FullSWOF_2D

v1.02.00

Generated by Doxygen 1.7.4

Fri Jan 20 2012 11:29:56

Contents

Chapter 1

Class Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 bc_discharge_imp Class Reference

Class that computes boundary conditions where the discharge is imposed.

```
#include <bc_imp_discharge.hpp>
```

4.1.1 Detailed Description

Class that computes boundary conditions where the discharge is imposed.

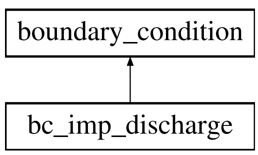
The documentation for this class was generated from the following file:

- Headers/libboundaryconditions/bc_imp_discharge.hpp

4.2 bc_imp_discharge Class Reference

```
#include <bc_imp_discharge.hpp>
```

Inheritance diagram for bc_imp_discharge:



Public Member Functions

- bc_imp_discharge (parameters &, TAB &, int, int)

    Constructor.

- SCALAR getValueOfPolynomial (const SCALAR, const SCALAR, const SCALAR, const SCALAR, const SCALAR, int, int) const
- SCALAR getValueofDerivativeOfPolynomial (const SCALAR, const SCALAR, const SCALAR, const SCALAR, int, int) const
- SCALAR NewtonSolver (const SCALAR, const SCALAR, const SCALAR, const SCALAR, const SCALAR, int, int) const
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int, int)

    Calculates boundary condition.
- virtual ∼bc_imp_discharge ()

    Destructor.

Protected Attributes

- SCALAR tol
- int maxiter

4.2.1 Detailed Description

Definition at line 61 of file bc_imp_discharge.hpp.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 bc_imp_discharge::bc_imp_discharge (parameters & *par*, TAB & *z*, int *n1*, int *n2*)

Constructor.

Definition at line 49 of file bc_imp_discharge.cpp.

4.2.2.2 bc_imp_discharge::∼bc_imp_discharge ( ) [virtual]

Destructor.

Definition at line 162 of file bc_imp_discharge.cpp.

4.2.3 Member Function Documentation

4.2.3.1 void bc_imp_discharge::calcul (SCALAR *hin*, SCALAR *unorm_in*, SCALAR *utan_in*, SCALAR *hfix*, SCALAR *qfix*, SCALAR *hin_oppbound*, SCALAR *unorm_in_oppbound*, SCALAR *utan_in_oppbound*, SCALAR *time*, int *n1*, int *n2*) [virtual]

Calculates boundary condition.

Implements boundary_condition.

Definition at line 125 of file bc_imp_discharge.cpp.

4.2.3.2 SCALAR bc_imp_discharge::getValueofDerivativeOfPolynomial (const SCALAR *hin*, const SCALAR *unorm_in*, const SCALAR *utan_in*, const SCALAR *h*, int *n1*, int *n2*) const

Definition at line 93 of file bc_imp_discharge.cpp.

4.2.3.3 SCALAR bc_imp_discharge::getValueOfPolynomial (const SCALAR *hin*, const SCALAR *unorm_in*, const SCALAR *utan_in*, const SCALAR *qfix*, const SCALAR *h*, int *n1*, int *n2*) const

Definition at line 85 of file bc_imp_discharge.cpp.

4.2.3.4 SCALAR bc_imp_discharge::NewtonSolver (const SCALAR *hin*, const SCALAR *unorm_in*, const SCALAR *utan_in*, const SCALAR *qfix*, const SCALAR *h_init*, int *n1*, int *n2*) const

Definition at line 101 of file bc_imp_discharge.cpp.

4.2.4 Member Data Documentation

4.2.4.1 int bc_imp_discharge::maxiter `[protected]`

Definition at line 65 of file bc_imp_discharge.hpp.

4.2.4.2 SCALAR bc_imp_discharge::tol `[protected]`

Definition at line 64 of file bc_imp_discharge.hpp.

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

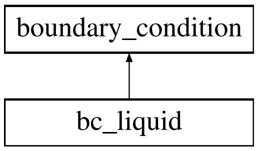
- Headers/libboundaryconditions/bc_imp_discharge.hpp
- Sources/libboundaryconditions/bc_imp_discharge.cpp

4.3 bc_liquid Class Reference

Class which allows to calculate right liquid boundary thanks to the method of caracteristics.

`#include <bc_liquid.hpp>`

Inheritance diagram for bc_liquid:



Public Member Functions

- bc_liquid (parameters &, TAB &, int, int)

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int, int)

  Calculates boundary condition.
- virtual ∼bc_liquid ()

  Destructor.

4.3.1 Detailed Description

Class which allows to calculate right liquid boundary thanks to the method of caracteristics.

Definition at line 61 of file bc_liquid.hpp.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 bc_liquid::bc_liquid (**parameters** & *par*, **TAB** & *z*, int *n1*, int *n2*)

Constructor.

Definition at line 50 of file bc_liquid.cpp.

4.3.2.2 bc_liquid::∼bc_liquid ( ) `[virtual]`

Destructor.

Definition at line 123 of file bc_liquid.cpp.

4.3.3 Member Function Documentation

4.3.3.1 void bc_liquid::calcul (**SCALAR** *hin*, **SCALAR** *unorm_in*, **SCALAR** *utan_in*, **SCALAR** *hfix*, **SCALAR** *qfix*, **SCALAR** *hin_oppbound*, **SCALAR** *unorm_in_oppbound*, **SCALAR** *utan_in_oppbound*, **SCALAR** *time*, int *n1*, int *n2*) `[virtual]`

Calculates boundary condition.

Implements boundary_condition.

Definition at line 86 of file bc_liquid.cpp.

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

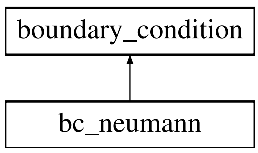
- Headers/libboundaryconditions/bc_liquid.hpp
- Sources/libboundaryconditions/bc_liquid.cpp

4.4 bc_neumann Class Reference

Class which allows to calculate Neumann boundary conditions.

```
#include <bc_neumann.hpp>
```

Inheritance diagram for bc_neumann:



Public Member Functions

- bc_neumann (parameters &, TAB &, int, int)

    Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int, int)

    Calculates boundary condition.
- virtual ~bc_neumann ()

    Destructor.

4.4.1 Detailed Description

Class which allows to calculate Neumann boundary conditions.

Definition at line 61 of file bc_neumann.hpp.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 bc_neumann::bc_neumann (**parameters** & *par*, **TAB** & *z*, int *n1*, int *n2*)

Constructor.

Definition at line 48 of file bc_neumann.cpp.

4.4.2.2 bc_neumann::∼bc_neumann ( ) [virtual]

Destructor.

Definition at line 92 of file bc_neumann.cpp.

4.4.3 Member Function Documentation

4.4.3.1 void bc_neumann::calcul (SCALAR *hin*, SCALAR *unorm_in*, SCALAR *utan_in*, SCALAR *hfix*, SCALAR *qfix*, SCALAR *hin_oppbound*, SCALAR *unorm_in_oppbound*, SCALAR *utan_in_oppbound*, SCALAR *time*, int *n1*, int *n2*) [virtual]

Calculates boundary condition.

Implements boundary_condition.

Definition at line 86 of file bc_neumann.cpp.

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

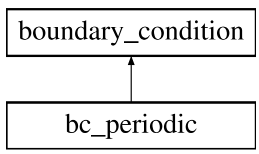
- Headers/libboundaryconditions/bc_neumann.hpp
- Sources/libboundaryconditions/bc_neumann.cpp

4.5 bc_periodic Class Reference

Class which allows to calculate the left periodic boundary condition.

`#include <bc_periodic.hpp>`

Inheritance diagram for bc_periodic:



Public Member Functions

- bc_periodic (parameters &, TAB &, int, int)

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int, int)

  Calculates boundary condition.
- virtual ∼bc_periodic ()

  Destructor.

4.5.1 Detailed Description

Class which allows to calculate the left periodic boundary condition.

Definition at line 62 of file bc_periodic.hpp.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 bc_periodic::bc_periodic (parameters & *par*, TAB & *z*, int *n1*, int *n2*)

Constructor.

Definition at line 48 of file bc_periodic.cpp.

4.5.2.2 bc_periodic::∼bc_periodic ( ) [virtual]

Destructor.

Definition at line 86 of file bc_periodic.cpp.

4.5.3 Member Function Documentation

4.5.3.1 void bc_periodic::calcul (SCALAR *hin*, SCALAR *unorm_in*, SCALAR *utan_in*, SCALAR *hfix*, SCALAR *qfix*, SCALAR *hin_oppbound*, SCALAR *unorm_in_oppbound*, SCALAR *utan_in_oppbound*, SCALAR *time*, int *n1*, int *n2*) [virtual]

Calculates boundary condition.

Implements boundary_condition.

Definition at line 80 of file bc_periodic.cpp.

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

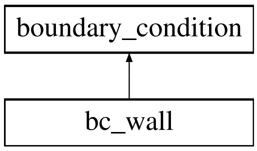
- Headers/libboundaryconditions/bc_periodic.hpp
- Sources/libboundaryconditions/bc_periodic.cpp

4.6 bc_wall Class Reference

Class which allows to calculate wall boundary condition.

`#include <bc_wall.hpp>`

Inheritance diagram for bc_wall:



Public Member Functions

- bc_wall (parameters &, TAB &, int, int)

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int, int)

  Calculates boundary condition.
- virtual ∼bc_wall ()

  Destructor.

4.6.1 Detailed Description

Class which allows to calculate wall boundary condition.

Definition at line 61 of file bc_wall.hpp.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 bc_wall::bc_wall (**parameters** & *par*, **TAB** & *z*, int *n1*, int *n2*)

Constructor.

Definition at line 48 of file bc_wall.cpp.

4.6.2.2 bc_wall::∼bc_wall ( ) `[virtual]`

Destructor.

Definition at line 88 of file bc_wall.cpp.

4.6.3 Member Function Documentation

4.6.3.1 void bc_wall::calcul (**SCALAR** *hin*, **SCALAR** *unorm_in*, **SCALAR** *utan_in*, **SCALAR** *hfix*, **SCALAR** *qfix*, **SCALAR** *hin_oppbound*, **SCALAR** *unorm_in_oppbound*, **SCALAR** *utan_in_oppbound*, **SCALAR** *time*, int *n1*, int *n2*) `[virtual]`

Calculates boundary condition.

Implements boundary_condition.

Definition at line 81 of file bc_wall.cpp.

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

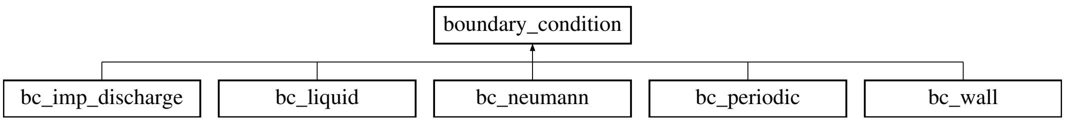
- Headers/libboundaryconditions/bc_wall.hpp
- Sources/libboundaryconditions/bc_wall.cpp

4.7 boundary_condition Class Reference

Class which allows to calculate boundary conditions.

`#include <boundary_condition.hpp>`

Inheritance diagram for boundary_condition:



Public Member Functions

- boundary_condition (parameters &)

    Constructor.
- virtual void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int, int)=0
- SCALAR get_hbound () const

    Returns the height of water at the boundary.
- SCALAR get_unormbound () const

    Returns the normal velocity of the flow at the boundary.
- SCALAR get_utanbound () const

    Returns the tangent velocity of the flow at the boundary.
- virtual ∼boundary_condition ()

    Destructor.

Protected Attributes

- const int Nxcell
- const int Nycell
- SCALAR hbound
- SCALAR unormbound
- SCALAR utanbound
- SCALAR unormfix

4.7.1 Detailed Description

Class which allows to calculate boundary conditions.

Definition at line 59 of file boundary_condition.hpp.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 boundary_condition::boundary_condition (parameters & *par*)

Constructor.

Definition at line 47 of file boundary_condition.cpp.

4.7.2.2 boundary_condition::~boundary_condition ( ) [virtual]

Destructor.

Definition at line 62 of file boundary_condition.cpp.

4.7.3 Member Function Documentation

4.7.3.1 virtual void boundary_condition::calcul (SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , int , int) [pure virtual]

Implemented in bc_imp_discharge, bc_liquid, bc_neumann, bc_periodic, and bc_wall.

4.7.3.2 SCALAR boundary_condition::get_hbound ( ) const

Returns the height of water at the boundary.

Definition at line 49 of file boundary_condition.cpp.

4.7.3.3 SCALAR boundary_condition::get_unormbound ( ) const

Returns the normal velocity of the flow at the boundary.

Definition at line 53 of file boundary_condition.cpp.

4.7.3.4 SCALAR boundary_condition::get_utanbound ( ) const

Returns the tangent velocity of the flow at the boundary.

Definition at line 57 of file boundary_condition.cpp.

4.7.4 Member Data Documentation

4.7.4.1 SCALAR boundary_condition::hbound [protected]

Definition at line 63 of file boundary_condition.hpp.

4.7.4.2 const int boundary_condition::Nxcell [protected]

Definition at line 61 of file boundary_condition.hpp.

4.7.4.3 const int boundary_condition::Nycell [protected]

Definition at line 62 of file boundary_condition.hpp.

4.7.4.4 SCALAR boundary_condition::unormbound [protected]

Definition at line 63 of file boundary_condition.hpp.

4.7.4.5 SCALAR boundary_condition::unormfix [protected]

Definition at line 63 of file boundary_condition.hpp.

4.7.4.6 SCALAR boundary_condition::utanbound [protected]

Definition at line 63 of file boundary_condition.hpp.

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

- Headers/libboundaryconditions/boundary_condition.hpp
- Sources/libboundaryconditions/boundary_condition.cpp

4.8 choice_condition Class Reference

Class which allows to choose right boundary conditions.

`#include <choice_condition.hpp>`

Public Member Functions

- choice_condition (int, parameters &, TAB &, int, int)

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int, int)

Calculates right boundary condition.

- SCALAR get_hbound ()

  Returns the height of water at the boundary.
- SCALAR get_unormbound ()

  Returns the normal velocity of the flow at the boundary.
- SCALAR get_utanbound ()

  Returns the tangent velocity of the flow at the boundary.
- virtual ∼choice_condition ()

  Destructor.

4.8.1 Detailed Description

Class which allows to choose right boundary conditions.

Definition at line 82 of file choice_condition.hpp.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 choice_condition::choice_condition (int *choice*, parameters & *par*, TAB & *z*, int *n1*, int *n2*)

Constructor.

Definition at line 47 of file choice_condition.cpp.

4.8.2.2 choice_condition::∼choice_condition ( ) [virtual]

Destructor.

Definition at line 87 of file choice_condition.cpp.

4.8.3 Member Function Documentation

4.8.3.1 void choice_condition::calcul (SCALAR *hin*, SCALAR *unorm_in*, SCALAR *utan_in*, SCALAR *hfix*, SCALAR *qfix*, SCALAR *hin_oppbound*, SCALAR *unorm_in_oppbound*, SCALAR *utan_in_oppbound*, SCALAR *time*, int *n1*, int *n2*)

Calculates right boundary condition.

Definition at line 71 of file choice_condition.cpp.

4.8.3.2 SCALAR choice_condition::get_hbound ( )

Returns the height of water at the boundary.

Definition at line 75 of file choice_condition.cpp.

4.8.3.3 SCALAR choice_condition::get_unormbound ( )

Returns the normal velocity of the flow at the boundary.

Definition at line 79 of file choice_condition.cpp.

4.8.3.4 SCALAR choice_condition::get_utanbound ( )

Returns the tangent velocity of the flow at the boundary.

Definition at line 83 of file choice_condition.cpp.

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

- Headers/libboundaryconditions/choice_condition.hpp
- Sources/libboundaryconditions/choice_condition.cpp

4.9 choice_flux Class Reference

class which allows to choose and calculate the numerical flux.

`#include <choice_flux.hpp>`

Public Member Functions

- choice_flux (int)

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates the numerical flux.
- void set_tx (SCALAR)

  Sets the variable tx.
- SCALAR get_f1 ()

  Returns the first component of the numerical flux.
- SCALAR get_f2 ()

  Returns the second component of the numerical flux.
- SCALAR get_f3 ()

  Returns the third component of the numerical flux.
- SCALAR get_cfl ()

  Returns the CFL.
- virtual ~choice_flux ()

  Destructor.

4.9.1 Detailed Description

class which allows to choose and calculate the numerical flux.

Definition at line 74 of file choice_flux.hpp.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 choice_flux::choice_flux (int *choice*)

Constructor.

Definition at line 46 of file choice_flux.cpp.

4.9.2.2 choice_flux::∼choice_flux ( ) [virtual]

Destructor.

Definition at line 90 of file choice_flux.cpp.

4.9.3 Member Function Documentation

4.9.3.1 void choice_flux::calcul (SCALAR *hg*, SCALAR *ug*, SCALAR *vg*, SCALAR *hd*, SCALAR *ud*, SCALAR *vd*)

Calculates the numerical flux.

Definition at line 66 of file choice_flux.cpp.

4.9.3.2 SCALAR choice_flux::get_cfl ( )

Returns the CFL.

Definition at line 86 of file choice_flux.cpp.

4.9.3.3 SCALAR choice_flux::get_f1 ( )

Returns the first component of the numerical flux.

Definition at line 74 of file choice_flux.cpp.

4.9.3.4 SCALAR choice_flux::get_f2 ( )

Returns the second component of the numerical flux.

Definition at line 78 of file choice_flux.cpp.

4.9.3.5 SCALAR choice_flux::get_f3 ( )

Returns the third component of the numerical flux.

Definition at line 82 of file choice_flux.cpp.

4.9.3.6 void choice_flux::set_tx (SCALAR *tx*)

Sets the variable tx.

Definition at line 70 of file choice_flux.cpp.

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

- Headers/libflux/choice_flux.hpp
- Sources/libflux/choice_flux.cpp

4.10 choice_friction Class Reference

class which allows to choose and calculate the friction term.

`#include <choice_friction.hpp>`

Public Member Functions

- choice_friction (int)

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates the friction term.
- SCALAR get_q1mod ()

  Returns the discharge in direction 1 after friction.
- SCALAR get_q2mod ()

  Returns the discharge in direction 2 after friction.
- void calculSf (SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates explicit formula of the friction term.
- SCALAR get_Sf1 ()

  Returns the explicit friction term in direction 1.
- SCALAR get_Sf2 ()

  Returns the explicit friction term in direction 2.
- virtual ∼choice_friction ()

  Destructor.

4.10.1 Detailed Description

class which allows to choose and calculate the friction term.

Definition at line 71 of file choice_friction.hpp.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 choice_friction::choice_friction (int *choice*)

Constructor.

Definition at line 45 of file choice_friction.cpp.

4.10.2.2 choice_friction::∼choice_friction ( ) [virtual]

Destructor.

Definition at line 88 of file choice_friction.cpp.

4.10.3 Member Function Documentation

4.10.3.1 void choice_friction::calcul (SCALAR *uold*, SCALAR *vold*, SCALAR *hnew*, SCALAR *q1new*, SCALAR *q2new*, SCALAR *dt*, SCALAR *cf*)

Calculates the friction term.

Definition at line 63 of file choice_friction.cpp.

4.10.3.2 void choice_friction::calculSf (SCALAR *h*, SCALAR *u*, SCALAR *v*, SCALAR *cf*)

Calculates explicit formula of the friction term.

Definition at line 76 of file choice_friction.cpp.

4.10.3.3 SCALAR choice_friction::get_q1mod ( )

Returns the discharge in direction 1 after friction.

Definition at line 67 of file choice_friction.cpp.

4.10.3.4 SCALAR choice_friction::get_q2mod ( )

Returns the discharge in direction 2 after friction.

Definition at line 71 of file choice_friction.cpp.

4.10.3.5 SCALAR choice_friction::get_Sf1 ( )

Returns the explicit friction term in direction 1.

Definition at line 80 of file choice_friction.cpp.

4.10.3.6 SCALAR choice_friction::get_Sf2 ( )

Returns the explicit friction term in direction 2.

Definition at line 84 of file choice_friction.cpp.

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

- Headers/libfrictions/choice_friction.hpp
- Sources/libfrictions/choice_friction.cpp

4.11 choice_infiltration Class Reference

Class which allows to choose the infiltration and to calculate the infiltrated volume the volume of water available.

`#include <choice_infiltration.hpp>`

Public Member Functions

- choice_infiltration (parameters &)

  Constructor.
- SCALAR get_hmod ()
- SCALAR get_Vin ()
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates the infiltrated volume and the volume of water available.
- virtual ∼choice_infiltration ()

  Destructor.

4.11.1 Detailed Description

Class which allows to choose the infiltration and to calculate the infiltrated volume the volume of water available.

Definition at line 66 of file choice_infiltration.hpp.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 choice_infiltration::choice_infiltration (parameters & *par*)

Constructor.

Definition at line 46 of file choice_infiltration.cpp.

4.11.2.2 choice_infiltration::~choice_infiltration ( ) [virtual]

Destructor.

Definition at line 73 of file choice_infiltration.cpp.

4.11.3 Member Function Documentation

4.11.3.1 void choice_infiltration::calcul (SCALAR *h*, SCALAR *Vin*, SCALAR *dt*, SCALAR *Kc*, SCALAR *Ks*, SCALAR *dtheta*, SCALAR *Psi*, SCALAR *zcrust*)

Calculates the infiltrated volume and the volume of water available.

Definition at line 61 of file choice_infiltration.cpp.

4.11.3.2 SCALAR choice_infiltration::get_hmod ( )

Definition at line 65 of file choice_infiltration.cpp.

4.11.3.3 SCALAR choice_infiltration::get_Vin ( )

Definition at line 69 of file choice_infiltration.cpp.

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

- Headers/librain_infiltration/choice_infiltration.hpp
- Sources/librain_infiltration/choice_infiltration.cpp

4.12 choice_init_huv Class Reference

Class which allows to choose and calculate the initial conditions on the water height and the velocity.

`#include <choice_init_huv.hpp>`

Public Member Functions

- choice_init_huv (parameters &)

  Constructor.
- void initialization (TAB &, TAB &, TAB &)

  Performs the scheme.
- virtual ~choice_init_huv ()

  Destructor.

4.12.1 Detailed Description

Class which allows to choose and calculate the initial conditions on the water height and the velocity.

