordvalue, [15](#)
- readstringkeywordvalue, [15](#)
- mprocessor, [16](#)
  - applybc, [17](#)
  - assmb, [17](#)
  - drchlt, [18](#)
  - formkf, [19](#)
- mscalar, [20](#)
  - localelem, [20](#)
- msetup, [21](#)
  - read\_elems, [21](#)
  - read\_nodes, [22](#)
- mshapefunctions, [23](#)
  - setint, [24](#)
  - shpf1d, [24](#)
  - w, [25](#)
  - xi, [25](#)
- mutilities, [25](#)
  - f1, [26](#)
  - linspace, [26](#)
  - print\_matrix, [26](#)
  - quad1, [27](#)
  - test\_shpf1d, [27](#)
- nelem
  - meshstructure::mesh, [33](#)
- nelems
  - meshstructure::mesh, [33](#)
- nen
  - meshstructure::mesh, [33](#)
- nintp
  - meshstructure::mesh, [33](#)
- nnodes
  - meshstructure::mesh, [34](#)
- nsd
  - meshstructure::mesh, [34](#)
- numat
  - meshstructure::mesh, [34](#)
- number\_of\_lines
  - minputreader, [16](#)
- preparefilelines
  - minputreader, [12](#)
- print\_matrix
  - mutilities, [26](#)
- quad1
  - mutilities, [27](#)
- read\_elems
  - msetup, [21](#)
- read\_nodes
  - msetup, [22](#)
- readinputfiles
  - minputreader, [13](#)
- readintarraykeywordvalue
  - minputreader, [14](#)
- readintegerkeywordvalue
  - minputreader, [14](#)
- readrealkeywordvalue
  - minputreader, [15](#)
- readstringkeywordvalue
  - minputreader, [15](#)
- rhelem
  - scalarstructure::scalarstructuresystem, [36](#)
- rhsys
  - scalarstructure::scalarstructuresystem, [36](#)
- scalarstructure, [28](#)
  - mallocelemb, [29](#)
  - malloglobalkf, [29](#)
- scalarstructure::scalarstructuresystem, [35](#)
  - lhelem, [36](#)
  - lhsys, [36](#)
  - rhelem, [36](#)
  - rhsys, [36](#)
  - u, [36](#)
  - valbc, [37](#)
- setint
  - mshapefunctions, [24](#)
- shpf1d
  - mshapefunctions, [24](#)
- si
  - meshstructure::mesh, [34](#)
- sj
  - meshstructure::mesh, [34](#)
- sk
  - meshstructure::mesh, [34](#)
- src/driver.F90, [39](#)
- src/mInputReader.F90, [40](#)
- src/meshStructure.F90, [40](#)
- src/processor.F90, [41](#)
- src/scalar.F90, [41](#)
- src/scalarStructure.F90, [41](#)
- src/setup.F90, [42](#)
- src/shapeFunctions.F90, [42](#)
- src/utilities.F90, [43](#)
- test\_shpf1d
  - mutilities, [27](#)
- u
  - scalarstructure::scalarstructuresystem, [36](#)
- valbc
  - scalarstructure::scalarstructuresystem, [37](#)
- w
  - mshapefunctions, [25](#)
- x
  - meshstructure::mesh, [34](#)
- xi
  - mshapefunctions, [25](#)
- xv
  - meshstructure::mesh, [34](#)
- y
  - meshstructure::mesh, [35](#)

yv

meshstructure::mesh, [35](#)