Definition at line 81 of file choice_init_huv.hpp.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 choice_init_huv::choice_init_huv (**parameters &** *par*)

Constructor.

Definition at line 45 of file choice_init_huv.cpp.

4.12.2.2 choice_init_huv::∼choice_init_huv ( ) `[virtual]`

Destructor.

Definition at line 74 of file choice_init_huv.cpp.

4.12.3 Member Function Documentation

4.12.3.1 void choice_init_huv::initialization (**TAB &** *h*, **TAB &** *u*, **TAB &** *v*)

Performs the scheme.

Definition at line 70 of file choice_init_huv.cpp.

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

- Headers/libinitialization/choice_init_huv.hpp
- Sources/libinitialization/choice_init_huv.cpp

4.13 choice_init_topo Class Reference

Class which allows to choose and calculate the initial conditions on the topography.

`#include <choice_init_topo.hpp>`

Public Member Functions

- choice_init_topo (parameters &)
  Constructor.
- void initialization (TAB &)
  Performs the scheme.
- virtual ∼choice_init_topo ()

Destructor.

4.13.1 Detailed Description

Class which allows to choose and calculate the initial conditions on the topography.

Definition at line 73 of file choice_init_topo.hpp.

4.13.2 Constructor & Destructor Documentation

4.13.2.1 choice_init_topo::choice_init_topo (parameters & *par*)

Constructor.

Definition at line 45 of file choice_init_topo.cpp.

4.13.2.2 choice_init_topo::∼choice_init_topo ( ) [virtual]

Destructor.

Definition at line 68 of file choice_init_topo.cpp.

4.13.3 Member Function Documentation

4.13.3.1 void choice_init_topo::initialization (TAB & *h_sol*)

Performs the scheme.

Definition at line 64 of file choice_init_topo.cpp.

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

- Headers/libinitialization/choice_init_topo.hpp
- Sources/libinitialization/choice_init_topo.cpp

4.14 choice_limiter Class Reference

class which allows to choose and calculate the slope limiter.

`#include <choice_limiter.hpp>`

Public Member Functions

- choice_limiter (int)

  Constructor.
- void calcul (SCALAR, SCALAR)

Performs the computation.

- SCALAR get_rec () const

  Gets the reconstruction parameter.

- virtual ~choice_limiter ()

  Destructor.

4.14.1 Detailed Description

class which allows to choose and calculate the slope limiter.

Definition at line 73 of file choice_limiter.hpp.

4.14.2 Constructor & Destructor Documentation

4.14.2.1 choice_limiter::choice_limiter (int *choice*)

Constructor.

Definition at line 45 of file choice_limiter.cpp.

4.14.2.2 choice_limiter::~choice_limiter ( ) [virtual]

Destructor.

Definition at line 72 of file choice_limiter.cpp.

4.14.3 Member Function Documentation

4.14.3.1 void choice_limiter::calcul (SCALAR *a*, SCALAR *b*)

Performs the computation.

Definition at line 64 of file choice_limiter.cpp.

4.14.3.2 SCALAR choice_limiter::get_rec ( ) const

Gets the reconstruction parameter.

Definition at line 68 of file choice_limiter.cpp.

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

- Headers/liblimitations/choice_limiter.hpp
- Sources/liblimitations/choice_limiter.cpp

4.15 choice_output Class Reference

Class which allows to choose the style of the output file.

`#include <choice_output.hpp>`

Public Member Functions

- choice_output (parameters &)

  Constructor.
- void write (TAB, TAB, TAB, TAB, SCALAR)

  Write the result in the output file.
- void fc_var (SCALAR, SCALAR)
- void check_vol (SCALAR, SCALAR, TAB, TAB, SCALAR, SCALAR &, SCALAR &, SCALAR)
- void result (SCALAR, SCALAR, SCALAR, SCALAR, const SCALAR, const int, SCALAR)
- void initial (SCALAR, SCALAR, TAB, TAB, TAB, TAB)
- void final (SCALAR, SCALAR, TAB, TAB, TAB, TAB)
- SCALAR boundaries_flux (SCALAR, TAB &, TAB &, SCALAR, SCALAR, int, int, SCALAR &, TAB &)
- void boundaries_flux_LR (SCALAR, TAB)
- void boundaries_flux_BT (SCALAR, TAB)
- virtual ∼choice_output ()

  Destructor.

4.15.1 Detailed Description

Class which allows to choose the style of the output file.

Definition at line 69 of file choice_output.hpp.

4.15.2 Constructor & Destructor Documentation

4.15.2.1 choice_output::choice_output (parameters & *par*)

Constructor.

Definition at line 45 of file choice_output.cpp.

4.15.2.2 choice_output::∼choice_output ( ) `[virtual]`

Destructor.

Definition at line 97 of file choice_output.cpp.

4.15.3 Member Function Documentation

4.15.3.1 SCALAR choice_output::boundaries_flux (SCALAR *time*, TAB & *flux_u*, TAB & *flux_v*, SCALAR *dt*, SCALAR *dt_first*, int *ORDER*, int *verif*, SCALAR & *Rain_tot*, TAB & *Rain*)

Definition at line 87 of file choice_output.cpp.

4.15.3.2 void choice_output::boundaries_flux_BT (SCALAR *time*, TAB *BT_flux*)

Definition at line 93 of file choice_output.cpp.

4.15.3.3 void choice_output::boundaries_flux_LR (SCALAR *time*, TAB *LR_flux*)

Definition at line 90 of file choice_output.cpp.

4.15.3.4 void choice_output::check_vol (SCALAR *time*, SCALAR *dt*, TAB *h*, TAB *Vin*, SCALAR *Vol_rain_tot*, SCALAR & *Vol_inf*, SCALAR & *Vol_of*, SCALAR *bound_flux_tot*)

Definition at line 68 of file choice_output.cpp.

4.15.3.5 void choice_output::fc_var (SCALAR , SCALAR)

4.15.3.6 void choice_output::final (SCALAR *dx*, SCALAR *dy*, TAB *z*, TAB *h*, TAB *u*, TAB *v*)

Definition at line 82 of file choice_output.cpp.

4.15.3.7 void choice_output::initial (SCALAR *dx*, SCALAR *dy*, TAB *z*, TAB *h*, TAB *u*, TAB *v*)

Definition at line 77 of file choice_output.cpp.

4.15.3.8 void choice_output::result (SCALAR *time*, SCALAR *Vol_rain*, SCALAR *Vol_inf*, SCALAR *Vol_of*, const SCALAR *Froude*, const int *nbiter*, SCALAR *vol_output*)

Definition at line 72 of file choice_output.cpp.

4.15.3.9 void choice_output::write (TAB *h*, TAB *u*, TAB *v*, TAB *z*, SCALAR *tps*)

Write the result in the output file.

Definition at line 64 of file choice_output.cpp.

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

- Headers/libsave/choice_output.hpp
- Sources/libsave/choice_output.cpp

4.16 choice_rain Class Reference

Class which allows to choose and calculate the initial conditions of the rain.

`#include <choice_rain.hpp>`

Public Member Functions

- choice_rain (parameters &)
  Constructor.
- void Rain_func (SCALAR, TAB &)
  Performs the scheme.
- virtual ∼choice_rain ()
  Destructor.

4.16.1 Detailed Description

Class which allows to choose and calculate the initial conditions of the rain.

Definition at line 70 of file choice_rain.hpp.

4.16.2 Constructor & Destructor Documentation

4.16.2.1 choice_rain::choice_rain (**parameters** & *par*)

Constructor.

Definition at line 45 of file choice_rain.cpp.

4.16.2.2 choice_rain::∼choice_rain ( ) `[virtual]`

Destructor.

Definition at line 65 of file choice_rain.cpp.

4.16.3 Member Function Documentation

4.16.3.1 void choice_rain::Rain_func (**SCALAR** *time*, **TAB** & *Tab_rain*)

Performs the scheme.

Definition at line 61 of file choice_rain.cpp.

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

- Headers/librain_infiltration/choice_rain.hpp
- Sources/librain_infiltration/choice_rain.cpp

4.17 choice_reconstruction Class Reference

Class which allows to choose and calculate the second order reconstruction in space.

`#include <choice_reconstruction.hpp>`

Public Member Functions

- choice_reconstruction (parameters &, TAB &)

  Constructor.
- void calcul (TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &)

  Calculates the second order reconstructions in space.
- virtual ∼choice_reconstruction ()

  Destructor.

4.17.1 Detailed Description

Class which allows to choose and calculate the second order reconstruction in space.

Definition at line 74 of file choice_reconstruction.hpp.

4.17.2 Constructor & Destructor Documentation

4.17.2.1 choice_reconstruction::choice_reconstruction (parameters & *par*, TAB & *z*)

Constructor.

Definition at line 45 of file choice_reconstruction.cpp.

4.17.2.2 choice_reconstruction::∼choice_reconstruction ( ) [virtual]

Destructor.

Definition at line 68 of file choice_reconstruction.cpp.

4.17.3 Member Function Documentation

4.17.3.1 void choice_reconstruction::calcul (TAB & *h*, TAB & *u*, TAB & *v*, TAB & *z*, TAB & *delzc1*, TAB & *delzc2*, TAB & *delz1*, TAB & *delz2*, TAB & *h1r*, TAB & *u1r*, TAB & *v1r*, TAB & *h1l*, TAB & *u1l*, TAB & *v1l*, TAB & *h2r*, TAB & *u2r*, TAB & *v2r*, TAB & *h2l*, TAB & *u2l*, TAB & *v2l*)

Calculates the second order reconstructions in space.

Definition at line 64 of file choice_reconstruction.cpp.

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

- Headers/libreconstructions/choice_reconstruction.hpp
- Sources/libreconstructions/choice_reconstruction.cpp

4.18 choice_scheme Class Reference

class which allows to choose the scheme and to perform calculations.

`#include <choice_scheme.hpp>`

Public Member Functions

- choice_scheme (parameters &)

  Constructor.
- void calcul ()

  Performs the scheme.
- virtual ∼choice_scheme ()

  Destructor.

4.18.1 Detailed Description

class which allows to choose the scheme and to perform calculations.

Definition at line 70 of file choice_scheme.hpp.

4.18.2 Constructor & Destructor Documentation

4.18.2.1 choice_scheme::choice_scheme (parameters & *par*)

Constructor.

Definition at line 45 of file choice_scheme.cpp.

4.18.2.2 choice_scheme::∼choice_scheme ( ) [virtual]

Destructor.

Definition at line 65 of file choice_scheme.cpp.

4.18.3 Member Function Documentation

4.18.3.1 void choice_scheme::calcul ( )

Performs the scheme.

Definition at line 61 of file choice_scheme.cpp.

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

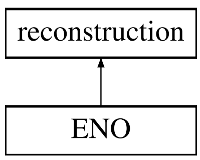
- Headers/libschemes/choice_scheme.hpp
- Sources/libschemes/choice_scheme.cpp

4.19 ENO Class Reference

class which allows to calculate the ENO reconstruction in space

`#include <ENO.hpp>`

Inheritance diagram for ENO:



Public Member Functions

- ENO (parameters &, TAB &)

  Constructor.
- void calcul (TAB &, TAB &)

  Calculates the ENO reconstruction in space.
- ∼ENO ()

  Destructor.

Protected Attributes

- int Nxcell
- int Nycell
- SCALAR ddh1
- SCALAR ddz1
- SCALAR ddu1
- SCALAR ddv1
- SCALAR ddh2
- SCALAR ddz2
- SCALAR ddu2
- SCALAR ddv2
- SCALAR hh1
- SCALAR uu1
- SCALAR vv1
- SCALAR hh2
- SCALAR uu2
- SCALAR vv2
- SCALAR amortENO
- TAB som_z1
- TAB som_z2

4.19.1 Detailed Description

class which allows to calculate the ENO reconstruction in space

In order to get a second order in space, we can use a ENO reconstruction. A scalar fonction $s \in R$ is reconstructed by the ENO reconstruction (Essentially Non Oscillatory), which writes

$$s_{i-1/2+} = s_i - \tfrac{\Delta x}{2} D_{eno} s_i \;\; and \quad s_{i+1/2-} = s_i + \tfrac{\Delta x}{2} D_{eno} s_i$$

where D_{eno} writes

$$D_{eno} s_i = minmod\left(\frac{s_i - s_{i-1}}{\Delta x} + \frac{\Delta x}{2} D^2 s_{i-1/2}, \frac{s_{i+1} - s_i}{\Delta x} - \frac{\Delta x}{2} D^2 s_{i+1/2}\right)$$

with the Minmod slope limiter and with

$$D^2 s_{i+1/2} = minmod\left(\frac{s_{i+1} - 2s_i + s_{i-1}}{\Delta x^2}, \frac{s_{i+2} - 2s_{i+1} + s_i}{\Delta x^2}\right)$$

.

Definition at line 76 of file ENO.hpp.

4.19.2 Constructor & Destructor Documentation

4.19.2.1 ENO::ENO (parameters & *par*, TAB & *z*)

Constructor.

Definition at line 45 of file ENO.cpp.

4.19.2.2 ENO::∼ENO ( )

Destructor.

Definition at line 351 of file ENO.cpp.

4.19.3 Member Function Documentation

4.19.3.1 void ENO::calcul (TAB & *h*, TAB & *u*, TAB & *v*, TAB & *z*, TAB & *delzc1*, TAB & *delzc2*, TAB & *delz1*, TAB & *delz2*, TAB & *h1r*, TAB & *u1r*, TAB & *v1r*, TAB & *h1l*, TAB & *u1l*, TAB & *v1l*, TAB & *h2r*, TAB & *u2r*, TAB & *v2r*, TAB & *h2l*, TAB & *u2l*, TAB & *v2l*) [virtual]

Calculates the ENO reconstruction in space.

Implements reconstruction.

Definition at line 103 of file ENO.cpp.

4.19.4 Member Data Documentation

4.19.4.1 SCALAR ENO::amortENO [protected]

Definition at line 82 of file ENO.hpp.

4.19.4.2 SCALAR ENO::ddh1 [protected]

Definition at line 80 of file ENO.hpp.

4.19.4.3 SCALAR ENO::ddh2 [protected]

Definition at line 80 of file ENO.hpp.

4.19.4.4 SCALAR ENO::ddu1 [protected]

Definition at line 80 of file ENO.hpp.

4.19.4.5 SCALAR ENO::ddu2 [protected]

Definition at line 80 of file ENO.hpp.

4.19.4.6 SCALAR ENO::ddv1 [protected]

Definition at line 80 of file ENO.hpp.

4.19.4.7 SCALAR ENO::ddv2 [protected]

Definition at line 80 of file ENO.hpp.

4.19.4.8 SCALAR ENO::ddz1 [protected]

Definition at line 80 of file ENO.hpp.

4.19.4.9 SCALAR ENO::ddz2 [protected]

Definition at line 80 of file ENO.hpp.

4.19.4.10 SCALAR ENO::hh1 [protected]

Definition at line 81 of file ENO.hpp.

4.19.4.11 SCALAR ENO::hh2 [protected]

Definition at line 81 of file ENO.hpp.

4.19.4.12 int ENO::Nxcell [protected]

Definition at line 79 of file ENO.hpp.

4.19.4.13 int ENO::Nycell [protected]

Definition at line 79 of file ENO.hpp.

4.19.4.14 TAB ENO::som_z1 [protected]

Definition at line 83 of file ENO.hpp.

4.19.4.15 TAB ENO::som_z2 [protected]

Definition at line 83 of file ENO.hpp.

4.19.4.16 SCALAR ENO::uu1 [protected]

Definition at line 81 of file ENO.hpp.

4.19.4.17 SCALAR ENO::uu2 [protected]

Definition at line 81 of file ENO.hpp.

4.19.4.18 SCALAR ENO::vv1 [protected]

Definition at line 81 of file ENO.hpp.

4.19.4.19 SCALAR ENO::vv2 [protected]

Definition at line 81 of file ENO.hpp.

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

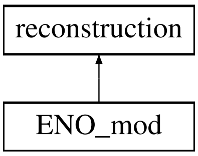
- Headers/libreconstructions/ENO.hpp
- Sources/libreconstructions/ENO.cpp

4.20 ENO_mod Class Reference

class which allows to calculate the modified ENO reconstruction in space.

```
#include <ENO_mod.hpp>
```

Inheritance diagram for ENO_mod:



Public Member Functions

- ENO_mod (parameters &, TAB &)

  Constructor.
- void calcul (TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &)

  Calculates the modified ENO reconstruction in space.
- ∼ENO_mod ()

  Destructor.

Protected Attributes

- int Nxcell
- int Nycell
- SCALAR ddh1
- SCALAR ddz1
- SCALAR ddu1
- SCALAR ddv1
- SCALAR ddh2
- SCALAR ddz2
- SCALAR ddu2
- SCALAR ddv2
- SCALAR hh1
- SCALAR uu1
- SCALAR vv1
- SCALAR hh2
- SCALAR uu2
- SCALAR vv2
- SCALAR amortENO
- SCALAR modifENO
- TAB som_z1
- TAB som_z2

4.20.1 Detailed Description

class which allows to calculate the modified ENO reconstruction in space.

Definition at line 64 of file ENO_mod.hpp.

4.20.2 Constructor & Destructor Documentation

4.20.2.1 ENO_mod::ENO_mod (parameters & *par*, TAB & *z*)

Constructor.

Definition at line 46 of file ENO_mod.cpp.

4.20.2.2 ENO_mod::∼ENO_mod ( )

Destructor.

Definition at line 380 of file ENO_mod.cpp.

4.20.3 Member Function Documentation

4.20.3.1 void ENO_mod::calcul (TAB & *h*, TAB & *u*, TAB & *v*, TAB & *z*, TAB & *delzc1*, TAB & *delzc2*, TAB & *delz1*, TAB & *delz2*, TAB & *h1r*, TAB & *u1r*, TAB & *v1r*, TAB & *h1l*, TAB & *u1l*, TAB & *v1l*, TAB & *h2r*, TAB & *u2r*, TAB & *v2r*, TAB & *h2l*, TAB & *u2l*, TAB & *v2l*) [virtual]

Calculates the modified ENO reconstruction in space.

Implements reconstruction.

Definition at line 102 of file ENO_mod.cpp.

4.20.4 Member Data Documentation

4.20.4.1 SCALAR ENO_mod::amortENO [protected]

Definition at line 70 of file ENO_mod.hpp.

4.20.4.2 SCALAR ENO_mod::ddh1 [protected]

Definition at line 68 of file ENO_mod.hpp.

4.20.4.3 SCALAR ENO_mod::ddh2 [protected]

Definition at line 68 of file ENO_mod.hpp.

4.20.4.4 SCALAR ENO_mod::ddu1 [protected]

Definition at line 68 of file ENO_mod.hpp.

4.20.4.5 SCALAR ENO_mod::ddu2 [protected]

Definition at line 68 of file ENO_mod.hpp.

4.20.4.6 SCALAR ENO_mod::ddv1 [protected]

Definition at line 68 of file ENO_mod.hpp.

4.20.4.7 SCALAR ENO_mod::ddv2 [protected]

Definition at line 68 of file ENO_mod.hpp.

4.20.4.8 SCALAR ENO_mod::ddz1 [protected]

Definition at line 68 of file ENO_mod.hpp.

4.20.4.9 SCALAR ENO_mod::ddz2 [protected]

Definition at line 68 of file ENO_mod.hpp.

4.20.4.10 SCALAR ENO_mod::hh1 [protected]

Definition at line 69 of file ENO_mod.hpp.

4.20.4.11 SCALAR ENO_mod::hh2 [protected]

Definition at line 69 of file ENO_mod.hpp.

4.20.4.12 SCALAR ENO_mod::modifENO [protected]

Definition at line 70 of file ENO_mod.hpp.

4.20.4.13 int ENO_mod::Nxcell [protected]

Definition at line 67 of file ENO_mod.hpp.

4.20.4.14 int ENO_mod::Nycell [protected]

Definition at line 67 of file ENO_mod.hpp.

4.20.4.15 TAB ENO_mod::som_z1 [protected]

Definition at line 71 of file ENO_mod.hpp.

4.20.4.16 TAB ENO_mod::som_z2 [protected]

Definition at line 71 of file ENO_mod.hpp.

4.20.4.17 SCALAR ENO_mod::uu1 [protected]

Definition at line 69 of file ENO_mod.hpp.

4.20.4.18 SCALAR ENO_mod::uu2 [protected]

Definition at line 69 of file ENO_mod.hpp.

4.20.4.19 SCALAR ENO_mod::vv1 [protected]

Definition at line 69 of file ENO_mod.hpp.

4.20.4.20 SCALAR ENO_mod::vv2 [protected]

Definition at line 69 of file ENO_mod.hpp.

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

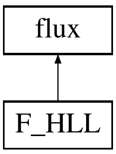
- Headers/libreconstructions/ENO_mod.hpp
- Sources/libreconstructions/ENO_mod.cpp

4.21 F_HLL Class Reference

class which allows to calculate HLL flux

`#include <F_HLL.hpp>`

Inheritance diagram for F_HLL:



Public Member Functions

- F_HLL ()

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates HLL flux.
- virtual ∼F_HLL ()

  Destructor.

4.21.1 Detailed Description

class which allows to calculate HLL flux

$$\mathscr{F}(U_G,U_D)=\begin{cases} F(U_G)\ if\ 0<c_1 \\ \frac{c_1F(U_G)-c_2F(U_D)}{c_2-c_1}+\frac{c_1c_2}{c_2-c_1}.(U_D-U_G)\ if\ c_1<0<c_2 \\ F(U_D)\ if\ c_2<0 \end{cases},$$

with

$$c_1=\inf_{U=U_G,U_D}\left(\inf_{j\in\{1,2\}}|\lambda_j(U)|\right)\ and\ c_2=\sup_{U=U_G,U_D}\left(\sup_{j\in\{1,2\}}|\lambda_j(U)|\right),$$

where $\lambda_1(U)=u-\sqrt{gh}$ et $\lambda_2(U)=u+\sqrt{gh}$ are the eigenvalues of the shallow water system.

Definition at line 71 of file F_HLL.hpp.

4.21.2 Constructor & Destructor Documentation

4.21.2.1 F_HLL::F_HLL ( )

Constructor.

Definition at line 48 of file F_HLL.cpp.

4.21.2.2 F_HLL::~F_HLL ( ) [virtual]

Destructor.

Definition at line 93 of file F_HLL.cpp.

4.21.3 Member Function Documentation

4.21.3.1 void F_HLL::calcul (SCALAR *hg*, SCALAR *ug*, SCALAR *vg*, SCALAR *hd*, SCALAR *ud*, SCALAR *vd*) [virtual]

Calculates HLL flux.

Implements flux.

Definition at line 52 of file F_HLL.cpp.

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

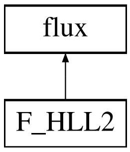
- Headers/libflux/F_HLL.hpp
- Sources/libflux/F_HLL.cpp

4.22 F_HLL2 Class Reference

class which allows to calculate HLL flux written under a shorter form

`#include <F_HLL2.hpp>`

Inheritance diagram for F_HLL2:



Public Member Functions

- F_HLL2 ()

    Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

    Calculates HLL flux.
- virtual ∼F_HLL2 ()

    Destructor.

4.22.1 Detailed Description

class which allows to calculate HLL flux written under a shorter form

$$\mathscr{F}(U_G,U_D) = t_1 F(U_D) + t_2 F(U_G) - t_3(U_D - U_G),$$

with

$$t_1 = \frac{\min(c_2,0) - \min(c_1,0)}{c_2 - c_1},\ t_2 = 1 - t_1,\ t_3 = \frac{c_2|c_1| - c_1|c_2|}{2(c_2 - c_1)}.$$

Definition at line 65 of file F_HLL2.hpp.

4.22.2 Constructor & Destructor Documentation

4.22.2.1 F_HLL2::F_HLL2 ( )

Constructor.

Definition at line 48 of file F_HLL2.cpp.

4.22.2.2 F_HLL2::∼F_HLL2 ( ) [virtual]

Destructor.

Definition at line 80 of file F_HLL2.cpp.

4.22.3 Member Function Documentation

4.22.3.1 void F_HLL2::calcul (SCALAR *hg*, SCALAR *ug*, SCALAR *vg*, SCALAR *hd*, SCALAR *ud*, SCALAR *vd*) [virtual]

Calculates HLL flux.

Implements flux.

Definition at line 52 of file F_HLL2.cpp.

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

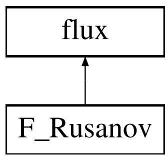
- Headers/libflux/F_HLL2.hpp
- Sources/libflux/F_HLL2.cpp

4.23 F_Rusanov Class Reference

class which allows to calculate Rusanov flux

```
#include <F_Rusanov.hpp>
```

Inheritance diagram for F_Rusanov:



Public Member Functions

- F_Rusanov ()

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates Rusanov flux.
- virtual ~F_Rusanov ()

  Destructor.

4.23.1 Detailed Description

class which allows to calculate Rusanov flux

$$\mathscr{F}(U_G,U_D) = \frac{F(U_G)+F(U_D)}{2} - c.\frac{U_D-U_G}{2},$$

with

$$c = \max(|u_G| + \sqrt{gh_G}, |u_D| + \sqrt{gh_D}).$$

Definition at line 65 of file F_Rusanov.hpp.

4.23.2 Constructor & Destructor Documentation

4.23.2.1 F_Rusanov::F_Rusanov ( )

Constructor.

Definition at line 48 of file F_Rusanov.cpp.

4.23.2.2 F_Rusanov::∼F_Rusanov ( ) [virtual]

Destructor.

Definition at line 72 of file F_Rusanov.cpp.

4.23.3 Member Function Documentation

4.23.3.1 void F_Rusanov::calcul (SCALAR *hg*, SCALAR *ug*, SCALAR *vg*, SCALAR *hd*, SCALAR *ud*, SCALAR *vd*) [virtual]

Calculates Rusanov flux.

Implements flux.

Definition at line 51 of file F_Rusanov.cpp.

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

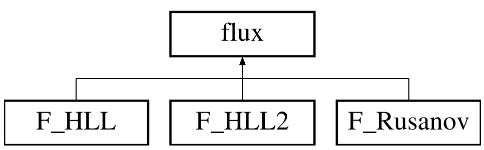
- Headers/libflux/F_Rusanov.hpp
- Sources/libflux/F_Rusanov.cpp

4.24 flux Class Reference

class which allows to calculate numerical flux.

`#include <flux.hpp>`

Inheritance diagram for flux:



Public Member Functions

- flux ()

  Constructor.
- virtual void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)=0
- void set_tx (SCALAR)

  Sets the variable tx.
- SCALAR get_f1 () const

  Returns the first component of the numerical flux.
- SCALAR get_f2 () const

  Returns the second component of the numerical flux.
- SCALAR get_f3 () const

  Returns the third component of the numerical flux.
- SCALAR get_cfl () const

  Returns the CFL.
- virtual ~flux ()

  Destructor.

Protected Attributes

- SCALAR f1
- SCALAR f2
- SCALAR f3
- SCALAR cfl
- SCALAR tx

4.24.1 Detailed Description

class which allows to calculate numerical flux.

Definition at line 59 of file flux.hpp.

4.24.2 Constructor & Destructor Documentation

4.24.2.1 flux::flux ( )

Constructor.

Definition at line 45 of file flux.cpp.

4.24.2.2 flux::~flux ( ) [virtual]

Destructor.

Definition at line 72 of file flux.cpp.

4.24.3 Member Function Documentation

4.24.3.1 virtual void flux::calcul (SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR) [pure virtual]

Implemented in F_HLL, F_HLL2, and F_Rusanov.

4.24.3.2 SCALAR flux::get_cfl ( ) const

Returns the CFL.

Definition at line 68 of file flux.cpp.

4.24.3.3 SCALAR flux::get_f1 ( ) const

Returns the first component of the numerical flux.

Definition at line 56 of file flux.cpp.

4.24.3.4 SCALAR flux::get_f2 ( ) const

Returns the second component of the numerical flux.

Definition at line 60 of file flux.cpp.

4.24.3.5 SCALAR flux::get_f3 ( ) const

Returns the third component of the numerical flux.

Definition at line 64 of file flux.cpp.

4.24.3.6 void flux::set_tx (SCALAR *tx*)

Sets the variable tx.

Definition at line 52 of file flux.cpp.

4.24.4 Member Data Documentation

4.24.4.1 SCALAR flux::cfl [protected]

Definition at line 61 of file flux.hpp.

4.24.4.2 SCALAR flux::f1 [protected]

Definition at line 61 of file flux.hpp.

4.24.4.3 SCALAR flux::f2 [protected]

Definition at line 61 of file flux.hpp.

4.24.4.4 SCALAR flux::f3 [protected]

Definition at line 61 of file flux.hpp.

4.24.4.5 SCALAR flux::tx [protected]

Definition at line 62 of file flux.hpp.

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

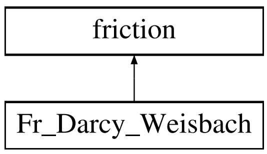
- Headers/libflux/flux.hpp
- Sources/libflux/flux.cpp

4.25 Fr_Darcy_Weisbach Class Reference

class which allows to calculate Darcy-Weisbach friction term

`#include <Fr_Darcy_Weisbach.hpp>`

Inheritance diagram for Fr_Darcy_Weisbach:



Public Member Functions

- Fr_Darcy_Weisbach ()

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates the Darcy-Weisbach friction term.
- void calculSf (SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates explicit formula of the Darcy-Weisbach friction term.
- virtual ∼Fr_Darcy_Weisbach ()

  Destructor.

4.25.1 Detailed Description

class which allows to calculate Darcy-Weisbach friction term

$$S_f = \frac{fu|u|}{gh}$$

thanks to a semi-implicit method :

$$q_i^{n+1} = \frac{q_{i*}^{n+1}}{1+\Delta t \frac{f|u_i^n|}{8h_i^{n+1}}}.$$

where f is the friction coefficient.

Definition at line 67 of file Fr_Darcy_Weisbach.hpp.

4.25.2 Constructor & Destructor Documentation

4.25.2.1 Fr_Darcy_Weisbach::Fr_Darcy_Weisbach ( )

Constructor.

Definition at line 45 of file Fr_Darcy_Weisbach.cpp.

4.25.2.2 Fr_Darcy_Weisbach::∼Fr_Darcy_Weisbach ( ) `[virtual]`

Destructor.

Definition at line 63 of file Fr_Darcy_Weisbach.cpp.

4.25.3 Member Function Documentation

4.25.3.1 void Fr_Darcy_Weisbach::calcul (SCALAR *uold*, SCALAR *vold*, SCALAR *hnew*, SCALAR *q1new*, SCALAR *q2new*, SCALAR *dt*, SCALAR *cf*) `[virtual]`

Calculates the Darcy-Weisbach friction term.

Implements friction.

Definition at line 49 of file Fr_Darcy_Weisbach.cpp.

4.25.3.2 void Fr_Darcy_Weisbach::calculSf (SCALAR *h*, SCALAR *u*, SCALAR *v*, SCALAR *cf*) `[virtual]`

Calculates explicit formula of the Darcy-Weisbach friction term.

Implements friction.

Definition at line 55 of file Fr_Darcy_Weisbach.cpp.

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

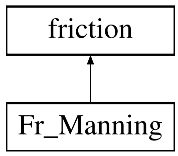
- Headers/libfrictions/Fr_Darcy_Weisbach.hpp
- Sources/libfrictions/Fr_Darcy_Weisbach.cpp

4.26 Fr_Manning Class Reference

class which allows to calculate Manning friction term

`#include <Fr_Manning.hpp>`

Inheritance diagram for Fr_Manning:



Public Member Functions

- Fr_Manning ()

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates the Manning friction term.
- void calculSf (SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates explicit formula of the Manning friction term.
- virtual ∼Fr_Manning ()

  Destructor.

4.26.1 Detailed Description

class which allows to calculate Manning friction term

$$S_f = \frac{u|u|}{Nycell^2 h^{4/3}}$$

thanks to a semi-implicit method :

$$q_i^{n+1} = \frac{q_{i*}^{n+1}}{1+\Delta t \frac{g|u_i^n|}{Nycell^2 (h_i^{n+1})^{4/3}}}.$$

where $Nycell$ is the friction coefficient.

Definition at line 67 of file Fr_Manning.hpp.

4.26.2 Constructor & Destructor Documentation

4.26.2.1 Fr_Manning::Fr_Manning ( )

Constructor.

Definition at line 45 of file Fr_Manning.cpp.

4.26.2.2 Fr_Manning::∼Fr_Manning ( ) [virtual]

Destructor.

Definition at line 62 of file Fr_Manning.cpp.

4.26.3 Member Function Documentation

4.26.3.1 void Fr_Manning::calcul (SCALAR *uold,* SCALAR *vold,* SCALAR *hnew,* SCALAR *q1new,* SCALAR *q2new,* SCALAR *dt,* SCALAR *cf*) [virtual]

Calculates the Manning friction term.

Implements friction.

Definition at line 49 of file Fr_Manning.cpp.

4.26.3.2 void Fr_Manning::calculSf (SCALAR *h,* SCALAR *u,* SCALAR *v,* SCALAR *cf*) [virtual]

Calculates explicit formula of the Manning friction term.

Implements friction.

Definition at line 55 of file Fr_Manning.cpp.

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

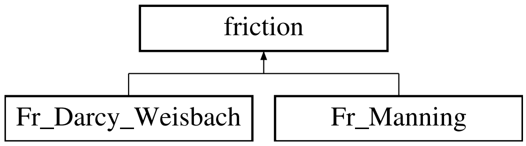
- Headers/libfrictions/Fr_Manning.hpp
- Sources/libfrictions/Fr_Manning.cpp

4.27 friction Class Reference

class which allows to calculate the friction term.

`#include <friction.hpp>`

Inheritance diagram for friction:



Public Member Functions

- friction ()

  Constructor.
- virtual void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)=0
- virtual SCALAR get_q1mod () const

  Returns the discharge q1 after friction.
- virtual SCALAR get_q2mod () const

  Returns the discharge q2 after friction.
- virtual void calculSf (SCALAR, SCALAR, SCALAR, SCALAR)=0

  Calculates explicit formula of the friction term.
- virtual SCALAR get_Sf1 () const

  Returns the explicit friction term in direction 1.
- virtual SCALAR get_Sf2 () const

  Returns the explicit friction term in direction 2.
- virtual ∼friction ()

  Destructor.

Protected Attributes

- SCALAR q1mod
- SCALAR q2mod
- SCALAR Sf1
- SCALAR Sf2

4.27.1 Detailed Description

class which allows to calculate the friction term.

Definition at line 60 of file friction.hpp.

4.27.2 Constructor & Destructor Documentation

4.27.2.1 friction::friction ( )

Constructor.

Definition at line 45 of file friction.cpp.

4.27.2.2 friction::~friction ( ) [virtual]

Destructor.

Definition at line 65 of file friction.cpp.

4.27.3 Member Function Documentation

4.27.3.1 virtual void friction::calcul (SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR) [pure virtual]

Implemented in Fr_Darcy_Weisbach, and Fr_Manning.

4.27.3.2 virtual void friction::calculSf (SCALAR , SCALAR , SCALAR , SCALAR) [pure virtual]

Calculates explicit formula of the friction term.

Implemented in Fr_Darcy_Weisbach, and Fr_Manning.

4.27.3.3 SCALAR friction::get_q1mod ( ) const [virtual]

Returns the discharge q1 after friction.

Definition at line 47 of file friction.cpp.

4.27.3.4 SCALAR friction::get_q2mod ( ) const [virtual]

Returns the discharge q2 after friction.

Definition at line 51 of file friction.cpp.

4.27.3.5 SCALAR friction::get_Sf1 ( ) const [virtual]

Returns the explicit friction term in direction 1.

Definition at line 57 of file friction.cpp.

4.27.3.6 SCALAR friction::get_Sf2 ( ) const [virtual]

Returns the explicit friction term in direction 2.

Definition at line 61 of file friction.cpp.

4.27.4 Member Data Documentation

4.27.4.1 SCALAR friction::q1mod [protected]

Definition at line 62 of file friction.hpp.

4.27.4.2 SCALAR friction::q2mod [protected]

Definition at line 62 of file friction.hpp.

4.27.4.3 SCALAR friction::Sf1 [protected]

Definition at line 63 of file friction.hpp.

4.27.4.4 SCALAR friction::Sf2 [protected]

Definition at line 63 of file friction.hpp.

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

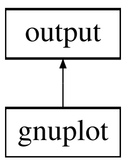
- Headers/libfrictions/friction.hpp
- Sources/libfrictions/friction.cpp

4.28 gnuplot Class Reference

Class which allows to write the result in the output file for gnuplot.

```
#include <gnuplot.hpp>
```

Inheritance diagram for gnuplot:



Public Member Functions

- gnuplot (parameters &)

  Constructor.
- void write (TAB, TAB, TAB, TAB, SCALAR)
- virtual ∼gnuplot ()

  Destructor.

4.28.1 Detailed Description

Class which allows to write the result in the output file for gnuplot.

Definition at line 61 of file gnuplot.hpp.

4.28.2 Constructor & Destructor Documentation

4.28.2.1 gnuplot::gnuplot (parameters & *par*)

Constructor.

Definition at line 46 of file gnuplot.cpp.

4.28.2.2 gnuplot::~gnuplot ( ) [virtual]

Destructor.

Definition at line 88 of file gnuplot.cpp.

4.28.3 Member Function Documentation

4.28.3.1 void gnuplot::write (TAB *h*, TAB *u*, TAB *v*, TAB *z*, SCALAR *tps*) [virtual]

Implements output.

Definition at line 66 of file gnuplot.cpp.

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

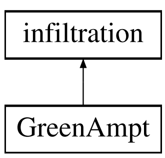
- Headers/libsave/gnuplot.hpp
- Sources/libsave/gnuplot.cpp

4.29 GreenAmpt Class Reference

Class which allows to calculate the infiltration capacity and infiltrated volum.

```
#include <GreenAmpt.hpp>
```

Inheritance diagram for GreenAmpt:



Public Member Functions

- GreenAmpt (parameters &)

  Constructor.
- SCALAR capacity (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates infiltration capacity.
- void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)

  Calculates infiltrated volum.
- virtual ∼GreenAmpt ()

  Destructor.

4.29.1 Detailed Description

Class which allows to calculate the infiltration capacity and infiltrated volum.

$$I_C^n = \begin{cases} Kc(1+\frac{h_f-h_{sur}^n}{Z_f^n})\ if\ Z_f^n <= Zc \\ \\ K_e^n(1+\frac{h_f-h_{sur}^n}{Z_f^n}) \end{cases},$$

with hydraulic conductivity

$$K_e^n = \frac{Z_f^n}{[(Z_f^n - Z_c)/K_s] + Z_c/K_c},$$

Where -h^{n}_{sur} 0 is the water height at the topography surface (obtained thanks to Shallow Water model)

Definition at line 73 of file GreenAmpt.hpp.

4.29.2 Constructor & Destructor Documentation

4.29.2.1 GreenAmpt::GreenAmpt (parameters & *par*)

Constructor.

Definition at line 45 of file GreenAmpt.cpp.

4.29.2.2 GreenAmpt::∼GreenAmpt ( ) [virtual]

Destructor.

Definition at line 90 of file GreenAmpt.cpp.

4.29.3 Member Function Documentation

4.29.3.1 void GreenAmpt::calcul (SCALAR *h*, SCALAR *Vin_tot*, SCALAR *dt*, SCALAR *Kc*, SCALAR *Ks*, SCALAR *dtheta*, SCALAR *Psi*, SCALAR *zcrust*) [virtual]

Calculates infiltrated volum.

Implements infiltration.

Definition at line 63 of file GreenAmpt.cpp.

4.29.3.2 SCALAR GreenAmpt::capacity (SCALAR *h*, SCALAR *Vin_tot*, SCALAR *dt*, SCALAR *Kc*, SCALAR *Ks*, SCALAR *dtheta*, SCALAR *Psi*, SCALAR *zcrust*)

Calculates infiltration capacity.

Definition at line 49 of file GreenAmpt.cpp.

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

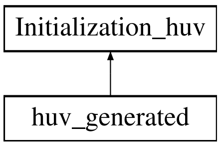
- Headers/librain_infiltration/GreenAmpt.hpp
- Sources/librain_infiltration/GreenAmpt.cpp

4.30 huv_generated Class Reference

Class for the initialization of the water height and velocity thanks to a function.

`#include <huv_generated.hpp>`

Inheritance diagram for huv_generated:



Public Member Functions

- huv_generated (parameters &)

  Constructor.
- void initialization (TAB &, TAB &, TAB &)
- virtual ~huv_generated ()

4.30.1 Detailed Description

Class for the initialization of the water height and velocity thanks to a function.

Class for the initialization of the water height and velocity for Thacker's benchmark.

Class for the initialization of the water height and velocity for Dressler's Dam-break benchmark.

Definition at line 61 of file huv_generated.hpp.

4.30.2 Constructor & Destructor Documentation

4.30.2.1 huv_generated::huv_generated (**parameters & *par***)

Constructor.

Definition at line 47 of file huv_generated.cpp.

4.30.2.2 huv_generated::∼huv_generated ( ) `[virtual]`

Definition at line 50 of file huv_generated.cpp.

4.30.3 Member Function Documentation

4.30.3.1 void huv_generated::initialization (**TAB & *h*, TAB & *u*, TAB & *v***) `[virtual]`

Implements Initialization_huv.

Definition at line 52 of file huv_generated.cpp.

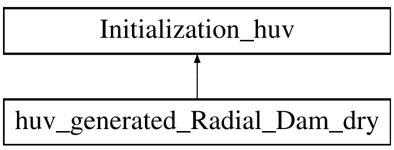
The documentation for this class was generated from the following files:

- Headers/libinitialization/huv_generated.hpp
- Sources/libinitialization/huv_generated.cpp

4.31 huv_generated_Radial_Dam_dry Class Reference

`#include <huv_generated_Radial_Dam_dry.hpp>`

Inheritance diagram for huv_generated_Radial_Dam_dry:



Public Member Functions

- huv_generated_Radial_Dam_dry (parameters &)

  Constructor.
- void initialization (TAB &, TAB &, TAB &)
- virtual ∼huv_generated_Radial_Dam_dry ()

4.31.1 Detailed Description

Definition at line 61 of file huv_generated_Radial_Dam_dry.hpp.

4.31.2 Constructor & Destructor Documentation

4.31.2.1 huv_generated_Radial_Dam_dry::huv_generated_Radial_Dam_dry (parameters & *par*)

Constructor.

Definition at line 47 of file huv_generated_Radial_Dam_dry.cpp.

4.31.2.2 huv_generated_Radial_Dam_dry::∼huv_generated_Radial_Dam_dry ( ) [virtual]

Definition at line 59 of file huv_generated_Radial_Dam_dry.cpp.

4.31.3 Member Function Documentation

4.31.3.1 void huv_generated_Radial_Dam_dry::initialization (TAB & *h*, TAB & *u*, TAB & *v*) [virtual]

Implements Initialization_huv.

Definition at line 61 of file huv_generated_Radial_Dam_dry.cpp.

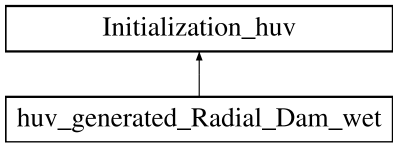
The documentation for this class was generated from the following files:

- Headers/libinitialization/huv_generated_Radial_Dam_dry.hpp
- Sources/libinitialization/huv_generated_Radial_Dam_dry.cpp

4.32 huv_generated_Radial_Dam_wet Class Reference

`#include <huv_generated_Radial_Dam_wet.hpp>`

Inheritance diagram for huv_generated_Radial_Dam_wet:



Public Member Functions

- huv_generated_Radial_Dam_wet (parameters &)

  Constructor.
- void initialization (TAB &, TAB &, TAB &)
- virtual ∼huv_generated_Radial_Dam_wet ()

4.32.1 Detailed Description

Definition at line 62 of file huv_generated_Radial_Dam_wet.hpp.

4.32.2 Constructor & Destructor Documentation

4.32.2.1 huv_generated_Radial_Dam_wet::huv_generated_Radial_Dam_wet (parameters & *par*)

Constructor.

Definition at line 47 of file huv_generated_Radial_Dam_wet.cpp.

4.32.2.2 huv_generated_Radial_Dam_wet::∼huv_generated_Radial_Dam_wet ( ) [virtual]

Definition at line 60 of file huv_generated_Radial_Dam_wet.cpp.

4.32.3 Member Function Documentation

4.32.3.1 void huv_generated_Radial_Dam_wet::initialization (TAB & *h*, TAB & *u*, TAB & *v*) [virtual]

Implements Initialization_huv.

Definition at line 62 of file huv_generated_Radial_Dam_wet.cpp.

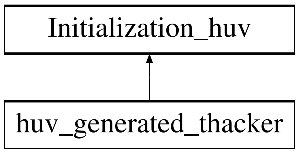
The documentation for this class was generated from the following files:

- Headers/libinitialization/huv_generated_Radial_Dam_wet.hpp
- Sources/libinitialization/huv_generated_Radial_Dam_wet.cpp

4.33 huv_generated_thacker Class Reference

`#include <huv_generated_thacker.hpp>`

Inheritance diagram for huv_generated_thacker:



Public Member Functions

- huv_generated_thacker (parameters &)

  Constructor.
- void initialization (TAB &, TAB &, TAB &)
- virtual ∼huv_generated_thacker ()

4.33.1 Detailed Description

Definition at line 62 of file huv_generated_thacker.hpp.

4.33.2 Constructor & Destructor Documentation

4.33.2.1 huv_generated_thacker::huv_generated_thacker (**parameters & *par***)

Constructor.

Definition at line 47 of file huv_generated_thacker.cpp.

4.33.2.2 huv_generated_thacker::∼huv_generated_thacker ( ) `[virtual]`

Definition at line 62 of file huv_generated_thacker.cpp.

4.33.3 Member Function Documentation

4.33.3.1 void huv_generated_thacker::initialization (**TAB & *h*, TAB & *u*, TAB & *v***) `[virtual]`

Implements Initialization_huv.

Definition at line 64 of file huv_generated_thacker.cpp.

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

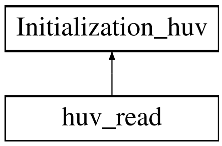
- Headers/libinitialization/huv_generated_thacker.hpp
- Sources/libinitialization/huv_generated_thacker.cpp

4.34 huv_read Class Reference

Class for the initialization of the water height and velocity read in a file.

#include <huv_read.hpp>

Inheritance diagram for huv_read:



Public Member Functions

- huv_read (parameters &)

    Constructor.
- virtual ∼huv_read ()

    Destructor.
- void initialization (TAB &, TAB &, TAB &)

4.34.1 Detailed Description

Class for the initialization of the water height and velocity read in a file.

Definition at line 60 of file huv_read.hpp.

4.34.2 Constructor & Destructor Documentation

4.34.2.1 huv_read::huv_read (**parameters &** *par*)

Constructor.

Definition at line 46 of file huv_read.cpp.

4.34.2.2 huv_read::∼huv_read ( ) `[virtual]`

Destructor.

Definition at line 49 of file huv_read.cpp.

4.34.3 Member Function Documentation

4.34.3.1 void huv_read::initialization (TAB & *h*, TAB & *u*, TAB & *v*) [virtual]

Implements Initialization_huv.

Definition at line 52 of file huv_read.cpp.

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

- Headers/libinitialization/huv_read.hpp
- Sources/libinitialization/huv_read.cpp

4.35 hydrostatic Class Reference

class which allows to perform the hydrostatic reconstruction.

```
#include <hydrostatic.hpp>
```

Public Member Functions

- hydrostatic ()

  Constructor.
- void calcul (SCALAR, SCALAR, SCALAR)

  Calculates the hydrostatic reconstruction.
- SCALAR hg ()

  Returns left height of water reconstruction.
- SCALAR hd ()

  Returns right height of water reconstruction.

4.35.1 Detailed Description

class which allows to perform the hydrostatic reconstruction.

At first order, we get $U_{\bullet L}$ and $U_{\bullet R}$ variables from $U_{\bullet}$ variables thanks to the hydrostatic reconstruction

$$\left\{ \begin{array}{l} U_{i+1/2L} = (h_{i+1/2L}, h_{i+1/2L}u_i), \; h_{i+1/2L} = (h_i + z_i - \max(z_i, z_{i+1}))_+ \\ U_{i+1/2R} = (h_{i+1/2R}, h_{i+1/2R}u_i), \; h_{i+1/2R} = (h_{i+1} + z_{i+1} - \max(z_i, z_{i+1}))_+ \end{array} \right.$$

Definition at line 66 of file hydrostatic.hpp.

4.35.2 Constructor & Destructor Documentation

4.35.2.1 hydrostatic::hydrostatic ( )

Constructor.

Definition at line 49 of file hydrostatic.cpp.

4.35.3 Member Function Documentation

4.35.3.1 void hydrostatic::calcul (SCALAR *hg*, SCALAR *hd*, SCALAR *dz*)

Calculates the hydrostatic reconstruction.

Definition at line 53 of file hydrostatic.cpp.

4.35.3.2 SCALAR hydrostatic::hd ( )

Returns right height of water reconstruction.

Definition at line 64 of file hydrostatic.cpp.

4.35.3.3 SCALAR hydrostatic::hg ( )

Returns left height of water reconstruction.

Definition at line 59 of file hydrostatic.cpp.

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

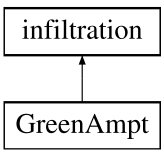
- Headers/libreconstructions/hydrostatic.hpp
- Sources/libreconstructions/hydrostatic.cpp

4.36 infiltration Class Reference

Class which allows to calculate the infiltration rate thanks to Green-Ampt model.

`#include <infiltration.hpp>`

Inheritance diagram for infiltration:



Public Member Functions

- infiltration (parameters &)

  Constructor.
- virtual void calcul (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR)=0

- SCALAR get_hmod () const
- SCALAR get_Vin () const
- virtual ∼infiltration ()

Protected Attributes

- const int Nxcell
- const int Nycell
- SCALAR hmod
- SCALAR Vin

4.36.1 Detailed Description

Class which allows to calculate the infiltration rate thanks to Green-Ampt model.

Definition at line 62 of file infiltration.hpp.

4.36.2 Constructor & Destructor Documentation

4.36.2.1 infiltration::infiltration (**parameters & *par***)

Constructor.

Definition at line 45 of file infiltration.cpp.

4.36.2.2 infiltration::∼infiltration ( ) `[virtual]`

Definition at line 58 of file infiltration.cpp.

4.36.3 Member Function Documentation

4.36.3.1 virtual void infiltration::calcul (**SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR , SCALAR**) `[pure virtual]`

Implemented in GreenAmpt.

4.36.3.2 SCALAR infiltration::get_hmod ( ) const

Definition at line 50 of file infiltration.cpp.

4.36.3.3 SCALAR infiltration::get_Vin ( ) const

Definition at line 54 of file infiltration.cpp.

4.36.4 Member Data Documentation

4.36.4.1 SCALAR infiltration::hmod [protected]

Definition at line 66 of file infiltration.hpp.

4.36.4.2 const int infiltration::Nxcell [protected]

Definition at line 64 of file infiltration.hpp.

4.36.4.3 const int infiltration::Nycell [protected]

Definition at line 65 of file infiltration.hpp.

4.36.4.4 SCALAR infiltration::Vin [protected]

Definition at line 66 of file infiltration.hpp.

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

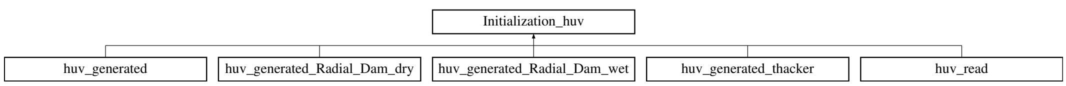
- Headers/librain_infiltration/infiltration.hpp
- Sources/librain_infiltration/infiltration.cpp

4.37 Initialization_huv Class Reference

Class for the initialization of the water height and velocity.

`#include <Initialization_huv.hpp>`

Inheritance diagram for Initialization_huv:



Public Member Functions

- Initialization_huv (parameters &)
- virtual void initialization (TAB &, TAB &, TAB &)=0
- virtual ∼Initialization_huv ()

Protected Attributes

- TAB h

- TAB u
- TAB v
- int Nxcell
- int Nycell
- SCALAR dx
- SCALAR dy

4.37.1 Detailed Description

Class for the initialization of the water height and velocity.

Definition at line 60 of file Initialization_huv.hpp.

4.37.2 Constructor & Destructor Documentation

4.37.2.1 Initialization_huv::Initialization_huv (parameters & *par*)

Definition at line 45 of file Initialization_huv.cpp.

4.37.2.2 Initialization_huv::∼Initialization_huv ( ) [virtual]

Definition at line 61 of file Initialization_huv.cpp.

4.37.3 Member Function Documentation

4.37.3.1 virtual void Initialization_huv::initialization (TAB & , TAB & , TAB &) [pure virtual]

Implemented in huv_generated, huv_generated_Radial_Dam_dry, huv_generated_Radial_-Dam_wet, huv_generated_thacker, and huv_read.

4.37.4 Member Data Documentation

4.37.4.1 SCALAR Initialization_huv::dx [protected]

Definition at line 64 of file Initialization_huv.hpp.

4.37.4.2 SCALAR Initialization_huv::dy [protected]

Definition at line 64 of file Initialization_huv.hpp.

4.37.4.3 TAB Initialization_huv::h [protected]

Definition at line 62 of file Initialization_huv.hpp.

4.37.4.4 int Initialization_huv::Nxcell [protected]

Definition at line 63 of file Initialization_huv.hpp.

4.37.4.5 int Initialization_huv::Nycell [protected]

Definition at line 63 of file Initialization_huv.hpp.

4.37.4.6 TAB Initialization_huv::u [protected]

Definition at line 62 of file Initialization_huv.hpp.

4.37.4.7 TAB Initialization_huv::v [protected]

Definition at line 62 of file Initialization_huv.hpp.

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

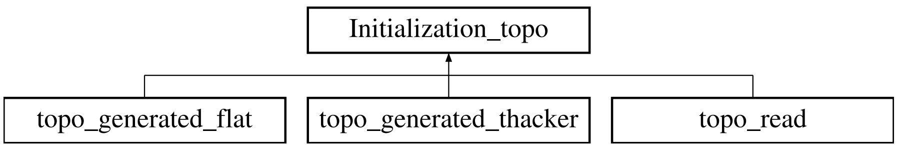
- Headers/libinitialization/Initialization_huv.hpp
- Sources/libinitialization/Initialization_huv.cpp

4.38 Initialization_topo Class Reference

Class for the initialization of the topography.

`#include <Initialization_topo.hpp>`

Inheritance diagram for Initialization_topo:



Public Member Functions

- Initialization_topo (parameters &)
- virtual void initialization (TAB &)=0
- virtual ∼Initialization_topo ()

Protected Attributes

- int Nxcell

- int Nycell
- SCALAR dx
- SCALAR dy

4.38.1 Detailed Description

Class for the initialization of the topography.

Definition at line 60 of file Initialization_topo.hpp.

4.38.2 Constructor & Destructor Documentation

4.38.2.1 Initialization_topo::Initialization_topo (parameters & *par*)

Definition at line 45 of file Initialization_topo.cpp.

4.38.2.2 Initialization_topo::~Initialization_topo ( ) [virtual]

Definition at line 53 of file Initialization_topo.cpp.

4.38.3 Member Function Documentation

4.38.3.1 virtual void Initialization_topo::initialization (TAB &) [pure virtual]

Implemented in topo_generated_flat, topo_generated_thacker, and topo_read.

4.38.4 Member Data Documentation

4.38.4.1 SCALAR Initialization_topo::dx [protected]

Definition at line 63 of file Initialization_topo.hpp.

4.38.4.2 SCALAR Initialization_topo::dy [protected]

Definition at line 63 of file Initialization_topo.hpp.

4.38.4.3 int Initialization_topo::Nxcell [protected]

Definition at line 62 of file Initialization_topo.hpp.

4.38.4.4 int Initialization_topo::Nycell [protected]

Definition at line 62 of file Initialization_topo.hpp.

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

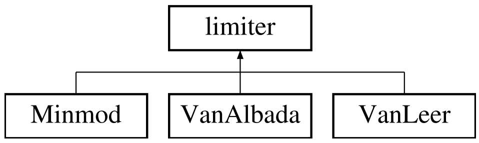
- Headers/libinitialization/Initialization_topo.hpp
- Sources/libinitialization/Initialization_topo.cpp

4.39 limiter Class Reference

class which allows to calculate the slope limiter.

`#include <limiter.hpp>`

Inheritance diagram for limiter:



Public Member Functions

- limiter ()

  Constructor.
- virtual void calcul (SCALAR, SCALAR)=0
- SCALAR get_rec () const
- virtual ∼limiter ()

  Destructor.

Protected Attributes

- SCALAR rec

4.39.1 Detailed Description

class which allows to calculate the slope limiter.

Definition at line 59 of file limiter.hpp.

4.39.2 Constructor & Destructor Documentation

4.39.2.1 limiter::limiter ( )

Constructor.

Definition at line 45 of file limiter.cpp.

4.39.2.2 limiter::∼limiter ( ) [virtual]

Destructor.

Definition at line 53 of file limiter.cpp.

4.39.3 Member Function Documentation

4.39.3.1 virtual void limiter::calcul (SCALAR , SCALAR) [pure virtual]

Implemented in Minmod, VanAlbada, and VanLeer.

4.39.3.2 SCALAR limiter::get_rec ( ) const

Definition at line 49 of file limiter.cpp.

4.39.4 Member Data Documentation

4.39.4.1 SCALAR limiter::rec [protected]

Definition at line 61 of file limiter.hpp.

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

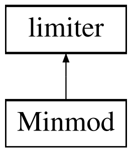
- Headers/liblimitations/limiter.hpp
- Sources/liblimitations/limiter.cpp

4.40 Minmod Class Reference

Class which allows to calculate the minmod of two real numbers (x, y).

`#include <Minmod.hpp>`

Inheritance diagram for Minmod:



Public Member Functions

- Minmod ()

  Constructor.
- void calcul (SCALAR, SCALAR)

  calculates the minmod of a and b
- virtual ∼Minmod ()

  Destructor.

4.40.1 Detailed Description

Class which allows to calculate the minmod of two real numbers (x,y).

[**?**]

$$minmod(x,y) = \begin{cases} \min(x,y)\ if\ x,y \geq 0 \\ \max(x,y)\ if\ x,y \leq 0 \\ 0\ else \end{cases}.$$

Definition at line 68 of file Minmod.hpp.

4.40.2 Constructor & Destructor Documentation

4.40.2.1 Minmod::Minmod ( )

Constructor.

Definition at line 45 of file Minmod.cpp.

4.40.2.2 Minmod::∼Minmod ( ) [virtual]

Destructor.

Definition at line 58 of file Minmod.cpp.

4.40.3 Member Function Documentation

4.40.3.1 void Minmod::calcul (SCALAR *a*, SCALAR *b*) [virtual]

calculates the minmod of a and b

Implements limiter.

Definition at line 48 of file Minmod.cpp.

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

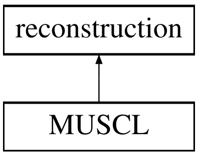
- Headers/liblimitations/Minmod.hpp
- Sources/liblimitations/Minmod.cpp

4.41 MUSCL Class Reference

class which allows to calculate MUSCL reconstruction in space

```
#include <MUSCL.hpp>
```

Inheritance diagram for MUSCL:



Public Member Functions

- MUSCL (parameters &, TAB &)

    Constructor.
- void calcul (TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &)

    Calculates the MUSCL reconstruction in space.
- ∼MUSCL ()

    Destructor.

Protected Attributes

- int Nxcell
- int Nycell

4.41.1 Detailed Description

class which allows to calculate MUSCL reconstruction in space

In order to get a second order in space, we can use a MUSCL reconstruction. A scalar fonction $s \in R$ is reconstructed by the MUSCL reconstruction (Monotonic Upwind

Scheme Conservation Law)

$$s_{i-1/2+} = s_i - \frac{\Delta x}{2} D_{mm} s_i \ \ and \quad s_{i+1/2-} = s_i + \frac{\Delta x}{2} D_{mm} s_i$$

where D_{mm} writes

$$D_{mm} s_i = minmod\left(\frac{s_i - s_{i-1}}{\Delta x}, \frac{s_{i+1} - s_i}{\Delta x}\right)$$

with the Minmod slope limiter.

Definition at line 73 of file MUSCL.hpp.

4.41.2 Constructor & Destructor Documentation

4.41.2.1 MUSCL::MUSCL (parameters & *par*, TAB & *z*)

Constructor.

Definition at line 46 of file MUSCL.cpp.

4.41.2.2 MUSCL::∼MUSCL ( )

Destructor.

Definition at line 226 of file MUSCL.cpp.

4.41.3 Member Function Documentation

4.41.3.1 void MUSCL::calcul (TAB & *h*, TAB & *u*, TAB & *v*, TAB & *z*, TAB & *delzc1*, TAB & *delzc2*, TAB & *delz1*, TAB & *delz2*, TAB & *h1r*, TAB & *u1r*, TAB & *v1r*, TAB & *h1l*, TAB & *u1l*, TAB & *v1l*, TAB & *h2r*, TAB & *u2r*, TAB & *v2r*, TAB & *h2l*, TAB & *u2l*, TAB & *v2l*) [virtual]

Calculates the MUSCL reconstruction in space.

Implements reconstruction.

Definition at line 96 of file MUSCL.cpp.

4.41.4 Member Data Documentation

4.41.4.1 int MUSCL::Nxcell [protected]

Definition at line 76 of file MUSCL.hpp.

4.41.4.2 int MUSCL::Nycell [protected]

Definition at line 76 of file MUSCL.hpp.

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

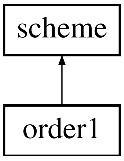
- Headers/libreconstructions/MUSCL.hpp
- Sources/libreconstructions/MUSCL.cpp

4.42 order1 Class Reference

class which allows to perform the first order scheme

`#include <order1.hpp>`

Inheritance diagram for order1:



Public Member Functions

- order1 (parameters &)

  Constructor.
- virtual ∼order1 ()

  Destructor.
- void calcul ()

4.42.1 Detailed Description

class which allows to perform the first order scheme

Definition at line 61 of file order1.hpp.

4.42.2 Constructor & Destructor Documentation

4.42.2.1 order1::order1 (parameters & *par*)

Constructor.

Definition at line 45 of file order1.cpp.

4.42.2.2 order1::∼order1 ( ) [virtual]

Destructor.

Definition at line 47 of file order1.cpp.

4.42.3 Member Function Documentation

4.42.3.1 void order1::calcul ( ) [virtual]

Implements scheme.

Definition at line 53 of file order1.cpp.

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

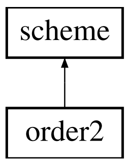
- Headers/libschemes/order1.hpp
- Sources/libschemes/order1.cpp

4.43 order2 Class Reference

class which allows to perform the second order scheme

`#include <order2.hpp>`

Inheritance diagram for order2:



Public Member Functions

- order2 (parameters &)

  Constructor.
- virtual ∼order2 ()

  Destructor.
- void calcul ()

Protected Attributes

- SCALAR tmp
- TAB hsa
- TAB usa
- TAB vsa
- TAB qsa1
- TAB qsa2
- TAB Vin1
- TAB Vin2
- choice_reconstruction ∗ rec

4.43.1 Detailed Description

class which allows to perform the second order scheme

Definition at line 64 of file order2.hpp.

4.43.2 Constructor & Destructor Documentation

4.43.2.1 order2::order2 (**parameters & *par***)

Constructor.

Definition at line 45 of file order2.cpp.

4.43.2.2 order2::∼order2 ( ) [virtual]

Destructor.

Definition at line 80 of file order2.cpp.

4.43.3 Member Function Documentation

4.43.3.1 void order2::calcul ( ) [virtual]

Implements scheme.

Definition at line 109 of file order2.cpp.

4.43.4 Member Data Documentation

4.43.4.1 TAB order2::hsa [protected]

Definition at line 67 of file order2.hpp.

4.43.4.2 TAB order2::qsa1 [protected]

Definition at line 67 of file order2.hpp.

4.43.4.3 TAB order2::qsa2 [protected]

Definition at line 67 of file order2.hpp.

4.43.4.4 choice_reconstruction∗ order2::rec [protected]

Definition at line 70 of file order2.hpp.

4.43.4.5 SCALAR order2::tmp [protected]

Definition at line 66 of file order2.hpp.

4.43.4.6 TAB order2::usa [protected]

Definition at line 67 of file order2.hpp.

4.43.4.7 TAB order2::Vin1 [protected]

Definition at line 68 of file order2.hpp.

4.43.4.8 TAB order2::Vin2 [protected]

Definition at line 68 of file order2.hpp.

4.43.4.9 TAB order2::vsa [protected]

Definition at line 67 of file order2.hpp.

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

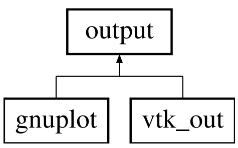
- Headers/libschemes/order2.hpp
- Sources/libschemes/order2.cpp

4.44 output Class Reference

Class which allows to write the result in the output file.

`#include <output.hpp>`

Inheritance diagram for output:



Public Member Functions

- output (parameters &)

  Constructor.
- virtual void write (TAB, TAB, TAB, TAB, SCALAR)=0

- void check_vol (SCALAR, SCALAR, TAB, TAB, SCALAR, SCALAR &, SCALAR &, SCALAR)
- void result (SCALAR, SCALAR, SCALAR, SCALAR, const SCALAR, const int, SCALAR)
- void initial (SCALAR, SCALAR, TAB, TAB, TAB, TAB)
- void final (SCALAR, SCALAR, TAB, TAB, TAB, TAB)
- SCALAR boundaries_flux (SCALAR, TAB &, TAB &, SCALAR, SCALAR, int, int, SCALAR &, TAB &)
- void boundaries_flux_LR (SCALAR, TAB)
- void boundaries_flux_BT (SCALAR, TAB)
- virtual ∼output ()

  Destructor.

Protected Attributes

- int Nxcell
- int Nycell
- SCALAR dx
- SCALAR dy
- string outputDirectory
- string namefile
- string namefile_check_volum
- string namefile_res
- string namefile_init
- string namefile_final
- string namefile_Bound_flux
- string namefile_Bound_flux_BT
- string namefile_Bound_flux_LR
- SCALAR fluxNycell_cum_T
- SCALAR fluxNxcell_cum_T
- SCALAR fluxy0_cum_T
- SCALAR fluxx0_cum_T
- SCALAR difference_f1
- SCALAR difference_g1
- SCALAR Vol_inf_tot

4.44.1 Detailed Description

Class which allows to write the result in the output file.

Definition at line 61 of file output.hpp.

4.44.2 Constructor & Destructor Documentation

4.44.2.1 output::output (parameters & *par*)

Constructor.

Definition at line 46 of file output.cpp.

4.44.2.2 output::∼output ( ) [virtual]

Destructor.

Definition at line 131 of file output.cpp.

4.44.3 Member Function Documentation

4.44.3.1 SCALAR output::boundaries_flux (SCALAR *time*, TAB & *flux_u*, TAB & *flux_v*, SCALAR *dt*, SCALAR *dt_first*, int *ORDER*, int *verif*, SCALAR & *Rain_tot*, TAB & *Rain*)

Definition at line 216 of file output.cpp.

4.44.3.2 void output::boundaries_flux_BT (SCALAR *time*, TAB *BT_flux*)

Definition at line 264 of file output.cpp.

4.44.3.3 void output::boundaries_flux_LR (SCALAR *time*, TAB *LR_flux*)

Definition at line 253 of file output.cpp.

4.44.3.4 void output::check_vol (SCALAR *time*, SCALAR *dt*, TAB *h*, TAB *Vin*, SCALAR *Vol_rain_tot*, SCALAR & *Vol_inf*, SCALAR & *Vol_of*, SCALAR *bound_flux_tot*)

Definition at line 160 of file output.cpp.

4.44.3.5 void output::final (SCALAR *dx*, SCALAR *dy*, TAB *z*, TAB *h*, TAB *u*, TAB *v*)

Definition at line 276 of file output.cpp.

4.44.3.6 void output::initial (SCALAR *dx*, SCALAR *dy*, TAB *z*, TAB *h*, TAB *u*, TAB *v*)

Definition at line 135 of file output.cpp.

4.44.3.7 void output::result (SCALAR *time*, SCALAR *Vol_rain*, SCALAR *Vol_inf*, SCALAR *Vol_of*, const SCALAR *Froude*, const int *nbiter*, SCALAR *vol_output*)

Definition at line 179 of file output.cpp.

4.44.3.8 virtual void output::write (TAB , TAB , TAB , TAB , SCALAR) [pure virtual]

Implemented in gnuplot, and vtk_out.

4.44.4 Member Data Documentation

4.44.4.1 SCALAR output::difference_f1 [protected]

Definition at line 69 of file output.hpp.

4.44.4.2 SCALAR output::difference_g1 [protected]

Definition at line 69 of file output.hpp.

4.44.4.3 SCALAR output::dx [protected]

Definition at line 64 of file output.hpp.

4.44.4.4 SCALAR output::dy [protected]

Definition at line 64 of file output.hpp.

4.44.4.5 SCALAR output::fluxNxcell_cum_T [protected]

Definition at line 68 of file output.hpp.

4.44.4.6 SCALAR output::fluxNycell_cum_T [protected]

Definition at line 68 of file output.hpp.

4.44.4.7 SCALAR output::fluxx0_cum_T [protected]

Definition at line 68 of file output.hpp.

4.44.4.8 SCALAR output::fluxy0_cum_T [protected]

Definition at line 68 of file output.hpp.

4.44.4.9 string output::namefile [protected]

Definition at line 65 of file output.hpp.

4.44.4.10 string output::namefile_Bound_flux [protected]

Definition at line 67 of file output.hpp.

4.44.4.11 string output::namefile_Bound_flux_BT [protected]

Definition at line 67 of file output.hpp.

4.44.4.12 string output::namefile_Bound_flux_LR [protected]

Definition at line 67 of file output.hpp.

4.44.4.13 string output::namefile_check_volum [protected]

Definition at line 66 of file output.hpp.

4.44.4.14 string output::namefile_final [protected]

Definition at line 66 of file output.hpp.

4.44.4.15 string output::namefile_init [protected]

Definition at line 66 of file output.hpp.

4.44.4.16 string output::namefile_res [protected]

Definition at line 66 of file output.hpp.

4.44.4.17 int output::Nxcell [protected]

Definition at line 63 of file output.hpp.

4.44.4.18 int output::Nycell [protected]

Definition at line 63 of file output.hpp.

4.44.4.19 string output::outputDirectory [protected]

Definition at line 65 of file output.hpp.

4.44.4.20 SCALAR output::Vol_inf_tot [protected]

Definition at line 70 of file output.hpp.

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

- Headers/libsave/output.hpp
- Sources/libsave/output.cpp

4.45 parameters Class Reference

Class which gets the initialization parameters.

`#include <parameters.hpp>`

Public Member Functions

- parameters ()
- void setparameters (const char ∗)
- virtual ∼parameters ()
- int get_Nxcell () const
- int get_Nycell () const
- SCALAR get_T () const
- int get_scheme_type () const
- SCALAR get_dtfix () const
- SCALAR get_cflfix () const
- SCALAR get_dx () const
- SCALAR get_dy () const
- int get_Lbound () const
- int get_Rbound () const
- int get_Bbound () const
- int get_Tbound () const
- int get_flux () const
- int get_order () const
- int get_rec () const
- int get_fric () const
- int get_lim () const
- int get_inf () const
- SCALAR get_amortENO () const
- SCALAR get_modifENO () const
- SCALAR get_Kc () const

- SCALAR get_Ks () const
- SCALAR get_dtheta () const
- SCALAR get_Psi () const
- SCALAR get_zcrust () const
- SCALAR get_imax () const
- SCALAR get_c () const
- int get_topo () const
- int get_huv () const
- int get_rain () const
- string get_topographyNameFile (void) const
- string get_topographyNameFileS () const
- string get_huvNameFile (void) const
- string get_huvNameFileS (void) const
- string get_rainNameFile (void) const
- string get_rainNameFileS (void) const
- string get_outputDirectory (void) const
- string get_suffix (void) const
- int get_output () const
- SCALAR get_left_imp_discharge () const
- SCALAR get_left_imp_h () const
- SCALAR get_right_imp_discharge () const
- SCALAR get_right_imp_h () const
- SCALAR get_bottom_imp_discharge () const
- SCALAR get_bottom_imp_h () const
- SCALAR get_top_imp_discharge () const
- SCALAR get_top_imp_h () const

Protected Attributes

- SCALAR cfl_fix
- SCALAR dt_fix
- int scheme_type
- int Nxcell
- int Nycell
- SCALAR T
- SCALAR dx
- SCALAR dy
- SCALAR L
- SCALAR l
- int Lbound
- int Rbound
- int Bbound
- int Tbound
- int flux
- int order
- int rec

- int fric
- int lim
- int inf
- int topo
- int huv_initialization
- int rain
- SCALAR amortENO
- SCALAR modifENO
- SCALAR frotcoef
- SCALAR c
- string topography_namefile
- string topography_namefile_short
- string huv_namefile
- string huv_namefile_short
- string rain_namefile
- string rain_namefile_short
- string output_directory
- string suffix_outputs
- char ∗ tempo_topo
- char ∗ tempo_huv
- char ∗ tempo_rain
- char ∗ tempo_suffix
- char ∗ suffix
- SCALAR Ks
- SCALAR dtheta
- SCALAR Psi
- SCALAR zcrust
- SCALAR Kc
- SCALAR imax
- string namefile
- int output_format
- SCALAR left_imp_discharge
- SCALAR left_imp_h
- SCALAR right_imp_discharge
- SCALAR right_imp_h
- SCALAR bottom_imp_discharge
- SCALAR bottom_imp_h
- SCALAR top_imp_discharge
- SCALAR top_imp_h

4.45.1 Detailed Description

Class which gets the initialization parameters.

Definition at line 59 of file parameters.hpp.

4.45.2 Constructor & Destructor Documentation

4.45.2.1 parameters::parameters ( )

Definition at line 53 of file parameters.cpp.

4.45.2.2 parameters::∼parameters ( ) [virtual]

Definition at line 453 of file parameters.cpp.

4.45.3 Member Function Documentation

4.45.3.1 SCALAR parameters::get_amortENO ( ) const

Definition at line 521 of file parameters.cpp.

4.45.3.2 int parameters::get_Bbound ( ) const

Definition at line 492 of file parameters.cpp.

4.45.3.3 SCALAR parameters::get_bottom_imp_discharge ( ) const

Definition at line 637 of file parameters.cpp.

4.45.3.4 SCALAR parameters::get_bottom_imp_h ( ) const

Definition at line 643 of file parameters.cpp.

4.45.3.5 SCALAR parameters::get_c ( ) const

Definition at line 561 of file parameters.cpp.

4.45.3.6 SCALAR parameters::get_cflfix ( ) const

Definition at line 472 of file parameters.cpp.

4.45.3.7 SCALAR parameters::get_dtfix ( ) const

Definition at line 469 of file parameters.cpp.

4.45.3.8 SCALAR parameters::get_dtheta ( ) const

Definition at line 541 of file parameters.cpp.

4.45.3.9 SCALAR parameters::get_dx ( ) const

Definition at line 476 of file parameters.cpp.

4.45.3.10 SCALAR parameters::get_dy ( ) const

Definition at line 480 of file parameters.cpp.

4.45.3.11 int parameters::get_flux ( ) const

Definition at line 500 of file parameters.cpp.

4.45.3.12 int parameters::get_fric ( ) const

Definition at line 512 of file parameters.cpp.

4.45.3.13 int parameters::get_huv ( ) const

Definition at line 579 of file parameters.cpp.

4.45.3.14 string parameters::get_huvNameFile (void) const

Definition at line 583 of file parameters.cpp.

4.45.3.15 string parameters::get_huvNameFileS (void) const

Definition at line 588 of file parameters.cpp.

4.45.3.16 SCALAR parameters::get_imax ( ) const

Definition at line 553 of file parameters.cpp.

4.45.3.17 int parameters::get_inf ( ) const

Definition at line 529 of file parameters.cpp.

4.45.3.18 SCALAR parameters::get_Kc ( ) const

Definition at line 533 of file parameters.cpp.

4.45.3.19 SCALAR parameters::get_Ks ( ) const

Definition at line 537 of file parameters.cpp.

4.45.3.20 int parameters::get_Lbound ( ) const

Definition at line 484 of file parameters.cpp.

4.45.3.21 SCALAR parameters::get_left_imp_discharge ( ) const

Definition at line 619 of file parameters.cpp.

4.45.3.22 SCALAR parameters::get_left_imp_h ( ) const

Definition at line 625 of file parameters.cpp.

4.45.3.23 int parameters::get_lim ( ) const

Definition at line 517 of file parameters.cpp.

4.45.3.24 SCALAR parameters::get_modifENO ( ) const

Definition at line 525 of file parameters.cpp.

4.45.3.25 int parameters::get_Nxcell ( ) const

Definition at line 455 of file parameters.cpp.

4.45.3.26 int parameters::get_Nycell ( ) const

Definition at line 459 of file parameters.cpp.

4.45.3.27 int parameters::get_order ( ) const

Definition at line 504 of file parameters.cpp.

4.45.3.28 int parameters::get_output ( ) const

Definition at line 615 of file parameters.cpp.

4.45.3.29 string parameters::get_outputDirectory (void) const

Definition at line 607 of file parameters.cpp.

4.45.3.30 SCALAR parameters::get_Psi ( ) const

Definition at line 545 of file parameters.cpp.

4.45.3.31 int parameters::get_rain ( ) const

Definition at line 593 of file parameters.cpp.

4.45.3.32 string parameters::get_rainNameFile (void) const

Definition at line 597 of file parameters.cpp.

4.45.3.33 string parameters::get_rainNameFileS (void) const

Definition at line 602 of file parameters.cpp.

4.45.3.34 int parameters::get_Rbound ( ) const

Definition at line 488 of file parameters.cpp.

4.45.3.35 int parameters::get_rec ( ) const

Definition at line 508 of file parameters.cpp.

4.45.3.36 SCALAR parameters::get_right_imp_discharge ( ) const

Definition at line 628 of file parameters.cpp.

4.45.3.37 SCALAR parameters::get_right_imp_h ( ) const

Definition at line 634 of file parameters.cpp.

4.45.3.38 int parameters::get_scheme_type ( ) const

Definition at line 466 of file parameters.cpp.

4.45.3.39 string parameters::get_suffix (void) const

Definition at line 611 of file parameters.cpp.

4.45.3.40 SCALAR parameters::get_T ( ) const

Definition at line 463 of file parameters.cpp.

4.45.3.41 int parameters::get_Tbound ( ) const

Definition at line 496 of file parameters.cpp.

4.45.3.42 SCALAR parameters::get_top_imp_discharge ( ) const

Definition at line 646 of file parameters.cpp.

4.45.3.43 SCALAR parameters::get_top_imp_h ( ) const

Definition at line 652 of file parameters.cpp.

4.45.3.44 int parameters::get_topo ( ) const

Definition at line 575 of file parameters.cpp.

4.45.3.45 string parameters::get_topographyNameFile (void) const

Definition at line 566 of file parameters.cpp.

4.45.3.46 string parameters::get_topographyNameFileS (void) const

Definition at line 571 of file parameters.cpp.

4.45.3.47 SCALAR parameters::get_zcrust ( ) const

Definition at line 549 of file parameters.cpp.

4.45.3.48 void parameters::setparameters (const char * *filename*)

Definition at line 62 of file parameters.cpp.

4.45.4 Member Data Documentation

4.45.4.1 SCALAR parameters::amortENO [protected]

Definition at line 67 of file parameters.hpp.

4.45.4.2 int parameters::Bbound [protected]

Definition at line 65 of file parameters.hpp.

4.45.4.3 SCALAR parameters::bottom_imp_discharge [protected]

Definition at line 83 of file parameters.hpp.

4.45.4.4 SCALAR parameters::bottom_imp_h [protected]

Definition at line 83 of file parameters.hpp.

4.45.4.5 SCALAR parameters::c [protected]

Definition at line 68 of file parameters.hpp.

4.45.4.6 SCALAR parameters::cfl_fix [protected]

Definition at line 61 of file parameters.hpp.

4.45.4.7 SCALAR parameters::dt_fix [protected]

Definition at line 61 of file parameters.hpp.

4.45.4.8 SCALAR parameters::dtheta [protected]

Definition at line 77 of file parameters.hpp.

4.45.4.9 SCALAR parameters::dx [protected]

Definition at line 64 of file parameters.hpp.

4.45.4.10 SCALAR parameters::dy [protected]

Definition at line 64 of file parameters.hpp.

4.45.4.11 int parameters::flux [protected]

Definition at line 66 of file parameters.hpp.

4.45.4.12 int parameters::fric [protected]

Definition at line 66 of file parameters.hpp.

4.45.4.13 SCALAR parameters::frotcoef [protected]

Definition at line 67 of file parameters.hpp.

4.45.4.14 int parameters::huv_initialization [protected]

Definition at line 66 of file parameters.hpp.

4.45.4.15 string parameters::huv_namefile [protected]

Definition at line 72 of file parameters.hpp.

4.45.4.16 string parameters::huv_namefile_short [protected]

Definition at line 72 of file parameters.hpp.

4.45.4.17 SCALAR parameters::imax [protected]

Definition at line 77 of file parameters.hpp.

4.45.4.18 int parameters::inf [protected]

Definition at line 66 of file parameters.hpp.

4.45.4.19 SCALAR parameters::Kc [protected]

Definition at line 77 of file parameters.hpp.

4.45.4.20 SCALAR parameters::Ks [protected]

Definition at line 77 of file parameters.hpp.

4.45.4.21 SCALAR parameters::L [protected]

Definition at line 64 of file parameters.hpp.

4.45.4.22 SCALAR parameters::l [protected]

Definition at line 64 of file parameters.hpp.

4.45.4.23 int parameters::Lbound [protected]

Definition at line 65 of file parameters.hpp.

4.45.4.24 SCALAR parameters::left_imp_discharge [protected]

Definition at line 83 of file parameters.hpp.

4.45.4.25 SCALAR parameters::left_imp_h [protected]

Definition at line 83 of file parameters.hpp.

4.45.4.26 int parameters::lim [protected]

Definition at line 66 of file parameters.hpp.

4.45.4.27 SCALAR parameters::modifENO [protected]

Definition at line 67 of file parameters.hpp.

4.45.4.28 string parameters::namefile [protected]

Definition at line 80 of file parameters.hpp.

4.45.4.29 int parameters::Nxcell [protected]

Definition at line 63 of file parameters.hpp.

4.45.4.30 int parameters::Nycell [protected]

Definition at line 63 of file parameters.hpp.

4.45.4.31 int parameters::order [protected]

Definition at line 66 of file parameters.hpp.

4.45.4.32 string parameters::output_directory [protected]

Definition at line 74 of file parameters.hpp.

4.45.4.33 int parameters::output_format [protected]

Definition at line 81 of file parameters.hpp.

4.45.4.34 SCALAR parameters::Psi [protected]

Definition at line 77 of file parameters.hpp.

4.45.4.35 int parameters::rain [protected]

Definition at line 66 of file parameters.hpp.

4.45.4.36 string parameters::rain_namefile [protected]

Definition at line 73 of file parameters.hpp.

4.45.4.37 string parameters::rain_namefile_short [protected]

Definition at line 73 of file parameters.hpp.

4.45.4.38 int parameters::Rbound [protected]

Definition at line 65 of file parameters.hpp.

4.45.4.39 int parameters::rec [protected]

Definition at line 66 of file parameters.hpp.

4.45.4.40 SCALAR parameters::right_imp_discharge [protected]

Definition at line 83 of file parameters.hpp.

4.45.4.41 SCALAR parameters::right_imp_h [protected]

Definition at line 83 of file parameters.hpp.

4.45.4.42 int parameters::scheme_type [protected]

Definition at line 62 of file parameters.hpp.

4.45.4.43 char * parameters::suffix [protected]

Definition at line 75 of file parameters.hpp.

4.45.4.44 string parameters::suffix_outputs [protected]

Definition at line 74 of file parameters.hpp.

4.45.4.45 SCALAR parameters::T [protected]

Definition at line 64 of file parameters.hpp.

4.45.4.46 int parameters::Tbound [protected]

Definition at line 65 of file parameters.hpp.

4.45.4.47 char * parameters::tempo_huv [protected]

Definition at line 75 of file parameters.hpp.

4.45.4.48 char * parameters::tempo_rain [protected]

Definition at line 75 of file parameters.hpp.

4.45.4.49 char * parameters::tempo_suffix [protected]

Definition at line 75 of file parameters.hpp.

4.45.4.50 char* parameters::tempo_topo [protected]

Definition at line 75 of file parameters.hpp.

4.45.4.51 SCALAR parameters::top_imp_discharge [protected]

Definition at line 83 of file parameters.hpp.

4.45.4.52 SCALAR parameters::top_imp_h [protected]

Definition at line 83 of file parameters.hpp.

4.45.4.53 int parameters::topo [protected]

Definition at line 66 of file parameters.hpp.

4.45.4.54 string parameters::topography_namefile [protected]

Definition at line 71 of file parameters.hpp.

4.45.4.55 string parameters::topography_namefile_short [protected]

Definition at line 71 of file parameters.hpp.

4.45.4.56 SCALAR parameters::zcrust [protected]

Definition at line 77 of file parameters.hpp.

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

- Headers/libparameters/parameters.hpp
- Sources/libparameters/parameters.cpp

4.46 parser Class Reference

Class that reads the input file writen as description <variable>:: value # comment and keep the values after the "::" ignoring the comments that begin with a "#".

```
#include <parser.hpp>
```

Public Member Functions

- parser (const char ∗)

  Constructor: read the input parameter and copy the data in a tabular Inputs: filename, the name of the paramters file Outputs: data, tabular that contain the strings "description <variable>:: value" nblines, the length of data.
- string GetValue (const char ∗)

For each variable, return the value of the variable writen in the input tabular Inputs: tag, the name of the variable with delimiters Outputs: value, the value of the variable as a string.

- virtual ~parser ()

  Destructor.

Protected Attributes

- int i
- int j
- int nblines
- string ch
- string * data
- string value
- size_t found

4.46.1 Detailed Description

Class that reads the input file writen as description <variable>:: value # comment and keep the values after the "::" ignoring the comments that begin with a "#".

Definition at line 73 of file parser.hpp.

4.46.2 Constructor & Destructor Documentation

4.46.2.1 parser::parser (const char * *fileName*)

Constructor: read the input parameter and copy the data in a tabular Inputs: filename, the name of the paramters file Outputs: data, tabular that contain the strings "description <variable>:: value" nblines, the length of data.

Definition at line 54 of file parser.cpp.

4.46.2.2 parser::~parser ( ) `[virtual]`

Destructor.

Definition at line 140 of file parser.cpp.

4.46.3 Member Function Documentation

4.46.3.1 string parser::GetValue (const char * *tag*)

For each variable, return the value of the variable writen in the input tabular Inputs: tag, the name of the variable with delimiters Outputs: value, the value of the variable as a string.

Definition at line 102 of file parser.cpp.

4.46.4 Member Data Documentation

4.46.4.1 string parser::ch [protected]

Definition at line 76 of file parser.hpp.

4.46.4.2 string* parser::data [protected]

Definition at line 77 of file parser.hpp.

4.46.4.3 size_t parser::found [protected]

Definition at line 79 of file parser.hpp.

4.46.4.4 int parser::i [protected]

Definition at line 75 of file parser.hpp.

4.46.4.5 int parser::j [protected]

Definition at line 75 of file parser.hpp.

4.46.4.6 int parser::nblines [protected]

Definition at line 75 of file parser.hpp.

4.46.4.7 string parser::value [protected]

Definition at line 78 of file parser.hpp.

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

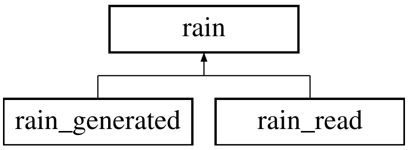
- Headers/libparser/parser.hpp
- Sources/libparser/parser.cpp

4.47 rain Class Reference

Class which allows to choose the rain.

`#include <rain.hpp>`

Inheritance diagram for rain:



Public Member Functions

- rain (parameters &)
- virtual void Rain_func (SCALAR, TAB &)=0
- virtual ∼rain ()

Protected Attributes

- int Nxcell
- int Nycell
- SCALAR dx
- SCALAR dy

4.47.1 Detailed Description

Class which allows to choose the rain.

Definition at line 60 of file rain.hpp.

4.47.2 Constructor & Destructor Documentation

4.47.2.1 rain::rain (**parameters &** *par*)

Definition at line 45 of file rain.cpp.

4.47.2.2 rain::∼rain ( ) [virtual]

Definition at line 54 of file rain.cpp.

4.47.3 Member Function Documentation

4.47.3.1 virtual void rain::Rain_func (**SCALAR**, **TAB &**) [pure virtual]

Implemented in rain_generated, and rain_read.

4.47.4 Member Data Documentation

4.47.4.1 SCALAR rain::dx [protected]

Definition at line 63 of file rain.hpp.

4.47.4.2 SCALAR rain::dy [protected]

Definition at line 63 of file rain.hpp.

4.47.4.3 int rain::Nxcell [protected]

Definition at line 62 of file rain.hpp.

4.47.4.4 int rain::Nycell [protected]

Definition at line 62 of file rain.hpp.

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

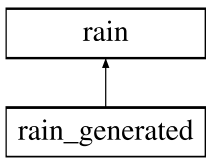
- Headers/librain_infiltration/rain.hpp
- Sources/librain_infiltration/rain.cpp

4.48 rain_generated Class Reference

Class for the initialization of the rain.

`#include <rain_generated.hpp>`

Inheritance diagram for rain_generated:



Public Member Functions

- rain_generated (parameters &)

  Constructor.
- void Rain_func (SCALAR, TAB &)
- virtual ∼rain_generated ()

4.48.1 Detailed Description

Class for the initialization of the rain.

Definition at line 62 of file rain_generated.hpp.

4.48.2 Constructor & Destructor Documentation

4.48.2.1 rain_generated::rain_generated (**parameters** & *par*)

Constructor.

Definition at line 47 of file rain_generated.cpp.

4.48.2.2 rain_generated::∼rain_generated ( ) [virtual]

Definition at line 50 of file rain_generated.cpp.

4.48.3 Member Function Documentation

4.48.3.1 void rain_generated::Rain_func (**SCALAR** *time*, **TAB** & *Tab_rain*) [virtual]

Implements rain.

Definition at line 52 of file rain_generated.cpp.

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

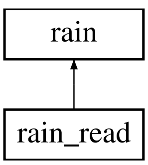
- Headers/librain_infiltration/rain_generated.hpp
- Sources/librain_infiltration/rain_generated.cpp

4.49 rain_read Class Reference

Class for the initialization of the rain from the file.

`#include <rain_read.hpp>`

Inheritance diagram for rain_read:



Public Member Functions

- rain_read (parameters &)

    Constructor.
- virtual ∼rain_read ()

    Destructor.
- void Rain_func (SCALAR, TAB &)

4.49.1 Detailed Description

Class for the initialization of the rain from the file.

Definition at line 60 of file rain_read.hpp.

4.49.2 Constructor & Destructor Documentation

4.49.2.1 rain_read::rain_read (**parameters** & *par*)

Constructor.

Definition at line 46 of file rain_read.cpp.

4.49.2.2 rain_read::∼rain_read ( ) `[virtual]`

Destructor.

Definition at line 76 of file rain_read.cpp.

4.49.3 Member Function Documentation

4.49.3.1 void rain_read::Rain_func (**SCALAR** *time*, **TAB** & *Tab_rain*) `[virtual]`

Implements rain.

Definition at line 79 of file rain_read.cpp.

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

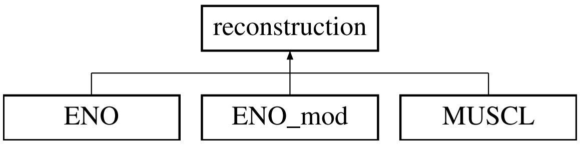
- Headers/librain_infiltration/rain_read.hpp
- Sources/librain_infiltration/rain_read.cpp

4.50 reconstruction Class Reference

class which allows to calculate the second-order reconstruction in space

`#include <reconstruction.hpp>`

Inheritance diagram for reconstruction:



Public Member Functions

- reconstruction ()

  Constructor.
- virtual void calcul (TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &)=0
- virtual ∼reconstruction ()

  Destructor.

Protected Attributes

- SCALAR delta_h1
- SCALAR delta_u1
- SCALAR delta_v1
- SCALAR delta_h2
- SCALAR delta_u2
- SCALAR delta_v2
- SCALAR dh
- SCALAR du
- SCALAR dv
- SCALAR dz_h
- SCALAR a1
- SCALAR a2
- SCALAR a3
- SCALAR a4
- TAB z1r
- TAB z1l
- TAB z2r
- TAB z2l
- TAB delta_z1
- TAB delta_z2
- choice_limiter ∗ limiter

4.50.1 Detailed Description

class which allows to calculate the second-order reconstruction in space

In order to get a second order in space, we can use a MUSCL reconstruction (Monotonic Upwind Scheme for Conservation Law), a ENO reconstruction (Essentially Non Oscillatory) or a ENO_mod reconstruction (modified Essentially Non Oscillatory). A scalar fonction $s \in R$ is reconstructed by the linear reconstruction

$$s_{i-1/2+} = s_i - \frac{\Delta x}{2} Ds_i \ \ and \quad s_{i+1/2-} = s_i + \frac{\Delta x}{2} Ds_i$$

where D is a reconstruction operator.

Definition at line 73 of file reconstruction.hpp.

4.50.2 Constructor & Destructor Documentation

4.50.2.1 reconstruction::reconstruction ( )

Constructor.

Definition at line 45 of file reconstruction.cpp.

4.50.2.2 reconstruction::∼reconstruction ( ) [virtual]

Destructor.

Definition at line 47 of file reconstruction.cpp.

4.50.3 Member Function Documentation

4.50.3.1 virtual void reconstruction::calcul (TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &) [pure virtual]

Implemented in ENO, ENO_mod, and MUSCL.

4.50.4 Member Data Documentation

4.50.4.1 SCALAR reconstruction::a1 [protected]

Definition at line 77 of file reconstruction.hpp.

4.50.4.2 SCALAR reconstruction::a2 [protected]

Definition at line 77 of file reconstruction.hpp.

4.50.4.3 SCALAR reconstruction::a3 [protected]

Definition at line 77 of file reconstruction.hpp.

4.50.4.4 SCALAR reconstruction::a4 [protected]

Definition at line 77 of file reconstruction.hpp.

4.50.4.5 SCALAR reconstruction::delta_h1 [protected]

Definition at line 75 of file reconstruction.hpp.

4.50.4.6 SCALAR reconstruction::delta_h2 [protected]

Definition at line 75 of file reconstruction.hpp.

4.50.4.7 SCALAR reconstruction::delta_u1 [protected]

Definition at line 75 of file reconstruction.hpp.

4.50.4.8 SCALAR reconstruction::delta_u2 [protected]

Definition at line 75 of file reconstruction.hpp.

4.50.4.9 SCALAR reconstruction::delta_v1 [protected]

Definition at line 75 of file reconstruction.hpp.

4.50.4.10 SCALAR reconstruction::delta_v2 [protected]

Definition at line 75 of file reconstruction.hpp.

4.50.4.11 TAB reconstruction::delta_z1 [protected]

Definition at line 79 of file reconstruction.hpp.

4.50.4.12 TAB reconstruction::delta_z2 [protected]

Definition at line 79 of file reconstruction.hpp.

4.50.4.13 SCALAR reconstruction::dh [protected]

Definition at line 76 of file reconstruction.hpp.

4.50.4.14 SCALAR reconstruction::du [protected]

Definition at line 76 of file reconstruction.hpp.

4.50.4.15 SCALAR reconstruction::dv [protected]

Definition at line 76 of file reconstruction.hpp.

4.50.4.16 SCALAR reconstruction::dz_h [protected]

Definition at line 76 of file reconstruction.hpp.

4.50.4.17 choice_limiter* reconstruction::limiter [protected]

Definition at line 80 of file reconstruction.hpp.

4.50.4.18 TAB reconstruction::z1l [protected]

Definition at line 78 of file reconstruction.hpp.

4.50.4.19 TAB reconstruction::z1r [protected]

Definition at line 78 of file reconstruction.hpp.

4.50.4.20 TAB reconstruction::z2l [protected]

Definition at line 78 of file reconstruction.hpp.

4.50.4.21 TAB reconstruction::z2r [protected]

Definition at line 78 of file reconstruction.hpp.

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

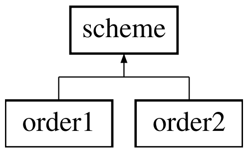
- Headers/libreconstructions/reconstruction.hpp
- Sources/libreconstructions/reconstruction.cpp

4.51 scheme Class Reference

class which allows to perform the scheme.

```
#include <scheme.hpp>
```

Inheritance diagram for scheme:



Public Member Functions

- scheme (parameters &)

  Constructor.
- void bloc1 (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR &, SCALAR)
- void bloc2 (TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, TAB &, SCALAR, SCALAR, int, SCALAR)
- SCALAR froude_number (TAB, TAB, TAB)

  Returns the Froude number.
- void allocation ()
- void desallocation ()
- virtual void calcul ()=0
- void boundary (TAB &, TAB &, TAB &, SCALAR, const int, const int)
- virtual ∼scheme ()

  Destructor.

Protected Attributes

- const int Nxcell
- const int Nycell
- const int ORDER
- int M
- int scheme_type
- SCALAR T
- SCALAR tx
- SCALAR dx
- SCALAR ty
- SCALAR dy
- SCALAR dt_fix
- SCALAR cfl_new
- SCALAR cfl_fix
- SCALAR Ks

- SCALAR dtheta
- SCALAR Psi
- SCALAR c
- SCALAR zcrust
- SCALAR Kc
- SCALAR Vol_inf
- SCALAR Vol_of
- SCALAR Vol_rain
- SCALAR source
- SCALAR f
- TAB Fric_tab
- TAB z
- TAB h
- TAB u
- TAB v
- TAB q1
- TAB q2
- TAB Vin_tot
- TAB hs
- TAB us
- TAB vs
- TAB qs1
- TAB qs2
- TAB f1
- TAB f2
- TAB f3
- TAB g1
- TAB g2
- TAB g3
- TAB h1g
- TAB h1d
- TAB h2g
- TAB h2d
- TAB delz1
- TAB delz2
- TAB h1r
- TAB u1r
- TAB v1r
- TAB h1l
- TAB u1l
- TAB v1l
- TAB h2r
- TAB u2r
- TAB v2r
- TAB h2l
- TAB u2l

- TAB v2l
- TAB delzc1
- TAB delzc2
- TAB Rain
- hydrostatic rec_hydro
- choice_condition ∗ Lbound
- choice_condition ∗ Rbound
- choice_condition ∗ Bbound
- choice_condition ∗ Tbound
- choice_flux ∗ flux_num
- choice_friction ∗ fric
- choice_infiltration ∗ I
- choice_init_topo ∗ topo
- choice_init_huv ∗ huv_init
- choice_rain ∗ Prain
- choice_output ∗ out
- string outputDirectory
- string namefile
- time_t start
- time_t end
- SCALAR timecomputation
- int n
- SCALAR dt1
- SCALAR tps
- SCALAR dt_max
- int condition
- SCALAR Fr
- int hours
- int minutes
- int seconds
- SCALAR minutesreal
- SCALAR hoursreal
- int n1
- int n2
- SCALAR left_imp_discharge
- SCALAR left_imp_h
- SCALAR right_imp_discharge
- SCALAR right_imp_h
- SCALAR bottom_imp_discharge
- SCALAR bottom_imp_h
- SCALAR top_imp_discharge
- SCALAR top_imp_h
- SCALAR fluxNycell
- SCALAR fluxNxcell
- SCALAR fluxy0
- SCALAR fluxx0

- int nchoice_Lbound
- int nchoice_Rbound
- int nchoice_Bbound
- int nchoice_Tbound
- int verif
- SCALAR dt_first
- SCALAR Total_discharge_outflow
- SCALAR Volrain_Tot

4.51.1 Detailed Description

class which allows to perform the scheme.

Definition at line 100 of file scheme.hpp.

4.51.2 Constructor & Destructor Documentation

4.51.2.1 scheme::scheme (**parameters & *par***)

Constructor.

Definition at line 48 of file scheme.cpp.

4.51.2.2 scheme::∼scheme ( ) [virtual]

Destructor.

Definition at line 174 of file scheme.cpp.

4.51.3 Member Function Documentation

4.51.3.1 void scheme::allocation ( )

Definition at line 414 of file scheme.cpp.

4.51.3.2 void scheme::bloc1 (**SCALAR *cflfix*, SCALAR *T*, SCALAR *tps*, SCALAR *dt_max*, SCALAR *dt*, SCALAR & *dt_cal*, SCALAR *cfl_new***)

Definition at line 188 of file scheme.cpp.

4.51.3.3 void scheme::bloc2 (**TAB & *he*, TAB & *ve1*, TAB & *ve2*, TAB & *qe1*, TAB & *qe2*, TAB & *hes*, TAB & *ves1*, TAB & *ves2*, TAB & *qes1*, TAB & *qes2*, TAB & *Vin*, SCALAR *tps*, SCALAR *dt*, int *n*, SCALAR *dtheta***)

Definition at line 260 of file scheme.cpp.

4.51.3.4 void scheme::boundary (TAB & *h_tmp*, TAB & *u_tmp*, TAB & *v_tmp*, SCALAR *time_tmp*, const int *nodex*, const int *nodey*)

Definition at line 352 of file scheme.cpp.

4.51.3.5 virtual void scheme::calcul ( ) [pure virtual]

Implemented in order1, and order2.

4.51.3.6 void scheme::desallocation ( )

Definition at line 555 of file scheme.cpp.

4.51.3.7 SCALAR scheme::froude_number (TAB *h_s*, TAB *u_s*, TAB *v_s*)

Returns the Froude number.

Definition at line 382 of file scheme.cpp.

4.51.4 Member Data Documentation

4.51.4.1 choice_condition* scheme::Bbound [protected]

Definition at line 138 of file scheme.hpp.

4.51.4.2 SCALAR scheme::bottom_imp_discharge [protected]

Definition at line 169 of file scheme.hpp.

4.51.4.3 SCALAR scheme::bottom_imp_h [protected]

Definition at line 169 of file scheme.hpp.

4.51.4.4 SCALAR scheme::c [protected]

Definition at line 107 of file scheme.hpp.

4.51.4.5 SCALAR scheme::cfl_fix [protected]

Definition at line 106 of file scheme.hpp.

4.51.4.6 SCALAR scheme::cfl_new [protected]

Definition at line 106 of file scheme.hpp.

4.51.4.7 int scheme::condition [protected]

Definition at line 157 of file scheme.hpp.

4.51.4.8 TAB scheme::delz1 [protected]

Definition at line 128 of file scheme.hpp.

4.51.4.9 TAB scheme::delz2 [protected]

Definition at line 128 of file scheme.hpp.

4.51.4.10 TAB scheme::delzc1 [protected]

Definition at line 131 of file scheme.hpp.

4.51.4.11 TAB scheme::delzc2 [protected]

Definition at line 131 of file scheme.hpp.

4.51.4.12 SCALAR scheme::dt1 [protected]

Definition at line 155 of file scheme.hpp.

4.51.4.13 SCALAR scheme::dt_first [protected]

Definition at line 175 of file scheme.hpp.

4.51.4.14 SCALAR scheme::dt_fix [protected]

Definition at line 106 of file scheme.hpp.

4.51.4.15 SCALAR scheme::dt_max [protected]

Definition at line 155 of file scheme.hpp.

4.51.4.16 SCALAR scheme::dtheta [protected]

Definition at line 107 of file scheme.hpp.

4.51.4.17 SCALAR scheme::dx [protected]

Definition at line 106 of file scheme.hpp.

4.51.4.18 SCALAR scheme::dy [protected]

Definition at line 106 of file scheme.hpp.

4.51.4.19 time_t scheme::end [protected]

Definition at line 150 of file scheme.hpp.

4.51.4.20 SCALAR scheme::f [protected]

Definition at line 111 of file scheme.hpp.

4.51.4.21 TAB scheme::f1 [protected]

Definition at line 123 of file scheme.hpp.

4.51.4.22 TAB scheme::f2 [protected]

Definition at line 123 of file scheme.hpp.

4.51.4.23 TAB scheme::f3 [protected]

Definition at line 123 of file scheme.hpp.

4.51.4.24 choice_flux* scheme::flux_num [protected]

Definition at line 140 of file scheme.hpp.

4.51.4.25 SCALAR scheme::fluxNxcell [protected]

Definition at line 171 of file scheme.hpp.

4.51.4.26 SCALAR scheme::fluxNycell [protected]

Definition at line 171 of file scheme.hpp.

4.51.4.27 SCALAR scheme::fluxx0 [protected]

Definition at line 171 of file scheme.hpp.

4.51.4.28 SCALAR scheme::fluxy0 [protected]

Definition at line 171 of file scheme.hpp.

4.51.4.29 SCALAR scheme::Fr [protected]

Definition at line 159 of file scheme.hpp.

4.51.4.30 choice_friction* scheme::fric [protected]

Definition at line 141 of file scheme.hpp.

4.51.4.31 TAB scheme::Fric_tab [protected]

Definition at line 116 of file scheme.hpp.

4.51.4.32 TAB scheme::g1 [protected]

Definition at line 124 of file scheme.hpp.

4.51.4.33 TAB scheme::g2 [protected]

Definition at line 124 of file scheme.hpp.

4.51.4.34 TAB scheme::g3 [protected]

Definition at line 124 of file scheme.hpp.

4.51.4.35 TAB scheme::h [protected]

Definition at line 119 of file scheme.hpp.

4.51.4.36 TAB scheme::h1d [protected]

Definition at line 126 of file scheme.hpp.

4.51.4.37 TAB scheme::h1g [protected]

Definition at line 126 of file scheme.hpp.

4.51.4.38 TAB scheme::h1l [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.39 TAB scheme::h1r [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.40 TAB scheme::h2d [protected]

Definition at line 127 of file scheme.hpp.

4.51.4.41 TAB scheme::h2g [protected]

Definition at line 127 of file scheme.hpp.

4.51.4.42 TAB scheme::h2l [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.43 TAB scheme::h2r [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.44 int scheme::hours [protected]

Definition at line 161 of file scheme.hpp.

4.51.4.45 SCALAR scheme::hoursreal [protected]

Definition at line 163 of file scheme.hpp.

4.51.4.46 TAB scheme::hs [protected]

Definition at line 121 of file scheme.hpp.

4.51.4.47 choice_init_huv* scheme::huv_init [protected]

Definition at line 144 of file scheme.hpp.

4.51.4.48 choice_infiltration* scheme::I [protected]

Definition at line 142 of file scheme.hpp.

4.51.4.49 SCALAR scheme::Kc [protected]

Definition at line 107 of file scheme.hpp.

4.51.4.50 SCALAR scheme::Ks [protected]

Definition at line 107 of file scheme.hpp.

4.51.4.51 choice_condition* scheme::Lbound [protected]

Definition at line 136 of file scheme.hpp.

4.51.4.52 SCALAR scheme::left_imp_discharge [protected]

Definition at line 169 of file scheme.hpp.

4.51.4.53 SCALAR scheme::left_imp_h [protected]

Definition at line 169 of file scheme.hpp.

4.51.4.54 int scheme::M [protected]

Definition at line 105 of file scheme.hpp.

4.51.4.55 int scheme::minutes [protected]

Definition at line 161 of file scheme.hpp.

4.51.4.56 SCALAR scheme::minutesreal [protected]

Definition at line 163 of file scheme.hpp.

4.51.4.57 int scheme::n [protected]

Definition at line 153 of file scheme.hpp.

4.51.4.58 int scheme::n1 [protected]

Definition at line 167 of file scheme.hpp.

4.51.4.59 int scheme::n2 [protected]

Definition at line 167 of file scheme.hpp.

4.51.4.60 string scheme::namefile [protected]

Definition at line 148 of file scheme.hpp.

4.51.4.61 int scheme::nchoice_Bbound [protected]

Definition at line 173 of file scheme.hpp.

4.51.4.62 int scheme::nchoice_Lbound [protected]

Definition at line 173 of file scheme.hpp.

4.51.4.63 int scheme::nchoice_Rbound [protected]

Definition at line 173 of file scheme.hpp.

4.51.4.64 int scheme::nchoice_Tbound [protected]

Definition at line 173 of file scheme.hpp.

4.51.4.65 const int scheme::Nxcell [protected]

Definition at line 102 of file scheme.hpp.

4.51.4.66 const int scheme::Nycell [protected]

Definition at line 103 of file scheme.hpp.

4.51.4.67 const int scheme::ORDER [protected]

Definition at line 104 of file scheme.hpp.

4.51.4.68 choice_output* scheme::out [protected]

Definition at line 147 of file scheme.hpp.

4.51.4.69 string scheme::outputDirectory [protected]

Definition at line 148 of file scheme.hpp.

4.51.4.70 choice_rain* scheme::Prain [protected]

Definition at line 145 of file scheme.hpp.

4.51.4.71 SCALAR scheme::Psi [protected]

Definition at line 107 of file scheme.hpp.

4.51.4.72 TAB scheme::q1 [protected]

Definition at line 120 of file scheme.hpp.

4.51.4.73 TAB scheme::q2 [protected]

Definition at line 120 of file scheme.hpp.

4.51.4.74 TAB scheme::qs1 [protected]

Definition at line 122 of file scheme.hpp.

4.51.4.75 TAB scheme::qs2 [protected]

Definition at line 122 of file scheme.hpp.

4.51.4.76 TAB scheme::Rain [protected]

Definition at line 132 of file scheme.hpp.

4.51.4.77 choice_condition* scheme::Rbound [protected]

Definition at line 137 of file scheme.hpp.

4.51.4.78 hydrostatic scheme::rec_hydro [protected]

Definition at line 135 of file scheme.hpp.

4.51.4.79 SCALAR scheme::right_imp_discharge [protected]

Definition at line 169 of file scheme.hpp.

4.51.4.80 SCALAR scheme::right_imp_h [protected]

Definition at line 169 of file scheme.hpp.

4.51.4.81 int scheme::scheme_type [protected]

Definition at line 105 of file scheme.hpp.

4.51.4.82 int scheme::seconds [protected]

Definition at line 161 of file scheme.hpp.

4.51.4.83 SCALAR scheme::source [protected]

Definition at line 111 of file scheme.hpp.

4.51.4.84 time_t scheme::start [protected]

Definition at line 150 of file scheme.hpp.

4.51.4.85 SCALAR scheme::T [protected]

Definition at line 106 of file scheme.hpp.

4.51.4.86 choice_condition* scheme::Tbound [protected]

Definition at line 139 of file scheme.hpp.

4.51.4.87 SCALAR scheme::timecomputation [protected]

Definition at line 152 of file scheme.hpp.

4.51.4.88 SCALAR scheme::top_imp_discharge [protected]

Definition at line 169 of file scheme.hpp.

4.51.4.89 SCALAR scheme::top_imp_h [protected]

Definition at line 169 of file scheme.hpp.

4.51.4.90 choice_init_topo* scheme::topo [protected]

Definition at line 143 of file scheme.hpp.

4.51.4.91 SCALAR scheme::Total_discharge_outflow [protected]

Definition at line 176 of file scheme.hpp.

4.51.4.92 SCALAR scheme::tps [protected]

Definition at line 155 of file scheme.hpp.

4.51.4.93 SCALAR scheme::tx [protected]

Definition at line 106 of file scheme.hpp.

4.51.4.94 SCALAR scheme::ty [protected]

Definition at line 106 of file scheme.hpp.

4.51.4.95 TAB scheme::u [protected]

Definition at line 119 of file scheme.hpp.

4.51.4.96 TAB scheme::u1l [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.97 TAB scheme::u1r [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.98 TAB scheme::u2l [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.99 TAB scheme::u2r [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.100 TAB scheme::us [protected]

Definition at line 121 of file scheme.hpp.

4.51.4.101 TAB scheme::v [protected]

Definition at line 119 of file scheme.hpp.

4.51.4.102 TAB scheme::v1l [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.103 TAB scheme::v1r [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.104 TAB scheme::v2l [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.105 TAB scheme::v2r [protected]

Definition at line 130 of file scheme.hpp.

4.51.4.106 int scheme::verif [protected]

Definition at line 174 of file scheme.hpp.

4.51.4.107 TAB scheme::Vin_tot [protected]

Definition at line 120 of file scheme.hpp.

4.51.4.108 SCALAR scheme::Vol_inf [protected]

Definition at line 110 of file scheme.hpp.

4.51.4.109 SCALAR scheme::Vol_of [protected]

Definition at line 110 of file scheme.hpp.

4.51.4.110 SCALAR scheme::Vol_rain [protected]

Definition at line 110 of file scheme.hpp.

4.51.4.111 SCALAR scheme::Volrain_Tot [protected]

Definition at line 176 of file scheme.hpp.

4.51.4.112 TAB scheme::vs [protected]

Definition at line 121 of file scheme.hpp.

4.51.4.113 TAB scheme::z [protected]

Definition at line 118 of file scheme.hpp.

4.51.4.114 SCALAR scheme::zcrust [protected]

Definition at line 107 of file scheme.hpp.

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

- Headers/libschemes/scheme.hpp
- Sources/libschemes/scheme.cpp

4.52 topo_generated Class Reference

Class for the initialization of Thacker's topography.

```
#include <topo_generated_thacker.hpp>
```

4.52.1 Detailed Description

Class for the initialization of Thacker's topography.

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

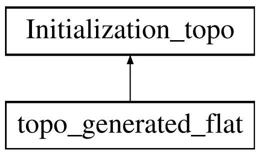
- Headers/libinitialization/topo_generated_thacker.hpp

4.53 topo_generated_flat Class Reference

Class for the initialization of the flat topography.

```
#include <topo_generated_flat.hpp>
```

Inheritance diagram for topo_generated_flat:



Public Member Functions

- topo_generated_flat (parameters &)

    Constructor.
- void initialization (TAB &)
- virtual ~topo_generated_flat ()

4.53.1 Detailed Description

Class for the initialization of the flat topography.

Definition at line 61 of file topo_generated_flat.hpp.

4.53.2 Constructor & Destructor Documentation

4.53.2.1 topo_generated_flat::topo_generated_flat (**parameters** & *par*)

Constructor.

Definition at line 47 of file topo_generated_flat.cpp.

4.53.2.2 topo_generated_flat::∼topo_generated_flat ( ) [virtual]

Definition at line 50 of file topo_generated_flat.cpp.

4.53.3 Member Function Documentation

4.53.3.1 void topo_generated_flat::initialization (TAB & *topo*) [virtual]

Implements Initialization_topo.

Definition at line 52 of file topo_generated_flat.cpp.

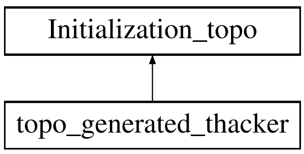
The documentation for this class was generated from the following files:

- Headers/libinitialization/topo_generated_flat.hpp
- Sources/libinitialization/topo_generated_flat.cpp

4.54 topo_generated_thacker Class Reference

#include <topo_generated_thacker.hpp>

Inheritance diagram for topo_generated_thacker:



Public Member Functions

- topo_generated_thacker (parameters &)

    Constructor.
- void initialization (TAB &)
- virtual ∼topo_generated_thacker ()

4.54.1 Detailed Description

Definition at line 61 of file topo_generated_thacker.hpp.

4.54.2 Constructor & Destructor Documentation

4.54.2.1 topo_generated_thacker::topo_generated_thacker (parameters & *par*)

Constructor.

Definition at line 46 of file topo_generated_thacker.cpp.

4.54.2.2 topo_generated_thacker::~topo_generated_thacker ( ) [virtual]

Definition at line 56 of file topo_generated_thacker.cpp.

4.54.3 Member Function Documentation

4.54.3.1 void topo_generated_thacker::initialization (TAB & *topo*) [virtual]

Implements Initialization_topo.

Definition at line 58 of file topo_generated_thacker.cpp.

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

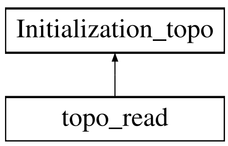
- Headers/libinitialization/topo_generated_thacker.hpp
- Sources/libinitialization/topo_generated_thacker.cpp

4.55 topo_read Class Reference

Class for the initialization of the topography read in a file.

```
#include <topo_read.hpp>
```

Inheritance diagram for topo_read:



Public Member Functions

- topo_read (parameters &)

  Constructor.
- virtual ~topo_read ()

  Destructor.
- void initialization (TAB &)

4.55.1 Detailed Description

Class for the initialization of the topography read in a file.

Definition at line 61 of file topo_read.hpp.

4.55.2 Constructor & Destructor Documentation

4.55.2.1 topo_read::topo_read (**parameters & *par***)

Constructor.

Definition at line 45 of file topo_read.cpp.

4.55.2.2 topo_read::~topo_read ( ) `[virtual]`

Destructor.

Definition at line 51 of file topo_read.cpp.

4.55.3 Member Function Documentation

4.55.3.1 void topo_read::initialization (**TAB & *topo***) `[virtual]`

Implements Initialization_topo.

Definition at line 54 of file topo_read.cpp.

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

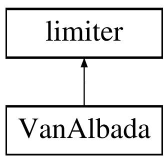
- Headers/libinitialization/topo_read.hpp
- Sources/libinitialization/topo_read.cpp

4.56 VanAlbada Class Reference

class which allows to calculate the Van Albada slope limiter of two real numbers

`#include <VanAlbada.hpp>`

Inheritance diagram for VanAlbada:



Public Member Functions

- VanAlbada ()

  Constructor.
- void calcul (SCALAR, SCALAR)

  calculates the VanAlbada of a and b
- virtual ∼VanAlbada ()

  Destructor.

4.56.1 Detailed Description

class which allows to calculate the Van Albada slope limiter of two real numbers (x,y)

$$VanAlbada(x,y) = \left\{ \begin{array}{l} 0 \; if \; sign(x) \neq sign(y) \\ \frac{x(y^2+\varepsilon)+y(x^2+\varepsilon)}{x^2+y^2+2\varepsilon} \; else \end{array} \right. ,$$

with $0 \leq \varepsilon << 1$.

Definition at line 67 of file VanAlbada.hpp.

4.56.2 Constructor & Destructor Documentation

4.56.2.1 VanAlbada::VanAlbada ( )

Constructor.

Definition at line 45 of file VanAlbada.cpp.

4.56.2.2 VanAlbada::∼VanAlbada ( ) [virtual]

Destructor.

Definition at line 57 of file VanAlbada.cpp.

4.56.3 Member Function Documentation

4.56.3.1 void VanAlbada::calcul (SCALAR *a*, SCALAR *b*) [virtual]

calculates the VanAlbada of a and b

Implements limiter.

Definition at line 49 of file VanAlbada.cpp.

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

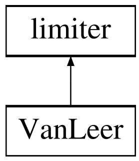
- Headers/liblimitations/VanAlbada.hpp
- Sources/liblimitations/VanAlbada.cpp

4.57 VanLeer Class Reference

class which allows to calculate the VanLeer of two real numbers (x,y)

`#include <VanLeer.hpp>`

Inheritance diagram for VanLeer:



Public Member Functions

- VanLeer ()

  Constructor.
- void calcul (SCALAR, SCALAR)

  calculates the VanLeer of a and b
- virtual ∼VanLeer ()

  Destructor.

4.57.1 Detailed Description

class which allows to calculate the VanLeer of two real numbers (x,y)

$$VL(x,y) = \begin{cases} 0 & \text{if } xy \leq 0, \\ \dfrac{2xy}{x+y} & \text{else.} \end{cases}$$

Definition at line 68 of file VanLeer.hpp.

4.57.2 Constructor & Destructor Documentation

4.57.2.1 VanLeer::VanLeer ( )

Constructor.

Definition at line 45 of file VanLeer.cpp.

4.57.2.2 VanLeer::∼VanLeer ( ) `[virtual]`

Destructor.

Definition at line 56 of file VanLeer.cpp.

4.57.3 Member Function Documentation

4.57.3.1 void VanLeer::calcul (SCALAR *a*, SCALAR *b*) [virtual]

calculates the VanLeer of a and b

Implements limiter.

Definition at line 48 of file VanLeer.cpp.

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

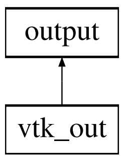
- Headers/liblimitations/VanLeer.hpp
- Sources/liblimitations/VanLeer.cpp

4.58 vtk_out Class Reference

Class which allows to write the result in the output file for vtk.

`#include <vtk_out.hpp>`

Inheritance diagram for vtk_out:



Public Member Functions

- vtk_out (parameters &)

    Constructor.
- void write (TAB, TAB, TAB, TAB, SCALAR)
- virtual ~vtk_out ()

    Destructor.

4.58.1 Detailed Description

Class which allows to write the result in the output file for vtk.

Definition at line 60 of file vtk_out.hpp.

4.58.2 Constructor & Destructor Documentation

4.58.2.1 vtk_out::vtk_out (parameters & *par*)

Constructor.

Definition at line 46 of file vtk_out.cpp.

4.58.2.2 vtk_out::~vtk_out ( ) [virtual]

Destructor.

Definition at line 211 of file vtk_out.cpp.

4.58.3 Member Function Documentation

4.58.3.1 void vtk_out::write (TAB *h*, TAB *u*, TAB *v*, TAB *z*, SCALAR *tps*) [virtual]

Implements output.

Definition at line 59 of file vtk_out.cpp.

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

- Headers/libsave/vtk_out.hpp
- Sources/libsave/vtk_out.cpp

Chapter 5

File Documentation

5.1 Headers/libboundaryconditions/bc_imp_discharge.hpp File Reference

```
#include "boundary_condition.hpp"
```

Classes

- class bc_imp_discharge

Defines

- #define Class_bc_imp_discharge

5.1.1 Define Documentation

5.1.1.1 #define Class_bc_imp_discharge

Definition at line 48 of file bc_imp_discharge.hpp.

5.2 Headers/libboundaryconditions/bc_liquid.hpp File Reference

Right liquid boundary condition.

```
#include "boundary_condition.hpp"
```

Classes

- class bc_liquid

Class which allows to calculate right liquid boundary thanks to the method of caracteristics.

5.2.1 Detailed Description

Right liquid boundary condition.

Author

Olivier Delestre

Definition in file bc_liquid.hpp.

5.3 Headers/libboundaryconditions/bc_neumann.hpp File Reference

Neumann boundary condition.

```
#include "boundary_condition.hpp"
```

Classes

- class bc_neumann

  Class which allows to calculate Neumann boundary conditions.

5.3.1 Detailed Description

Neumann boundary condition.

Author

Olivier Delestre

Definition in file bc_neumann.hpp.

5.4 Headers/libboundaryconditions/bc_periodic.hpp File Reference

Left periodic boundary condition.

```
#include "boundary_condition.hpp"
```

Classes

- class bc_periodic

  Class which allows to calculate the left periodic boundary condition.

5.4.1 Detailed Description

Left periodic boundary condition.

Author

Pierre-Antoine Ksinant

Definition in file bc_periodic.hpp.

5.5 Headers/libboundaryconditions/bc_wall.hpp File Reference

Wall boundary condition.

```
#include "boundary_condition.hpp"
```

Classes

- class bc_wall

  Class which allows to calculate wall boundary condition.

5.5.1 Detailed Description

Wall boundary condition.

Author

Olivier Delestre

Definition in file bc_wall.hpp.

5.6 Headers/libboundaryconditions/boundary_condition.hpp File Reference

Boundary condition.

```
#include "parameters.hpp"
```

Classes

- class boundary_condition

  Class which allows to calculate boundary conditions.

5.6.1 Detailed Description

Boundary condition.

Author

Olivier Delestre

Definition in file boundary_condition.hpp.

5.7 Headers/libboundaryconditions/choice_condition.hpp File Reference

Allows to choose and calculate right boundary condition.

```
#include "boundary_condition.hpp"
#include "bc_liquid.hpp"
#include "bc_wall.hpp"
#include "bc_neumann.hpp"
#include "bc_periodic.hpp"
#include "bc_imp_discharge.hpp"
```

Classes

- class choice_condition

    Class which allows to choose right boundary conditions.

Defines

- #define Class_choice_condition

5.7.1 Detailed Description

Allows to choose and calculate right boundary condition.

Author

Olivier Delestre

Definition in file choice_condition.hpp.

5.7.2 Define Documentation

5.7.2.1 #define Class_choice_condition

Definition at line 69 of file choice_condition.hpp.

5.8 Headers/libflux/choice_flux.hpp File Reference

Allows to choose and calculate numerical flux.

```
#include "flux.hpp"
#include "F_Rusanov.hpp"
#include "F_HLL.hpp"
#include "F_HLL2.hpp"
```

Classes

- class choice_flux

    class which allows to choose and calculate the numerical flux.

Defines

- #define Class_choice_flux

5.8.1 Detailed Description

Allows to choose and calculate numerical flux.

Author

Olivier Delestre

Definition in file choice_flux.hpp.

5.8.2 Define Documentation

5.8.2.1 #define Class_choice_flux

Definition at line 61 of file choice_flux.hpp.

5.9 Headers/libflux/F_HLL.hpp File Reference

Calculates the HLL flux.

```
#include "flux.hpp"
```

Classes

- class F_HLL

  class which allows to calculate HLL flux

5.9.1 Detailed Description

Calculates the HLL flux.

Author

Olivier Delestre

Definition in file F_HLL.hpp.

5.10 Headers/libflux/F_HLL2.hpp File Reference

Calculates the HLL2 flux.

```
#include "flux.hpp"
```

Classes

- class F_HLL2

  class which allows to calculate HLL flux written under a shorter form

5.10.1 Detailed Description

Calculates the HLL2 flux.

Author

Olivier Delestre

Definition in file F_HLL2.hpp.

5.11 Headers/libflux/F_Rusanov.hpp File Reference

Rusanov flux.

```
#include "flux.hpp"
```

Classes

- class F_Rusanov

  class which allows to calculate Rusanov flux

5.11.1 Detailed Description

Rusanov flux.

Author

Olivier Delestre

Definition in file F_Rusanov.hpp.

5.12 Headers/libflux/flux.hpp File Reference

Numerical flux.

```
#include "misc.hpp"
```

Classes

- class flux

  class which allows to calculate numerical flux.

5.12.1 Detailed Description

Numerical flux.

Author

Olivier Delestre

Definition in file flux.hpp.

5.13 Headers/libfrictions/choice_friction.hpp File Reference

Allows to choose and calculate the friction term.

```
#include "friction.hpp"
#include "Fr_Manning.hpp"
#include "Fr_Darcy_Weisbach.hpp"
```

Classes

- class choice_friction

  class which allows to choose and calculate the friction term.

Defines

- #define Class_choice_friction

5.13.1 Detailed Description

Allows to choose and calculate the friction term. by the semi-implicit method

Author

Olivier Delestre

Definition in file choice_friction.hpp.

5.13.2 Define Documentation

5.13.2.1 #define Class_choice_friction

Definition at line 57 of file choice_friction.hpp.

5.14 Headers/libfrictions/Fr_Darcy_Weisbach.hpp File Reference

Calculates the Darcy-Weisbach friction term.

```
#include "friction.hpp"
```

Classes

- class Fr_Darcy_Weisbach

  class which allows to calculate Darcy-Weisbach friction term

5.14.1 Detailed Description

Calculates the Darcy-Weisbach friction term. by the semi-implicit method

Author

Olivier Delestre

Definition in file Fr_Darcy_Weisbach.hpp.

5.15 Headers/libfrictions/Fr_Manning.hpp File Reference

Calculates the Manning friction term.

```
#include "friction.hpp"
```

Classes

- class Fr_Manning

  class which allows to calculate Manning friction term

5.15.1 Detailed Description

Calculates the Manning friction term. by the semi-implicit method

Author

Olivier Delestre

Definition in file Fr_Manning.hpp.

5.16 Headers/libfrictions/friction.hpp File Reference

Calculates the friction term.

```
#include "misc.hpp"
```

Classes

- class friction

  class which allows to calculate the friction term.

5.16.1 Detailed Description

Calculates the friction term. by the semi-implicit method

Author

Olivier Delestre

Definition in file friction.hpp.

5.17 Headers/libinitialization/choice_init_huv.hpp File Reference

Allows to choose and calculate the initial conditions on the water height and the velocity.

```
#include "Initialization_huv.hpp"
#include "huv_read.hpp"
#include "huv_generated.hpp"
#include "huv_generated_thacker.hpp"
#include "huv_generated_Radial_Dam_dry.hpp"
#include "huv_generated_Radial_Dam_wet.hpp"
```

Classes

- class choice_init_huv

    Class which allows to choose and calculate the initial conditions on the water height and the velocity.

Defines

- #define Class_init_huv

5.17.1 Detailed Description

Allows to choose and calculate the initial conditions on the water height and the velocity.

Author

Olivier Delestre

Definition in file choice_init_huv.hpp.

5.17.2 Define Documentation

5.17.2.1 #define Class_init_huv

Definition at line 68 of file choice_init_huv.hpp.

5.18 Headers/libinitialization/choice_init_topo.hpp File Reference

Allows to choose and calculate the initial conditions on the topography.

```
#include "Initialization_topo.hpp"
#include "topo_read.hpp"
#include "topo_generated_flat.hpp"
#include "topo_generated_thacker.hpp"
```

Classes

- class choice_init_topo

  Class which allows to choose and calculate the initial conditions on the topography.

Defines

- #define Class_init_topo

5.18.1 Detailed Description

Allows to choose and calculate the initial conditions on the topography.

Author

Olivier Delestre

Definition in file choice_init_topo.hpp.

5.18.2 Define Documentation

5.18.2.1 #define Class_init_topo

Definition at line 60 of file choice_init_topo.hpp.

5.19 Headers/libinitialization/huv_generated.hpp File Reference

Initialization of the water height and velocity thanks to a function.

```
#include "Initialization_huv.hpp"
```

Classes

- class huv_generated

  Class for the initialization of the water height and velocity thanks to a function.

5.19.1 Detailed Description

Initialization of the water height and velocity thanks to a function. Initialization of the water height and velocity for Thacker's benchmark.

Initialization of the water height and velocity for Dressler's Dam-break benchmark.

Author

Olivier Delestre
Christian Laguerre

Definition in file huv_generated.hpp.

5.20 Headers/libinitialization/huv_generated_Radial_Dam_dry.hpp File Reference

```
#include "Initialization_huv.hpp"
```

Classes

- class huv_generated_Radial_Dam_dry

5.21 Headers/libinitialization/huv_generated_Radial_Dam_wet.hpp File Reference

```
#include "Initialization_huv.hpp"
```

Classes

- class huv_generated_Radial_Dam_wet

5.22 Headers/libinitialization/huv_generated_thacker.hpp File Reference

```
#include "Initialization_huv.hpp"
```

Classes

- class huv_generated_thacker

5.23 Headers/libinitialization/huv_read.hpp File Reference

Initialization of the water height and velocity read in a file.

```
#include "Initialization_huv.hpp"
```

Classes

- class huv_read

  Class for the initialization of the water height and velocity read in a file.

5.23.1 Detailed Description

Initialization of the water height and velocity read in a file.

Author

Olivier Delestre

Definition in file huv_read.hpp.

5.24 Headers/libinitialization/Initialization_huv.hpp File Reference

Initialization of the water height and velocity.

```
#include "parameters.hpp"
```

Classes

- class Initialization_huv

  Class for the initialization of the water height and velocity.

5.24.1 Detailed Description

Initialization of the water height and velocity.

Author

Olivier Delestre

Definition in file Initialization_huv.hpp.

5.25 Headers/libinitialization/Initialization_topo.hpp File Reference

Initialization of the topography.

```
#include "parameters.hpp"
```

Classes

- class Initialization_topo

  Class for the initialization of the topography.

5.25.1 Detailed Description

Initialization of the topography.

Author

Olivier Delestre

Definition in file Initialization_topo.hpp.

5.26 Headers/libinitialization/topo_generated_flat.hpp File Reference

Initialization of the flat topography.

```
#include "Initialization_topo.hpp"
```

Classes

- class topo_generated_flat

  Class for the initialization of the flat topography.

5.26.1 Detailed Description

Initialization of the flat topography.

Author

Christian Laguerre

Definition in file topo_generated_flat.hpp.

5.27 Headers/libinitialization/topo_generated_thacker.hpp File Reference

```
#include "Initialization_topo.hpp"
```

Classes

- class topo_generated_thacker

5.28 Headers/libinitialization/topo_read.hpp File Reference

Initialization of the topography read in a file.

```
#include "Initialization_topo.hpp"
```

Classes

- class topo_read
  Class for the initialization of the topography read in a file.

5.28.1 Detailed Description

Initialization of the topography read in a file.

Author

Olivier Delestre

Definition in file topo_read.hpp.

5.29 Headers/liblimitations/choice_limiter.hpp File Reference

Allows to choose and calculate a slope limiter.

```
#include "limiter.hpp"
#include "Minmod.hpp"
#include "VanAlbada.hpp"
#include "VanLeer.hpp"
```

Classes

- class choice_limiter

    class which allows to choose and calculate the slope limiter.

Defines

- #define Class_choice_limiter

5.29.1 Detailed Description

Allows to choose and calculate a slope limiter.

Author

Olivier Delestre

Definition in file choice_limiter.hpp.

5.29.2 Define Documentation

5.29.2.1 #define Class_choice_limiter

Definition at line 60 of file choice_limiter.hpp.

5.30 Headers/liblimitations/limiter.hpp File Reference

Allows to calculate the slope limiter.

```
#include "misc.hpp"
```

Classes

- class limiter

    class which allows to calculate the slope limiter.

5.30.1 Detailed Description

Allows to calculate the slope limiter.

Author

Olivier Delestre

Definition in file limiter.hpp.

5.31 Headers/liblimitations/Minmod.hpp File Reference

Minmod slope limiter.

```
#include "limiter.hpp"
```

Classes

- class Minmod

  Class which allows to calculate the minmod of two real numbers (x, y).

5.31.1 Detailed Description

Minmod slope limiter.

Author

Olivier Delestre

Definition in file Minmod.hpp.

5.32 Headers/liblimitations/VanAlbada.hpp File Reference

VanAlbada slope limiter.

```
#include "limiter.hpp"
```

Classes

- class VanAlbada

  class which allows to calculate the Van Albada slope limiter of two real numbers

5.32.1 Detailed Description

VanAlbada slope limiter.

Author

Olivier Delestre

Definition in file VanAlbada.hpp.

5.33 Headers/liblimitations/VanLeer.hpp File Reference

VanLeer slope limiter.

```
#include "limiter.hpp"
```

Classes

- class VanLeer

  class which allows to calculate the VanLeer of two real numbers (x, y)

5.33.1 Detailed Description

VanLeer slope limiter.

Author

Olivier Delestre

Definition in file VanLeer.hpp.

5.34 Headers/libparameters/misc.hpp File Reference

```
#include <vector>
#include <iostream>
#include <cmath>
#include <fstream>
#include <string>
#include <cstring>
#include <cstdlib>
#include <iomanip>
#include <sstream>
#include <cfloat>
```

Defines

- #define max(a, b) (a>=b?a:b)
- #define min(a, b) (a<=b?a:b)
- #define grav 9.81
- #define grav_dem 4.905
- #define const_cfl_x 0.5
- #define const_cfl_y 0.5
- #define he_ca 1.e-12
- #define ve_ca 1.e-12
- #define max_cfl_x 0.
- #define max_cfl_y 0.
- #define NB_CHAR 256
- #define zero 0.
- #define le_ca 1.e-8
- #define version "FullSWOF_2D - version v1.02.00 - January, 20, 2012"

Typedefs

- typedef double SCALAR
- typedef vector< vector< SCALAR > > TAB

5.34.1 Define Documentation

5.34.1.1 #define const_cfl_x 0.5

Definition at line 59 of file misc.hpp.

5.34.1.2 #define const_cfl_y 0.5

Definition at line 60 of file misc.hpp.

5.34.1.3 #define grav 9.81

Definition at line 57 of file misc.hpp.

5.34.1.4 #define grav_dem 4.905

Definition at line 58 of file misc.hpp.

5.34.1.5 #define he_ca 1.e-12

Definition at line 61 of file misc.hpp.

5.34.1.6 #define le_ca 1.e-8

Definition at line 67 of file misc.hpp.

5.34.1.7 #define max(*a*, *b*) (a>=b?a:b)

Definition at line 54 of file misc.hpp.

5.34.1.8 #define max_cfl_x 0.

Definition at line 63 of file misc.hpp.

5.34.1.9 #define max_cfl_y 0.

Definition at line 64 of file misc.hpp.

5.34.1.10 #define min(*a*, *b*) (a<=b?a:b)

Definition at line 55 of file misc.hpp.

5.34.1.11 #define NB_CHAR 256

Definition at line 65 of file misc.hpp.

5.34.1.12 #define ve_ca 1.e-12

Definition at line 62 of file misc.hpp.

5.34.1.13 #define version "FullSWOF_2D - version v1.02.00 - January, 20, 2012"

Definition at line 68 of file misc.hpp.

5.34.1.14 #define zero 0.

Definition at line 66 of file misc.hpp.

5.34.2 Typedef Documentation

5.34.2.1 typedef double SCALAR

Definition at line 72 of file misc.hpp.

5.34.2.2 typedef vector< vector< SCALAR > > TAB

Definition at line 73 of file misc.hpp.

5.35 Headers/libparameters/parameters.hpp File Reference

Gets the paramters.

`#include "misc.hpp"`

Classes

- class parameters

  Class which gets the initialization parameters.

5.35.1 Detailed Description

Gets the paramters.

Author

Olivier Delestre

Definition in file parameters.hpp.

5.36 Headers/libparser/parser.hpp File Reference

`#include "misc.hpp"`

Classes

- class parser

  Class that reads the input file writen as description <variable>:: value # comment and keep the values after the "::" ignoring the comments that begin with a "#".

5.37 Headers/librain_infiltration/choice_infiltration.hpp File Reference

Allows to choose the infiltration and to calculate the infiltrated volume and the volume of water available.

`#include "infiltration.hpp"`

`#include "GreenAmpt.hpp"`

Classes

- class choice_infiltration

  Class which allows to choose the infiltration and to calculate the infiltrated volume the volume of water available.

Defines

- #define Class_choice_infiltration

5.37.1 Detailed Description

Allows to choose the infiltration and to calculate the infiltrated volume and the volume of water available.

Author

Marie Rousseau

Definition in file choice_infiltration.hpp.

5.37.2 Define Documentation

5.37.2.1 #define Class_choice_infiltration

Definition at line 53 of file choice_infiltration.hpp.

5.38 Headers/librain_infiltration/choice_rain.hpp File Reference

Allows to choose and calculate the initial conditions of the rain.

```
#include "rain.hpp"
#include "rain_read.hpp"
#include "rain_generated.hpp"
```

Classes

- class choice_rain

  Class which allows to choose and calculate the initial conditions of the rain.

Defines

- #define Class_choice_rain

5.38.1 Detailed Description

Allows to choose and calculate the initial conditions of the rain.

Author

Christian Laguerre

Definition in file choice_rain.hpp.

5.38.2 Define Documentation

5.38.2.1 #define Class_choice_rain

Definition at line 57 of file choice_rain.hpp.

5.39 Headers/librain_infiltration/GreenAmpt.hpp File Reference

Calculates the infiltration capacity and infiltrated volum.

```
#include "infiltration.hpp"
```

Classes

- class GreenAmpt

  Class which allows to calculate the infiltration capacity and infiltrated volum.

5.39.1 Detailed Description

Calculates the infiltration capacity and infiltrated volum.

Author

Marie Rousseau

Definition in file GreenAmpt.hpp.

5.40 Headers/librain_infiltration/infiltration.hpp File Reference

```
#include "parameters.hpp"
```

Classes

- class infiltration

  Class which allows to calculate the infiltration rate thanks to Green-Ampt model.

5.40.1 Detailed Description

Calculates the infiltration capacity and the volum infiltrated by Green-Ampt model

Author

Marie Rousseau

Definition in file infiltration.hpp.

5.41 Headers/librain_infiltration/rain.hpp File Reference

Allows to choose the rain.

```
#include "parameters.hpp"
```

Classes

- class rain

  Class which allows to choose the rain.

5.41.1 Detailed Description

Allows to choose the rain.

Author

Marie Rousseau

Definition in file rain.hpp.

5.42 Headers/librain_infiltration/rain_generated.hpp File Reference

Initialization of the rain.

```
#include "rain.hpp"
```

Classes

- class rain_generated

  Class for the initialization of the rain.

5.42.1 Detailed Description

Initialization of the rain.

Author

Christian Laguerre

Definition in file rain_generated.hpp.

5.43 Headers/librain_infiltration/rain_read.hpp File Reference

Initialization of the rain from the file.

```
#include "rain.hpp"
```

Classes

- class rain_read
  Class for the initialization of the rain from the file.

5.43.1 Detailed Description

Initialization of the rain from the file.

Author

Christian Laguerre

Definition in file rain_read.hpp.

5.44 Headers/libreconstructions/choice_reconstruction.hpp File Reference

Allows to choose and calculate the second order reconstruction in space.

```
#include "reconstruction.hpp"
#include "MUSCL.hpp"
#include "ENO.hpp"
#include "ENO_mod.hpp"
```

Classes

- class choice_reconstruction
  Class which allows to choose and calculate the second order reconstruction in space.

Defines

- #define Class_choice_reconstruction

5.44.1 Detailed Description

Allows to choose and calculate the second order reconstruction in space.

Author

Olivier Delestre

Definition in file choice_reconstruction.hpp.

5.44.2 Define Documentation

5.44.2.1 #define Class_choice_reconstruction

Definition at line 61 of file choice_reconstruction.hpp.

5.45 Headers/libreconstructions/ENO.hpp File Reference

Allows to perform the ENO reconstruction in space.

```
#include "reconstruction.hpp"
```

Classes

- class ENO

  class which allows to calculate the ENO reconstruction in space

5.45.1 Detailed Description

Allows to perform the ENO reconstruction in space.

Author

Olivier Delestre

Definition in file ENO.hpp.

5.46 Headers/libreconstructions/ENO_mod.hpp File Reference

Allows to perform the modified ENO reconstruction in space.

```
#include "reconstruction.hpp"
```

Classes

- class ENO_mod
  class which allows to calculate the modified ENO reconstruction in space.

5.46.1 Detailed Description

Allows to perform the modified ENO reconstruction in space.

Author

Olivier Delestre

Definition in file ENO_mod.hpp.

5.47 Headers/libreconstructions/hydrostatic.hpp File Reference

Performs the hydrostatic reconstruction.

```
#include "misc.hpp"
```

Classes

- class hydrostatic
  class which allows to perform the hydrostatic reconstruction.

5.47.1 Detailed Description

Performs the hydrostatic reconstruction.

Author

Olivier Delestre

Definition in file hydrostatic.hpp.

5.48 Headers/libreconstructions/MUSCL.hpp File Reference

Allows to perform the MUSCL reconstruction in space.

```
#include "reconstruction.hpp"
```

Classes

- class MUSCL
  class which allows to calculate MUSCL reconstruction in space

5.48.1 Detailed Description

Allows to perform the MUSCL reconstruction in space.

Author

Olivier Delestre

Definition in file MUSCL.hpp.

5.49 Headers/libreconstructions/reconstruction.hpp File Reference

Allows to perform second-order reconstruction in space.

```
#include "parameters.hpp"
#include "choice_limiter.hpp"
```

Classes

- class reconstruction

  class which allows to calculate the second-order reconstruction in space

5.49.1 Detailed Description

Allows to perform second-order reconstruction in space.

Author

Olivier Delestre

Definition in file reconstruction.hpp.

5.50 Headers/libsave/choice_output.hpp File Reference

Allows to choose the style of the output file.

```
#include "output.hpp"
#include "gnuplot.hpp"
#include "vtk_out.hpp"
```

Classes

- class choice_output

  Class which allows to choose the style of the output file.

Defines

- #define Class_choice_output

5.50.1 Detailed Description

Allows to choose the style of the output file.

Author

Christian Laguerre

Definition in file choice_output.hpp.

5.50.2 Define Documentation

5.50.2.1 #define Class_choice_output

Definition at line 56 of file choice_output.hpp.

5.51 Headers/libsave/gnuplot.hpp File Reference

Write the result in the output file for gnuplot.

```
#include "output.hpp"
```

Classes

- class gnuplot

  Class which allows to write the result in the output file for gnuplot.

5.51.1 Detailed Description

Write the result in the output file for gnuplot.

Author

Christian Laguerre

Definition in file gnuplot.hpp.

5.52 Headers/libsave/output.hpp File Reference

Write the result in the output file.

```
#include "parameters.hpp"
```

Classes

- class output

  Class which allows to write the result in the output file.

5.52.1 Detailed Description

Write the result in the output file.

Author

Christian Laguerre

Definition in file output.hpp.

5.53 Headers/libsave/vtk_out.hpp File Reference

Write the result in the output file for vtk.

```
#include "output.hpp"
```

Classes

- class vtk_out

  Class which allows to write the result in the output file for vtk.

5.53.1 Detailed Description

Write the result in the output file for vtk.

Author

Christian Laguerre

Definition in file vtk_out.hpp.

5.54 Headers/libschemes/choice_scheme.hpp File Reference

Allows to choose the scheme and to perform calculations.

```
#include "scheme.hpp"
#include "order1.hpp"
#include "order2.hpp"
```

Classes

- class choice_scheme

  class which allows to choose the scheme and to perform calculations.

Defines

- #define Class_choice_scheme

5.54.1 Detailed Description

Allows to choose the scheme and to perform calculations.

Author

Olivier Delestre

Definition in file choice_scheme.hpp.

5.54.2 Define Documentation

5.54.2.1 #define Class_choice_scheme

Definition at line 57 of file choice_scheme.hpp.

5.55 Headers/libschemes/order1.hpp File Reference

Performs the first order scheme.

```
#include "scheme.hpp"
```

Classes

- class order1

  class which allows to perform the first order scheme

5.55.1 Detailed Description

Performs the first order scheme.

Author

Olivier Delestre

Definition in file order1.hpp.

5.56 Headers/libschemes/order2.hpp File Reference

Performs the second order scheme.

```
#include "choice_reconstruction.hpp"
#include "scheme.hpp"
```

Classes

- class order2

  class which allows to perform the second order scheme

5.56.1 Detailed Description

Performs the second order scheme.

Author

Olivier Delestre

Definition in file order2.hpp.

5.57 Headers/libschemes/scheme.hpp File Reference

Performs the scheme.

```
#include "parameters.hpp"
#include "hydrostatic.hpp"
#include "choice_condition.hpp"
#include "choice_flux.hpp"
#include "choice_friction.hpp"
#include "choice_infiltration.hpp"
#include "choice_init_topo.hpp"
#include "choice_init_huv.hpp"
#include "choice_rain.hpp"
#include "choice_output.hpp"
```

Classes

- class scheme

  class which allows to perform the scheme.

5.57.1 Detailed Description

Performs the scheme.

Author

Olivier Delestre, Christian Laguerre and Pierre-Antoine Ksinant

Definition in file scheme.hpp.

5.58 Sources/FullSWOF_2D.cpp File Reference

Main function Allows to declare the scheme and Execute the program.

```
#include <iostream>
#include <stdio.h>
#include <string>
#include <fstream>
#include "choice_scheme.hpp"
#include "parameters.hpp"
```

Functions

- int main (int argc, char **argv)

5.58.1 Detailed Description

Main function Allows to declare the scheme and Execute the program.

Author

Christian Laguerre - First version : Olivier Delestre

Version

1.0

Definition in file FullSWOF_2D.cpp.

5.58.2 Function Documentation

5.58.2.1 int main (int *argc*, char ** *argv*)

Definition at line 62 of file FullSWOF_2D.cpp.

5.59 Sources/libboundaryconditions/bc_imp_discharge.cpp File Reference

```
#include <iostream>
#include <cmath>
#include <complex>
#include "bc_imp_discharge.hpp"
```

5.60 Sources/libboundaryconditions/bc_liquid.cpp File Reference

```
#include <iostream>
#include <cmath>
#include "bc_liquid.hpp"
```

5.61 Sources/libboundaryconditions/bc_neumann.cpp File Reference

```
#include <iostream>
#include <cmath>
#include <complex>
#include "bc_neumann.hpp"
```

5.62 Sources/libboundaryconditions/bc_periodic.cpp File Reference

```
#include <iostream>
#include <cmath>
#include <complex>
#include "bc_periodic.hpp"
```

5.63 Sources/libboundaryconditions/bc_wall.cpp File Reference

```
#include <iostream>
#include <cmath>
#include <complex>
#include "bc_wall.hpp"
```

5.64 Sources/libboundaryconditions/boundary_condition.cpp File Reference

```
#include <iostream>
#include <cmath>
#include "boundary_condition.hpp"
```

5.65 Sources/libboundaryconditions/choice_condition.cpp File Reference

```
#include <iostream>
#include <cmath>
#include "choice_condition.hpp"
```

5.66 Sources/libflux/choice_flux.cpp File Reference

```
#include "choice_flux.hpp"
```

5.67 Sources/libflux/F_HLL.cpp File Reference

```
#include "F_HLL.hpp"
```

5.68 Sources/libflux/F_HLL2.cpp File Reference

```
#include "F_HLL2.hpp"
```

5.69 Sources/libflux/F_Rusanov.cpp File Reference

```
#include "F_Rusanov.hpp"
```

5.70 Sources/libflux/flux.cpp File Reference

```
#include "flux.hpp"
```

5.71 Sources/libfrictions/choice_friction.cpp File Reference

```
#include "choice_friction.hpp"
```

5.72 Sources/libfrictions/Fr_Darcy_Weisbach.cpp File Reference

```
#include "Fr_Darcy_Weisbach.hpp"
```

5.73 Sources/libfrictions/Fr_Manning.cpp File Reference

```
#include "Fr_Manning.hpp"
```

5.74 Sources/libfrictions/friction.cpp File Reference

```
#include "friction.hpp"
```

5.75 Sources/libinitialization/choice_init_huv.cpp File Reference

```
#include "choice_init_huv.hpp"
```

5.76 Sources/libinitialization/choice_init_topo.cpp File Reference

```
#include "choice_init_topo.hpp"
```

5.77 Sources/libinitialization/huv_generated.cpp File Reference

```
#include "huv_generated.hpp"
```

5.78 Sources/libinitialization/huv_generated_Radial_Dam_dry.cpp File Reference

```
#include "huv_generated_Radial_Dam_dry.hpp"
```

5.79 Sources/libinitialization/huv_generated_Radial_Dam_wet.cpp File Reference

```
#include "huv_generated_Radial_Dam_wet.hpp"
```

5.80 Sources/libinitialization/huv_generated_thacker.cpp File Reference

```
#include "huv_generated_thacker.hpp"
```

5.81 Sources/libinitialization/huv_read.cpp File Reference

```
#include "huv_read.hpp"
```

5.82 Sources/libinitialization/Initialization_huv.cpp File Reference

```
#include "Initialization_huv.hpp"
```

5.83 Sources/libinitialization/Initialization_topo.cpp File Reference

```
#include "Initialization_topo.hpp"
```

5.84 Sources/libinitialization/topo_generated_flat.cpp File Reference

```
#include "topo_generated_flat.hpp"
```

5.85 Sources/libinitialization/topo_generated_thacker.cpp File Reference

```
#include "topo_generated_thacker.hpp"
```

5.86 Sources/libinitialization/topo_read.cpp File Reference

```
#include "topo_read.hpp"
```

5.87 Sources/liblimitations/choice_limiter.cpp File Reference

```
#include "choice_limiter.hpp"
```

5.88 Sources/liblimitations/limiter.cpp File Reference

```
#include "limiter.hpp"
```

5.89 Sources/liblimitations/Minmod.cpp File Reference

```
#include "Minmod.hpp"
```

5.90 Sources/liblimitations/VanAlbada.cpp File Reference

```
#include "VanAlbada.hpp"
```

5.91 Sources/liblimitations/VanLeer.cpp File Reference

```
#include "VanLeer.hpp"
```

5.92 Sources/libparameters/parameters.cpp File Reference

```
#include <fstream>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <vector>
#include <iostream>
#include <unistd.h>
#include "parameters.hpp"
#include "parser.hpp"
```

5.93 Sources/libparser/parser.cpp File Reference

```
#include <stdio.h>
```

```
#include <vector>
#include <fstream>
#include <cstdlib>
#include <iostream>
#include <cmath>
#include "parser.hpp"
```

5.94 Sources/librain_infiltration/choice_infiltration.cpp File Reference

```
#include "choice_infiltration.hpp"
```

5.95 Sources/librain_infiltration/choice_rain.cpp File Reference

```
#include "choice_rain.hpp"
```

5.96 Sources/librain_infiltration/GreenAmpt.cpp File Reference

```
#include "GreenAmpt.hpp"
```

5.97 Sources/librain_infiltration/infiltration.cpp File Reference

```
#include "infiltration.hpp"
```

5.98 Sources/librain_infiltration/rain.cpp File Reference

```
#include "rain.hpp"
```

5.99 Sources/librain_infiltration/rain_generated.cpp File Reference

```
#include "rain_generated.hpp"
```

5.100 Sources/librain_infiltration/rain_read.cpp File Reference

```
#include "rain_read.hpp"
```

5.101 Sources/libreconstructions/choice_reconstruction.cpp File Reference

```
#include "choice_reconstruction.hpp"
```

5.102 Sources/libreconstructions/ENO.cpp File Reference

```
#include "ENO.hpp"
```

5.103 Sources/libreconstructions/ENO_mod.cpp File Reference

```
#include "ENO_mod.hpp"
```

5.104 Sources/libreconstructions/hydrostatic.cpp File Reference

```
#include "hydrostatic.hpp"
```

5.105 Sources/libreconstructions/MUSCL.cpp File Reference

```
#include "MUSCL.hpp"
```

5.106 Sources/libreconstructions/reconstruction.cpp File Reference

```
#include "reconstruction.hpp"
```

5.107 Sources/libsave/choice_output.cpp File Reference

```
#include "choice_output.hpp"
```

5.108 Sources/libsave/gnuplot.cpp File Reference

```
#include "gnuplot.hpp"
```

5.109 Sources/libsave/output.cpp File Reference

```
#include "output.hpp"
```

5.110 Sources/libsave/vtk_out.cpp File Reference

```
#include "vtk_out.hpp"
```

5.111 Sources/libschemes/choice_scheme.cpp File Reference

```
#include "choice_scheme.hpp"
```

5.112 Sources/libschemes/order1.cpp File Reference

```
#include "order1.hpp"
```

5.113 Sources/libschemes/order2.cpp File Reference

```
#include "order2.hpp"
```

5.114 Sources/libschemes/scheme.cpp File Reference

```
#include "scheme.hpp"
```

Index